Otto-von-Guericke-University Magdeburg
Faculty for Electrical Engineering and Information Technology
Institute for Information Technology and Communications
Mobile Dialog Systems

Master Thesis

Building of Conversational Agents using Goal-oriented Dialog Managers

submitted on: 16.03.2020

Submitted by: Saran Karthikeyan(209735)
born on 04.02.1992
in Chidambaram, India

Supervisors: Jun.-Prof. Dr.-Ing. Ingo Siegert
Dipl.- Inf. Marcus Petersen (regiocom SE)
First Examiner: Jun.-Prof. Dr.-Ing. Ingo Siegert
Second Examiner: Prof. Dr. rer. nat. A. Wendemuth

Abstract

The call centers require automation for customer services to provide cost-efficient, reliable, and user-friendly self-services for domain-specific tasks in an interactive manner. The automation in customer services supports call-center agents to eliminate repetitive tasks and also reduce costs for the call center companies. Human-Human conversations are a reliable source to build conversational agents to generate natural responses with proper greetings to satisfy the customers. In conventional conversational agents, the domain-specific hand-crafting methods have employed to achieve specific goals. End-to-end conversational models escape from this hindrance by learning from the dialogues themselves, but they are limited to the dependency on the resource of the dataset.

The open-source dataset has selected, which has human to human conversations, and the domain-specific conversation has extracted. The restaurant reservation domain, which serves as a prototype to build Interactive Conversational Agents, has taken to build Interactive Conversational Agents along with handling business logic in the natural responses. Several components have trained separately to interpret the user's intention from the utterances, predict the next system action, and generate a proper system response for each turns in the conversation to attain the goal. Hence, the limitations of both conventional and End-to-End learning systems have avoided. The technical errors within that business logic integration also have handled by providing a suitable response for the technical error which has faced while having the conversation with the user.

The Recurrent Neural Networks play a prominent role in designing Interactive Conversational Agents for automation with the requirements mentioned above. Also, the advancements in deep learning techniques have motivated the sector of automation, and digitalization, which plays a crucial role in the Revenue growth of a company. Thus, a customized variant of sequence-to-sequence model architecture has employed to predict the system's next action, which uses Long Short Term Memory cells. The customized architecture has designed to learn progressively using Progressive Neural Network architecture. Hence, the gate mechanism has also modified according to the specifications of the Progressive Neural Network. Thus, the proposed **progressive sequence-to-sequence learning architecture** for **Dialogue Management** uses a transfer learning technique to train progressively of different dialogues either in a single domain or it can also be used to train multi-domain dialogues. To achieve naturalness in responses, the generation of responses has also automatized. To achieve **automated responses**, the **Progressive sequence-to-sequence learning with an Attention mechanism** has proposed. The limitations of the proposed strategy for achieving automated responses have discussed, and as an alternative, the sequence-to-sequence learning technique has implemented for generating automated responses. Two Interactive Conversational Agents have built with a difference in the design technique to enhance the performance. Also, the dataset has modified manually, and the influence of data distribution on the quality of the developed products have discussed.

The obtained results have compared and discussed with the existing methods. Then, the built conversational agents have evaluated in a live user setting by a total of 14 participants, with equivalent professional and naive participants. The demographic perspective has considered in choosing the participants with the factors, including age group, and sex as well. Each user has participated in two surveys after interacting with each system, one is a web-based survey, and the other is a document-based survey. The Attractiveness of each system has measured from the web-based survey, and the usability metrics complying with ISO standards have measured by the document-based survey. The objective measures from the collective subjective assessment have also determined from the document-based survey. The complete results that have obtained from both the surveys have depicted.

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG | EIT | FAKULTÄT FÜR ELEKTROTECHNIK UND INFORMATIONSTECHNIK

Master Thesis

for Mr. Saran Karthikeyan
(Matriculation Number: 209735)

Topic: Building of Conversational Agents using Goal-oriented Dialog Managers

Aim of the Thesis:
The use of conversational dialog systems with the goal-oriented Dialog Management strategies is an emergent task for a local call centers company. The aim is to get an interpretation of the user's intention from the dialog acts, and generate a proper response. To avoid inappropriate responses, the responses must be approved by humans. The conversational system has to include results from external APIs in responses and has to handle technical errors within that API. In the end the system has to identify the following tasks and should be able to transfer the conversations with the contexts of spoken dialogue to human agents if it is unable to handle the users' request with a proper response.

Tasks:
The main task of this thesis is to develop a stand-alone prototype for a domain specific and goal-oriented conversational agent with a Dialog Management strategy with requirements predefined by 'regiocom SE' and to evaluate whether the developed conversational agent is able to achieve the goal of the users'

- Selection of an available corpus, and classifying the Dialog Acts from the corpus data using available tools or an approach based on literature search
- Implementing a Dialog Management strategy using a statistical approach or Deterministic rule-based techniques based on literature research
- Building a domain specific conversational agent using Natural Language Processing techniques (using available tools or model-based training based on literature research) with a proposed Dialog Management strategy
- Dialog Management strategy has to be implemented according to the requirements from regiocom SE, which will be defined together with the Student at the beginning of the thesis
- Including results from external APIs in responses and handling technical errors within that API
- Evaluation of the developed model using suitable metrics (success rate, average turn number) within an live test with at least 10 (5 professional + 5 naive) participants

The results of this project have to be reported in a written thesis and defended during an oral presentation.

Magdeburg, 07.08.2019

Date of issue:	26.08.2019
Date of submission:	13.01.2020
First Examiner:	Jun.-Prof. Dr.-Ing. I. Siegert
Second Examiner:	Prof. Dr. rer. nat. A. Wendemuth
Supervisors:	Jun.-Prof. Dr.-Ing. I. Siegert
	Dipl.- Inf. Marcus Petersen (regiocom SE)

.....
Jun.-Prof. Dr.-Ing Ingo Siegert
Assignor of the task (First Examiner)

.....
Prof. Dr.-Ing. Roberto Leidhold
Chair of the Examination Board

Acknowledgement

I express my gratitude to Prof. Dr. rer. nat. Andreas Wendemuth and Jun.-Prof. Dr.-Ing. Ingo Siegert for making an arrangement to do my Master Thesis at snt-regiocom Magdeburg GmbH. I am also grateful to Dr. Florian Marquardt for giving me an opportunity to do my Master Thesis at snt-regiocom Magdeburg GmbH. I thank Dr. Steffen Jung and Dipl.- Inf. Marcus Petersen for accepting to offer me a Master Thesis topic. I thank Mr. Marcus Petersen and my colleagues for helping me in defining the Master Thesis topic suitable for business case and research work.

I thank Mr. Marcus Petersen and my colleague Matthias Busch for helping me to understand the business logics involved in my Master Thesis. I thank Mr. Marcus Petersen for supporting me by his suggestions for writing the report, presenting my implemented ideas to the audience and issues which I have faced while working with the sample machine learning architecture. I also thank Jun.-Prof. Dr.-Ing. Ingo Siegert and Mr. Normann Weisskirchen for helping me to understand the concepts to implement my own work. I am thankful to Jun.-Prof. Dr.-Ing. Ingo Siegert for his suggestions in implementing my work, evaluating my implemented work, and writing my report as well. I express my gratitude to the Examination office of Faculty of Electrical Engineering and Information Technology for the support and the administration. I am also thankful to my colleagues at the company for motivating me to complete my tasks.

I am grateful to my Parents for motivating me whenever I felt stressed. I also thank all my friends for their motivation and support.

Declaration of Authorship

I hereby declare that the proposed work is my own work and effort and it has not been submitted anywhere for any award. Whether other sources of informations have been used, they have been cited.
The work has not been presented in the same or a similar form to any other testing authority and has not been made public.

.....

Saran Karthikeyan,
Magdeburg, 16.03.2020

Contents

List of Abbreviations

AEB	Agent-Environment-Bodies
AI	Artificial Intelligence
ANN	Artificial Neural Network
AWS	Amazon Web Services
CAGR	Compound Annual Growth Rate
CF	Catastrophic Forgetting
DM	Dialogue Management
DST	Dialogue State Tracker
DSTC	Dialogue State Tracking Challenge
FSM	Finite State Machine
HCN	Hybrid Code Networks
ICA	Interactive Conversational Agents
ISCT	Incremental Sequence Classification Task
ISU	Information State Update
KB	Knowledge Base
KGD	Knowledge-Graph Driven
LSTM	Long Short Term Memory
LSTMN	Long Short Term Memory Network
MILU	Multiple-intent LU
NE	Named Entity
NER	Named Entity Recognition
NLG	Natural Language Generation
NLP	Natural Language Processing
NLU	Natural Language Understanding
RL	Reinforcement Learning
RNN	Recurrent Neural Networks
SDS	Spoken Dialog Systems
SGD	Stochastic Gradient Descent
STC	Semantic Tuple Classification
WOZ	Wizard-of-Oz

Glossary

Artificial Neural Network	The algorithm that has built inspired from the human brain to learn from the data, and capable to perform tasks specific to the learned data in a test environment
Attention	A mechanism that has used to pay attention to the information while processing the long sequences
Catastrophic Forgetting	The Deep Learning Networks tends to totally forget the learned representations upon learning incrementally
Dialog Acts or Intents	An information that has requested by the user, which has obtained from the uttered dialogue of the user.
Entities or Slots	The contextual information associated with the intents, which has uttered by the user
Deep Learning Network	The algorithm that has developed with multiple hidden layers with multiple lateral connections to learn the features, and improve the performance of the model suitable for testing environment
Interactive Conversation Agent	A system that has built to automate the process of holding a interactive conversation with a human
Long Short Term Memory Cell	An architecture that has proposed to overcome the vanishing and gradient problem, which has faced in Recurrent Neural Network. It has four gating mechanisms, such as Input, Output, Forget, and Cell state update gate to process the input sequences.
Policy or Strategy	A technique using hand-crafted rules or neural network algorithms to predict an appropriate system action corresponding to the user's input
Recurrent Neural Network	A neural network architecture that has proposed by feed-forwarding the inputs to the output layer along with the back propagation
Semantic Frames	The features which includes intents and the associated slot-value pairs for both user inputs and the system responses
Transfer Learning	The transfer of knowledge from the learned information of one domain to another

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Interactive Conversational Agents

Human-Computer Interaction (HCI) has dated back to six decades, and the business and the research interest in Interactive Conversational Agents (ICA) have increased due to technical advancements in Artificial Intelligence (AI). The implementation of Conversational agents, such as Chatbots, Spoken Dialogue Systems (SDS), or Personal Virtual Assistants, have attracted the customer service business sector, who have invested or ready to invest heavily in this technology [**1**]. Nowadays, customers require fast, reliable, convenient, and personal customer services, which pressurizes the customer service organizations to sustain customer loyalty and enhance revenue growth. Hence, many customer service organizations have planned, or they are planning to employ ICAs, which can offer customer service 24/7 and reduce the service cost of the number of necessary employees simultaneously [**1**].

The emotional labor of the call center workforce also influences the quality of customer service. The call centers are the representatives of the company over the phone or other communication channels to the customers. The qualitative studies have revealed that the call center work is a high-pressure environment due to the demands on high-quality customer service delivery [**2**]. The calls to the call centers are either in-bound, which have initiated by the customers or out-bound calls, which have initiated by call center workforce [**2**]. The call-centers are either in-house or out-sourced, where the customer services are for their firms or other companies, respectively. The out-sourced call centers are either in the same country or else it has located off-shore, where the customers and call centers have separated geographically, which pressurizes the call center agents in terms of gender, race, and personality based on the nationality[**2**]. Thus information and communication technologies play a prominent role in facilitating workflow and workforce management by using electronic performance monitoring metrics such as call handling time, after call time, compliance with schedule and script content, and the ratings of customer satisfaction. Thus call center workforce is under pressure to express fewer negative emotions since a high level of control has exercised in these workplaces [**2**].

Thus ICAs have attained the business and research interest to provide a potentially cost-effective solution and also to reduce the pressure in the call center workforce for reliable, fast, and cost-effective customer service. Though many ICAs have implemented on commercial websites, they cannot maintain their promises, and they have disappeared [**1**]. The realization of building a reliable ICAs for customer service remains a challenge because of designing ICAs according to the expectations of the users while interacting with ICAs. The industry and the research reports say that the ICAs fail to engage in the conversation to achieve certain goals by not holding a longer conversation, understand the conversations, differentiate whether the conversations have held in the desired directions, and act accordingly [**1**]. The attempts to make human-like rep-

resentations in ICAs have resulted in unrealistic expectations, which turned into frustration. Hence, the design techniques have to be improved to make ICAs effective and employ it for customer service [**1**].

1.2 Importance of Interactive Conversational Agents

In service innovation, digitalization plays an important aspect, which has paved the way for Information and Communication Technology-enabled (ICT-enabled) services, and it has introduced significant changes in the interaction between customers and the service providers [**1**]. The ICAs have become actors in the value creation process, which satisfies the roles of service employees. When compared to other technologies used in the service systems, ICAs have the potential to perform interpersonal interactions in online service encounters, and thus enhancing the service quality, efficiency, and experience of the customers. It plays a dual-role by representing the technology and taking an active part in interacting with the customers. Hence, it has to be designed with the abilities to represent human characteristics to play a role in customer service [**1**].

The use of technology for the delivery of services has brought changes in customer behavior, which has evolved to a personalized approach over the last two decades, and it has many direct and indirect interventions through multiple channels [**3**]. The digitalization has intervened the customer engagement with the following trends among the access of the customers as follows [**3**],

- **Natural Interactions** The experience of the customer is of the utmost importance, such that easier access, an appealing experience, and the respective actions for the user's input.
- **Flexibility** Accessing the service through multiple channels at any time
- **Responsive Service** A proper response according to the requirements and to provide individual attention
- **Clarity in the information** Delivery of concise and relevant information about the products to the customer.

Reports and Data, which is a Market Research and Consulting firm, have analyzed that the market size for chatbots had reached a value of 1.17 Billion United States dollars (USD) in 2018[1]. This has anticipated to reach up to 10.08 Billion by the year 2026[1], at a Compound Annual Growth Rate (CAGR) of 30.9%[1] and it has forecasted that the CAGR of 31.6% especially in the business sector of customer services [1]. Thus ICAs add monetary value to the company by reducing customer service costs, customer's experience by reliable service, and it also adds revenue by marketing the products and services through Technology, AI, and business process design, which have involved in implementing ICAs. The technology helps in exchanging the information between the customers, ICA and the internal systems by the information delivery through the mobile or web application services, AI is the core for ICAs to decode the messages and decides the interaction and the business process design chooses a standard process to access information, handles security of the information and product delivery to buy, sell or for inquiry [**3**]

[1]https://www.reportsanddata.com/press-release/global-chatbot-market, *Version August 2019*

1.3 Statement of the Problem

Though many of the current ICAs can answer simple questions, they cannot handle complex tasks efficiently to reach the goal. It is due to the limitations in Natural Language Processing (NLP) techniques, such as interpretation of the messages and generation of an appropriate response for the request throughout the interaction to reach a goal. Thus, the system must engage in conversation with the customers for the best experience in the provided service. To attain customer satisfaction, the decisions taken by the system for its response actions must be corresponding to the customer's requests'. Hence, the data source is significant while building ICAs, and the technological strategies remains a challenging task in building ICAs. The challenges which avails while implementing ICAs have discussed below as follows [**3**],

- **Service reliability** The customers and service employees conversations are complex since the requirements of each customer involve various tasks, and they expect individual attention and high-quality service delivery while handling their tasks. Hence, ICAs are often limited to perform basic tasks. It is unable to assure ICAs to handle complex problems
- **Cooperative principle in the conversation** According to the theory of H. Paul Grice, who proposed the cooperation principle of four maxims states that Quantity, Quality, Relation, and Manner are important to have a meaningful conversation. The Quantity defines the informative contributions, and quality represents truthfulness in the contribution, Relation means the relevant information in the responses and Manner defines the transparency in the information with proper delivery ethics [**3**]. Thus the ICAs must comply with these four maxim principles while handling the conversations
- **Responsiveness** The present ICAs are unable to deliver human-like conversations, as they provide general information. It makes the customers or the users have a feeling that they are talking to a robot. Additionally, the ICAs could not have long conversations to reach a specific goal and evaluating whether the customer requests' has handled in the desired direction. Hence, the interaction process is intrusive by using certain rules. The improper interpretation of users' requests results in an inappropriate response.

Hence building ICAs remains one of the technologically challenging tasks to provide accurate natural responses for longer conversations. The present ICAs uses rather simple pattern-matching techniques or else hard-coded rules, which lacks the natural language in the responses. Though there are several platforms such as Dialogflow, Microsoft bot framework, or IBM Watson, Wit.ai provide online services, it requires to pay for handling each dialogue [**3**]. The relevant business logic must involve while taking actions by integrating existing applications such as product or customer databases and APIs for an effective conversation. Hence, building ICAs considering the above-mentioned problematic factors to provide high-quality self-services the following measures have considered as follows [**3**],

- Deployment of advanced NLP techniques and integration of business logic to handle the requests' of the customer or the user in a natural manner with high-quality service
- The ability to have interactions with sufficient amount of information for the received requests'
- The ability to provide transparent and flexible conversations with the four maxim principles to attain a specific goal along with clarification, error-handling strategies to process the requests' irrespective of the misunderstandings
- The generation of natural responses, which fits to the context of the requests' and proper delivery of the responses satisfying the expectations of the user

1.4 Objective of the proposed work

The data is the crucial source for building ICAs to have interactions effectively. For high-quality customer services, it is more likely to have customer-service employee conversations about training ICAs. Hence, ICA relies on the resourceful data to provide natural responses during the interaction. The advanced AI methodologies have to be employed for NLP techniques to provide high-quality services with an appropriate response for each dialogue request. The NLP techniques include interpretation of the intention from the user's requests and process it to make an appropriate decision to take necessary actions with a response specific to the decision. The integration of business logic has achieved by including the results from the API calls to make the conversation effective. At last, the developed ICA has evaluated by using professional and naive participants.

The objectives to develop a prototype of an ICA has described below as follows,

- **Selection of dataset** The human-human conversations involve natural conversations, which provide resources to build ICA to deliver reliable and quality service. After analyzing the available open-source dataset available for domain-specific case, anyone has to be selected based on the quantity and the quality of the information
- **Natural Language Processing (NLP)** The advanced methodologies to interpret the messages or the requests of the user or customer for identifying the intention of the requests and processing of its associated information. From the processed interpretation of information, a Dialogue Management (DM) strategy has to be employed to decide on sending an appropriate response with each dialogue request, and then a generation of natural responses using AI techniques or standard templates based on the chosen decision.
- **Selection of tools** Analysis on various available frameworks and software tools to build ICAs and selection of an appropriate framework to tackle the complexity factors using NLP techniques
- **Business logic Integration** The integration of products database or API to provide efficient customer service. The technical errors which occur while handling the database or API have handled.
- **Evaluation** The evaluation of the developed ICA by live-users, who are both professionals like call-center agents and the naive users like students or ordinary persons, who has no experience in interacting with system agents to achieve specific tasks

The prototype of an ICA has built for an enterprise 'regiocom,' which has its main office at Magdeburg in Germany. It develops products and provides services in customer services for various branches like consumer goods and trade, automobiles, energy sector, transport and logistics, media and telecommunications, and healthcare.

1.5 Structure of the proposed work

The proposed work has summarised with the following chapters to have insights about the state-of-art techniques to build ICAs, literature research on the ideas behind the methodologies, the proposed strategy and the results of the evaluation

- **Chapter 2** describes the background and a brief survey on the conventional technical system of ICA have described. It also gives details about the fundamental theories associated in the state-of-the-work

- **Chapter 3** explains about the related work in state-of-art techniques to build ICAs and the requirements to build Interactive Conversational Agents along with the necessary software frameworks. The NLP strategies offered by these frameworks have also described, which serves as a baseline architecture to build an ICA
- **Chapter 4** explains the custom Dialogue Management and Natural Language Generation strategy employed to develop ICA, and its architecture has depicted.
- **Chapter 5** describes the evaluation strategy and the results of baseline architectures with the custom architectures in building ICA, and their comparison has discussed.
- **Chapter 6** gives a conclusion of the proposed work and the possible future work which has an association with this work has described.

Chapter 2

Background and Literature Research

2.1 Building of an Interactive Conversational Agents

The designing of ICAs must have the following factors, namely, functional intelligence growth of the technical systems and providing that intelligence while interacting with the user. ICAs have to adapt its functionality according to the requirements and preferences of the user for holding conversational flow. The ICAs should exhibit characteristics such as adaptability, availability, cooperativeness, and trustworthiness. The cognitive processes involve advanced perception, planning and decision making, and natural interaction capabilities, and it has used in implementing characteristics as mentioned above [**4**]. The functionality of ICAs should be context-sensitive, and it should have the ability to perceive and recognize the contextual information, which has relevance to the decisions of the system to take suitable action. It has to interact with users to hold meaningful conversations [**4**]. Figure 2.1 depicts the Interaction between the Human and Interactive Conversational Agent as follows, The important factors associated with

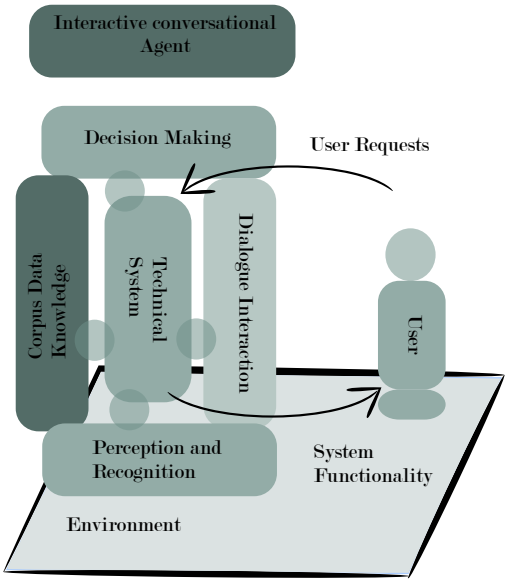


Figure 2.1: Interaction of Human- Conversational Agents [**4**]

the ICAs have described below [**4**],

- **Corpus Data Knowledge** In order to have conversations in an interactive manner, the

ICAs have to rely on comprehensive and multi-variant dialogue types depending on the specific domain and the tasks which it has to accomplish. The corpus data provides the static knowledge to the system for training such that the system can produce an effective response

- **Decision Making** The ICAs have to communicate to the user with its technological functionality to provide services to meet the user's requests. Thus the system's functional behavior has expected to be responsive, ability to perceive the user's request and then react to it, and flexibility to accomplish a specific task. The actions of the systems have decided based on the pre-conditions, which gives information about the applicability of the action for each state. From the beginning of an initial state, the system has to select appropriate actions coherent to the sub-goals to reach to complete the specific task. Sub-goals are the course of actions which enables the execution of relevant actions to achieve the end goal

- **Interaction and Dialogue** The significant importance of ICAs is the provide natural dialogues in its responses by sustaining the dialogue wit the user. An adaptive dialogue is a natural choice to understand the interaction between the user or multiple users, as it consists of a sequence of consecutive interaction steps. This cumulative information structure forms a basis for determining the intentions of the user using cognitive abilities. The current dialogue state is important to recognize an effective interaction.

- **Perception and Recognition of User's Dialogue state** The perception and recognition of user's requests are vital to ensure the functionality of the ICAs adapting to the user's dialogue state. The system has to predict the changes in the contextual information dialogue states, in order to have an appropriate interaction without any deviations.

The Realization of such ICAs must be based on real-world situational aspects, which are available in decision-relevant and actionable corpora. The corpora are the rich source of information to study the interaction activities in real-world interactions. The multi-variant data of dialogue types is available while having natural interaction. Corpora which meets these standards have to be selected to make the system learn through cognitive abilities.

The ICAs require three basic steps to interact with humans, namely interpretation of the message, tracking of information, selection of action, and then an appropriate response. The user inputs are the message of the user; the user wants to interact with the system, and from which the user's message has to be interpreted to understand the intention as to how the human brain process the input information, which it has received. Then the necessary information associated with the intention of the user has to be extracted in order to take the next system action by learning through certain strategies. After deciding the action, the system must provide an appropriate response corresponding to the chosen action.

Thus, to build such a system, various components associated with scientific techniques have to be involved. The conversations may include an open domain or a specific domain of application. In the proposed work, the techniques to build ICAs only for a specific domain has considered. The brief list of components and Terminologies which has used in the proposed work has explained in the forthcoming section. In Figure 2.2, a simple architecture has depicted to show the steps involved in building ICAs.

The ICAs have to service their functionality such that every user has served according to their specific needs and requests. They have to provide flexibility in the in its responsiveness according to the requests. In Figure 2.3, a sample conversational flow has depicted between the human and the system, such that certain goals have to be achieved.

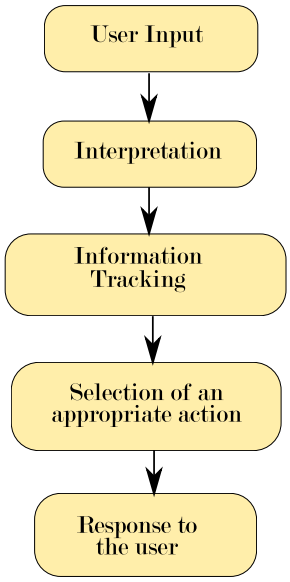


Figure 2.2: Building of Interactive Conversational Agents

2.1.1 Terminologies

The terminologies for building ICAs have described below as follows [**3**],

- **Utterances** refers to the message given by the user to get an appropriate response from the system. This information corresponds to the intentions of the user. It can be either any queries to the system, any requested information by the system or else any random chat messages
- **Intents** represents the intention of the user in the uttered messages. It is significant as it decides the conversation flow between the user and the system. In a supervised learning setting, the intents associated with each utterance have annotated, which will then be used to classify intents from the utterances. It has also referred to as Dialogue Acts
- **Entities** refer to the sub-ordinate values associated with each intent, which provides additional information to retain the context of the conversation. It has also referred to as slots values

2.1.2 Basic components to build Interactive Conversational Agents

The three basic components to build ICAs are Natural Language Understanding (NLU), Dialogue Management(DM), and Natural Language Generation(NLG).

- **NLU** is responsible for classifying the intents from the utterances of the user and its associated slots-value pairs, and it involves certain pre-processing techniques
- **DM** refers to a policy or strategy to co-ordinate the flow of the conversation for the obtained intents, and slots-values information from NLU component
- **NLG** refers to the generation of responses for chosen system action by Dialogue Management either from the pre-determined templates or from the corpus data, which has used to train the DM policy. There are also scientific approaches understudy for self-generation of texts, which is beyond the scope of this proposed work

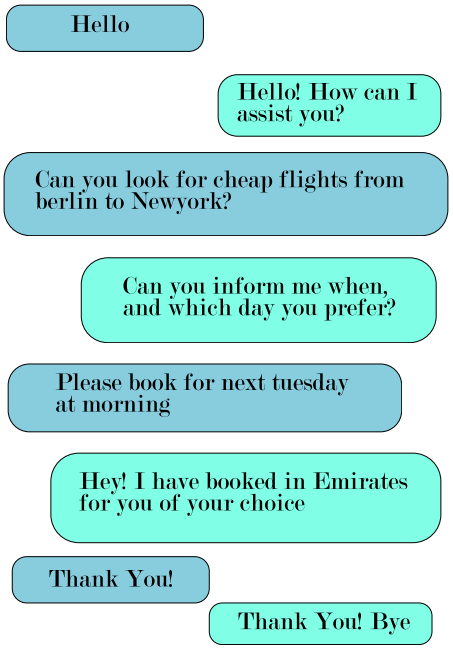


Figure 2.3: A sample Conversation flow for flight booking task. Boxes in 'Green' indicates a user's message and 'blue' indicates corresponding system response

2.2 Brief survey of conversational systems

2.2.1 Types of Conversational System

Social Conversational Systems serve the user's need for communication, affection, and social belonging. Hence, it must have the ability to recognize the emotions and track the emotional changes while having the conversation. It has to develop its functional abilities through cognitive strategies to perform the requests of the user. Unlike other Conversational systems, it has to be efficient. It means that the tasks have to be accomplished, and the conversation has to end soon. It performs tasks like conversing like a human, presenting results, and prompting new topics to keep the conversation live [**5**]

Goal-oriented Conversational Systems- As the name suggests, it was designed to accomplish certain tasks on specific domains. It was subdivided into two categories, namely Spoken Dialogue Systems and chatbots. The proposed work has concentrated on building Interactive Conversational Agents by holding a conversation through texts, and hence the description provided in the forthcoming chapters and section addresses the chatbots of task-completion conversational systems. The general illustration of task-oriented chatbots has shown in Figure 2.4 as follows, The NLU has to extract the core semantics, such as contextual information associated with the keys and values from the sequence of input utterances. It identifies the intention of the user and parses the semantic slot-value pairs of a specific domain of dialogues. It has to interact with the users in order to assist them in achieving certain goals. The DM acquires the semantic information and makes decisions to take appropriate action. It has to track the dialogue state and select an action from the learned policy or strategy with cognitive abilities. It may also include a knowledge database and access it to make more robust decisions. The recent works have proposed end-to-end learning systems to build task-oriented conversational systems, where multiple components have optimized jointly [**5**]

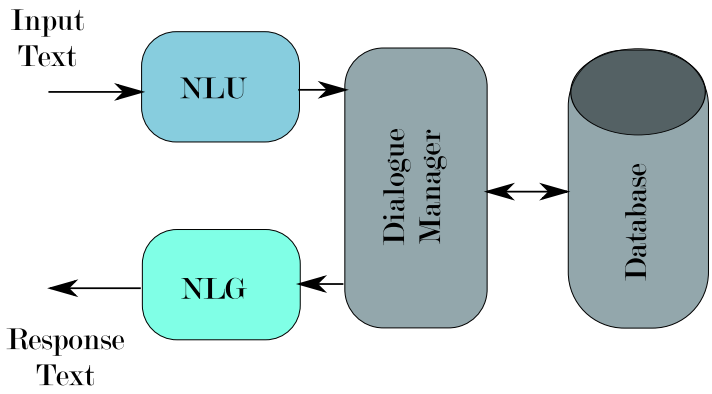


Figure 2.4: An Illustration of Task-Oriented Conversational Systems [**5**]

2.3 Basics of Machine Learning Models

- **Probability Theory** *Conditional Probability* It is the probability of an event, for which another event has occurred. The determination of conditional probability is possible if and only if an event has occurred, which means that an event has to happen to determine the conditional probability.

- **Gradient based optimization** The Deep learning models requires optimization by minimizing or maximizing the objective functions. The function which has to undergo minimization or maximization is known as the error function. The minimization has attained by determining the derivative of this function to get its slope, which has then used to reduce the corresponding function by moving its value in small steps gradually in the opposite direction of the obtained derivative. Thus, the goal of training the deep learning model is to achieve an optimized scaling for which a change occurred in the input influences obtaining the desired output.

  The criterion considered in any of the optimization techniques is a critical point, local minimum, local maximum, global minimum. When a derivative of a function is zero, then the point at which it occurs is called a critical point. A point where the value of a function is lower than all its neighboring points, and it is impossible to decrease the function gradually is **local minimum**, whereas **local maximum** is vice-versa of a local minimum. A point which is neither maximum or else minimum is a saddle point. The absolute lowest value of a function is the **Global Minimum**. A deep learning model can have multiple global and local minima. The multiple local minima, which are not optimal, and the saddle points with flat regions, have to be optimized while training the deep learning model. This optimization technique has known as Gradient descent optimization [**6**]. The gradient-based optimizers have discussed below.

  - ***RMSProp*** It is an optimization method, used along with momentum. The gradient information would be re-scaled based on the momentum, such that the parameters would be updated accordingly. It performs well for non-stationary settings, but it is not suitable for sparse gradients [**23**].

  - ***ADAM*** It is an abbreviation of ADAptive Moment estimation. The little memory is sufficient for an effective stochastic optimization using ADAM, as it uses only the first-order gradient information. Thus, the adaptive learning rates for each parameter use the first, and second momentum of gradients. The magnitude that has obtained for each parameter updates is invariant irrespective of the re-scaling of the gradients [**23**]. It is also adaptable for sparse gradients.

    The rule for the updates in ADAM optimizer is the careful selection of step sizes. The effective step size is proportional to the sparsity of the gradients, such that the estimation bounds to a *trust region*. It means that the estimation on the current gradient does not have any influence beyond the trust region. It helps in obtaining the right scaling for the value of α, which provides the optimization within fewer iterations [**23**].
  - ***ADADELTA*** It is also a gradient descent optimization technique, which is stochastic. It can adaptively tune the learning rate without any manual tuning, and it is also robust to noisy gradient information irrespective of modalities in the data, and the hyper-parameters. The optimization has achieved by the steepest descent method, which has the negative gradient information associated with a learning rate [**25**]. The learning rate decides the step size for each sample. It does not require that the learning rate has to set manually, it does not influence by hyper-parameters, and it provides less computation cost. It does not gather the squared gradients and sum it for all the time. It accumulates rather the window of past gradients for a fixed size, such that the learning has continued for many iterations . It does not store the squared gradients, but rather it accumulates the average of exponentially decaying squared gradients [**25**].

- **The Quantitative measure** It is important to estimate any machine learning algorithms for its performance measure of a specific task assigned to it. The **Accuracy** of the model has been used generally for quantitative analysis. It represents the number of samples for which the model is capable of producing the desired output, whereas the error rate is the number of samples for which the predicted output is not the desired one. To determine the performance of the machine learning model, the test set, which has separated from the training set, has used for estimation of the quantitative measure. It determines the performance of the trained model in a real-time scenario. The trained model of high-performance measures should be able to generalize well for the unobserved inputs as well.

  Additional quantitative measures include Precision, Recall, and F-Measure. The **Precision** is the ratio of 'True Positive' condition to the summation of 'True Positive,' and 'False Positive', whereas the **Recall** is the ratio of 'True Positive' condition to the summation of 'True Positive,' and 'False Negative'. The conditions have explained as follows, *True Positive* defines that the predicted label matches exactly to the true label, *True Negative* represents that the predicted label which is not the true label, which means that the predicted label is correct which is not a true label, *False Positive* represents that the non-desired label has predicted as a desired one, and *False Negative* represents that the desired label has predicted as a non-desired one. The $\mathbf{F}_\beta$**Measure** has calculated by the fraction of product of Precision and Recall, and the summation of the both multiplied by the β value. Thus obtained overall value has then multiplied with $(1+\beta^2)$

- **Experience** The training of a machine learning algorithm has categorized based on the experience of the data, namely **Supervised Learning** and **Unsupervised Learning** [**6**]

  - ***Supervised Learning*** Each input that has fed to the machine learning algorithm

has associated with a label. Each input has associated with a label. The features have extracted from each input, and learned by the machine learning algorithms. Then, the model can classify or predict the observed inputs. Also, it can classify or predict the unobserved inputs as well based on the obtained features and the features it has used when it has trained each associated with a corresponding label.

- ***Unsupervised Learning*** Each data that has fed to the machine learning algorithm has not annotated or labeled. The machine learning algorithms categorize each observed inputs and unobserved inputs based on the similarity between the features, which it has learned from the training data using metrics, such as Kullback-Leibler divergence.

- **Over fitting and Under fitting** The factors which influence the capability of the machine learning models are **Training error** and **Test error**. The ability of the machine learning algorithms is inversely proportional to the training error and the test error. Based on the factors mentioned above, the machine learning algorithms have to undergo two challenges, namely ***Under fitting*** and ***Over fitting*** [**6**]

  - ***Under fitting*** When the machine learning models are incapable of attaining a low error value on the set of data samples which it has trained, then it causes the machine learning model to underfit.
  - ***Over fitting*** When there is a large gap between the training error and the test error, then the machine learning model tends to overfit.

  The appropriate capacity for the machine learning model has to be chosen to overcome the issue mentioned above. The increase in training samples may help the model recover from underfitting, but when the large capacity has provided, then the system might overfit, and it could not comply with the desired task [**6**].

- **No Free Lunch Theorem** It states that every machine learning algorithms have the estimation of the same error rate for any unobserved inputs provided with an average of all possible data generating distributions. It means that it is not possible to conclude that one machine learning algorithm is the most sophisticated universally, but rather each machine learning algorithm has better sophistication depending on the distribution of data.

- **Regularization** This technique employed in deep learning algorithms to reduce the generalization error. It prevents the model from overfitting. Though it reduces the generalization error, it does not affect the test error. Drop out is one of the most used regularization techniques. Though there are other regularization techniques, such as L1-Regularization, and L2-Regularization, drop out averages the predictions of the parameters by randomly dropping out the hidden units along with their lateral connections for all the possible settings. This technique has proven to be effective [**24**].

  Moreover, the averaging of outputs for the separately trained larger neural networks is difficult. It is because that hyper-parameter fine-tuning for training different neural networks is a hard task to accomplish, and also it requires more computation power. Thus, it is possible to reduce overfitting, and the combination of models becomes feasible by using drop out technique, which also makes the machine learning model efficient. Also, the feasibility of minimizing the loss function has achieved by using drop out technique, which is stochastic in nature [**24**].

2.4 Deep Learning techniques

2.4.1 Recurrent Neural Networks

The interconnection of artificial neurons or layers of connected units has known as Artificial Neural networks (ANN). A shallow network has a input layer, an output layer, and at most one hidden layer. A shallow network can be either a feed-forward or recurrent networks. In feed-forward networks, input layers connect to hidden layer or multiple hidden layers, which has then the connection with the output layer. In recurrent networks, the hidden layers have the recurrent connection along with the forward connection to the output layer. A number of hidden layers with recurrent connection increases the complexity of the network as the depth of the network increases. Thus, it has various levels to represent the data and to extract the features, which has known as deep learning [**8**]. Recurrent Neural networks (RNN) have the capability to model sequential data for recognizing the sequences and prediction. RNN have two different architecture, the one has recurrent connections from the output layer, and the other has the recurrent connections from the hidden layer. This has named as Jordan network, and Elman network respectively. RNNs are the supervised machine learning models which have artificial neurons with one or multiple feedback loops [**8**]. A simple Elman RNN architecture has shown in Figure 2.5

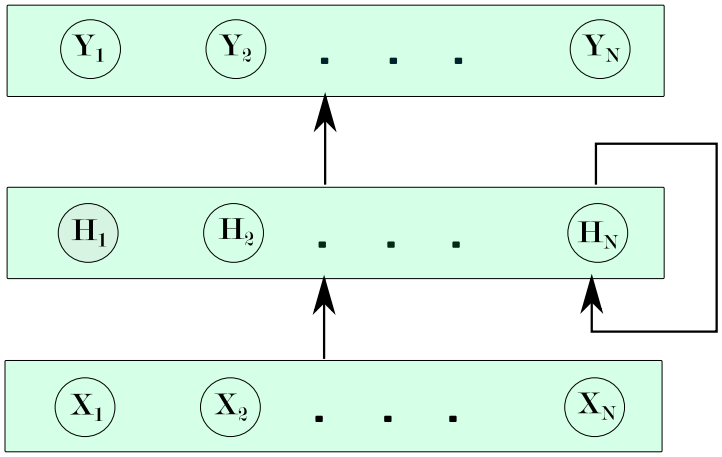


Figure 2.5: Architecture of simple Recurrent Neural Network [**8**]

2.4.2 Long Short Term Memory

The standard RNN architectures have the limitations of accessing the range of context. The input influences the output, such that it gets decayed exponentially for the cycles of recurrent connections. This problem has defined as a vanishing gradient problem [**8**], which has shown in Figure 2.6. Several strategies have proposed to address this problem like discrete error propagation, explicit introduction of time delays and time constraints, and hierarchical sequence compression [**8**]. Later, in 1997 Long Short Term Memory has proposed to reduce vanishing gradient problem, which has proven for its effectiveness when compared to other approaches [**8**]. The shadowed region is an indication of the sensitivity to the corresponding inputs at each

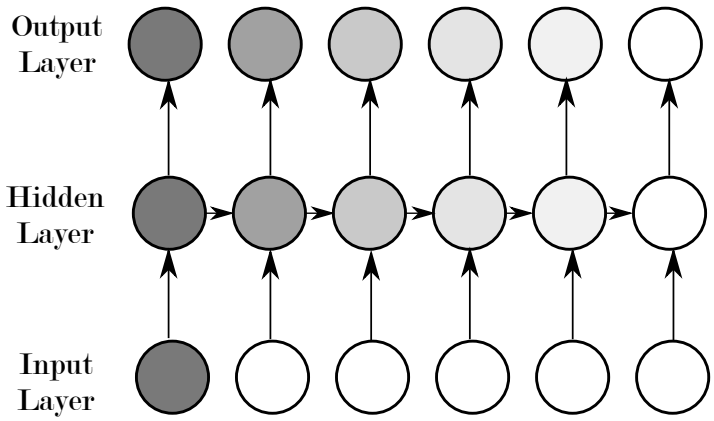


Figure 2.6: An Illustration of Vanishing gradient Problem [**8**]

time. The sensitivity is greater when the shade is darker[**8**]. The sensitivity decays over time, and the hidden layer activations have overwritten by the newly arrived inputs [**8**].

The LSTM has a set of recurrent connections, which are known as memory blocks. It is similar to the memory chips in any digitalized computer, but they are differentiable. Each memory block consists of either one or more memory cells that are self-connected. It has an additional three multiplicative units- input, output, and forget gates for the continuous write, read, and reset operations for each cell [**8**]. It is similar to RNN, but the summation units have replaced by memory blocks. The gates that have mentioned above allow LSTM memory cells to store and access information for long periods, which reduces the vanishing gradient problem [**8**]. The new inputs to the network do not overwrite the activation of the cell until the input gate remains closed. Thus, the information sequence has utilized later when the output gate has opened [**8**].

The three gates are non-linear summation units, which collects the information inside and outside the block. The control of activations has controlled by the multiplications. The multiplication of inputs and outputs has handled by input and output gates. The previous state information of the cell has used by Forget gate to multiply it [**8**]. The logistic sigmoid activation function has used for gate activation. Hence the gate activations are in between '0' and '1'. Figure 2.7 illustrates the LSTM memory unit with a single cell. The output to other blocks of the network comes only from the multiplication of output gates [**8**].

The gradient information has preserved by remembering each input until the forget gate has opened, and the input gate has closed. For simplification, only two options have depicted in Figure 2.8, either entirely sensitive or else entirely insensitive by using black and white shades respectively [**8**]. The state of the input gate, forget gate, and output gate to the left and above the hidden layer have depicted in Figure 2.8. The symbol 'o' represents that the gate remains open and '-' represents that the gate has closed [**8**]. LSTM is a differentiable function approximation, which has trained using a gradient descent approach. In the original LSTM, there were no forget gates, which has added later with additional weights to connect the gates to the memory cell. The sole purpose of the forget gate is to reset the memory cells by themselves. LSTM has also made bidirectional using the network architecture of Bi-directional recurrent Neural Network, which is useful in the long-range context in both the input directions. The

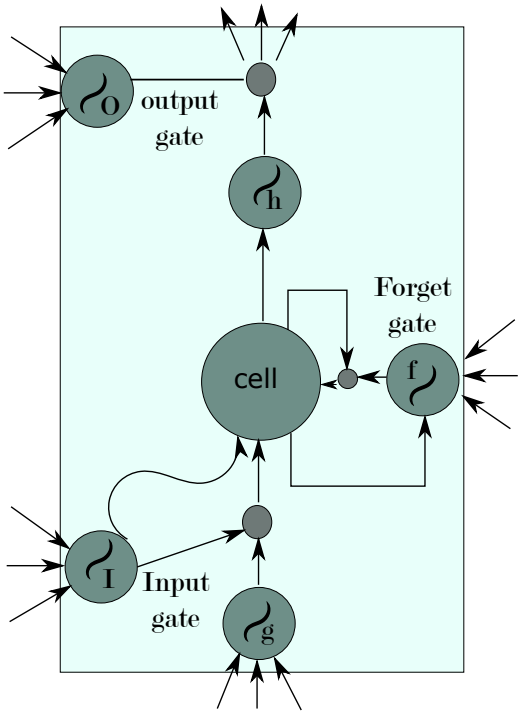


Figure 2.7: LSTM memory block with single-cell [**8**]

gating mechanism that have involved in LSTM cell has given in equation 2.1, 2.2, 2.3, and 2.4

$$i_t = sigmoid[(U_i h_{t-1} + W_i x_t) + b_i) \tag{2.1}$$

$$f_t = sigmoid[(U_f h_{t-1} + W_f x_t) + b_f) \tag{2.2}$$

$$\hat{c} = tanh[(U_c h_{t-1} + W_c x_t) + b_c) \tag{2.3}$$

$$o_t = sigmoid[(U_o h_{t-1} + W_o x_t) + b_o) \tag{2.4}$$

At last, the final cell state and the hidden state has calculated as given in equation 2.5, 2.6

$$c_t = f_t * c_{t-1} + i_t * \hat{c}_t \tag{2.5}$$

$$h_t = o_t * tanh(c_t) \tag{2.6}$$

2.4.3 Self-Attention Mechanism

The RNN performs better in the contextual representation of sentences by using the word-level embeddings. The recursive computation has performed on each word along with their previous memory. Thus, the entire meaning of the sentence has extracted. However, these networks face challenges, such as vanishing and exploding gradient problems, memory compression problems, and lack of a mechanism for handling the input structure. The vanishing and gradient problems have tackled partially by LSTM cells. As the sequences of the inputs have compressed into a single dense dimensional vector, it requires large memory space to store the past information,

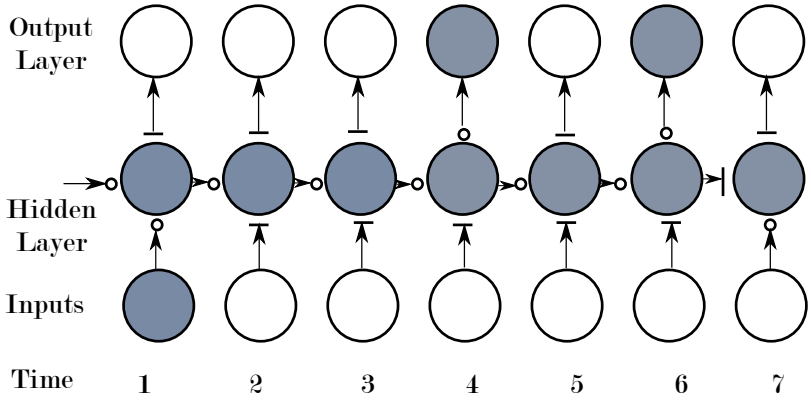


Figure 2.8: Prevention of gradient information in LSTM [**8**]

which leads to a memory compression problem. Hence, a machine reader has proposed which has designed to process the structured input, which uses LSTM with extended memory for an implicit analysis between the tokens by using attention-based memory mechanism at each time step [**9**].

The list of state representation has produced by LSTM during composition, which uses the present state to compute the next state. For the present state, the next state is conditionally independent of states, and tokens. LSTM maintains the unbounded memory while performing the recursive update in a Markov manner. This assumption may fail while performing with long sequences due to problems in modeling structured input or lack of a mechanism to model the relations between the tokens. Thus Long Short Term Memory Network (LSTMN) has proposed for storing the contextual information for each input tokens in a different memory slot. Thus, the memory size has enlarged until a maximum memory capacity has reached. The attention layer has induced to enable LSTMs to relate between the tokens and to perform non-Markov state updates. The read mechanism has upgraded by providing attentive linking of current tokens to the past memories and then selecting useful contents for processing the sequence. The LSTMN uses two vectors, namely, hidden state tape and a memory tape. The hidden state tape has used in computing the attention, and the memory tape represents the information that has stored in it. Thus, each token has an association with a hidden vector and a memory vector. The self-attention or intra-attention mechanism has computed using the following equation 2.7, and 2.8 and the architecture of self-attention mechanism has depicted in Figure 2.9 as follows [**9**],

$$a_i^t = v^t tanh(w_h h_i + w_x x_t + w_{\tilde{h}} \tilde{h}_{t-1}) \tag{2.7}$$

$$s_i^t = softmax(a_i^t) \tag{2.8}$$

For each time step, the relation between x_t and the hidden state h_T has computed to determine the probability distribution for previous tokens over the hidden state vectors. Then, the vector information has determined for previously hidden tape and memory tape, which has denoted as $\tilde{c}_t$ and $\tilde{h}_t$ in equation 2.9 as follows [**9**],

$$\begin{bmatrix} \tilde{h}_t \\ \tilde{c}_t \end{bmatrix} = \sum_{i=1}^{t-1} s_i^t . \begin{bmatrix} h_i \\ c_i \end{bmatrix} \tag{2.9}$$

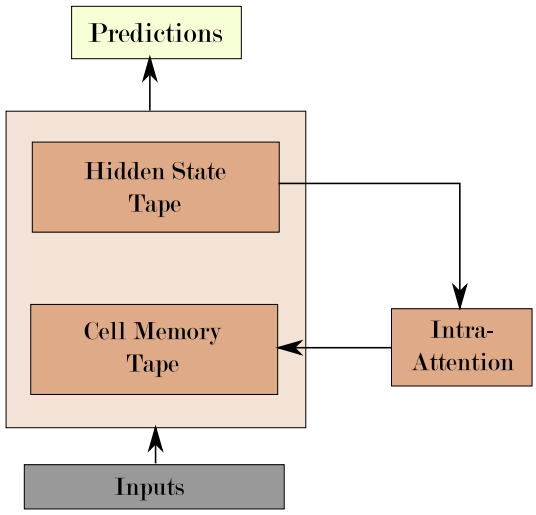


Figure 2.9: Architecture of Self Attention Mechanism [**9**]

The cell formulation of input, forget, output and Cell gates have then computed using the equation 2.10 as follows [**9**],

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ \widehat{c}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{bmatrix} W. \begin{bmatrix} \tilde{h}_t & x_t \end{bmatrix} \tag{2.10}$$

where v, W_h, W_x, and $W_{\tilde{h}}$ are updated weight matrices.

The final cell state and the hidden state vector have calculated similarly in LSTM from the matrices obtained above.

2.5 Sequence to sequence model architecture

The end-to-end methods offer a successful solution for task-oriented dialogues by connecting the dialogue system components. The encoder-decoder model approach, which is a supervised learning algorithm, has employed for non-task oriented ICAs using a fixed dialogue corpora. This sequence to sequence learning models has achieved promising performance on complex tasks. It uses one LSTM for encoding the inputs, and another LSTM for decoding the corresponding target outputs. At first, one LSTM has used to extract the fixed 3D vector state information for an input sequence, and then another LSTM has used to decode the target sequences that map to the conditional input. It has employed for Machine Translation tasks, and non-task oriented ICAs [**17**].

The sequence to sequence models using LSTMs has designed to overcome the challenge faced by sequence problems of the fixed dimensionality of inputs and outputs. The key idea is to take one input sequence, reading sequence by sequence for each time step, and obtain a large fixed 3-Dimensional vector representation. The obtained vector representation has then used by another LSTM to decode a target sequence conditioned on the read input sequence. LSTMs can learn from data with long-range temporal dependencies [**17**]. The architecture of the Sequence-sequence learning model has depicted in Figure 2.10 below. The following equation 2.7 and 2.8 has used by RNN for decoding the target outputs conditioned for a respective dimensional

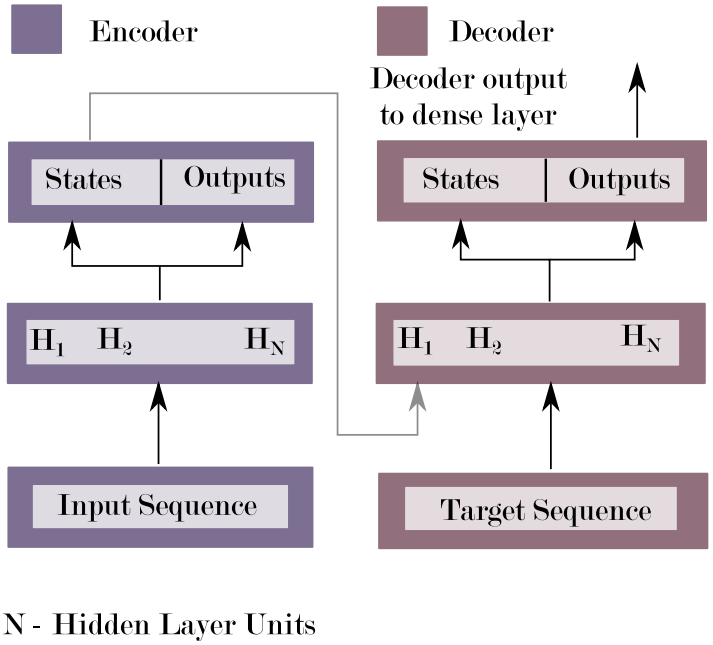


Figure 2.10: Architecture of Sequence to sequence learning, which has unfolded for each time step. The grey lines denote the encoder state vector information

input vector representation over iteration [**17**].

$$h_t = \sigma(W^{hx}x_t + W^{hh}h_{t-1}) \tag{2.11}$$

$$y_t = W^{yh}h_t \tag{2.12}$$

The aim of LSTM in this model is to estimate the conditional probability. The length of the target sequence may differ from the length of the input sequence. The fixed dimensionality representation of a vector has computed by the hidden state of LSTM, which has then fed as the initial state for the computation of the conditional probability of the target sequence. It has denoted in the equation 2.13 below [**17**],

$$p(y_1, y_2, y_T^{'}|x_1, x_2, ..., x_T) = \prod_{t=1}^{T^{'}} p(y_t|v, y_1, ..., y_{t-1}) \tag{2.13}$$

The output probability distribution has represented by a softmax layer. Then, the sampling of the tokens has performed over all the words that are available in the vocabulary. To have a probability distribution of variable length, a different end of the sentence sequence has introduced in the target sentence, which has used to stop sampling the tokens. From the depicted figure, for an obtained length of the input sequence, LSTM computes the encoder state-space vector representation, which has utilized to compute the conditional probability distribution of an output sequence [**17**]. The three factors to enhance the computation has described below [**17**],

- Two different LSTMs have employed to handle input and target sequence. It increases the model parameters at a low computational cost for training on multiple sequences
- The deep LSTMs provide better efficiency than shallow LSTMs

- The reversal of the order of the input sequence to a straight target sequence, such that the initial sequence of input has a close relation to the initial sequence of the target, and for subsequent sequences. This optimization helps SGD algorithms to establish communication between the input and the target sequence easily.

The training objective of the decoder is to produce an appropriate maximized log probability of the correct target sequence for the given input sequence. After the training has completed, the most likely decoded sequence has given by using equation 2.14 [**17**],

$$\hat{T} = argmax_T P(T|S) \tag{2.14}$$

It has also shown that this model performs well even for long sentences, which makes it applicable for Natural Language Generation for ICAs.

2.6 Progressive Neural Network

To learn the complex sequence of tasks, Progressive Neural Network has proposed, which has the ability to both transfers the knowledge and avoid CF to give human-like intelligence. It has stated that CF has overcome the leverage of prior knowledge through the lateral connections, which has previously learned. The transfer of knowledge between the neural networks plays a key role in multi-taking. Hence, Progressive neural Network has proposed, which has the support for transferring the knowledge across various domains and simultaneously overcoming CF. The hidden layer connections of updated weight values from the previously learned tasks have used to extract useful features for the new task. It has helped in achieving the feature hierarchy at each layer [**18**].

The continual learning remains a challenging task in machine learning, where the agents not only learn to generalize for the series of tasks that have the experience but also the ability to transfer the learned knowledge from one domain to another domain. The model architecture of Progressive Neural Networks have depicted in Figure 2.11 below, A PNN starts training with a single column, which is a deep neural network with stacked hidden layers and the corresponding hidden activations. When training the second column of another deep neural network of the same column height, the parameters of the previous column have frozen, and new parameters have updated. The hidden layer h_i^k receives the input from both h_{i-1}^k and h_{i-1}^{k-1} through lateral connections. Hence the hidden layer calculation for each column in PNN has given by the equation 2.15 [**18**],

$$h_i^k = f(w_i^k h_{i-1}^k + \sum_{j<k} U_i^{k:j} h_{i-1}^j) \tag{2.15}$$

where $W_i^k \in \mathbb{R}^{n_i x n_{i-1}}$ represents the weight matrix of the layer i in the column k,
$U_i^{k:j \in \mathbb{R}^{n_i, n_j}}$ represents the lateral connections from layer $i-1$ of column j to layer i of column k,
h_0 represents the network input,
and f denotes the element-wise non-linearity.

Adapter Function

The hidden layers in PNN have augmented with a non-linear lateral connection, which has called adapters to improve dimensionality reduction and initial conditioning. The lateral connections have multiplied by a scalar before feeding it to the network. The scalar is a small random value, which has used for different inputs to adjust the scale values. The non-linear adapter of each hidden layer is the projection of a n_i dimensional sub-space. The number of parameters from

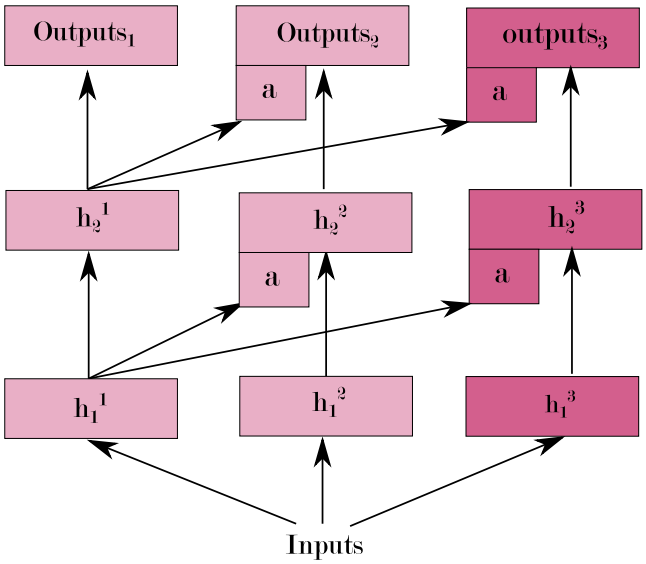


Figure 2.11: Architecture of Progressive Neural Network, which has depicted for three multi-tasks. The light shaded region represents the frozen layers, and the dark region represents the required region the network should get trained

the lateral connections is proportional to the chosen height of the column. Thus, the hidden layer with non-linear adapter function has given by equation 2.16 [**18**],

$$h_i^k = \sigma(W_i^k h_{i-1}^k + U_i^{k:j} \sigma(V_i^{k:j} \alpha_{i-1}^{<k} h_{i-1}^{<k})) \tag{2.16}$$

where, $V_i \in \mathbb{R}^{n_{t-1} X n_{t-1}^{<k}}$ is a projection matrix [**18**].

The limitations of PNN are the growth in the number of parameters with the increase in tasks. Hence, the stacking of fewer hidden layers has performed while training a network. The choice of tasks to select for an inference requires knowledge of the label of each task [**18**].

Chapter 3

Related Work and Frameworks

3.1 Analysis on datasets

The dataset have collected based on the interaction type, such as written, spoken, and multi-modal. As the proposed techniques has focused on building ICAs based on written approach, a brief analysis on written text based corpora have explained in the following sections.

3.1.1 Natural vs Unnatural Corpora

It is of significant importance the way of dialogues in the conversations have recorded and collected. The human-human dialogues have the natural conversation interacted between the humans, such that representation of dialogues are richer to represent with whom the dialogue system must have an interaction [**19**]

In unnatural dialogues, the humans have informed that the conversations will be recorded and asked to perform a particular task by interacting with a dialogue system. In this setting, a true intents of the user cannot be recorded, as human tends to behave in a different manner when they knew that their conversations will be recorded, and also the platform through which the interactions happens also have an influence [**20**].

Thus Wizard-of-Oz (WOZ) experiments plays a prominent role in collecting a most natural-like conversations. In this setting, a human has informed that they are speaking to a machine, but actually a human operator controls the dialogue system. Thus the generated dialogues are more natural, but this process is expensive and time-constraint [**20**].

3.1.2 Human-Human vs Human-Machine Conversations

Human-Machine Corpora - A practical application of Human-Machine corpora based dialogue system is the Let's Go Bus information system, which gives the live information about the bus schedule over a phone, which has proposed in the first Dialogue State Tracking Challenge (DSTC). Though it is an obvious solution, the possibility is only with an existing working system which lacks from domain adaptability. One of the limitation is that the initial system introduce biases to the data which has collected leading to mismatch in training and test sets, and lacks from the naturalness of dialogues in a conversation [**20**].

Human-Human Corpora - The involvement of interaction either between humans or between human-machine has also significant importance due to the complexity constraints of artificial dialogue systems. The human-human conversations does not have same distribution of understanding errors, and also these conversations have richer dialogues when compared to human-machine dialogues. Due to the richness involved in the dialogues, it is the best type to

build ICAs. When the artificial dialogue systems trained on this type of corpora, the human-like behaviour will be achieved in natural responses, and the robustness of a system by understanding the natural utterances can be enhanced using suitable research techniques. Though several large scale corpora have developed, such as Twitter dataset, Reddit conversations, and Ubuntu technical support corpus, they does not have grounding conversations to an existing Knowledge base or APIs. This limitation reduces the usability of such developed dialogue systems. Also, lack of explicit goal in the conversation affects from generation of consistent and diverse responses, and they are hard to evaluate [**20**].

Machine-Machine corpora has collected using a simulated user, and thus large amount of dialogue templates can be generated. The generated templates have then mapped to a natural language using pre-defined rules or else crowd-workers. Though it offers diversity of most of the dialogue turns involved in specific domain, the naturalness depends completely on engineering set-up. This creates the risks of low interaction quality because of mismatch between the training data and the real interactions [**20**].

Thus to overcome the limitations mentioned in different interaction type corpora Multi-WOZ, a Human-Human conversations collected in a multi-domain setting by Wizard-of-Oz (WOZ) experiments, including dialogue collection by Amazon Mechanical Turk to suit for crowd-sourcing have selected for the proposed work.

3.2 Natural Language Understanding

3.2.1 Star space Algorithm

The above-mentioned models such as LSA and skip-gram model have significant drawbacks. Though LSA efficiently extracts statistical information, it shows poor performance on word analogy tasks, which indicates sub-optimal vector space structure. On the other side, skip-gram model performs better on word analogy tasks, but the utilization of the statistics of the corpus is poor, as the model trains on local context window rather than global co-occurrence counts, which makes these models fail to utilize the repetitions in the data. The discrete feature representations of the entities have learned from the relations between them for ranking or classification tasks. It embeds the different types of entities into a same embedding space and the comparison takes place within that space. Hence, it has named as 'Starspace', where '* - star' represents all the types, which gets embed into a space. Thus, this neural embedding model has applied for various problems such as Text classification or labelling tasks, Ranking of set of entities, collaborative-filtering based recommendation, content-based recommendation, and to learn word, sentence, or document embeddings. Embedding entity has implemented by the following equation 3.1 [**10**],

$$embedding = \sum_{i \in a} F_i \tag{3.1}$$

where, F - Dictionary of features, which is D x d matrix,
F_i represents the indexes of ith feature row, which yield its d-dimensional embedding 'a' represents entity.

3.3 Existing Dialogue Management Strategy

Dialogue Management has a key role in ICAs to predict the best system action for the current dialogue state. Thus, Dialogue State Tracking is a key phenomenon for Dialogue Management, which keep tracks of the informations in a discourse history along with the currently

specified user inputs. The Dialogue Management has achieved by three approaches, namely Knowledge-based dialogue management, data-driven dialogue management and hybrid dialogue management. [**11**]

- **Knowledge-based DM** requires the developers who have domain specific knowledge in order to define a regularized language set for the dialogues which are structured. This approach uses Finite State Machine (FSM) involving hand-crafted rules to perform the structured tasks. Though this approach is in use for practical environments of rapid prototype but it is difficult to hand-craft every rules in advance, which makes the flow inflexible and the process becomes cumbersome for the complex products in customer-centric environments. It gives poor domain portability and hence the design process for different domains has to be processed from the beginning. In the agenda and task models, the larger tasks have segmented into smaller sub-tasks for which RavenClaw framework is one of the popular one, which has designed with a control logic for domain-dependent and domain-independent tasks, which has handled by hierarchical tree structure and specifications of dialogue tasks respectively [**11**].

- **Data-driven DM** In data-driven approach, the training has performed automatically from the corpus data, and it requires human-involvement in annotation and developing algorithms using a statistical approach to train. By modifications in the developed algorithms, the domain adaptability can be achieved and it can be re-trained on a different corpus data. This reduces time and effort made for Knowledge-based approach. The Reinforcement Learning algorithms uses optimization technique by reward and cost functions to choose a dialogue strategy for the current dialogue state. The limitations of RL is that the policy may lose control and refinement of the dialogue control is difficult [**11**]

- **Hybrid-based DM** In order to avoid the limitations in both Knowledge-based and RL-based Dialogue Management, hybrid of both approach has suggested to reduce the large policy space and include the business rules [**11**]

3.3.1 Knowledge-Graph Driven Dialogue Management

It has proposed for practical task application by using Knowledge-Graph Driven (KGD) based on Information State Update(ISU). The performance of this approach has analysed in food-ordering domain. In ISU approach, the state information of the dialogue has updated in its state itself rather than a pre-determined dialogue state with a fixed transition in a series manner. It involves employing a series of rules over a structured data and extracting the semantic frames from the current dialogue state. However implementation of this functionality with a FSM-based approach becomes complex due to the requirement of the all combinations of inputs and its corresponding transitions between all turns in a conversation. The flexibility has achieved by incorporating the machine learning based approach for a better user experience. The ISU system has designed by a set of rules with pre- and post-conditions for a dialogue engine to execute these rules to select a action based on the better match between the state and the pre-condition. The KGD-based system has also incorporated to have domain-specific knowledge. In ISU-based approach *move* represents the intent specified by the user. But, in this approach it has used in the context of domain-based functions along with the information from KGD-based approach which defines the modifications in the information state. For each turn, a set of possible moves of the user has generated based on the current information state of the dialogue and static Knowledge-Graph for the particular domain. It requires a domain expert to generate a Knowledge-Graph for each domain [**12**]. The changes in the selection of *move* has made in KGD-approach by scoring the automatically generated set of possible move either by a rule-based or data-driven approach. Then , N-best hypothesis have calculated from the interpretation module, confident scores and priori probabilities from domain-specific information. Whereas, in

ISU-based approach, move selection depends on a best match between the interpreted utterances and the pre-determined conditions [**12**].

Representation of domain

A domain expert creates a KGD-based domain description to take domain-specific dialogue turns and the initialization strategy has performed by a graphical based domain information. Each node in KGD corresponds to a decision, form or a leaf node. While selecting a move, the templates for NLU and NLG has used which are associated for each node. By using matching algorithms, the intents and the slots have matched to NLU templates at each turns which corresponds to a root node and a respective response has selected from a NLG templates which matches to the NLU requests [**12**].

The Form nodes are stored in the information state which represents the domain information in order to exchange between the user and the system. A leaf node represented as a child node for an individual slots to a form node, in which the slots and its corresponding values have associated. The NLU and NLG templates have properties, such that either a property can set or removed [**12**]

Representation of states

The ISU-based approach has used to maintain a information state between the system and the user, whereas KGD-based approach has used to store the information state with a set of shared belief state and the dialogue history, and it also includes the previous state information held in a conversation. The shared belief states corresponds to a form node, which has stored as a json object. The agenda of a system has considered to be static, which has extracted from the domain knowledge graph. For a state S_o, a set of N-possible moves have taken into account M to generate a new state S_1 [**12**]

Dialogue Manager

The dialogue Management of KGD-based ISU approach has following sub-modules as follows [**12**],

- **Move Generator** has initialized with a domain knowledge graph and current information state. Based on definition of domain, it generates the possible user moves by deterministic rules. The set of possible moves have generated according to specific domain, which have semantics and action to be taken for the current state of the dialogue
- **Encoder** converts the extracted semantics from the user utterances into a vector representation, which utilizes domain-specific intents, labels, and labels as the dimensions of vector
- **Move Scorer** scores each possible move by the computation of likelihood, which has the match respective to the semantics in the user's utterances. It performs probability estimation of a match between an utterance and a move. Though NLU output has errors associated with its semantics, the correct move with a highest score based on the context will be selected
- **Move Selector** selects the set of moves, which have to be executed based on the obtained scores. The heuristics have applied for multiple moves of a user's utterances in a single turn
- **Move Executor** executes a selected action matching the user's semantics based on the selected moves

- **System Move** has executed to get an answer for an open question or missing slot information. If there are incomplete node present in an information state, an appropriate request has selected from the NLG templates and if all the information state has satisfied the root node will be executed to make a response from NLG templates

The architecture of KGD-based ISU approach has depicted in Figure 3.1 below as follows,

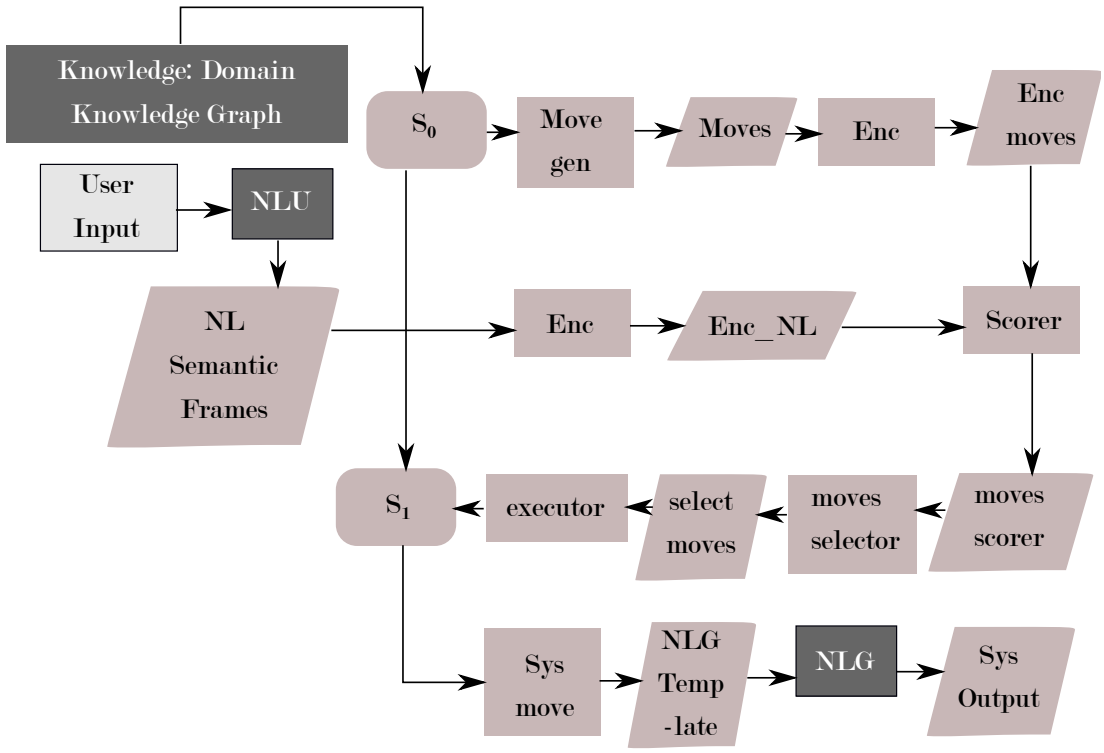


Figure 3.1: Architecture of Knowledge-Graph Driven based Information State Update approach for a specific domain [**12**]

3.4 Existing techniques for Neural Network based approach

3.4.1 Hybrid Code Networks (HCN)

In the dialogue turns of conversations, there is possibility of multiple valid next utterances in the text. The present end-to-end methods assume that at any time there is only one correct next utterance. This leads to different paths for reaching a goal. The human-human dialogue conversations have the complexity that for any given dialogue state or input, there is a probability of multiple correct utterances. The Reinforcement Learning (RL) algorithms learns through the trial and error method either from a human or a simulator. It is really cumbersome to reward a dialogue for every turns, which makes the Reinforcement Learning a difficult problem. In Supervised Learning, the corpus data collected from human-human conversations relies as a source for the neural network algorithm to learn. The network learns such that a particular dialogue in the dataset has relevance only to one of the utterance in the dialogue history and the current dialogue state. Hybrid Code Networks (HCN) combines RNN architecture with domain-specific knowledge which has encoded as a software and the system actions as templates. This is an end-to-end learning approach which neglected the dependencies of each modules for NLU, state

tracking, action selection and NLG such that the complexities have reduced. But the end-to-end learning models lacks the methods to introduce the domain knowledge and constraints as it learns directly from the text transcripts. Thus software has used to provide explicit information about the domain along with the action templates. The architecture of HCN has shown in Figure 3.2 [**13**],

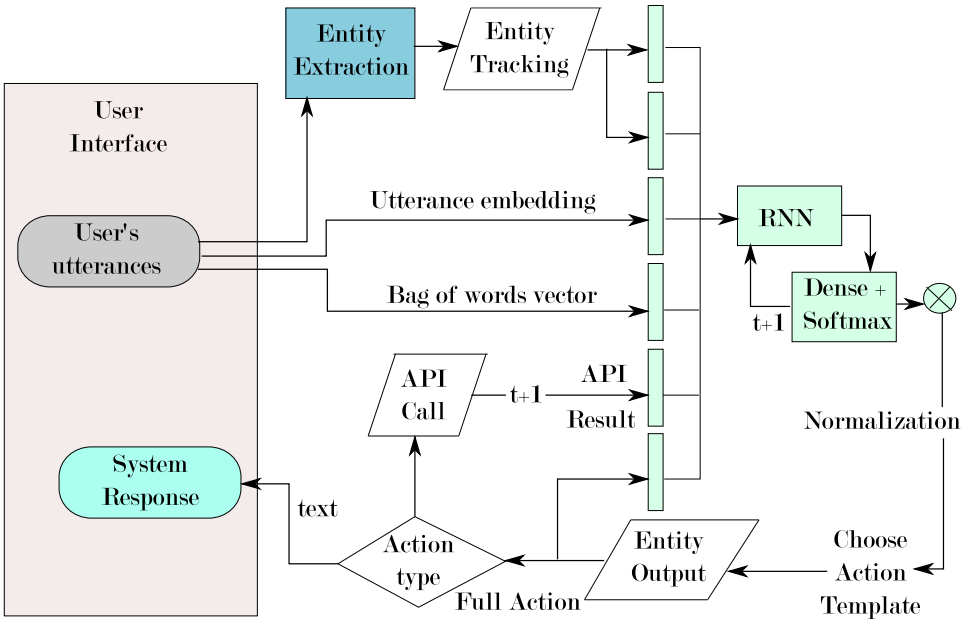


Figure 3.2: Architecture of Hybrid Code Network[**13**]

The components used in HCN are RNN, domain-specific software, domain-specific action templates and an entity extractor. At first, the utterances has featurized to form a bag-of-words vector, then the utterance embedding has created and the entities has extracted. This entire information has concatenated into a feature vector which has passed to RNN, such as LSTM or Gated Recurrent Unit (GRU) to compute the hidden state dimensionality vector. This state information has then passed to the dense layer with an activation function. The action mask has applied as an element-wise multiplication, and the obtained result has normalized to a probability distribution. The irrelevant actions has the probability of zero. The action with the highest probability has selected as output, in which the invoked entity output has added to make a full action.

In necessary cases, The API calls have made to render rich contextual information to the user. The contextual updates to the current dialogue of the user has updated by tracking the entities and it has achieved by the developed software [**13**]. The LSTM has fine-tuned with parameters like the optimizers, size of the hidden units, and the epochs. The bAbI dialogue dataset has used for the training and the evaluation process. The turn accuracy and the dialogue accuracy has calculated. The addition of domain knowledge has made the HCN to perform well with perfect accuracy similar to Rule-based approach.

3.4.2 ConvLab - Multi Domain dialogue system platform

The joint collaboration of Microsoft Research, USA and Tsinghua University, China have developed ConvLab, which is a multi-domain dialogue system using end-to-end learning method. They have used MultiWOZ dataset and they have annotated the utterances to train the models.

They offer two kind of architectures to develop ICAs. The one is the integration of multiple components, such as NLU, DST, Dialogue Policy learning, and NLG, and the second one is fully end-to-end learning model. The latter reduces the developing effort by manual coding. It has stated that it is the first platform to train using statistical models for all components, where as the other tool kits has used statistical models only for policy learning components and some tools uses only a baseline models.It has also stated that there is no platform which provides a mechanism to handle multi-domain and multi-intent utterances. The implementation design on ConvLab has depicted in Figure 3.3 below [**14**],

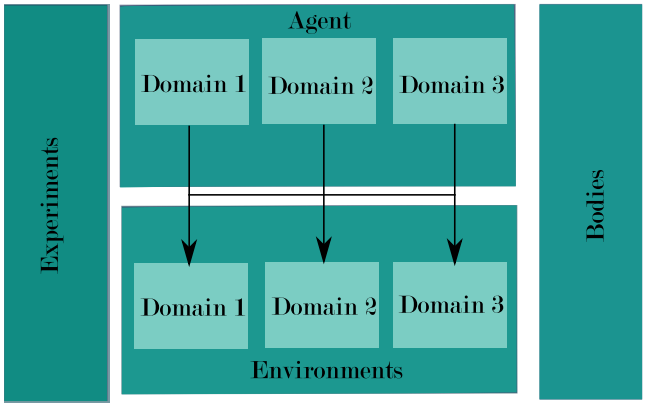


Figure 3.3: Implementation Design of ConvLab [**14**]

ConvLab has unique features such as open source multi-domain end-to-end dialogue system, tools which uses statistical models to build dialogue systems using different approaches, and evaluation using humans and simulators. They have designed Agent-Environment-Bodies (AEB) as depicted in Figure 3.5, where Agent represents the dialogue agent, Environment is the user simulator or human to evaluate the components, and Bodies are the integration of an agent in an environment. Each body has the capability to store the specific data. ConvLab has built with AEB design such that it can be trained not only in single domain but also in multi-domain and multi-agent conditions without any hand-coding logics [**14**]

- **Multi-agent Learning** In this approach, the utterances of the multiple domains has mapped to the corresponding actions of a specific domain through a joint observation of all domains. A centralized agent has used to achieve this. A disadvantage of this approach is that the observation points and the actions spaces grows exponentially with respect to the number of domains. To tackle this, a centralized agent has to be separated into two separate domains.

- **Multi-task learning** By this approach, transfer learning has applied to learn the common knowledge across the multiple domains. Hence, each bodies can present in the respective environments or in other environments to share the knowledge

- **Role Play** This has achieved through Reinforcement Learning and the agent to agent conversations takes place, such that one acts as an agent and the other as user. This experiments consists of multiple control layers such as Session, trial, and the Experiment. Session runs for pre-defined number of episodes by initializing agents and environments. Trial has the fixed set of parameter values to run multiple sessions, and then the results

have averaged. Experiment takes the hyper-parameters as input variables and the results have measured by metrics such as success rate, and the average reward.

Model Configuration

NLU - The NLU unit of ConvLab uses three models, namely, Semantic Tuple Classification (STC), OneNet, and Multiple-intent LU (MILU). MILU model has designed to handle Out-Of-Vocabulary words and multiple-intent dialogue acts [**14**].

Dialogue State Tracking - The rule-based technique has used to track the belief state and it has used to handle multi-domain interactions. The word level DST has performed by taking the system and the user's inputs to update the state of the dialogue.

System Policy - It has used hand-crafted policy, supervised learning models and the RL models to select the next system action for the present state of the dialogue.Word level policy uses sequence-to-sequence model to implement the dialogue management

NLG - It uses templates to generate the responses for the predicted actions or else Semantically Controlled- Long Short Term Memory has employed to generate natural responses.

Experiments and analysis From the experiments, it has stated that that word level DSTs have shown better performance. It has shown that the overall accuracies of rule-based DST and the word level DST have given 90.2 % and 89.7% overall accuracy, where as the end-to-end learning models success rates were 69.05% 16.67% [**14**].

3.5 Rasa Framework

3.5.1 Insight into Rasa framework

Rasa is an open-source Conversational AI interface, which requires the installation of multiple components to build conversational agents to accomplish specific tasks. The components of Rasa have built using python language, and hence it requires technical python coding skills to develop a bot using Rasa. The Rasa provide options to store the trained model in the cloud storage of Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Storage Each component of the Rasa tool have categorized into two main parts, such as Rasa Natural Language Understanding (NLU) and Rasa core [1]

- **Rasa NLU** Interpretation of the user's utterances and handles Feature selection and extraction
- **Rasa core** Tracking of the dialogues and fulfilment of the tasks by predicting system actions for each utterance

3.5.2 Rasa open source - Architecture overview

Rasa tool requires Feature selection and extraction from the utterances of the user to develop a bot. The dialogue tracker has used in order to track the state of the dialogue and select appropriate actions using a specific policy.The interpretations of the user's utterances come into the category of Rasa NLU, which requires the selection of the necessary features and it's extraction. The Dialogue State Tracking (DST) is helpful to track the current state of the uttered dialogue, and the policy is useful to predict the corresponding system action for the current state of

[1] https://rasa.com/docs/rasa/

the dialogue. DST and policy, which has known as dialogue learning or Dialogue Management together, come into the category of Rasa core [1].

Rasa tool offer to prepare templates for the list of all actions, which has to be taken by the bot, or else specific messages can be dispatched, which are approved by humans while developing a python script for the custom specific tasks. The machine-learning-based Natural Language Generation can also be integrated with the Rasa instead of templates or message dispatcher. End-to-End learning has not employed yet in Rasa, where the generation of responses relies completely on the corpus data. It uses a single strategy in order to generate the corresponding responses for the specific input utterances, where any mishaps in the annotation (in the case of supervised learning) of the utterances can't be handled, and it has to be neglected, Whereas Rasa tool offers components to handle these mishaps in order to build bots for the businesses, where customer satisfaction plays an important role, and hence fulfillment of the specific tasks in a reliable way is one of the key requirements [1].

Rasa NLU and Rasa core require separate training data for each. The training data for Rasa NLU consists of utterances of the user, and the entire one-to-one conversations are the training data for Rasa core. The training data for Rasa core has the name as stories, and the utterances of the user and the corresponding system responses have differences in their representation. The system responses have to be given either from the templates (or dispatch messages using python script) or from the trained NLG component. This architecture has shown in Figure 3.4 [1],
The NLG component can be trained externally from Rasa using a custom strategy and then loading the trained model for an appropriate response by providing an input sequence to the trained model.

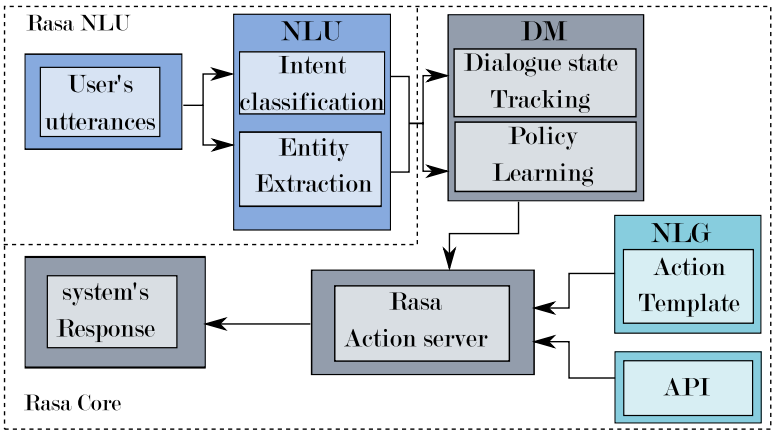


Figure 3.4: Architecture workflow for system responses using templates

3.5.3 Feature Selection and Extraction

The two main important features for the Natural Language Understanding techniques are Intents (Dialogue Acts) and Entities (slots). The training data provided to Rasa NLU have these features annotated. Intents represent the intention of the user in the utterances, and entities are the associated information to the corresponding intents, which gives additional information with that intent. Each entity has its value, and the utterances must have the annotation of its start and end value. Rasa offers a variety of pipelines to classify intents and extract entities, namely Spacy, MITIE, and Tensorflow embeddings. In the chosen pipelines, the corresponding sub-components such as Featurizer, Tokenizer, Intent classifiers, Entity extractors, and/or

selectors have to add to extract the required features [1].

3.5.4 Data Format - Rasa

NLU is a Natural Language Processing (NLP) technique to classify intents, extract entities, and retrieval of a response for building a conversational agent. The data format used to train Rasa NLU can be either in Markdown or JSON format. The training data has structured into different parts [1], such as

- **Common Examples** is the mandatory part and includes the list of examples of utterances (text) associated with their respective annotated intents and entity-value pairs. The utterances are mandatory, and intents or entity-value pairs are optional
- **Regex Features** is an optional part that has regular expressions involved in the entire training data. For example, Phone numbers or Postcodes which has fixed size and data type
- **Synonyms** is also optional which has synonyms for each entity involved in the training data
- **Lookup tables** is also optional, which categorizes the list of values for an entity. The lists of values can be provided either directly as lists or as text files. For example, the list of currency values for an entity 'currency'

3.5.5 Components of NLU - Rasa

To handle NLU with the above-mentioned data format, Rasa offers three different pipelines to choose the various corresponding sub-components. The three different pipelines are, namely, pre-trained embeddings spacy, supervised embeddings, and MITIE. As MITIE embedding will be deprecated in the future version of Rasa, it has not described here. The lists of sub-components are featurizers, intent classifiers, tokenizers, selectors, and entity extractors.

3.5.6 Dialogue Management policy

Rasa has introduced Recurrent Embedded Learning Policy (REDP) to embed system actions and the states of the dialogues in the same vector space as described in the Star space Algorithm. It has a memory component associated with a modified version of the Attention Mechanism proposed in Neural Turing Machines. It has stated that the performance of REDP is better than the baseline LSTM classifier. REDP has developed for significant key points, such as holding longer conversations to perform a goal and keeping track of the history of conversations. Since it is not possible to feed every possible dialogue types of a human to a conversational agent, REDP has implemented for a coherent response to new information [**15**].

In task-oriented dialogue systems, fully hand-crafted systems perform better and have manually coded by a developer but limited only to the logic developed. In end-to-end systems that have trained using non-annotated dialogue, and it does not have any domain-specific knowledge. Thus, REDP has developed using statistical approaches along with a domain-specific knowledge; especially, it has shown interest in learning from small datasets[**15**]. The slot filling requires the goal of the user, and the collection of that information to complete the task. The cooperative dialogue has stated that the response of the user to the prompts of the system with the information, which system requires. An uncooperative dialogue has stated that the response from the user with a lack of information, which has requested. In REDP, a classifier has not employed to select a system action. [**15**].

Featurization

At first, the user input, slots, and values have featurized. The intents and the entities extracted from the NLU system are the labels for the user's inputs, and the action names serve as the label for system actions. The token counts inside each user and system label have used to create a bag-of-words representation. The binary vector representation of the slots has used to determine the availability of the slots. The slot overwriting method has employed to track the most recently specified value for each slot [**15**]

Embedding Layer

The embedding of the feature vector has performed for each user utterances, system actions, and corresponding entities. The attention mechanism has employed, which uses the previously predicted action embedding and the present user's input for its determination [**15**]

Attention

The standard Attention Mechanism has the limit that it can attend to fixed memory size, and hence it cannot attend the future time steps. REDP corresponds to future time steps as well while training. But during inference, it is not possible to attend future time steps as it is unknown. Hence, attention mechanism can attend only to past inputs during inference, and so the distribution of attention has calculated over a truncated memory until current dialogue [**15**]. This policy has the modified version of Neural Turing Machines, where the interpolation gate has not applied to attention probabilities since the memory has truncated up to the current dialogue step. Thus, Interpolation has applied to Bahdanau scores for previous time steps, where the scores have used without Interpolation and calculated without using normalization. [**15**].

Memory Cell

For Memory cell, LSTM with Chrono initialization has used along with forgets and input biases. The recurrent drop-out has also used in the hidden state in order to achieve regularization. The embedded user input and user attention have summed and fed to the recurrent cell. It has then concatenated with embedded slots and fed into another embedding layer. Then the cell output for the current time step has obtained. Then the cosine similarity measure has used to determine the system action, provided with the attention of previously predicted system action and the user input. Thus, previously predicted system action has less similarity measure for the varied user utterances.

3.6 Insight into problems in the state of the Art methods

The KGD-based ISU based approach is one of the practical solutions for domain-specific dialogue systems. It has a limitation that it requires the domain-experts to create a Knowledge-Graph for every possible dialogue state, and system actions in complex domain-specific environments become cumbersome, and it experiences from domain adaptation problems. The developer has to create new heuristic rules for a new domain, which takes lots of effort, time and enhances the monetary value to develop for each domain with new heuristic rules and Knowledge-Graph

Because of the technical advancements in end-to-end solutions by applying supervised learning and the combination of deep learning with Reinforcement Learning algorithms, it is of both research and commercial interest to automatize the dialogue systems to attain a better user experience. The supervised algorithms of RNN architecture using LSTMs with several attention mechanisms have gained popularity for implementing dialogue systems that have the capability

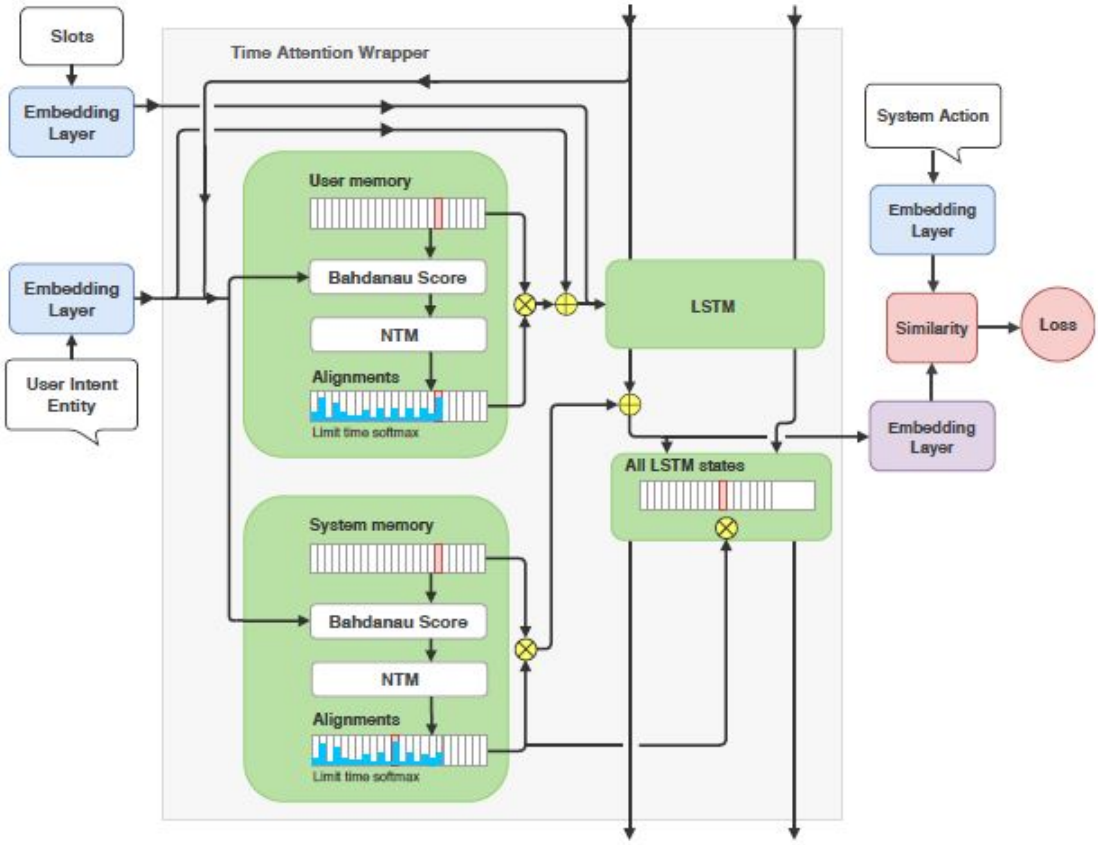


Figure 3.5: Recurrent Embedded Learning Policy for one time step [**15**]

to reduce the vanishing gradient problems, but it suffers from the limitation of Catastrophic Forgetting (CF). Catastrophic Forgetting has defined as an abrupt loss of previously acquired knowledge when re-training deep recurrent LSTM networks with new samples. From the Research analysis, it has stated that the CF occurs universally for all Deep Learning algorithms regardless of the construction of the sequences [**16**].

3.6.1 Catastrophic Forgetting

The sequence classification problems impose various challenges due to the variable length of inputs. The dynamic state of the recurrent systems represents not only to the present sequences but also to the previous sequences along with a linear read-out mechanism to infer the sequence class. As LSTMs trains based on gradient descent approach, it makes it more feasible to represent the dynamic state of the recurrent systems. This behavior enriched its application for NLP techniques, and especially for dialogue systems. The corpus-based supervised training of dialogue systems involves several complications, such as variants in dialogue turns and multi-variate corresponding response, multi-domain adaptability, and continual or incremental learning. As dialogue systems involve multiple sequence class, LSTMs are prone to forget the previously learned sequence pairs, when new multiple sequence pair has presented to it.

3.7 Existing techniques for the dialogue system evaluation

3.7.1 AttraktDiff Live user evaluation - Survey 1

AttraktDiff is used to measure the attractiveness of a product or a business software for comparison by asking the study subjects to fill out the Questionnaire for measuring the perceived combination of the quality aspects. The hedonic, and the pragmatic quality measure has used to analyse the attractiveness of the product. The overview of the model in determining the attractiveness of a product has depicted in Figure 3.6 as follows, The usability evaluation inves-

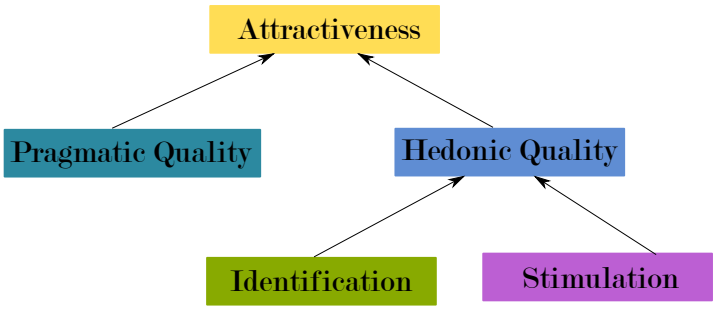


Figure 3.6: An overview of the model to measure attractiveness[**22**]

tigates the quality of a product by analysing the efficiency, and the effectiveness of a developed product or a software to decide whether it can be handled for the real-time applications. The user interfaces are obliged to appeal the users by its attractiveness. The pragmatic quality and hedonic quality measure have used to measure the overall attractiveness of a product. The efficiency, effectiveness,and the learnability have measured using the first, and the the qualities of indirect relations to the tasks such as originality, and the beauty have used in the latter quality measure. The attractiveness has determined by the average of perceived pragmatic, and the hedonic quality measure. Attractiveness of a product in proportion to the subjective importance of these quality factors which have mentioned above.

It has stated that the from the research hypothesis, the pragmatic quality has more influence on the attractiveness than the hedonic qualities, while the pragmatic quality directly involves on the tasks. The subjective measure might differ, when a subject decides to evaluate most probably on the usability conditions. Depending on the usability settings, the quality measure of hedonic or pragmatic quality differs in connection with the attractiveness of the product. Thus, number of subjects under study matters in determining the attractiveness of a product. Hence, an appealing product must be usable , which means that the dependency between the qualities which are subjective should have an influence in a positive way. Thus, it is important to choose the subjects in a wide range of perspective to get an influence of both quality measures on the attractiveness [**22**].

The questionnaire has prepared to measure the attractiveness, pragmatic, and the hedonic quality by using a structured list of semantic differentials of opposed adjectives. The subject can scale it between the seven points for seven items in each quality aspects. These ratings have performed into an online form.

Qualities involved in analysis for evaluation

The various quality aspects which have considered in the overall usability evaluation of the user experience of an interactive product are as follows [**21**],

- **Hedonic Quality** represents the qualities such as stimulation which denotes the improvement in the knowledge and skills, personal growth, Identification which represents the expressing oneself better, Interaction with an appropriate relevance in the dialogues with others, and evocation which represents self-maintenance and the memories.
- **Pragmatic Quality** represents the behaviour in the course of interaction to attain the specific tasks with clarification
- **Attractiveness** represents the higher level evaluative aspect in order to compare it with other evaluative products with the notation of goodness, and pleasantness. These aesthetic quality denotes the appeal of the developed product or business software in the user experience.

3.7.2 Subjective Assessment with usability standards

It is very significant for business product software to attract users, which plays a crucial role in the marketing of the product. Hence, the perception of a product by subjective assessment in terms of efficiency, its effectiveness, and the satisfaction has a correlation with the performance metrics, determined on the usability metrics affiliated to ISO standards. From the usability experts, eight factors have chosen for the assessment, such as completion of tasks and their quality, robustness, learnability, reliability, and ease of usability. The questionnaire has designed based on the factors mentioned above [**26**]. The Cronbach's alpha measure, the correlation coefficient, and the consensus measure have used to make the judgment based on the overall subjective measure of each characteristics in the qualities mentioned above [**27**].

Chapter 4

Implementation of the proposed strategy

4.1 Selection of Corpus

The statistical solutions for building ICAs have the complications such as, context-based decision making, integration of third party services to offer a service for each turns in the dialogue, natural understanding and generation of human language, and dialogue management. The corpora of dialogues is the essential requirement to develop such dialogue systems. Though there are different conversation corpora available, based on the annotation it has divided into two categories; namely, corpora which has structured labels associated with the dialogue and the corpora without structured labels but with an implicit intent represented in the dialogue. A brief analysis on different datasets has given and a suitable dataset has selected with the specific reasons have explained.

Multi-WOZ dataset has the collection of human-human conversation using WOZ experiments.The comparison of the Multi-WOZ dataset to other datasets have shown in Table 4.1 below, The

Table 4.1: Comparison of Multi-WOZ corpus with other similar datasets. The high-lighted numbers represents the best value of the respective metric [**20**]

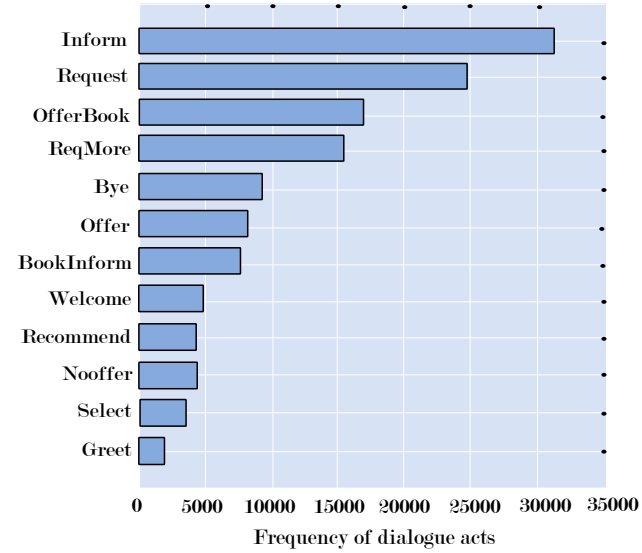
Metric	**SFX**	**WOZ 2.0**	**MultiWOZ**
Dialogues	1006	600	**8438**
Total turns	12,396	4472	**113,556**
Total tokens	108,975	50,264	**1,490,615**
Average number of Turns	12.32	7.45	**13.46**
Average Tokens per turn	8.79	11.24	**13.13**
Total unique tokens	1473	2142	**23,689**
Slots	14	4	**24**
Values	1847	99	**4510**

slots have divided in general into two categories, namely, inform slots and request slots. The inform slots represent the attributes, which allow the user to constrain the search, whereas the request slots give the additional information specific to a domain, the user may request [**20**].

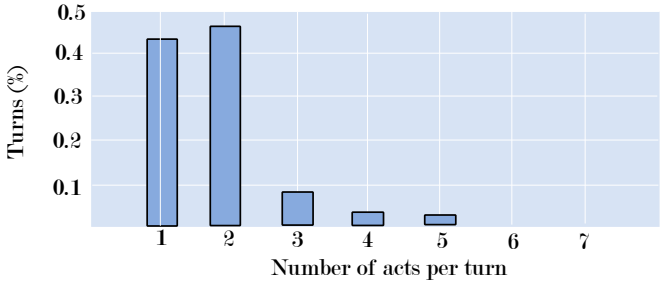
Annotation - The annotation is the most difficult process to make a dialogue corpora. Usually, each intent has its corresponding slots associated with it. The Amazon Mechanical Turk has used for the trial sets of subsets of the conversations. The discrepancies had occurred in the annotation of the dialogues. The workers who had performed well have screened from the

two-phase trials. The selected workers had chosen to annotate the long dialogues. It had solved the discrepancies in the annotation. The high-quality annotations have inspected, and the corrections have reported to the annotators. Then, the passed workers have allowed for annotating the original dialogues [**20**]

All the conversations have collected in two process sequences, and they have annotated. The errors in the intents have obtained and then corrected. A total of 10,438 dialogues have collected. There are 3406 single domain dialogues and 7032 multi-domain dialogues up to 2 to 5 domains. The average number of turns of 8.93 had seen for single-domain dialogues and 15.39 for multi-domain dialogues [**20**]. The distribution of intents across multi-domain and its corresponding number of turns has depicted in Figure 4.1 as follows, For reference, the Restaurant domain of



(a) Frequency of each intent in MultiWOZ corpus [**20**]



(b) Number of intents per turn in the collection of corpus [**20**]

Figure 4.1: Information of Intents in the MultiWOZ corpus [**20**]

the Multi-WOZ dataset has compared with the WOZ 2.0 corpus for the test accuracies. The results that have obtained are 89.7% and 96.5% of overall accuracy. The multiple intents of '1' and '2' have occurred frequently, and the joint of two intents have the highest occurrence frequency. The frequent occurrence of the intents that have involved in the conversations is 'inform' and 'request.' The encoder-decoder architecture with bi-directional GRU and LSTM cells have used to estimate the efficiency of Dialogue Management for Multi-WOZ and Cam676 dataset. The best results have obtained for the Cam676 dataset for bi-directional GRU cells.

4.2 Customization of the selected corpus

The selected corpus has multi-domain dialogues, which makes it too complex to build ICAs using rasa, and also the proposed work concentrates on building a domain-specific ICAs. The Python dictor package has used to extract necessary information from the JSON data. The restaurant reservation domain has selected to build ICAs. The customized logic has created to obtain the information of Restaurant reservation setting, and the algorithm has shown in Figure 4.2. The MultiWOZ corpus has multi-domain dialogues of conversation, such as restaurant,

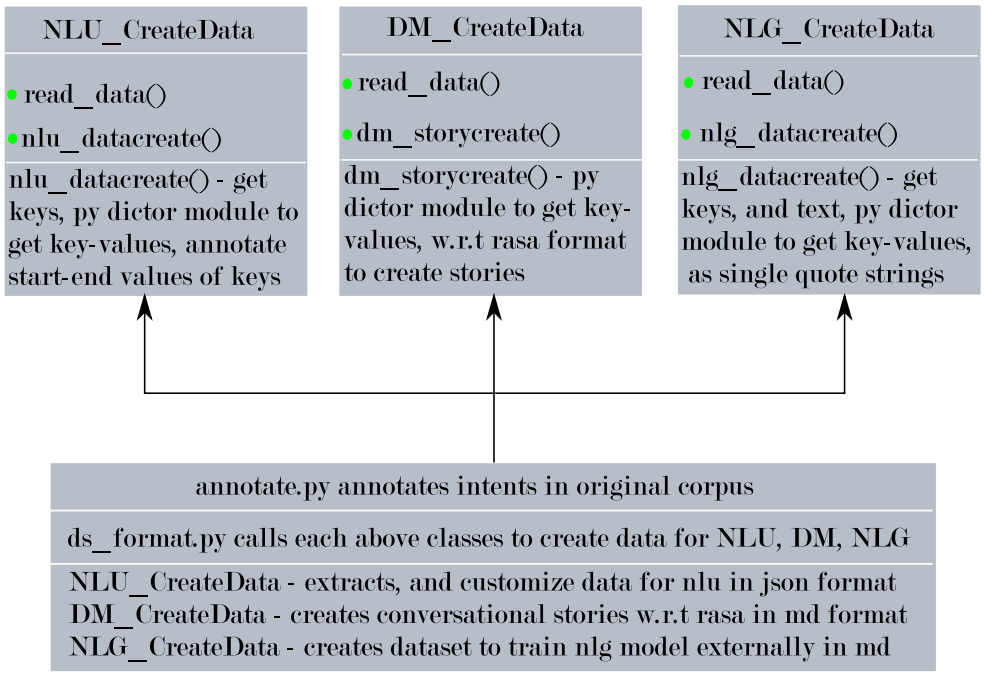


Figure 4.2: Algorithm to collect the data from Restaurant reservation setting

hotel, attraction, taxi, train, hospital, and police. The talks of Restaurant reservation domain have collected by analyzing the conversations involving in a restaurant reservation domain of MultiWOZ corpus. The custom logic has used to obtain domain-specific dialogues, which has shown in Figure 4.2. The annotated MultiWOZ corpus has multiple intents in a conversation. Though the associated entities have provided, the representation of their corresponding values in each dialogue has not specified. Hence each entity has identified in the dialogue, and it has represented with their 'start,' and 'end' values. The type of intents and the slots information, which have involved in the dialogues of the restaurant reservation domain has shown below in Table 4.2. It has identified that certain dialogues have not annotated, and certain dialogues have no entity-values information. As the dialogues in this corpus involve human-human conversations, it has taken into account that each subsequent dialogues are from different humans.

The distribution of dialogues of each intent involved in restaurant reservation domain has specified in Table 4.3 below, The dialogue without intents has named with an intent 'outofscope'. These dialogues represent that the information specified cannot be perceived, which also occurs in natural conversation. The dialogues specific to an intent 'general-greet' does not involve in the MultiWOZ dataset. It has included additionally while extracting the domain-specific conversation from the original corpus data. The metric information for the customized restaurant reservation data of MultiWOZ corpus has given in Table 4.4 below,

Table 4.2: The ontology Information for restaurant domain in MultiWOZ corpus [**20**]

Semantic Frames	**Specific information**
Intent	inform, request, select, recommend, not found, request booking info, booking offer, inform booking, decline booking, welcome, greet, bye, reqmore
Slots	address, postcode, phone, name, area, price range, number of people, reference number, day, time, food

Table 4.3: The dialogue distribution of intents in restaurant reservation domain of MultiWOZ corpus [**20**]

Intents	**Dialogue distribution**
general-bye	293
general-greet	25
general-thank	1171
general-welcome	76
Restaurant-Inform	5769
Restaurant-Request	1792
Restaurant-recommend	282
Restaurant-Select	274
Restaurant-Nooffer	462
Booking-Book	457
Booking-Nobook	241
Booking-Request	192
Booking-Inform	391
Without intents	566

4.2.1 Dataset preparation - customized approach

The domain-specific information which has extracted for restaurant reservation domain suits for end-to-end learning. In the proposed approach, four basic components have involved, namely NLU, DST, DM, and NLG, to build a domain-specific ICA. Hence it requires three different datasets for NLU, DM, and NLG component, which has to comply with the format supported by Rasa open-source framework. Though the customized data from the corpus has a human-human conversation, the dialogues between each human have referenced as 'user utterance,' and 'system utterance' for convenience to build a domain-specific ICA. This is the first version the datasets that have obtained from the original MultiWOZ corpus. A brief description about creating the datasets for all three components have specified below,

- **NLU** In the entire conversations available in the customized corpus data, the user has initiated the dialogues. At first, the user greets the system. Then, the user asks for information regarding the domain-specific scenario. The user-specified dialogue has segmented into semantic frames, which have intents associated with the corresponding entities and their value. These semantic frames have obtained from the annotated information of each dialogue. As the natural conversation has multiple intents associated with the corresponding entities and value pairs, and to reduce the training complexities for a neural network, the intents have modified which have specified in Table 4.5. At last, the user thank the system for providing the necessary information and end the conversation

- **NLG** In the corpus data, all the system utterance has followed by the user utterances.

Table 4.4: The Metric Information in restaurant reservation domain of MultiWOZ corpus [**20**]

Metric	**MultiWOZ (Restaurant)**
Dialogues	11537
Total turns	5759
The average number of turns	4.3928
slots	11

In the beginning, the system has to greet the user back when the user has greeted. Then the user utters a dialogue to get domain-specific information or else command the system to do domain-specific tasks. For a user uttered dialogue, the corresponding subsequent dialogue, which has given by the system, has segmented into semantic frames of intents and entities-value information similar to NLU semantic frames. When the user asks for specific information, the results from API calls has to update the corresponding entity values. Hence the entity-specific information uttered in the original system utterance has delexicalized to the corresponding entity name. At last, the system has to thank the user when all the user requests have satisfied.

- **DM** Dialogue Management plays a crucial role in building ICAs, as it decides the system action depending on the user utterances. For each user utterances, DST keeps track of each dialogue for every conversation to handle and update entities and their corresponding value information. For the current state of the dialogue, which includes previous events information, it is important to deliver a proper system action. It has achieved by mapping semantic frames information from the user's utterances to suitable semantic frames information, which has to involve in the system's utterance.

Table 4.5: The dialogue distribution of modified intents in the customized dataset

Intents	**Dialogue distribution**
generalgreet	21
restaurantinform	2608
restaurantrequest	1009
byetothesystem	1379
outofscope	505
informtobook	267

A sample dataflow of information to build the domain-specific ICAs has depicted in Figure 4.3,

4.2.2 Limitations - Original Multiwoz customized corpus data

Natural Language Understanding Corpus The data distribution of multiple intent dialogues is less. Hence, the multiple intent dialogues have generalized. The generalized dialogues, such as 'yes,' or 'no,' and the sentences without any contextual information have not annotated. Also, some of the dialogues have the same intent, which has annotated with variant names. Let's say that in dialogue, the user thank the system for the service, and say goodbye. These dialogue types have annotated with two intent variants, such as 'general-thank,' and 'general-bye.' It adds complexity to the system while learning from the data since there is also no balance data distribution. Additionally, some dialogues involve only the intention without any contextual information in it. In this case, only the specific intent has annotated without any contextual

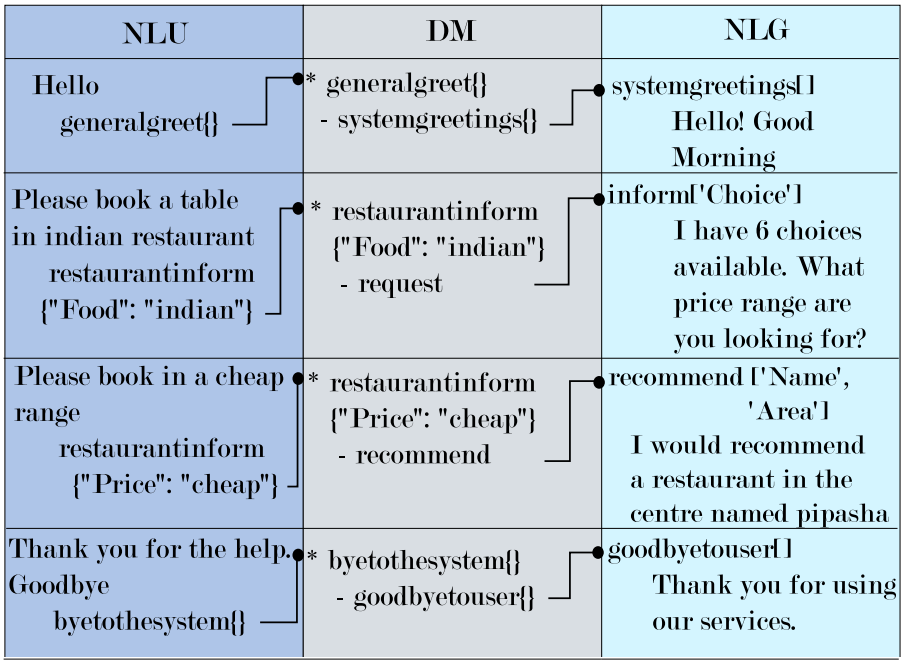


Figure 4.3: The dataflow in between NLU, DM, and NLG extracted from MultiWOZ corpus

information.

Dialogue Management Corpus The customized corpus data for DM has limitations, such as the stories for each conversation comply with the API database information. It means that when any API information is not available for any user specifications, then the system action has chosen as 'no offer' for that specific intent with the associated entities. When the dialogue stories comply with the available API database, then it isn't effortless to modify the entire dataset when there is a new entry in the API database. Let's say that there were no German restaurants available in the API database while building the system, but later it has added in the API database. In this scenario, the dialogues that have involved while training the system would have trained to make the correct system prediction as 'no offer.' After a new entry has included, the trained system could not handle to provide the available option, since the system has trained with the dialogue stories to inform the user that there are no offers available.

As said before, the multiple user intents which come into one general category have generalized to remove the complexity for the system while training. Thus the generalized user intents have various combinations of possible system actions. Let's say that the user intent is 'Restaurant Inform,' and the possible system actions are 'Request,' 'Request Inform,' 'Select,' 'Select Inform,' 'Recommend,' 'Recommend Inform,' 'Recommend Reqmore,' 'Reqmore Inform,' and 'Inform.' Also, for the user intent 'Restaurant Request,' the possible system actions are 'Inform' and 'Reqmore Inform.' The system actions for both the user intents have a coincidence. It makes the system hard to choose an appropriate system action for the perceived intent. The complexities that have involved in the DM dataset has depicted in Figure 4.4

Natural Language Generation Corpus As much contextual information has not annotated correctly in the original MultiWOZ corpus data, some of the contextual information in the sentences have not de-lexicalized properly. It has led to improper sentence delivery to the user, which had confused the user.

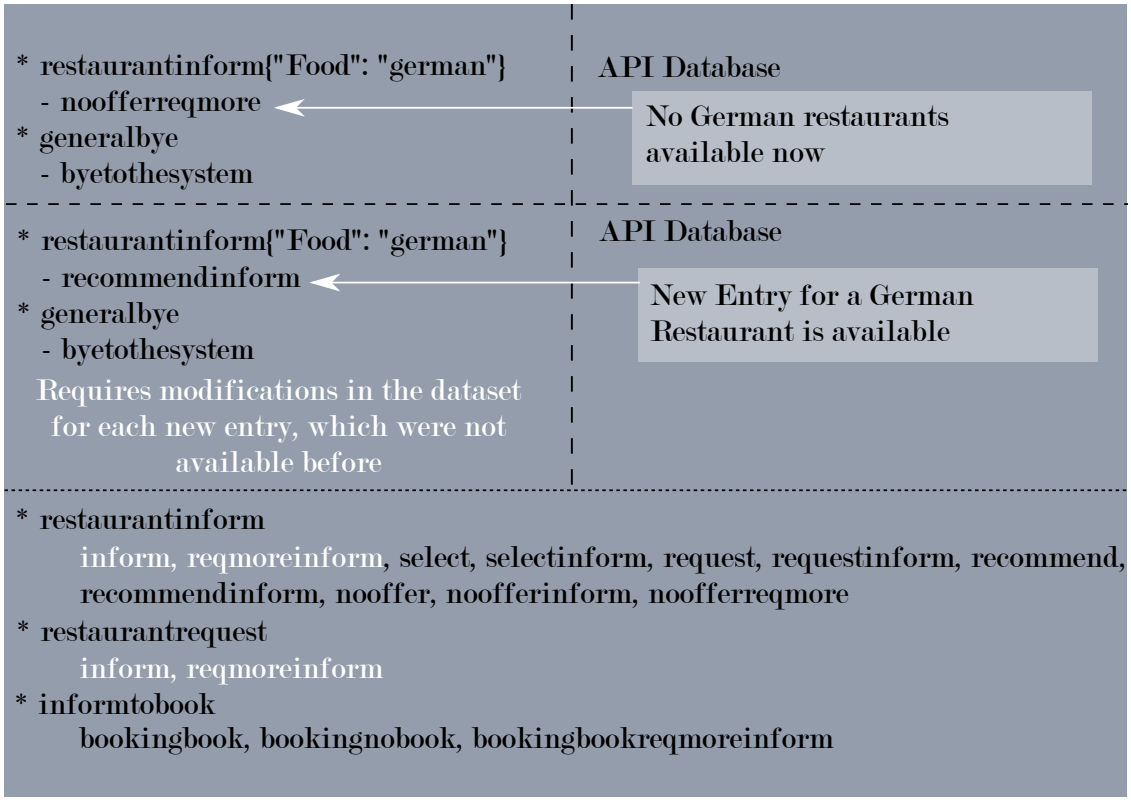


Figure 4.4: The complexities involved in the Dialogue Management Corpus data

4.2.3 Dataset Preparation - Manual approach

The dataset has modified manually for tackling the issues that have specified in the customized approach to increase the performance of the system by learning appropriately.

- **NLU** The original corpus data does not have any dialogues with the intent of 'greeting' at the beginning of the conversation. Hence, certain greeting dialogues have included while customizing the MultiWOZ corpus data with an intent named 'generalgreet.' The formal greeting dialogues only have provided. Still, various greeting dialogues have to be included in the corpus data, such that the system can handle it generously. Some entities that have involved in the conversations have not appropriately annotated. Let's say that there is an entity 'eastern' in the utterance, but it has annotated as 'east.' Hence, the associated start and end values have not annotated correctly. The incorrect annotations have manually handled since there were not many cases in this situation. The utterances that had no intents associated with it has annotated as 'outofscope' during the customization of the original corpus data. All the available restaurants available in the API database have included in the lookup table in the customized corpus, which has maintained for the manual approach as well. Though there were entities that have associated in the utterances, it has not annotated. Hence, these entities have annotated manually for a few samples. Due to time limitations, it has not followed for the entire dataset.

- **DM** Since the customized DM corpus has limitations, and the entire dataset has created manually. It has a strong influence on the quality of the developed product, and major importance has given to this part. To handle the new entries in the API dataset, a new input intent 'noavailablerestaurant' has added to the dataset. The idea behind this approach is that when there are no available restaurants for the user specifications, then the filtering technique has utilized to perceive the specific intent as 'noavailablerestaurant.'

The API database information has checked for availability, and when there is no available option, it has chosen as the perceived intent. Then, the possible system actions for the intent 'noavailablerestaurant,' are the one that is related to the system action 'no offer.'

To avoid the collision of possible system action for the perceived intent 'restaurantinform,' and 'restaurantrequest,' the system actions such as 'inform,' and 'reqmore inform' have restricted for the perceived user intent 'restaurantrequest.' All other possible system actions corresponding to the perceived intent 'restaurantinform' have maintained. Also, in the customized corpus, for the repetitive user input semantic frames, different possible system action is available in the various conversations. Hence, a unique combination of input semantic frames have created. Also, the possibility of the system action has segregated according to the user specifications. Let's say that the user asks the system for any cheaply available restaurants. Here, the user has not specified multiple contextual information. He had left some contextual information, such as type of food and price range. In this case, the dialogue stories have designed such that an appropriate system action has filtered by choosing the most suitable option from all the possibilities, has provided as a system action in the conversation stories. In this example case, the most suitable system actions are either 'request' or 'request inform' from all the other possibilities. When the user has specified two or more contextual information, then the conversation stories have designed such that anyone of the other suitable system action has chosen excluding the one that has mentioned above.

Also, attention has paid in balancing the distribution of the data. All the user input semantic frames have distributed according to the priority of the intent. Since it is a restaurant reservation domain, priority has ordered in the following way 'restaurant inform,' 'generalgreet' 'byetothesystem, 'restaurant request,' 'outofscope,' and 'informtobook.' Various priority has set to find an optimal distribution of the data to identify the effect of data distribution. Two versions of the dataset that has manually created have used to analyze the performance of the system finally.

- **NLG** The contextual information that has present in the customized corpus has manually modified. The contextual information has replaced by a suitable de-lexicalized word. The entities that have involved in the NLU dataset have used as the de-lexicalized words in the capitalized form. It has created to reduce the complexity of training the network model. It also helped to integrate the business logic in the response results. A customized sample has shown in Figure 4.5.

There is a moderate price European restaurant named zizzi cambridge at 18 Westhill Road. There are 5 other choices for your specifications. Would you like to go there or choose another one?

There is a PRICE price FOOD restaurant named NAME at ADDR. There are CHOICE other choices for your specifications. Would you like to go there or choose another one?

Figure 4.5: A sample dialogue that has customized for training NLG component. **The words highlighted in white represents the de-lexicalized words**

The relation between the user intents, and the possible system actions has shown in Figure 4.10

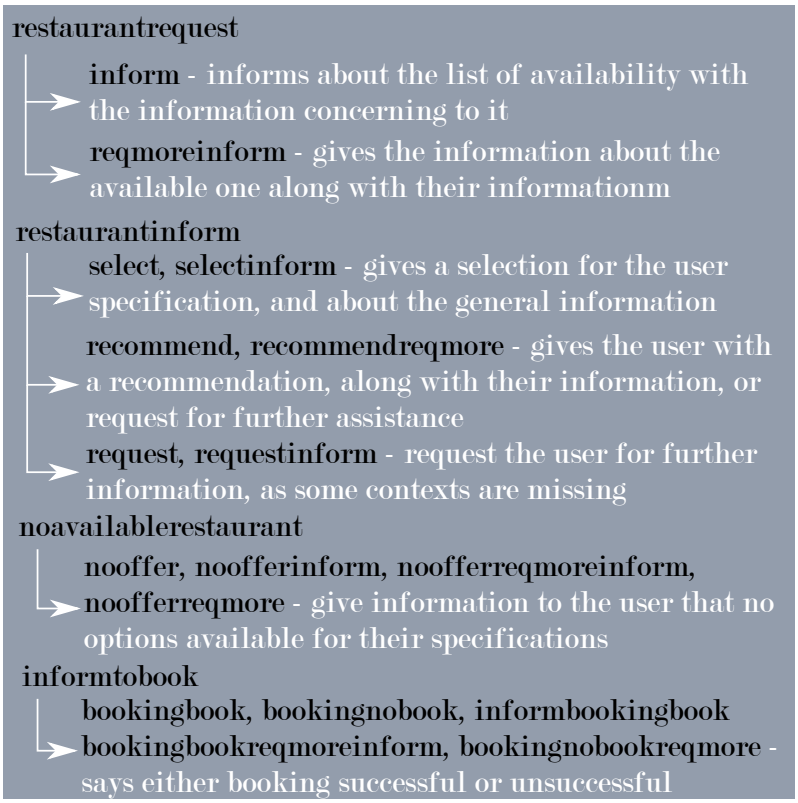


Figure 4.6: The relation between the user intents and the possible system actions for the Dialogue Management component

4.3 Customized architecture Overview of Rasa

The architecture that has used by rasa open source has customized for delivering automated responses. The architecture has modified to integrate business logic as well. The classification of intents and the extraction of the associated entities have performed by the rasa NLU component. The embedding intent classifier component and CRF entity mapper have used to perceive the user's intents and the contextual information. Along with these few other components from the rasa framework, such as count vector featurizer and the white space tokenizer have also used. The annotation of contextual information is the essential factor for classifying the intents, which has achieved according to the rasa format. The featurizer has applied to convert the input to features of vector for classifying intents, and extracting the entities.

Rasa core has used for Dialogue Management. The dialogue management policies have used by used to learn the possible dialogue stories that can involve in a domain-specific scenario. Rasa core has used default LSTM Keras policy, and REDP policy for learning the conversational dialogues to predict an appropriate system action. The custom DM policy has implemented in the Rasa core for predicting an appropriate system action.

The NLG component has trained externally using Keras deep learning library. A custom strategy has also proposed for training the NLG component to automatize the responses. Also, the existing neural network architecture has trained using the customized corpus, which had performed better. The trained model has used by the rasa action server to deliver the automated response to the user. The API database has hosted on a local server to integrate the suitable information in the responses that are available for the user's specifications. The customized architecture overview that has used to build ICAs has shown in Figure 4.7

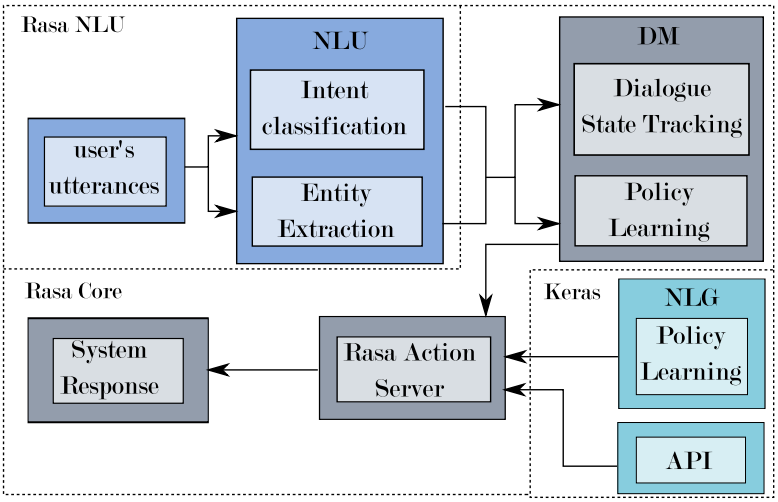


Figure 4.7: Architecture workflow for system responses by training Neural Network

4.4 Proposed strategy for Dialogue Management

The proposed strategy for Dialogue Management is a Progressive sequence-to-sequence learning architecture. The encoder and decoder have designed to learn progressively. It has inspired from sequence-to-sequence learning architecture, and Progressive Neural Network architecture [**17**] [**18**]. The encoder and decoder has separate LSTM as a hidden layer. It helps in transferring of knowledge that has learned in one domain to another domain. Hence, the proposed architecture has designed with a futuristic scope of adapting for multi-domain setting. In the proposed architecture, the hidden layers have stacked, and it has hyper-parameterized to learn progressively.

4.4.1 Progressive Learning of LSTM

Thus the proposed model architecture uses a modified LSTM hidden layer for computation, such that it has made flexible for transfer learning. For all the hidden layers in the first column, and the first hidden layers utilizes the gate mechanisms of input, forget, cell update, and output, which has derived to form an LSTM cell. Whereas, the stacked hidden layers of the succeeding column of the proposed architectures uses an adapter function for the cell formation. The adapter function performs the summation of resultant values of gate mechanism mentioned above and the previous hidden layer resultant values of the preceding columns. The calculation for progressive learning adapted in LSTM cells has derived in equations 4.1, 4.2, 4.3, and 4.4 below,

$$i_t = sigmoid[(U_i h_{t-1} + W_i x_t) + b_i) + \sum_{j<k} sigmoid(U_i^{k:j} h_{t-1}^j + W_i x_t^j + b_i^j)] \tag{4.1}$$

$$f_t = sigmoid[(U_f h_{t-1} + W_f x_t) + b_f) + \sum_{j<k} sigmoid(U_f^{k:j} h_{t-1}^j + W_f^j x_t^j + b_f^j)] \tag{4.2}$$

$$\hat{c} = tanh[(U_c h_{t-1} + W_c x_t) + b_c) + \sum_{j<k} tanh(U_c^{k:j} h_{t-1}^j + W_c^j x_t^j + b_i^j)] \tag{4.3}$$

$$o_t = sigmoid[(U_o h_{t-1} + W_o x_t) + b_o) + \sum_{j<k} sigmoid(U_o^{k:j} h_{t-1}^j + W_o^j x_t^j + b_o^j)] \tag{4.4}$$

Where, U_x^j and W_x^j represents the Recurrent Kernel Weight matrix and Kernel Weight matrix, which has obtained from the pre-trained information on corresponding corpus data.

The algorithm which has used to develop the progressive LSTM has depicted in Figure 4.8.

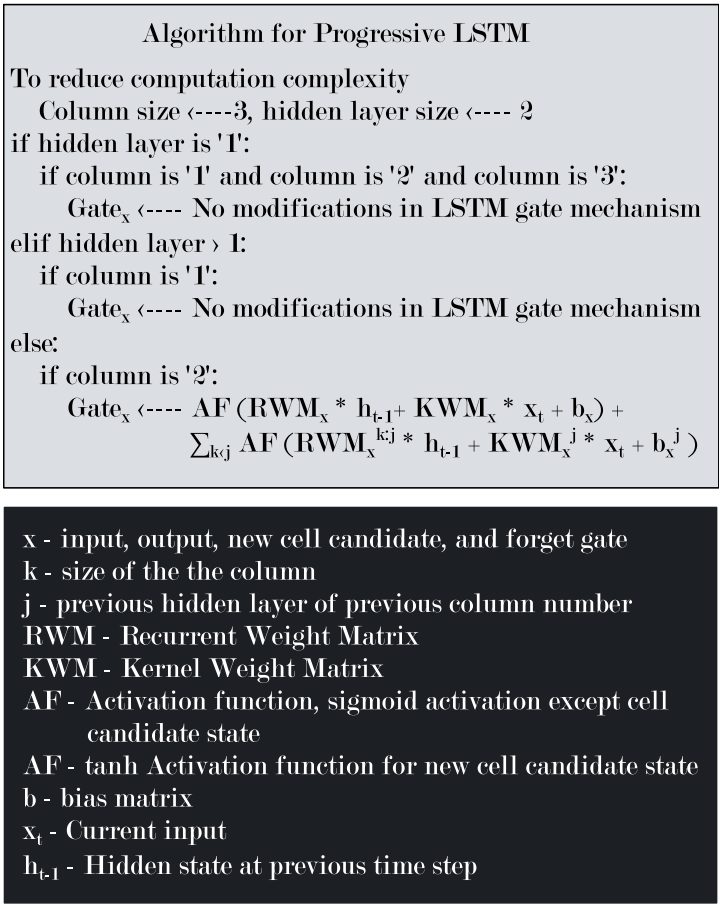


Figure 4.8: The algorithm for Progressive LSTM in Progressive Encoder Decoder Architecture

4.4.2 Custom Dialogue Management policy

The proposed architecture has the goal to estimate the conditional probability, which is similar to the sequence to sequence learning architecture. But, this Encoder-decoder architecture has modified to learn progressively for incremental learning to support transfer analysis, in order to develop a model suitable for domain-specific, and also for multi-domain scenarios. The proposed neural network has LSTMs stacked upto the size of '2' and has adapted for transfer analysis upto three domains.

It has stated that efficiency of learning of LSTMs is better, when the input sequences has reversed, such that the first sequence of the input, and the first sequence of the target sequence has close proximity in Neural Machine Translation tasks [**17**]. Hence, at first the source sequences has reversed, and its three dimensional vector state representation has calculated by first hidden layer of LSTM, which has then fed into a stacked LSTM with the returned sequences. The returned sequences from the first hidden layer has not reversed. Then, the encoder outputs from the stacked LSTM has neglected, and the state vector representation has

given to the decoder. Again, the target sequences has fed into the LSTM along with the state vector representation from the encoder in order to condition the target for the corresponding input. The returned sequences and the sate vector representation has then fed to another hidden layer of LSTM, and then the output has obtained by sampling each tokens corresponding to the input sequence until the stop character token has attained.

For the incremental learning, each upper stacked hidden layer in the succeeding column has trained not only from the previous hidden layer's states and outputs, but also from the preceding column's previous hidden layer outputs and states. Thus, the computation complexity increases, which is proportional to the increase in the size of the column. The current inputs used for training the succeeding column , has also fed into the LSTMs of previous columns which has frozen. By freezing the previous columns, the weight information does not get updated for the current input, but rather it utilizes the previously trained weight matrices to form the cell and hidden state outputs for the current inputs. The proposed architecture to learn progressively for single domain or multi domain transfer analysis has depicted in Figure 4.9. The proposed approach has also an advantage of using it for incremental learning for a single domain purpose, where the possibility of re-training the new domain-specific data becomes easier using the pre-trained old domain-specific data. This architecture has the flexibility to adapt itself for relative Multi-domain training, such that pre-trained information has its utility in the succeeding domain. Though the proposed approach supports Multi-domain setting, it

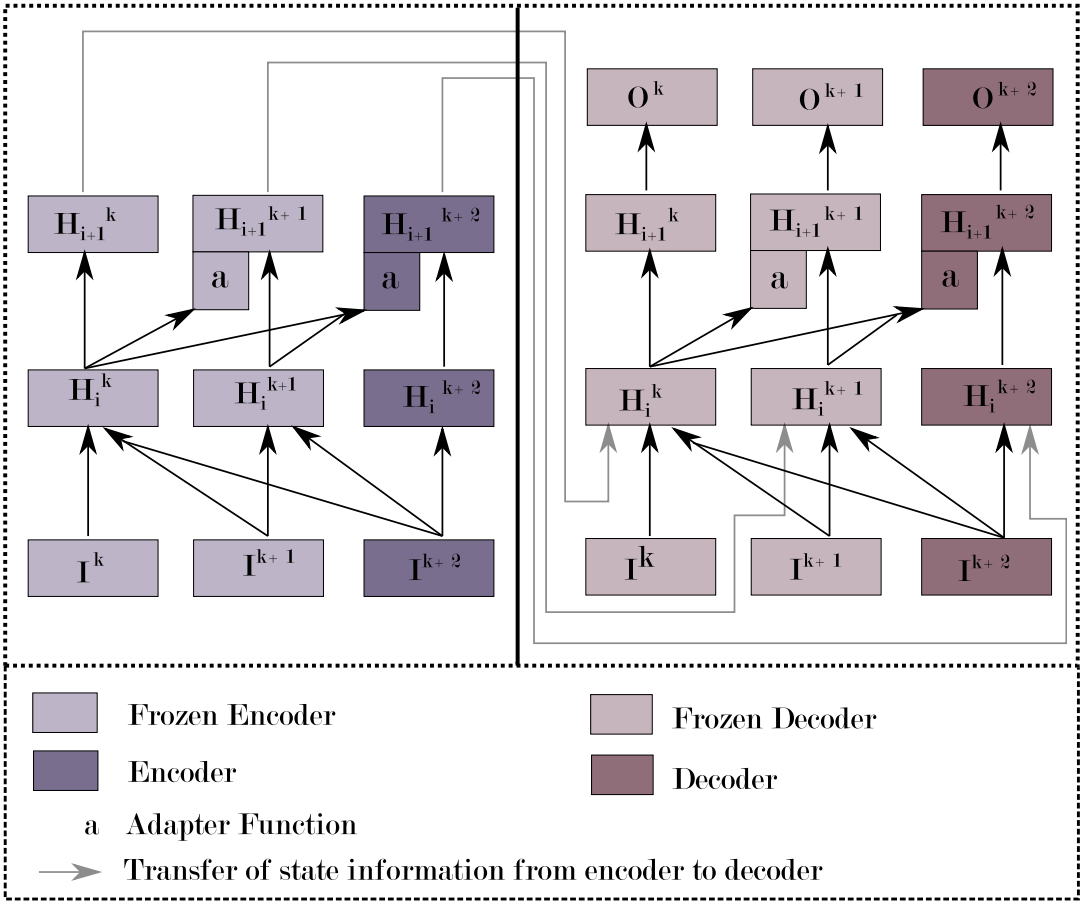


Figure 4.9: The Progressive Encoder-Decoder Architecture with two stacked LSTMs

is possible to use only single domain outputs at a time. The outputs of the specific domain,

and the necessary development settings have to be modified according to the requirements of the production settings. It means that for a production setting corresponding to the scenario trained in third column of the proposed approach, the outputs of the third column can only be used, the development settings for that specific domain has to be chosen.

4.4.3 Training of the Model

The proposed strategy has implemented in the keras policy class which has offered by rasa open source. The proposed custom architecture has combined from the sequence-to-sequence learning and progressive learning architectures [**17**] [**18**]. Hence, it has fine-tuned with various hyper-parameter settings to find an optimal model that performs better. The Stochastic optimization technique has employed, in which various optimizers that has mentioned in chapter 2 have used for the analysis. The best option for the proposed architecture has analyzed as 'Adadelta' optimizer. The suitable epoch value has found as '100' when compared to other epoch values, such as '150,' '120,' and '90.' The suitable batch size has found as '240.' Still, the epoch and batch size value can be optimized further by running various trials. The hardware specifications which has used for training the model has specified in Table 4.6. The dialogues involved in the

Table 4.6: The Technical specifications of the hardware used to train the model and the details of the used frameworks

Hardware and frameworks	**Specifications**
Hardware	HP Desktop -RAM - 8GB HP Personal Notebook RAM - 4GB, GPU - CUDA Compatibility 2.2
rasa	version 1.3.3
keras	2.3.1

NLU dataset is 5778,the user uttered semantic frames and corresponding system actions in the dataset for DM are 7507 each, and the system action semantic frames with the corresponding natural dialogues trained for NLG is 6000 which has specified in Table 4.7. The semantic frames beginning with '*' corresponds to user utterances, whereas the dialogues beginning with '-' corresponds to an appropriate system actions, which has occurred in the natural human-human conversations.

Table 4.7: The Technical specifications of the hardware used to train the model and the details of the used frameworks

Training component	**Number of Dialogues**
NLU	5778
DM	7507 for each semantic frames of user utterances and system actions
NLG	6000

4.4.4 Inference for selection of a system action

As the proposed strategy predicts the output depending on the conditioned input, it is essential to select an apt inference to predict the corresponding system action. As discussed before, in the preparation of the dataset, the user input semantic frames have an influence on predicting

the system action. Let's say that the user has requested any expensively priced restaurants that are in the center of town. Then, the system has to predict anyone from the possible the system actions, such as 'recommend inform,' 'select inform,' 'recommend,' or 'select,.' since the dataset has planned in this way, it is important to select an appropriate user input semantic frames for an inference.

At first, the input sequence has acquired from the user's utterance. The input sequence

```
Algorithm for the selection of system actions
during Inference

def inference_function
   intent ‹---- obtain intent from rasa tracker
   list_avail ‹---- api_function()
   inform_slots ‹---- obtain slots related to intent 'inform'
                      using rasa tracker
   request_slots ‹---- obtain slots related to intent 'request'
                      using rasa tracker
   book_slots ‹---- obtain slots related to intent 'book'
                      using rasa tracker
   int_wo_ent ‹---- list of intents without entities in dataset

   if intent in int_wo_int
      input_seq ‹---- string of obtained intent + "[]"
   elif list_avail is Null
      input_seq ‹---- explicit intent name
                     'noavailablerestaurant' + "[]"
   elif  intent name in 'inform' 'request' or 'book'
      input_seq ‹---- respective intent + corresponding slots

   split intents and entities by an appropriate token

   if length of entities is '1'
      if 'inform' in intent
         intent ‹---- 'noavailablerestaurant'

   for input, input_texts in the given inputs
      split intents and entities by an appropriate token for
      inferencing as well
```

Figure 4.10: Algorithm to perceive the user intents and their corresponding entities during the Inference

includes the intents of the user in the uttered dialogue, and the associated entities specific to an uttered intent. Let's say that the user intends to book a reservation. Then the associated entities are the number of people, booking time, and booking day. The dialogue state tracker tracks all the uttered entities from the beginning. Hence, the specific entities have filtered specific to the uttered intent, and it has considered as the input sequence for the current dialogue. The algorithm to handle this has shown in Figure 4.10.

It is not possible to infer all possible dialogues before-hand while building the system. The

representation of the dialogues differs for each user. Hence, the filtering mechanism is one of the requirements to handle this situation. If the user has specified only their intent without providing any slot value, then the system should also handle these scenarios. In this case, the system has perceived the intention of the user without any slot information. It checks the condition for the inference inputs concerning the uttered intent by neglecting the slot value information. It makes the prediction space larger for the system. The system can choose any one of the all possible system action of the intent uttered by the user.

If the user has uttered a dialogue with an intent that has associated with the slot values,

```
Algorithm for the selection of system actions
during Inference

for input, input_texts in the given inputs
    split intents and entities by an appropriate token for
    inferencing as well
    if length of entities in input_seq is '1'
        if length of intents in both input_seq and refernce
        input is same
            if length of system action predction less than '30'
                inference has chosen for this input condition
        elif 'inform' in intent
            if 'noavailablerestaurant' in the reference input
                if length of system action predction less than '30'
                    inference has chosen for this input condition
            else continue
        elif length of both intents are same
            if length of system action predction less than '30'
                inference has chosen for this input condition
        else continue

        Note: if length of prediction is greater than '30',then
              each condition breaks
    else the other condition has executed

                                        (...continued)
```

Figure 4.11: Algorithm to handle the system actions when the user has uttered dialogues without any slot values

then the filtering mechanism works effectively in sorting out the inputs corresponding to the utterance of the user. At first, the system checks for all the intents it has perceived, then it checks for all the slots specified by the user. If both the conditions have satisfied, then the corresponding input has chosen for inference. If the system cannot be able to find all the entities uttered by the user in the inference inputs, then the system should not fail to give a response. Hence, it checks for any of the slots values that the user has uttered in the inference. If this condition has satisfied, then the corresponding input has chosen for inference.

If any of these conditions have not satisfied, then the system chooses the perceived intent as out of scope. Thus, the system can take a fall back action. All the conditions have checked until the overall selected inference is less than or equal to '30.' Then, the system can choose any

one appropriate inference input from the appended list of data. It has finalized for selecting a suitable system action. This filtered input has fed to the decoder model to predict a system action by the sampling of tokens until the end sequence has found. The algorithm to handle these conditions has shown in Figure 4.12

```
Algorithm for the selection of system actions
during Inference
for input, input_texts in the given inputs
  if length of both the intents are same
    if all the intents are available in the inference
      if all the entities are available in the inference
        if length of system action predction less than '30'
          inference has chosen for this input condition

      elif any entities are available in the inference
        if length of system action predction less than '30'
          inference has chosen for this input condition

      elif length of both the entities are same
        if length of system action predction less than '30'
          inference has chosen for this input condition

      elif 'outofscope' in intent
        if length of system action predction less than '30'
          inference has chosen for this input condition

      Note: if length of prediction is greater than '30',then
       each condition breaks
    else continue
  else continue
```

Figure 4.12: Algorithm to handle the system actions by filtering mechanism during the Inference

4.5 Proposed strategy for Natural Language Generation

4.5.1 Architecture Overview

An attempt has made to generate the natural long sequences of sentences using an intra-attention (Self-Attention) mechanism, which has discussed in chapter 2. LSTMs produces a vector state representation, such that the next state has computed based on the current state, which means that for the current state h_t, the next state h_{t+1} conditionally independent in nature from previous states $h_1, ..., h_{t-1}$, and tokens $x_1, ..., x_t$. Since the update of states occurs in a Markov manner, LSTMs have assumed to summarize well for the current state for the tokens which have seen so far. It has stated that this assumption may fail, when the sequence is long. The information has processed subsequently in LSTMs by a sequential order, but there are no mechanisms to relate the tokens for modelling the structured output. Thus LSTMs with intra-attention mechanism has proposed to process the contextual representation for each input token, and retaining the contextual information between the tokens to produce the struc-

tured output [**9**]. The architecture of Progressive LSTM with an intra-attention Mechanism has depicted in Figure 4.13

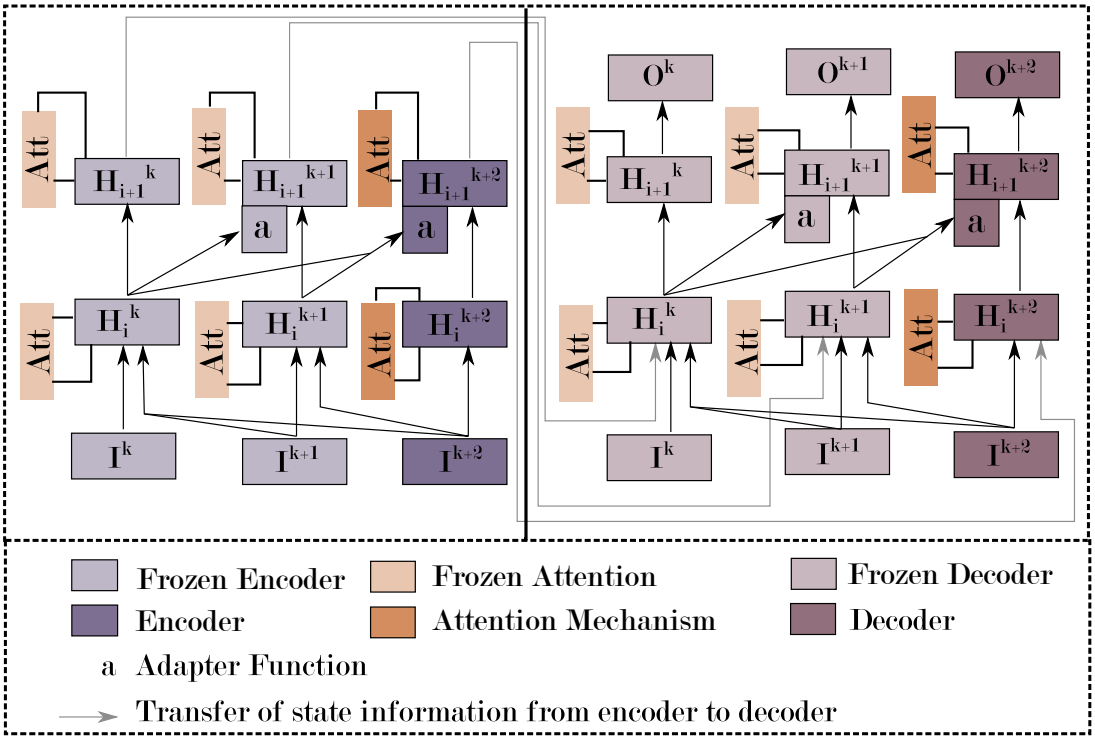


Figure 4.13: The Progressive Encoder-Decoder Architecture with two stacked LSTMs and self-attention mechanism

Hidden Layer Representation

The stacked LSTMs of progressive sequence-to-sequence architecture has provided with intra-attention mechanism. The attention mechanism for each gate mechanism has derived in equations 2.7 - 2.10 in chapter 2. Based on that, the gating mechanisms of each LSTM in the proposed architecture has given by the equation 4.5, 4.6, 4.7, and 4.8 as follows [**9**] [**17**] [**18**],

$$\begin{aligned} i_t = &\sum_{i+1}^{t-1} \sigma[U_i * h_i(softmax[v^T tanh(U_h h_i + W_x x_t + U_{\tilde{h}} \tilde{h}_{t-1})])] \\ &+ \sum_{k:j} \sigma[U_i^j * h_i(softmax[v^T tanh(U_h^j h_i + W_x^j x_t + U_{\tilde{h}}^j \tilde{h}_{t-1})])] \end{aligned} \tag{4.5}$$

$$\begin{aligned} f_t = &\sum_{i+1}^{t-1} \sigma[U_f * h_i(softmax[v^T tanh(U_h h_i + W_x x_t + U_{\tilde{h}} \tilde{h}_{t-1})])] \\ &+ \sum_{k:j} \sigma[U_f^j * h_i(softmax[v^T tanh(U_h^j h_i + W_x^j x_t + U_{\tilde{h}}^j \tilde{h}_{t-1})])] \end{aligned} \tag{4.6}$$

$$\begin{aligned} \hat{c}_t = &\sum_{i+1}^{t-1} \sigma[U_c * h_i(softmax[v^T tanh(U_h h_i + W_x x_t + U_{\tilde{h}} \tilde{h}_{t-1})])] \\ &+ \sum_{k:j} \sigma[U_c^j * h_i(softmax[v^T tanh(U_h^j h_i + W_x^j x_t + U_{\tilde{h}}^j \tilde{h}_{t-1})])] \end{aligned} \tag{4.7}$$

$$
\begin{aligned}
o_t = & \sum_{i+1}^{t-1} \sigma[U_o * h_i(softmax[v^T tanh(U_h h_i + W_x x_t + U_{\tilde{h}} \tilde{h}_{t-1})])] \\
& + \sum_{k:j} \sigma[U_o^j * h_i(softmax[v^T tanh(U_h^j h_i + W_x^j x_t + U_{\tilde{h}}^j \tilde{h}_{t-1})])]
\end{aligned} \tag{4.8}
$$

4.5.2 Existing network model for training NLG component

As the proposed strategy for NLG component has not produced the proper results, the sequence-to-sequence architecture, which has explained in Chapter 3 has used to train the model. The semantic frames with intents and slots-value pairs are the information for the input sequences, and the target sequences has the corresponding sentences. The values of each slots present in the sentences have delexicalized with the upper-case letters of the corresponding entity name. This has performed to imply API results in an appropriate places of the delexicalized entity name. The sequence-to-sequence network model has trained on 6000 samples, which have various intents-entities pairs, associated with the corresponding target sentences.

4.6 Handling of the predicted actions

4.6.1 Rasa custom action server

After training the model, the model file has stored in a path. A separate python script has developed to create the action server to execute the predicted action by the trained model of the custom architecture. For each custom action, which occurs in the dataset, a custom class has created. rasa action server invokes the corresponding predicted action class to generate a response. The input sequences has first transformed into a 3-dimensional one-hot encoded vector for each tokens. Then, the predicted action name associated with the corresponding entities has chosen as input sequence, which has fed to the encoder model. The encoder model produces the state space vector representation for the predicted actions, and entities. This state space vector representation has used by the decoder model to sample each tokens for the predicted target sequences until the stop character has seen in that target sequence. The sampling of tokens has performed based on the maximum log probability.

4.6.2 Handling of API calls

The responses have to include the results, which have obtained from the available database to give the user available information depending on the domain-specific scenario according to the requirements of the user. In restaurant reservation scenario, the system has to provide the queried results from the database as per the requests of the user, such as list of available restaurants, recommend a restaurant, list of restaurants in a specific price range, or else information about the specific restaurant. To handle this, the database of list of restaurants, and their corresponding information is the basic requirement. Though certain information about a restaurant is not available, the APIs have to handle it based on the available information. In real-time applications, the Application Programmable Interface (API) has connected through the server to collect the domain-specific information, and the available information has given to the user. Similarly, the database has hosted on a local server, and the necessary information has collected from it. The current belief state get updated from the present and the past user's utterances to achieve this. For the combined current state of the user's requirements, the API results have collected from the hosted local server. The collected API results have then included in selected candidate responses from the trained NLG model by replacing the delexicalized words. The algorithm to handle API has depicted in Figure 4.15 below

```
Algorithm for rasa action server

A new route to read data from URL
@app.route('/')
def f_1()
    with open ('data_path', 'r') as jsonf:
        data_file <---- load json file(jsonf.read())
    return jsonify(data_file)
-----------------------------------------------------------------
def api_exception()
    URL <---- "Local_Host_Server"
    response <---- requests.get(url = URL)
    if response.status_code == 200:
        msg = "ok"
    else:
        msg = "An appropriate error message for each code"
    return msg
-----------------------------------------------------------------
def state_track()
    dialogue_state <---- rasa_tracker.state()
    entity_i = dialogue_state[entity_i]
    in_int <---- dialogue_state[latest_message_intent]
    slots <---- slots.append(current_slot_values)
    slots_state <---- slots_state.append(entity_i)
    return in_int, slots_state, slots
-----------------------------------------------------------------
Functions for handling API
-----------------------------------------------------------------
```

Figure 4.14: Algorithm to track the current state of the dialogue in Rasa action server

4.6.3 Handling of technical errors within that API

The error handling techniques have to be performed to intimate the users whether there is a server problem from the client-side, or the server-side. Hence, the user gets an idea of the problem of error occurrence. The error handling has performed for the hosted server to deliver a response corresponding to the error status while having a conversation with the system. For a prototype two samples of errors in client-side, and the server-side have considered.

The error status code, which represents 4xx class, indicates that the error has come from the client-side. The server has to send an explanation for error situation to the user, and it has to state whether it is permanent or temporary. The selected error handling techniques for the client error has given below,

- **Error status '400'** Indication of **Bad Request**. It represents that the error has occurred from the client-side, and the server cannot process the request due to the factors, such as invalid request framing of the message, or malformed syntax in the request
- **Error status '404'** Indication of **Not Found**. It represents that the server could not find the target resource for its representation, which also could not intimate any information concerning that. It could not indicate whether this is temporary or permanent on the lack of representations.

The status code of error that belongs to the class 5xx indicates the server error. It represents the indication of whether the server is aware of an error, and it can perform the requested

```
Algorithm to handle API

Available slots in API database ----› area, price range, food,
                        postcode, address, phone number, name
-------------------------------------------------------------------------
def entval_track()
   dialogue_state ‹---- rasa_tracker.state()
   entity_1 ‹---- dialogue_state ['slots'][entity_1']
   .
   .
   entity_n ‹---- dialogue_state['slots'][entity_n']
   return entity_1, .., entity_n
-------------------------------------------------------------------------
def api_extract()
   entity_n ‹---- entval_track ()
   #system requires atleast one slot to check API
   if 'none' in entities (checks for food, price, area)
      area ‹---- random select [east, west, north, south,
                                          centre]
      price ‹---- random select [cheap, moderate, expensive]
   if specific_area has specified
      if specific_food has specified
         if specific_price has specified
            name ‹---- append restaurants name (3 criteria)
         else
            name ‹---- append restaurants name (area, food)
      elif specific_price has specified
         name ‹---- append restaurants name (area, price)
      else
         name ‹---- append restaurants name (area)
   elif specific_food has specified
      if specific_price has specified
         name ‹---- append restaurants name (food, price)
      else
         name ‹---- append restaurants name (food)
   else
      name ‹---- append restaurants name (price)
```

(a) Algorithm to handle API to list available restaurants for the user specific criteria

```
Algorithm to handle API

def api_extract()
   if specific_area has specified
      if specific_price has specified
         food ‹---- append available food types (area, price)
      else
         food ‹---- append available food types (area)
   else
      food ‹---- append available food types (price)
-------------------------------------------------------------------------
if specific_food has specified
      if specific_price has specified
         area ‹---- append available area types (food, price)
      else
         area ‹---- append available area types (food)
   else
      area ‹---- append available area types (price)
-------------------------------------------------------------------------
if specific_area has specified
      if specific_food has specified
         price ‹---- append available price types (area, food)
      else
         price ‹---- append available price types (area)
   else
      price ‹---- append available price types (food)
-------------------------------------------------------------------------
if name has specified
   specific_info ‹---- name [specific_info]
   specific_info are addr, phone, postcode,
   price, area, food
```

(b) Algorithm to handle API to list available food type, area, and price ranges for the user-specific criteria

Figure 4.15: Algorithm to handle API in the responses

method. The server has to send information explaining the error situation, and the response codes apply to any request methods.

- **Error status '500'** Indication of **Internal Server Error**. It indicates that an unexpected condition has encountered, which has prevented it from fulfilling the request
- **Error status '504'** Indication of **Gateway Timeout** It indicates that no timely response has received from an upstream server to complete the request. It happens when the server acts as a gateway or as a proxy.

4.6.4 Inference for selection of a candidate response

The inference for an appropriate response for delivering it to the user has made based on the predicted system action semantic frames. There are a totally 6000 samples of possible for overall intents and its associated entities. For specific intents that have predicted by the system does not involve any entity information. These types of actions include greeting the user or general request to the user for further help. Also, the perceived intents and the associated entities play a crucial role. The related entities relevant to the specific intent that have uttered by the user should also include in the selected responses after integrating the business logic.

At first, the predicted system action and the associated entities have tracked. Then, the

Algorithm for the selection of candidate response during Inference

```
def inference_function
   intent ‹---- obtain intent from rasa tracker
   inform_slots ‹---- list of relevant slots to intent 'inform'
   request_slots ‹---- list of relevant slots to intent 'request'
   book_slots ‹---- list of relevant slots to intent 'book'
   split intents and entities by an appropriate token

   for input, input_texts in all inference inputs
     split nlg intents and entities by an appropriate token
     if fallbackaction in intent
        if fallbackaction in nlg_intent
           if length of system prediction is less than '99'
              inference has chosen for this input condition
     if length of intent and nlg_intent is same
         if all intents are available in inference nlg_intent
            if all entities are available in nlg_entities
               if length of system prediction is less than '99'
                  inference has chosen for this input condition
            elif any entities are available in nlg_entities
               if length of system prediction is less than '99'
                  inference has chosen for this input condition
            elif length of entities is zero and any intent in
            nlg_intent
               if length of system prediction is less than '99'
                  inference has chosen for this input condition
             else continue
          else continue
       else continue
```

Figure 4.16: Algorithm to select a list of possible candidate responses

intents and the corresponding entities have split from each other. The inference input semantic frames have split into intents, and entities separately. If the length of both the intents is the same, then the next condition has processed to check. It checks whether all the entities that have specified by the user related to the particular intent are in the inference inputs. If this condition has satisfied, then the corresponding input has chosen for an inference. If that condition fails, then it checks for any of the user-specified entities are in the inference inputs. If this condition has fulfilled, then the corresponding input has chosen for inference. If the above condition could not be satisfied, then the system checks whether the entities' value corresponds to zero. If it is zero, then any of the systems predicted intent had in the inference inputs, then that input has chosen for inference. If all the conditions have failed, then the loop has continued for all the inference inputs. The maximum number of inference has set to '99.' If this condition has satisfied, then the loop has tripped. The algorithm to implement this has shown in Figure 4.16

There are more possible responses for each system predicted action. Since the source of the dataset is quite big for the NLG component, the likely candidate responses are also more. Hence, the filtering technique has to involve to select the most appropriate ones. The user intent has perceived by the dialogue state tracker, then the slots information specific to the perceived intent have chosen. As the user input context, specific values have de-lexicalized in the NLG component, the obtained entities have capitalized. Then, each selected candidate has

Algorithm for the selection of candidate response during Inference

```
def inference_function
    intent ‹---- obtain intent from rasa tracker
    inform_slots ‹---- list of relevant slots to intent 'inform'
    request_slots ‹---- list of relevant slots to intent 'request'
    book_slots ‹---- list of relevant slots to intent 'book'
    split intents and entities by an appropriate token

    prediction ‹---- decode the sentence for the selected lists

    Slots ‹---- Selection of specific slots relevent to perceived
                intent
    Slots ‹---- Capitalize all the specific slots obtained

    for output in prediction
        if all intent specific slots available in the output
            valid_output1 ‹---- append the selected candidate
        if any intent specific slot available in the output
            valid_output2 ‹---- append the selected candidate
        if length of entities is zero
            valid_output3 ‹---- append all the outputs
        for out1 in valid_output1:
            if 'NAME' in out1
                output1 ‹---- append corresponding candidate
        else continue
    if length of entities is '0' or not '0'
        if length of valid_output1 and valid_output2 is '0'
            valid_output4 ‹----append all the outputs
```

Figure 4.17: Algorithm for filtering the chosen list of candidate responses

checked for the availability of all relevant entities. The priority has given for the candidates that have all associated entities corresponding to the perceived intent. As a fallback option, any of the entities related to the perceived intent present in the candidate has provided. If this both condition fails, then all the selected outputs have approved. The algorithm to handle this situation has shown in Figure 4.17

At last, anyone from the list of filtered candidates has chosen to deliver it. The first condition mentioned above has checked for the availability of 'Name' in the filtered candidate. If it is available, then it has given priority to deliver it to the user. Also, the selected candidate has all the corresponding entities associated with the perceived intent. If an entity 'Name' is not available in the chosen candidate, then the candidates have filtered for the availability of all entities without the entity 'Name,' and an appropriate one has chosen. When all these conditions have failed, then the availability of any one of the entity-specific perceived intent has checked. If the state is true, then it has selected as a response. If all the conditions mentioned above have failed, one from the list of all possible candidates has decided. The algorithm to handle this has shown in Figure 4.18.

```
Algorithm for the selection of candidate response
                during Inference

def inference_function
   intent ‹---- obtain intent from rasa tracker
   inform_slots ‹---- list of relevant slots to intent 'inform'
   request_slots ‹---- list of relevant slots to intent 'request'
   book_slots ‹---- list of relevant slots to intent 'book'
   split intents and entities by an appropriate token

   if length of output1 is not '0'
      list_output ‹---- random selection of filtered candidate
                        (satisfies all conditions,includes
                         'Name' as well)
   elif length of valid_output1 is not '0'
      list_output ‹---- random selection of filtered candidate
                        (satisfies all condition, but either
                         includes 'Name' or does not include)
   elif length of valid_output2 is not '0
      list_output ‹---- random selection of filtered candidate
                        (Misses certain slot information)
   elif length of valid_output3 is not '0
      list_output ‹---- random selection of all possible
                       candidate
                       (either satisfies all conditions or
                        misses certain slot information)
   elif length of valid_output4 is not '0
      list_output ‹---- random selection of all possible
                       candidate
                       (either satisfies all conditions or
                        misses certain slot information)
```

Figure 4.18: Algorithm to select an appropriate candidate response after filtering

4.7 Generation of Responses with business logic Integration

The multiple intents have taken as a single system action as sequence-to-sequence prediction tasks have to comply with the equivalent number of input and target samples. The belief state update has achieved by rasa DST. The entities and the associated values have extracted from the obtained current belief state update. The selected system action name and the extracted current belief state update have combined into a single string. This single string has passed as an input sequence for an inference, which has used by the trained encoder model to predict the state space vector representation. This state-space vector has then used to decode the target sequence by the sampling of each token. The algorithm to create the rasa action server has shown in Figure 4.19

After selecting a candidate response as mentioned in the previous section, the de-lexicalized words in the chosen candidates have replaced. It has achieved from the results that have obtained from the API. The API results have the information, such as list of availability for user specifications, number of available choices, and the user-specific context. These values have replaced in the chosen candidate response. Let's say that in a chosen candidate have the de-lexicalized entities, such as Address, Phone number, Name, choices. The API results have all these relevant information. These results have either replaced by a complete list corresponding to the specific entity value, or else according to the number of repetitions of de-lexicalized

entities.

```
                Algorithm for rasa action server
Rasa custom actions
class action_1(Action):
   def name(self):
      return "action_1"
   def run():
      user_int, in_ent, dialogue_state ‹--- state_track()
      action_1 ‹---- int_1int_2int_3
      intent = ['int_1', ..., 'int_n']
      model_input ‹---- "int_1 int_n"+f"{dialogue_state}"
      process_msg ‹---- api_exception()
      if p_msg is "ok":
         api_info1, .., api_infon ‹---- API_function_1()
         prediction‹--Inference.predict(self, model_input,
                        intent, in_ent, user_int, api_info1)
         current_entity_values ‹---- API function_1()
         if 'None' not in current_entity_values:
            if 'Null' not in api_infos:
               select ‹---- random.choice(list_selection)
               if "API_INFO" in prediction:
                  prediction ‹---- prediction.replace("API_
                  INFO", api_info)
         msg ‹---- rasa_dispatcher.utter_message(str(
                                          prediction))
      else:
         msg ‹---- rasa_dispatcher.utter_message(str(
                                          p_msg))
         return msg
```

Figure 4.19: Algorithm to handle custom actions in Rasa action server

4.8 Comparison of existing approaches to the proposed DM strategy

The Knowledge Graph-based Information State Update approach requires the domain-specific expert to define a set of rules for pre- and postconditions. A dialogue engine has to select a set of actions. Then, it has to execute it by the match between the state update and the pre-defined rules. The domain expert has to create the information tree for all possible dialogue turns between the users and the system. It becomes cumbersome to re-use the model for a different domain. It becomes cumbersome to adapt it even for changes in the sub-domain, for which the domain expert has again re-model the information suitable for the requirements. Whereas, in the corpus-based training, though, the domain expert has not involved, but the availability of the domain-specific human-human conversations is vital to train the model. It requires the effort to annotate the corpus data by humans, which has specified in previous sections. The statistical proposed approach supports not only easy adaptation for multiple domains but also enhance the model by incremental training for domain-specific scenarios. It requires less human effort, and the ICAs can give automated responses from the trained data rather than a standard response. A brief comparison of KGD based ISU approach and the proposed strategy has shown in Table 4.8 below,

Table 4.8: Comparison of KGD-ISU approach and the proposed approach, a)- KGD-ISU approach, b)- Progressive Sequence to sequence learning approach

Technical background	**Advantages**	**Limitations**
a) ISU - A set of rules for pre- and post conditions. KGD - Information graph tree by a domain expert for all possible dialogue semantic frames	A ready-to-go development for static domain-specific applications. Guaranteed fulfillment for expected task specific dialogues	Lacks domain adaptation Lacks Incremental Learning for changes in domain-specific applications Requires human effort to develop and maintain the system
b) Encoder decoder architecture with progressive columns of adapter functions	Fully automatized for dynamic applications Multi-domain adaptation	No guaranteed fulfilment of tasks

Comparison of the proposed approach to Convlab, an open-source framework

The rasa open-source proposed strategy has compared with convlab, an open-source dialog system platform. The convlab is an open-source platform for building multi-domain dialogue systems. It has developed from the research perspective, which has a set of tools for various approaches under the same condition. It offers end-to-end evaluation through human and user simulation. It offers an architecture to develop a centralized agent who can handle multi-domain setting as the joint observations of all domains. It leads to a drawback of exponential growth in the action spaces proportional to the increase in the domain. It also provides multi-task learning by transfer learning approach, such that the multi-agents from different environments can share common knowledge specific to a domain, to adopt for multiple domains. It also has the option of role play, where a system itself can act as a user and an agent in a Reinforcement learning setting. Its NLU component can handle Out-Of-Vocabulary words and multiple intents in multiple domains. The DST component has developed using a hand-crafted rule to maintain the belief state. The word-level dialogue policy uses the sequence to sequence learning approach with dialogue state encoding mechanism, and querying a database for encoding additional features to the decoder. They have utilized RL algorithms for role-playing based system dialogue policy setting. For NLG, they have employed the Semantically Controlled-LSTM, and a template-based approach [**14**].

Whereas, in the proposed approach, it has attempted to satisfy the research focus and also the business-centric conditions, where user satisfaction plays an important role. In the proposed approach, an attempt has made to build domain-specific ICAs, which also have the flexibility to adapt to multi-domain through transfer learning. It also offers incremental learning for modifications/ and or additions in domain-specific conversations by re-training the model from the pre-trained model of previously specified corpus data. It has achieved by using the proposed architecture for policy learning. The NLU uses the Embedding Intent classifier, which has built based on the Star-space algorithm to handle multiple intents, and it also handles the Out-Of-Vocabulary dialogues. Similar to convlab, rasa open source also uses hand-crafted rules for DST. The Sequence-to-sequence architecture or template-based approach has used for NLG. The comparison between these approaches has shown in Table 4.9 below,

Comparison of Rasa REDP Policy and proposed architecture

It requires much effort either by humans or a user-simulator to train the model using a reward function for every system actions, which involves the expertise in the particular domain. The

Table 4.9: Comparison of convlab framework and Rasa with the proposed strategy, a)- convlab architecture, and b)- Progressive Sequence to sequence learning approach

Technical background	**Advantages**	**Limitations**
a) Multi-domain learning with joint observations for centralized agent Multi-task learning Role-play learning	end-to-end learning approach less human effort role-play learning Evaluation using user simulator or humans	Lowest success rate for end-to-end approach
b) Encoder decoder architecture with progressive columns of adapter functions	Multi-domain adaptation Incremental learning	No guaranteed fulfilment for single agent in multi-domain No guaranteed fulfilment No user simulators

RL based approaches require huge effort to process until much possible observation of user utterances and system actions with reward functions has executed to train the model to apply in a production setting. Hence, the supervised learning approach has chosen as an optimal choice for the proposed strategy and in the development of the REDP policy of Rasa, which can extend it for RL in the future. The domain adaptation is one of the main interests of the research focus for NLU, DST, DM, and also NLG. But both the policies have focused the part of domain adaptation for policy learning.

In REDP policy, the system actions have chosen based on a ranking approach rather than using classification. The ranking of each candidate has performed between the trained embeddings of the dialogue state and the system action. It has achieved by using a similarity function. The system actions have represented as bag-of-features, and the user utterances have represented as intents. A recurrent network has employed to get the representation of the dialogue state. The learned dialogue behavior from one domain has adapted to another. The semantic frames from the user utterances and the bag-of-features of system actions have converted to feature vectors, from which the embeddings have created. An attention mechanism has used for a user utterance and a previous predicted system action. An interpolation gate has employed to the truncated memory of the current time step to apply an attention mechanism. The concatenation of summation of the embedded user input and the system attention vector has given to an input RNN, and the output has fed again into an embedding layer to get an output. At last, the similarity between the dialogue level embedding and the system attention embedding has calculated by the negative sampling of incorrect target labels of system actions [**15**].

Whereas, in the proposed architecture, a many-to-many sequence prediction approach has employed to get the system actions. Each input and target sequence has converted into a 3-dimensional one-hot vector for each token of non-repetitive words, which has occurred in all input and target sequences. Then, for each input sequence, the state vector representation has passed to decoder from encoder. While decoding, the target labels have scored based on the maximum log probability of the target tokens, conditioned on the input tokens. This approach can sample the target sequences irrespective of the variable length in the input sequence and target sequence. For Dialogue Management, no attention mechanism has employed to pay attention to past predicted system actions in the proposed strategy. The proposed approach cannot predict multiple system actions, but multiple intents have predicted as a single system

action.

The comparison between both the approaches have specified in Table 4.10 below,

Table 4.10: Comparison of REDP policy learning and the proposed strategy, a) REDP policy of Rasa, and b)Progressive sequence to sequence learning approach

Technical background	**Advantages**	**Limitations**
a) Domain adaptation Ranking approach RNN cell and cosine similarity to predict target label	selection of multiple actions Handles uncooperative behavior Attention Mechanism for previous predictions	No guaranteed fulfillment No continual learning
b) Encoder decoder architecture with progressive columns of adapter functions	Multi-domain adaptation Multi-task learning Incremental learning Multiple intents prediction as a single system action	No joint observation for single-agent in multi-domain No guaranteed fulfilment

4.8.1 The challenges that support the proposed strategy

The major challenges that incite the proposed approaches are as follows,

- The Catastrophic Forgetting of LSTM or RNN is one of the major drawbacks in multi-domain adaptation learning approaches for developing dialogue systems. In incremental learning, every deep learning algorithms are prone to CF, such that all the previously learned information has diminished due to the learning of new representations
- The application of incremental learning/ continual learning for domain-specific settings without forgetting the previously learned data. The transfer learning for sharing the common knowledge between various domains to address the multi-domain adaptation problem.
- This approach reduces the human effort strictly to the creation of an appropriate domain-specific corpus data, such that the heuristic rules for learning the policies for taking system actions, and the responses have neglected. It reduces the cost of investing time and monetary values for business cases, in creating the domain expert knowledge-based information tree, which is static and suffer from the flexibility
- As the proposed strategy employ the prediction of multiple intents as a single system action, there is no requirement for similarity matching functions to predict the system actions of multiple intents

4.9 Architecture overview of developed ICAs

4.9.1 Design technique

The fully automatized ICA has not assisted with any pre-determined logic or rules. The system can decide in all dialogue turns, and it has designed to fulfill the end-goal of the domain-specific task. An overview of the design technique has described below, and the architecture has shown in Figure 4.21 and Figure 5.1.

- **NLU** Rasa NLU has used to classify the intents and the associated entities from the annotated corpus data that has employed to train the system. The supervised learning method has used, and hence it requires a lot of annotated data to learn effectively and perceive the intents and the associated entities. It uses a rasa dialogue state tracker to track the dialogue for the entire conversation. It means that the system can remember the entity values that have specified by the user for the entire conversation. The user has to update the entity in single or multiple dialogues while interacting with the system.

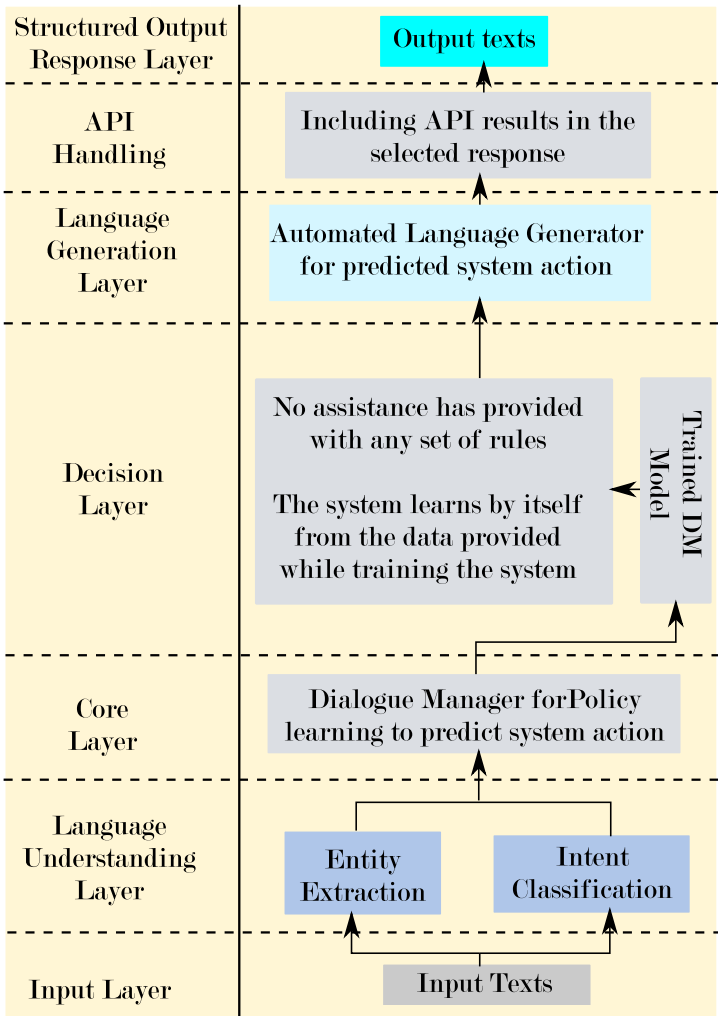


Figure 4.20: Architecture overview of an automatized ICA. **It has not provided any assistance, and the system can make its decision**

- **DM** Rasa REDP policy and the custom-developed deep learning strategy have employed to predict the system action. The responses from both strategies have depicted in Chapter 5. Each dialogue policy has trained using possible dialogue stories, as mentioned in the section of the Preparation of dataset. The policies should learn from the dialogue stories, and the trained model has stored for the inference. A filtering technique has used for selecting an appropriate input inference for the detected user's semantic frames.The system decides automatically concerning the system action for the perceived user's utterance. The policies have developed based on machine learning and deep learning algorithms to predict the system action. A action server has used to handle the predicted actions, where the integration of business logics have performed. The semi-automatized ICA has additional

decision layer, where the system has assisted when it has predicted a wrong system action in a dialogue turn.

- **NLG** A machine algorithm has employed to automate the responses for the predicted system action. The NLG component has trained separately from the rasa framework, and the trained model has used in the rasa action server to deliver a response after integration of the business logics. It involves filtering technique to choose one valid candidate from the list of possible candidates.
- **Business Logic Integration** The responses must have the information from API results to deliver a contextual response for the user specifications. The user satisfaction depends on this factor as well. Hence, to deliver a quality product the business logics have included in the automated responses, such as list of available options for the user specifications, quantity of available options, information specific to the available options. When there are no relevant information for the user specifications, it has also handled.

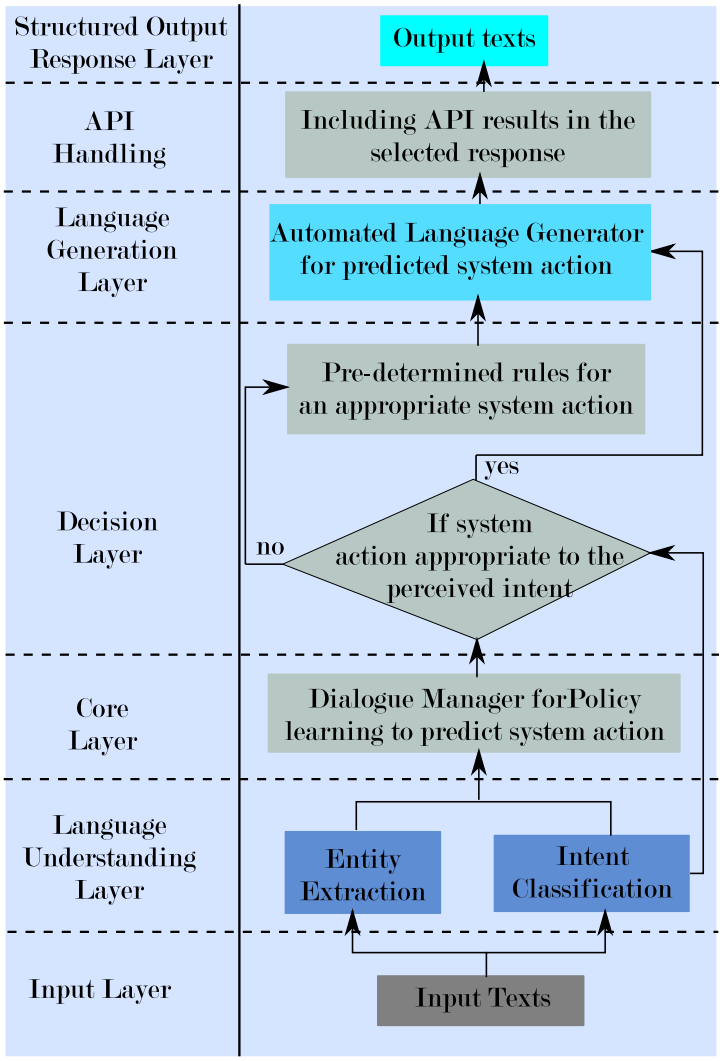


Figure 4.21: Architecture overview of a semi-automatized ICA. **It has assisted when a system has predicted the wrong action for the perceived user intent**

Chapter 5

Evaluation Results and Discussion

5.1 Evaluation method for live-user evaluation setting

The User experience is a significant aspect, which has used by Human-Computer Interaction (HCI) for the task- and work-related usability tests. The term User experience has close relation with usability, beauty, hedonic, affective, and experimental aspects of the use of technology. Though there are exclusive research is available for the behavior of the HCI in work settings, the instrumental value of the interactive products has not ensured by any research work. Beauty has considered as one of the aspects which satisfies the general human need, and hence it has considered as one of the aesthetic quality in evaluating the interactive products. The design of future HCI has to satisfy the behavioral goals of the developed product, which has to increase its knowledge and skills, and it must have the capability to express itself with a relevant interaction to the user. These multidimensional qualities express the needs and values of a product. To have a good user experience, the considerations of various quality aspects to develop a product are the hierarchy of their requirements, behavioral goals, the capability to increase the skills of the developed product, prioritization of context-driven interactions, and the dynamic quality [**21**].

Also, developed products have to pay attention to the emotions of the user. HCI systems have to perceive the user intents, adapt themselves to the user intents, and then it should give a corresponding response. Though it has perceived that the interaction between humans and the computer involves frequent negative emotions, it has become a great deal for HCI systems to aid the users with negative emotions towards automated systems. Affective computing plays a prominent role in call center software, where a motivational user interface is significant to reduce the negative impact. Thus emotions and affect in user experience have taken either as a consequence of product use or as antecedents of product use. The problems of affective computing have to be addressed by the developed interactive product since it has significance while dealing with the customers to satisfy them by politely aiding them to shift the negative emotions of the user positively. Also, it is important to address the problem of the influence of relative aspects, such as the influence of beauty or usability on the emotions and affect of the user, on the developed product[**21**].

The User experience has associated with the product according to their states, such as mood, expectations, and active goals. It extends over time, and all the elements within these aspects are inter-related. The experiential outcomes have affective outcomes, which help to transform and regulate it. Experience can either be articulated or named, which has a beginning and an end. The articulated experience represents experiential as complex, unique, and hard to repeat. Whereas, the named experience represents experiential as temporal aspects of experiences, their corresponding subjectivity, and the associated dynamics. It is hypothetical to develop a product

with considering all relevant elements to attain the positive experience or else considering the experiential aspects to design a product, for which there is no guarantee for particular experience [**21**].

The joint collection of various aspects and their influence on User experience has shown in Figure 5.1,

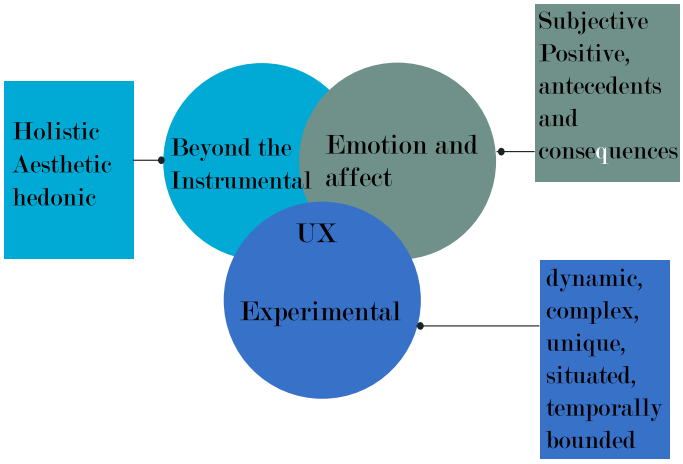


Figure 5.1: Influence of various aspects on User experience [**21**]

5.2 Results for Recurrent Embedded Learning Policy of Rasa

The training data for DM has modified with the different combinations of the distributions of data, and the corresponding results have analyzed. The REDP policy has trained with 100 epochs, and the default settings have maintained to train the model. The results which have obtained from the first version of training data has depicted in Figure 5.2 below. The system could greet back the user properly at the beginning and the end, but the dialogue turns, which has taken by the system, is inappropriate in certain scenarios. The unfulfilled dialogue turns happened in this sample when the user has asked the system to make a reservation, but rather the system has predicted to take action to provide the information about the restaurant, such as postal code or address. In the sample depicted in Figure 5.2, the system could not attain the end-goal of the user to make a reservation, but it has handled the errors by asking the user to make a request again. The trained NLG unit has predicted the candidate response for the chosen system action properly, but the wrong predictions mentioned above have occurred in the Dialogue Management unit. There are few errors in the NLG training data itself, such as improper sentence structure, and wrong system actions (inputs) associated with a list of candidate response (target). In version 2 of the training data for the DM component, all the user inputs and the system actions have approximately balanced. Additionally, multiple combinations of a user intent 'informtobook' and 'restaurantrequest' have included, which have followed by the either one from a list of corresponding system action ['informbookingbook,' 'bookingbook,' 'bookingnobook,' 'reqmorebookingbook,' 'bookingreqmoreinform'] and ['reqmoreinform,' 'inform'] respectively. The various combinations of user intent 'restaurantinform' has followed by either of one system action from the list ['request', 'requestinform', 'select', 'selectinform', 'recommend', 'recommendreqmore', 'recommendinform', 'inform'].Also, in this version of training data, for all the combination of user intent 'restaurantinform,' no related corresponding system action, 'nooffer' has occurred. A new intent 'noavailrestaurant' has created, for which

Hello
Hello, Good Morning. Welcome to Cambridge Restaurant reservation System
I am looking for european restaurants
no, there are no european restaurants in ['centre', 'north', 'west']. would you like to check another area ['centre', 'north', 'west'].
Please look in the centre part of town
I am sorry for the inconvenience. I could not get your request. could you request once more for the information?
Please look for european restaurants in the centre of town
There are 16 european restaurants, would you like me to book one for you?
Please reserve a table at 15:00 on monday
,16 are located in the centre part of town. would you like to book a reservation at one of these?
Yes, please make a reservation at one of these
I am sorry for the inconvenience. I could not get your request. could you request once more for the information?
Please reserve a table for 5 people at 14:00 on saturday
the post code is ['cb21rh', 'cb21qa', 'cb21su', 'cb21uw'], and is on the centre
Please give me their address and phone number
the post code is ['cb21rh', 'cb21qa', 'cb21su', 'cb21uw'], and is on the centre
Thank you, goodbye
Thank you for using our services. We welcome you. Have a great time!

Figure 5.2: A sample of results for the REDP policy of rasa, which have obtained using the first version of training data. **It includes wrong predictions in the system actions, and the unfulfilled goal of the user**

the related system action intent 'nooffer' has given. It is useful when making API calls before predicting the system actions.

A sample of results, which have obtained from the trained model of version 2 dataset mentioned above has depicted in Figure 5.5. All the system actions have predicted before making API calls to look for requested user-specific information. The API results have handled after selecting a candidate response from the NLG unit, in which the de-lexicalized words have replaced by the API results while responding to the user. The system is capable of predicting the correct system action for greetings at the beginning and the end. Though the system has predicted the system actions properly, it takes multiple dialogue turns. In the sample result, the system repeatedly asks for the specified slot value, which the user has mentioned. Also, the system could not fulfill the user requests, especially when the user has asked to provide the information about the chosen restaurant, or restaurants of specific criteria. Even for this version of the dataset, the wrong predictions have happened from the Dialogue Management Unit, where the system action 'reqmoreinform' has predicted for user input intent 'restaurantrequest.' Also, REDP policy has provided with attention mechanism for the previously taken system actions, such that an appropriate system action has given the highest score based on the current user input. It helps the REDP policy to provide end greetings properly to the user irrespective of wrong dialogue turns in the course of dialogue conversation to attain a goal. Though rasa open source supports to take multiple actions in sequential order, in all the versions of the dataset, the multiple system actions have combined. As the NLG unit has trained based on a dataset, where the multiple intents have associated together for a corresponding single sentence, the multiple system actions for a user intent have combined. Also, rasa open source has reconfigured to take one multi-intent system action at a time for user input of single or multi-intent and associated slot, and its value.

Version 3 of training data has modified, such that the data combination for the user input 'restaurantinform' dominates other user inputs, such as 'restaurantrequest,' 'informtobook,' and 'byetothesystem.' For this training data, the results which have obtained has depicted in

Figure 5.3: A sample result for REDP policy of rasa using the second version of training data, **which has wrong predictions in the system actions, but the fulfillment has attained**

Figure 5.4, which represents a happy path in attaining the end goal of the user. Figure 5.6 shows the happy path in attaining the end-goal of the user. The system is capable of providing the user-specific information, and it takes good Dialogue turns. It has also perceived the user intents properly for each input, and an appropriate output has predicted. The system also gives the user when information about the restaurants for specific criteria has asked. Also, the system has provided proper greetings at the beginning and the end. But there is false information, such as the day and the time of booking. The user-specified criteria have not perceived properly by the tracker of rasa, which has led to this false information.

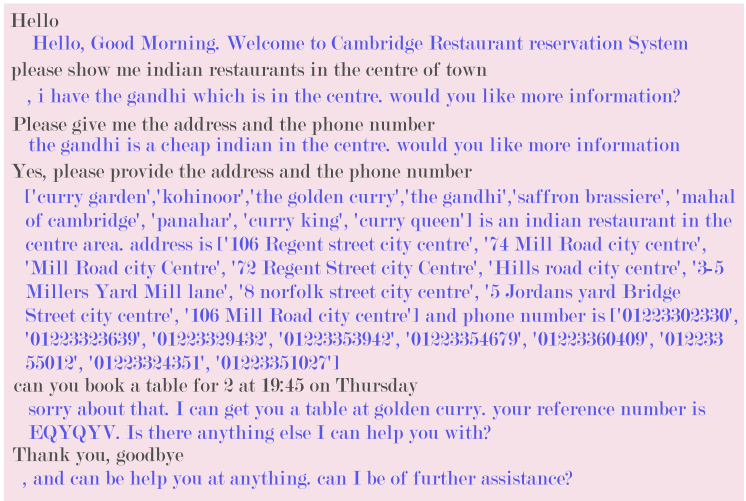
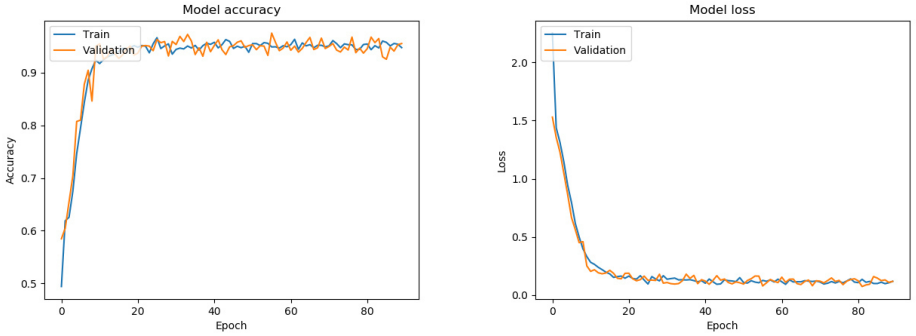


Figure 5.4: A sample of result for REDP policy of rasa using the third version of training data, in which the fulfillment has attained

5.3 Results for Keras Policy of Rasa

The default Keras policy of rasa uses a simple LSTM cell, and the default hyper-parameter settings have used to train the model. The data distribution is uneven in the first version of the training data, which has used to train the model. The user input intent 'restaurantinform' dominates all other intents, and also, the system action 'inform' and 'reqmoreinform' dominates all other system action intents. The accuracy and the loss of the trained model have depicted in Figure 5.4, and the sample results have depicted in 5.6 and 5.7. In Figure 5.6, the system action



(a) The Model accuracy of default keras policy-LSTM

(b) The Model loss of default Keras policy-LSTM

Figure 5.5: The details of the trained model of default Keras policy

has provided with the proper greetings, in the beginning, and in the end. The system has also taken good Dialogue turns at the beginning of a conversation. When the user has requested for 'Scandinavian' restaurants, which is not available in the database, the system has requested to select any other different option, which is available. The system has given the false prediction again, that there are no Japanese restaurants even though it is available in the API database. Then, again the system has given an appropriate response for the user requested information, by providing the list of 'Indian' restaurants with their address.

In Figure 5.7, the action taken by the system has an appropriate greeting, at the beginning, and the end of the conversation. Also, the system has taken good dialogue turns from the beginning and until the end. When the user has asked for the restaurants, which serve 'Jamaican' food, the system has said that it has no available restaurants, and it has asked the user to fill the specific slot. The user has changed the preference to 'North American' food with the specified location and the price range; the system is capable of giving appropriate information to the user. The APIs have handled only after predicting the system action and selection of a candidate response. The de-lexicalized words in the chosen candidate response have replaced by an appropriate value, which has obtained from an API.

5.4 Results for the proposed Dialogue Management strategy

The developed custom architecture provides both happy and unhappy paths for all the versions of the training data for DM units. The developed custom architecture has trained based on the default hyper-parameter settings. The proposed architecture is capable of learning faster by learning in fewer epochs depending on the size of the dataset, and the batch size provided for training. The model accuracy and the model loss, which has obtained from Progressive learning of encoder-decoder columns have depicted in Figure 5.8. The results which have obtained from training the first version of the dataset has depicted in Figure **??**, and Figure 5.10. The API calls have handled only after predicting the system actions, while using the trained model of

Hello
Hello, Good Morning. Welcome to Cambridge Restaurant reservation System
I am looking for european restaurants
no, there are no european restaurants in ['centre', 'north', 'west']. would you like to check another area ['centre', 'north', 'west'].
Please look in the centre part of town
I am sorry for the inconvenience. I could not get your request. could you request once more for the information?
Please look for european restaurants in the centre of town
There are 16 european restaurants, would you like me to book one for you?
Please reserve a table at 15:00 on monday
,16 are located in the centre part of town. would you like to book a reservation at one of these?
Yes, please make a reservation at one of these
I am sorry for the inconvenience. I could not get your request. could you request once more for the information?
Please reserve a table for 5 people at 14:00 on saturday
the post code is ['cb21rh', 'cb21qa', 'cb21su', 'cb21uw'], and is on the centre
Please give me their address and phone number
the post code is ['cb21rh', 'cb21qa', 'cb21su', 'cb21uw'], and is on the centre
Thank you, goodbye
Thank you for using our services. We welcome you. Have a great time!

Figure 5.6: A sample of result for default Keras policy of rasa using the first version of training data. **It represents a happy path with good dialogue turns**

first version of the dataset.

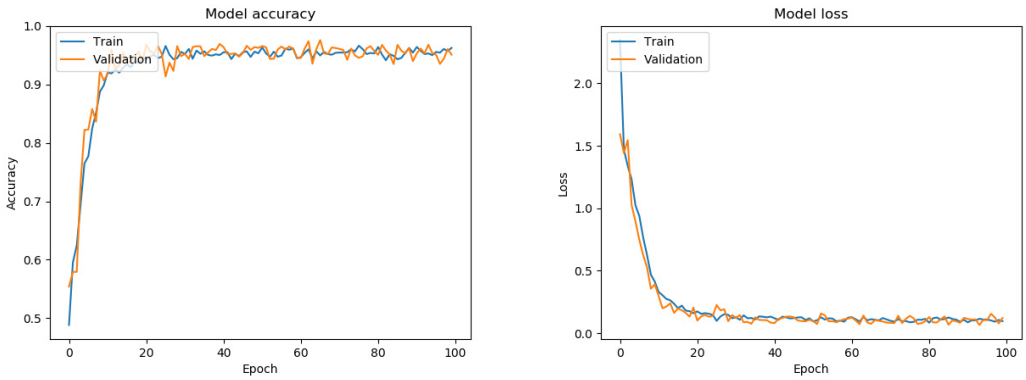
In Figure **??**, the system has greeted the user in the beginning properly, whereas it could not greet the user properly at the end. The system has took the good dialogue turns, by informing the user with the list of restaurants for the user specified criteria. The system could not provide the information of the restaurants for the user specific criteria, but rather it took a wrong dialogue turn to make a reservation. Thus, the system has took more wrong dialogue turns, while predicting the system actions.

In Figure 5.10, the unhappy path of the system actions with good and bad dialogue turns have depicted.The system could deliver a proper greetings at the beginning, but the system could not provide proper end greetings to the user. Though the system is not capable to provide the user-specific information for a single slot value specified, an appropriate response has given at the third dialogue turn for multiple slot values specified by the user without changing the first mentioned slot value. The system is also capable to provide information about the restaurant, the chosen candidate response has not all the de-lexicalized word requested by the user. For example, the user has requested for address and the phone number. In the chosen candidate response, there is no de-lexicalized word available to replace the address information, but the information about phone number has provided.

In second version of training data, the data distribution has balanced for each user input, and the corresponding system action. The system actions have reduced when compared to the first version of the data. Though the data distribution has balanced, it leads to the unhappy path in the predictions. For the trained model of this dataset, the API has handled before predicting the system action. If an user input has detected as 'restaurantinform' requesting for certain food type, or specific price range or area, then it has checked whether there are available restaurants for the user-specified criteria. If there are no information available, then the user intent 'noavailrestaurant' has taken for inference. In the training data, the suitable system action related to 'nooffer' has given for the conditioned input 'noavailrestaurant'. Irrespective

Hello
Hello, Good evening. Welcome to Cambridge Restaurant reservation System
please look for jamaican restaurants
there are 0 jamaican restaurants in the price range. do you prefer a certain part of town
Please look for north american restaurants in centre part of town, and it should be in cheap range
, i have 0. 0 are located in the centre and 0 is located in the centre. do you have a preference?
Please look for restaurants in an expensive price range
you looking for gourmet burger kitchen. it s a north american. would you like a reservation?
please give me the address.
address for gourmet burger kitchen is Regent Street City Centre. Would you like any additional information?
Please book a table for 4 people on saturday
you re booked. your reference number is B7DFJ3
Thank you, goodbye
are welcome. is there anything else i can assist you with today?

Figure 5.7: A sample result for LSTM policy using the first version of training data, **which has wrong actions for certain dialogues**



(a) The Model accuracy of Progressive Encoder-Decoder for final column

(b) The Model loss of Progressive Encoder-Decoder for final column

Figure 5.8: The details of trained model of custom architecture

of these taken measures, the system is incapable to take good dialogue turns to reach the end goal, and the system could not greet the user at the end.

In the third version of the dataset as mentioned above, the system is capable to take good dialogue turns, but the system could not provide end greetings to the user. There are exists certain unhappy paths, where the system provide wrong information instead of the user requested information. The results which have obtained from this dataset have depicted in Figure 5.11.

In Figure 5.11 depicted below, it has analysed that the system predicted actions took good dialogue turns for certain user dialogues, but it also took wrong dialogue turns. Though the system could request the user to change their specifications, as there are no availabilities for their criteria, the system could not reach the end-goal of making a reservation. Also, the end greetings cannot be provided to the user.

Hi
Hello, Good afernoon. Welcome to Cambridge Restaurant reservation System
please show me a cheap chinese restaurant
restaurant ['charlie chan', 'rice house', 'the lucky star', 'golden house'], is in the ['centre', 'south'] part of town, and is in the cheap price range. would you like their location?
Please give me their address and phone number
I booked a table for 2 at 19:55 on monday.at ['charlie chan', 'rice house', 'the lucky star', 'golden house'], an cheap chinese restaurant in the ['centre', 'south'] area of town
Thank you, that is all i need. goodbye
like booking was unsuccessful, would you like me to try another restaurant?

Figure 5.9: A sample result for Progressive encoder-decoder network using first version of training data. **It has wrong predictions in the system actions, and no end greeting has uttered**

Hello
Hello, Good afternoon. Welcome to Cambridge Restaurant reservation System
can you please look for cheap european restaurants in centre of town?
is ['Null']. It s cheap. would you like a reservation?
Please give me a list of restaurants which serve european food
, cheap, there is ['Null'] , european restaurant in the centre. is there an cheap i type that you prefer?
Please look for modern european restaurants in any part of town
Sorry. ['hotel du vin and bistro ', 'galleria'] is also located in the centre.
Please give me address and phone number of restaurant galleria
is ['hotel du vin and bistro ', 'galleria'] in the moderate price range.
ok, what is the address and the phone number of restaurant galleria
serves european food and is an moderate european restaurant. it s phone number is ['01223327330', '01223362054']
thank you for your services, goodbye
i booked you into the ['hotel du vin and bistro ', 'galleria'] in the moderate price range. your reference number is CDU00O

Figure 5.10: A sample of result for custom developed Progressive encoder-decoder network using first version of training data. **It includes good dialogue turns, and no end greetings have provided**

5.5 Discussion on results for the proposed strategy

From the analysis of the results which have obtained, it has noted that the data distribution in the dataset has an impact in selecting a system action. The changes in the prediction of system action has seen , when using the REDP policy of rasa for various versions of the training dataset for DM unit. When there are balanced data distribution, REDP policy takes more dialogue turns in attaining the end-goal of the user. Also, the sequence of dialogue turns in the each conversation, which has provided in the dataset has influence on taking the dialogue turns in test environment. The system actions have predicted based on a ranking approach rather than a classification based approach. As REDP policy has provided with attention mechanism for previously taken actions, the system action for the current input has optimized, and the dialogue flow has attained with the proper greetings.

The default keras policy of rasa has shown to provide the responses with both good and wrong dialogue turns for the current user specified inputs. As this architecture has no attention mech-

Hello
Hello, Good Afternoon. Welcome to Cambridge Restaurant reservation System
please show me some indian restaurants
, royal spice is a cheap indian restaurant in the north. would you like me to make a reservation for you?
No, I am looking for restaurants in the south
have 4 indian restaurants. would you like me to book you a table at one of them?
Yes, please book a table for 6 people on saturday
I booked a table for 6 at 19:33 on saturday. your reference number is 400654
thank you, goodbye
would be your try. taj tandoori serves indian and the address for which time. would you like me to book that for you?
no, thanks. that is it. good bye. See you again
I am sorry for the inconvenience. I could not get your request. could you request once more for the information?
goodbye
i booked a table for 6 at 19:33 on saturday. your reference number is 40B654

Figure 5.11: A sample of result for custom developed Progressive encoder-decoder network using third version of training data. **It includes good dialogue turns, and resulted in unhappy path without end greetings**

anism on previously taken system actions, it is prone to give wrong dialogue turns as the model has trained on various conversation of different dialogue turns with different dialogue length.

Whereas, the proposed strategy has implemented with an idea of predicting each system action conditioned on each inputs. The system actions have predicted solely conditioned on the provided input sequence. Irrespective of many combinations of the user intents and associated entity value pairs with the same target outputs, the trained model is capable to predict a suitable system action with exceptions. As the prediction of targets solely depends on the state vector representation of each conditioned inputs in the training data, and the inputs in the data has multiple repetitions of associated slot values and repetitive target sequences, the model cannot generalize well for each targets to its conditioned inputs irrespective of an optimization technique. The proposed strategy has an advantage of learning incrementally either for datasets of same domains, or it has the capability to use transfer learning approach for multi-domain scenarios.

In the beginning of implementing the proposed work, rasa has no available options of creating natural responses. In the proposed work, the NLG unit has trained separately and integrated into rasa open source, to select an appropriate response from the list of candidate responses for the predicted system actions. Hence in the proposed work, the ICA has fully automatised instead of relying on templates or standard responses.

5.6 Evaluation on Dialogue Statistics

The dialogues which have involved in training the model have evaluated based on the performance measures, such as confusion matrix, Precision, Recall, and F1-Score. Due to technical limitations, the dialogues that have involved only in the NLU component have measured for its statistics. The confusion matrix has shown in Figure 5.12, which shows the matrix of correctly predicted intents, and the 'True Negative' and 'False Positive' conditions of the corresponding intent. The higher the intensity of the region, the higher the number of samples and the correct

predicts. The incorrect predictions have surrounded by light shaded regions nearby the one-to-one correspondence of the matrix element. In the matrix of one-to-one correspondence for the label 'outofscope' from both the axes of true and predicted labels, '457' represents the correctly predicted samples, and the nearby matrix elements of '36' and '11' in the row of 'outofscope' along x-axis shows the 'False Negative condition,' whereas the nearby matrix elements of '21' and '13' in the column of 'outofscope' along y-axis represents the 'False Positive' condition. The intensity of the shade has shown low for the less number of samples in 'False Positive' or 'False Negative' condition. The confidence of each perceived intent has shown as histograms with the

Figure 5.12: The confusion Matrix for the perceived intents by Rasa NLU

number of hits and misses in Figure 5.13. The confidence score shows the correctness in the prediction of perceived intents. The perception of input intents should have a high confidence score, such that the system perceives the intent of the user appropriately. The hits represent the number of correctly perceived intents with the corresponding confidence score, and misses represent the incorrect prediction of intents along with their respective confidence.

From Figure 5.13, it has analyzed that the system has also perceived wrong intents with a high confidence score, which might influence the system from predicting an appropriate system action and delivering an appropriate response. The certain intents, which has predicted correct, has lesser confidence score. Also, the system has predicted the intents correctly with a lesser confidence score, and it has also missed perceiving the intents for some samples. Most of the samples have the highest confidence score greater than '0.95', which means that the system can clearly discriminate each intent, and select the one appropriately. The graph has depicted for the samples with the confidence score each within the difference of '0.05.' The samples with the less confidence score in each range less than '0.95' are below '100.' The total number of samples with the misses is in between the range of '100' and '350' samples. The total number of samples that have involved with six different intents are '5762.'

The Quantitative measure of Precision, Recall, and F1-Score has shown in Table 5.1. Each

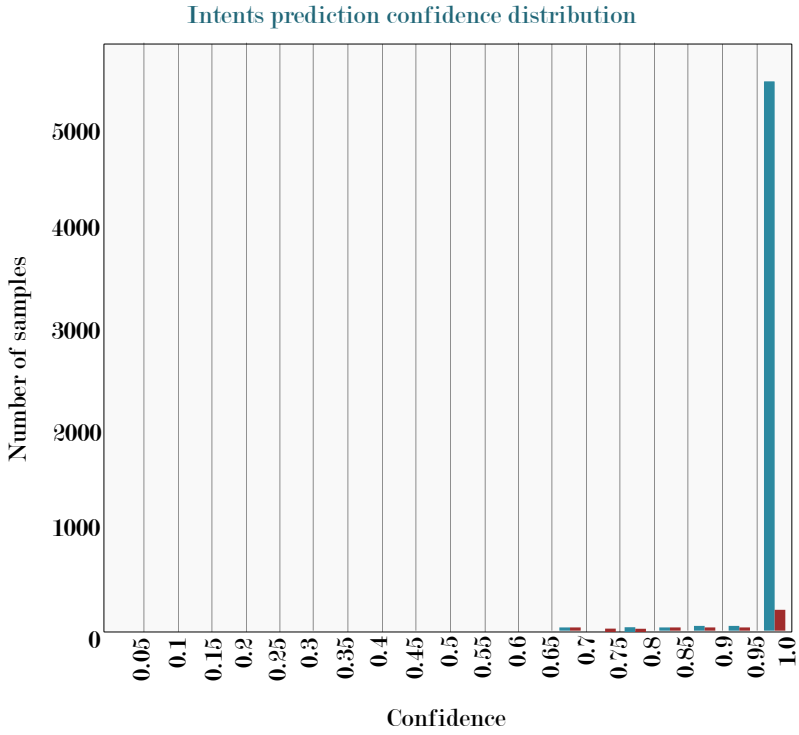


Figure 5.13: Histogram of intents prediction confidence distribution by hits and misses

intent has obtained the Precision, Recall , and F1-Score greater than '95%', except for an intent 'outofscope.' It has obtained only '92.9%', '90.5%' and '91.7%' of Precision, Recall, and F1-Score respectively. As mentioned chapter 2, these quantitative measures have determined from the conditions, such as 'True Positive,''True Negative,''False Positive,''False Negative.'

Table 5.1: Performance Measure on Intent Prediction

Intent	Precision	Recall	F1-Score	Samples
generalgreet	0.9545	1.0	0.9767	21
restaurantinform	0.9835	0.9884	0.9860	2607
restaurantrequest	0.978	0.9819	0.9799	996
outofscope	0.9288	0.9049	0.9167	505
informtobook	0.9880	0.9803	0.9841	254
byetothesystem	1.0	1.0	1.0	1379

5.7 Live user evaluation

The fully automatized ICA and a semi-automatized ICA have employed for the testing environment. In the semi-automatized ICA, the system has assisted whenever a system has predicted the wrong system action for the corresponding user input. Hence, the system has to deliver an appropriate response to the user along with the API results in responses.The assessment of the products have achieved by asking each user to fill out the Questionnaires, which have mentioned above. One measures the attractiveness of the system by measuring the pragmatic, and hedonic qualities of Identification, and stimulation. The other measures the Overall reaction,

effectiveness in performing the tasks, capabilities of the system, learnability, display of results, and visuals from the system, along with the real-time feedback. The fully automatized ICA has mentioned as product1, and semi-automatized ICA has mentioned as product2 in forthcoming sections.

In order to get a balanced experience rating, equivalent professional, and naive participants have asked to evaluate both ICAs. The professional participants have experience in building ICAs, whereas the naive participants does not have any experience in building ICAs. The naive participants are the general users. In total 15 participants have asked to evaluate the ICAs. One user has not shared the experience, and hence it has neglected. From other 14 participants, one user could not participate in the evaluation of product1, and one user could not participate in the evaluation of product2. Thus, it has considered as 14 candidates have participated in the survey, out of which '7' are professional, and '7' are naive candidates. In '7' professional candidates, '2' female, and '4' male have included. In '7' naive participants, '1' female, and '6' male have included. Due to lack of female candidates, more male candidates have selected. Also, only one participant is in the age range greater than 40 out of '14' participants, who comes into the professional category. The other all participants are in the age range 20 and 40. The ***product1*** has evaluated by '6' professional, and '7' naive participants in age range 20 and 40, whereas ***product2*** has evaluated by '7' professional, and '6' naive participants in the age range 20 and 40, and greater than 40. Totally, 13 participants have evaluated each product.

5.7.1 Survey Procedure

Each user has given a general introduction about the ICAs. Each user has attained the information about the domain of the ICAs for which the system has built, and then explanation about each intent, and their associated contextual information has provided, which the system can handle. The clarification has given that each user may handle the dialogues according to their way, and it has recommended to provide at least any one of the contextual information associated with each intent at the beginning. The assistance has provided by the developer of the product, to clarify the reaction of the system at each dialogue turn in a conversation. An information has provided the possible ways of reaction of the system to each intent, such that the user have an idea about the product. At the beginning of evaluation, no too much specifications of each product has provided, which might influence the rating of the product. It has requested to hold the conversation for five times with each product by each user, to have a clear insight about the experience with the system. After evaluating each product, the user has filled two standard questionnaire, which has mentioned above. The instructions has provided to the evaluator by the standard web-based assessment for assessing each product, and it has explained that it is also applicable for the other Questionnaire as well.

When each user had an interaction with each product, the guidance has provided by the developer to each user regarding the recognition of the user intents, and associated specific contexts, prediction of system action, selection of an appropriate candidate response, and delivering the response with the results from API. Also, a general insight about the system has given to each user while interacting with each product. Whenever the system cannot handle the dialogues, an assistance has provided with a reason, and a suggestion has given to the user. The user questions have also answered while holding the conversation about the behaviour of the system for each dialogues. It has requested from each user to hold five conversations or long conversation with multiple dialogue turns with each product, but it was not mandatory. When the user has felt that experience was enough to rate the product without holding five conversations, it has stopped. Then, the rating of the system has been followed. It took 14 days for all participants to interact, and evaluate the system.

5.7.2 Results of live user evaluation - survey 1

The survey 1 has accomplished by the web-based survey using AttraktDiff Questionnaire, which has mentioned in Chapter 3. The user has to rate based on the adjective word pairs according to their experience. The mean value of all the user's rating has obtained, and it has provided as a graph to determine the overall attractiveness of the system. The graphical result for each adjective word pairs has represented in Figure 5.14. The mean value for each quality has depicted in figure 5.15. The evaluation of the products has generally categorized into positive, neutral, and negative, and it has discussed based on the mean scale value comes into which category mentioned above.

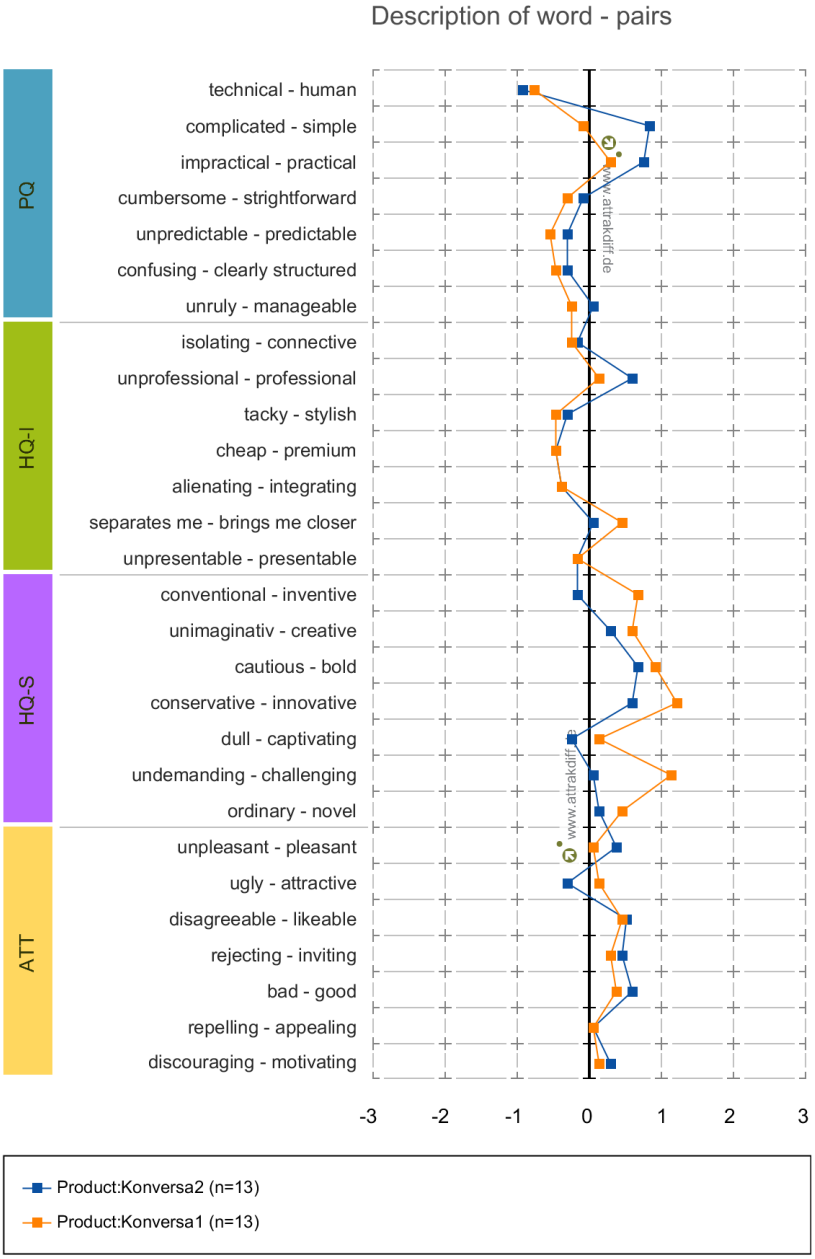


Figure 5.14: The mean value of each characteristics in a specific quality measure, number of participants n=13, ATT - Attractiveness, PQ - Pragmatic Quality, HQ-I - Hedonic Quality Identification, HQ-S - Hedonic Quality Stimulation

Product 1 - Discussion based on the rating scale

- **Pragmatic Quality** The product has considered as more technical, due to the limitations in the datasets. The system has misinterpreted the intents of the user, or it has took wrong system actions for which the responses delivered were not appropriate, and the lack of sentence structure responded to the user. Irrespective of the above mentioned limitations, the product has attained 'neutral', and 'positive' attitude for its simplicity, and practical nature respectively for the proposed DM strategy. It has attained a less negative impact due to the characteristics, such as cumbersome, unpredictable, confusing, and unruly. This negative impact is due to the limitations in the dataset, which has used to train all three components. The improper, and limited annotations in NLU dataset have led to the misinterpretations in perceiving the user intents, imbalance data distribution in DM dataset, and lack of attention to the obtained user input has led to wrong predictions of system actions, and improper sentence structure in NLG dataset has led in delivering a sentence which has confused a user.

- **Hedonic Quality** *Identification* : Due to the drawbacks mentioned above for pragmatic qualities, the experience of the system has obtained a negative impact for the characteristics, such as isolating, tacky, cheap, alienating, and unpresentable. As the system cannot predict all the system actions correct in all dialogue turns, and also the responses were quite artificial, it has created a negative impact for the characteristics mentioned above. In spite of this, the system has attained positive attitude because of its professional nature, and it has observed that the users feels that the system brings them closer to connect with the system.

  Stimulation : The system has attained positive attitude for its characteristics, such as inventive, creative, bold, innovative, captivating, challenging, and novel. The system has intrigued the user's stimulation towards the system. This less positive impact is due to the fact that the complexities involved in the design techniques have explained to the user while the user have asked about the system. The user got clarified at each dialogue turns for the possible outcomes of a response, and the user is capable to get an insight about the reason for the correct, or system behaviour.

- **Attractiveness** The system also has attained positive attitude towards all attractive characteristics, such as present, attractive, likeable, inviting, appealing, good, and motivating. Irrespective of the complexities involved in the design techniques, certain users have satisfied with the responses to reach the fulfilment of the task. Though there were wrong predictions, the developed automated system have attracted the users, and motivated them to have an interaction.

Product 2 - Discussion based on the rating scale

- **Pragmatic Quality** The system has perceived to be technical rather than human-like due to the limitations in the dataset, and also because of the limitations in the proposed strategy. The system has considered to be practical, and simple. The system has attained a neutral attitude towards manageable by the human user. The negative impact has shown for the characteristics, such as cumbersome, unpredictable, and confusing because of the limitations mentioned above.

- **Hedonic Quality** *Identification* : The attitude towards this product, which is semi-automatized is similar to the product 1 mentioned in previously, except that the system has obtained more positive attitude towards the professional characteristic, when compared to product 1. Also, the system has obtained a neutral attitude for a characteristic of brings

me closer, while certain users have rated towards the scale that the system separates from the people. *Stimulation* : The system has negative impact when compared to product 1, since the product is more conventional, and also it has rated that it is dull rather than captivating. Its because certain users have asked about the difference in the design techniques, and they have the knowledge that the system has applied with certain predetermined rules, and knew that the system is not fully automatized.

- **Attractiveness** The product has obtained similar attitude towards all the characteristics in product 1, except that the product has recognized as ugly rather than attractive.

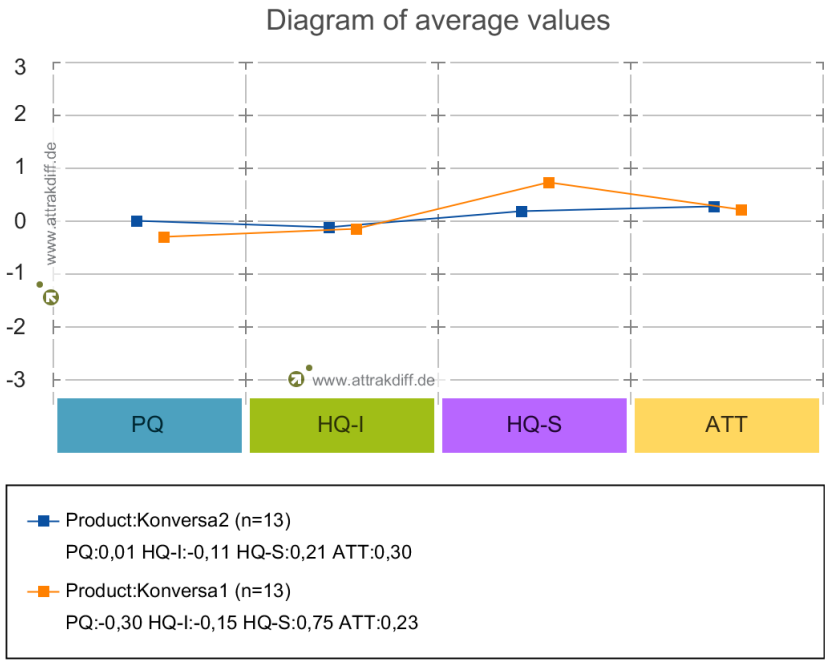


Figure 5.15: The average value of all the characteristics in a specific quality measure, number of participants n=13, ATT - Attractiveness, PQ - Pragmatic Quality, HQ-I - Hedonic Quality Identification, HQ-S - Hedonic Quality Stimulation

Overall reaction towards the attractiveness of the system

The combined results of the pragmatic quality, and the hedonic quality in the overall attractiveness of both the system have depicted in Figure 5.16 below. The fully automatized system, and the semi-automatized system have surpassed the negative impact, and stays more in a neutral position. The semi-automatized system has the capability to correct itself whenever the system takes the wrong dialogue turn. Both the systems have the capability to either ask the user anyone of the missing information for a respective intent or else it can make a recommendation based on the given specifications. Also the system has the capability to either list all the availabilities for the user specifications or else make a recommendation. But certain user expects that the system have to behave according to their expectations in a whole conversation, which is quite hard to satisfy even by humans in a real-time situations. Some user expects that the system has to list all the availabilities for a single contextual information provided in the input, whereas certain users expects to show all the availabilities for multiple contextual information. Hence designing such a system to handle all the expectations of the user is really a challenging task.

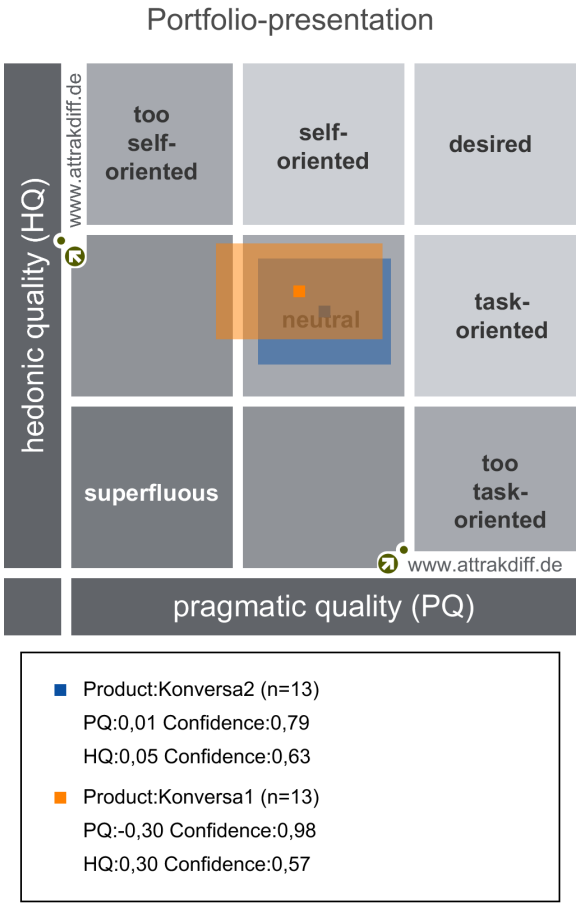


Figure 5.16: The overall attractiveness towards the system based on PQ, and HQ, number of participants n=13, ATT - Attractiveness, PQ - Pragmatic Quality, HQ-I - Hedonic Quality Identification, HQ-S - Hedonic Quality Stimulation

Also, the user expect a system to satisfy their requests' by exactly giving the corresponding contextual information in the responses after getting results from APIs. Though the predicted system action is appropriate for the correctly perceived user input associated with the contextual information, due to the limitations in the dataset of all the components, the delivered response is not sufficient to satisfy the user's request. This has led to the negative impact in certain characteristics of the system.

Finally, the user expects that the system should request the user the relevant information pertaining to the intent of the user in the current input, which has not specified by the user. Though the system has trained on the dialogues for certain intents to handle this kind of situations, certain intents have not enough dialogues in the dataset to train the system for this possibilities. Thus the resource of the dataset has a great influence on the satisfaction of the user to handle this situations. Hence, the fully automated system has surpassed the negative attitude, and stays in a neutral position. Also, the confidence of the quality of the product goes towards self-oriented, because of the complexities involved in designing the system. The semi-automatized system stays in a neutral position, and it has the possibility to get a desired attitude when a proper resource has provided in the dataset for training the network.

5.7.3 Results of live user evaluation - survey 2

The participants of the survey 2 are exactly similar to survey 1. The survey procedure has also followed exactly as in survey 1. Each user has assessed each Interactive system after interacting with each system. The only difference is that the rating is not anonymous in survey 2, whereas the rating is anonymous in survey 1. It means that the developer has not known about the individual rating of the user for assessing each system in survey 1.

Each user has rated each characteristics of the quality measure on a Likert scale of points '1' to '5'. The rating '1' shows the totally negative attitude of the user towards the system, and '5' shows the totally positive attitude towards the system. The mean value for each characteristics of the quality measure has calculated for both the products. It has depicted in Figure 5.17, and Figure 5.18 respectively. Thus, in the graph the total negative attitude has regarded as '-2,' less negative as '-1,' neutral as '0,' less positive as '1' and totally positive as '2', which has based on the Likert scale of '1' to '5' respectively. Then, these rating values have used to make objective measures on collective subjective assessment, such as cronbach's alpha measure, correlation coefficient, and consensus measure.

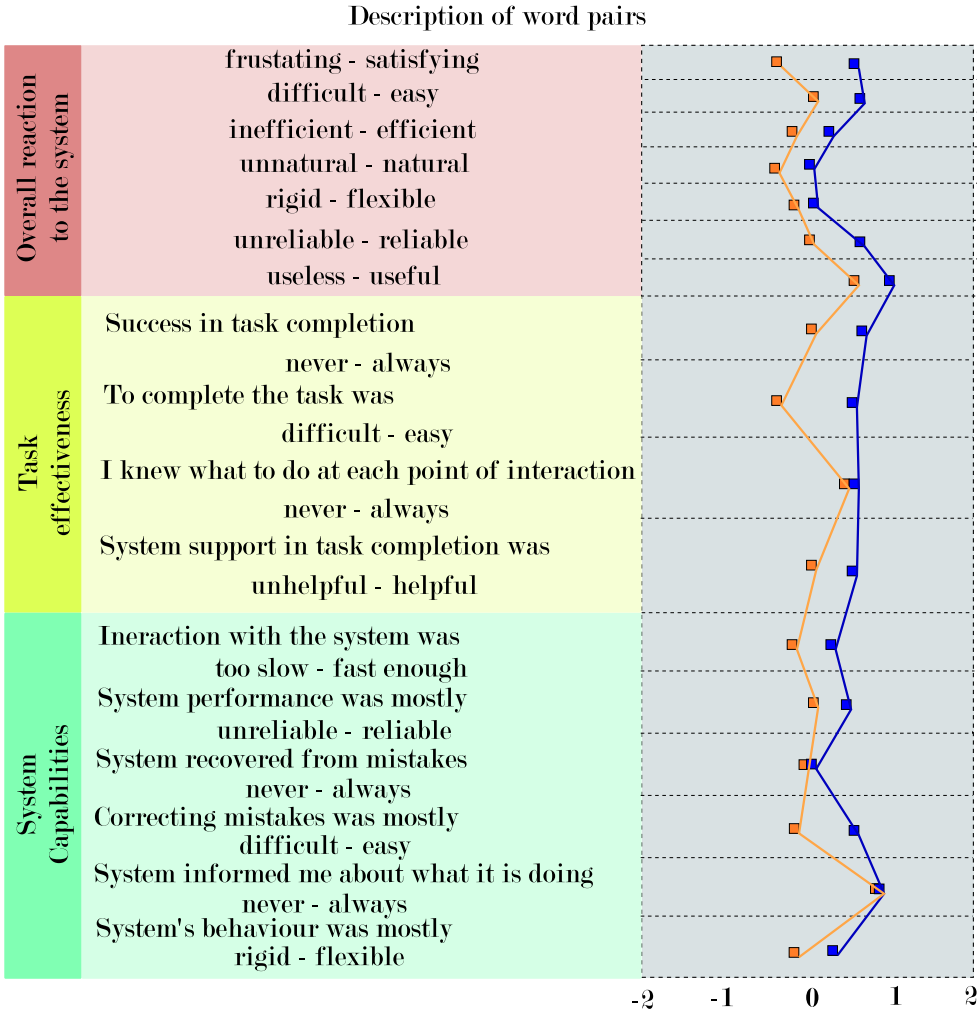


Figure 5.17: The mean value of each characteristics of specific quality measure, number of participants n=13

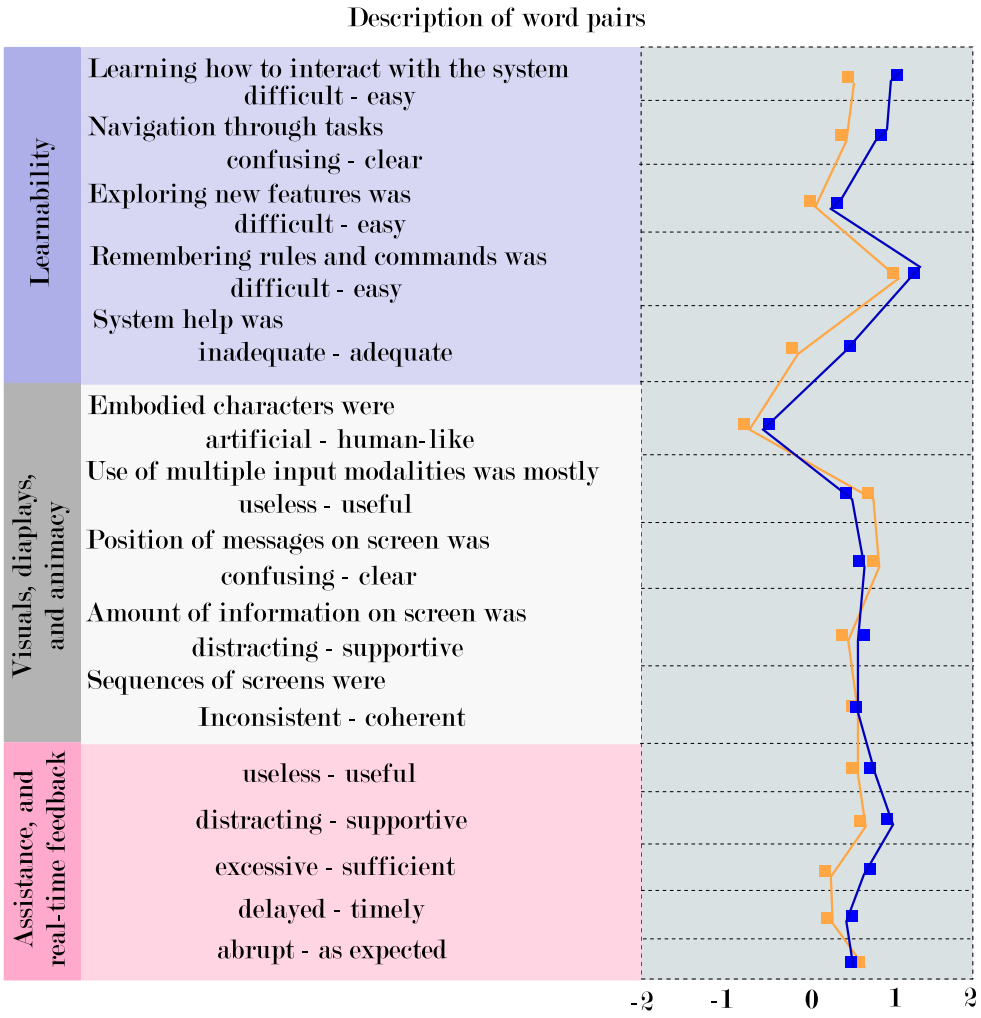


Figure 5.18: The mean value of each characteristics of specific quality measure, number of participants n=13

Product 1 - Discussion based on rating scale

- **Overall reaction to the System** From the average of all characteristics specific to the quality measure, the system has attained neutral towards negative attitude. The system has identified positive for the characteristics because of its reliability and ease of use. It has identified as useful, which has attained a more positive attitude of all other characteristics. The system has obtained a negative attitude because of its frustrating behavior, inefficiency, rigidity, and unnaturalness. The negative impact is because of the limitation mentioned in the dataset and the high expectations of the user.

- **Task Effectiveness** The system stays neutral from the collective subjective response. The system has described it as challenging to accomplish the respective tasks, which had created a negative impact upon the users. Whereas, the system has gained a positive attitude because of its successful fulfillment of the end goal of the user, the system's support, and the user had known to make an appropriate decision based on the previous response. The automatized domain-specific ICA has achieved to satisfy the end goal for the user specifications, irrespective of the wrong dialogue-turn in the middle of a conversation.

- **System Capabilities** The system has attained a positive attitude towards the charac-

teristics, such as performance reliability, and the delivery of the response. But the system remains in a negative attitude for its characteristics, such as rigidity, interaction speed, recovery from the mistakes. The technical issues with the framework which has chosen to build the system have influenced these negative impacts, especially in the interaction speed of the system. Also, the limitations in the dataset have played a role in the negative impact.

- **Learnability** The system behavior has identified with a more positive attitude for the characteristics associated with these behaviors, such as ease of interaction, clarity in the navigation of tasks, exploring new features, remembrance of rules and commands. But because of its inadequate help, the collective assessment has shown a negative impact from the users. The user got an insight into the agents before holding the conversations. Also, each user has learned the system's behavior quickly from the response of the user's requests, which has created this positive impact. As the user knew the general behavior of the system before the interaction, each user has found ease with interaction.

  Also, as the system has automatized, each user held a conversation without any complicated rules and commands. The user has interacted as how interacting with humans. These characteristics have created a tremendously positive impact on collective assessment from the users. It was difficult for the user to understand when the system has taken the wrong dialogue turn, and it could provide any satisfactory responses as expected by the user for its wrong behavior. When the user has tried to rectify the mistakes, the system has not recovered the mistakes in some cases. The technical issues with the framework used to design the system also influenced the negative impact.

- **Visuals and Display** The collective assessment has shown that the users had shown a negative attitude because the characters that have displayed by the system are too artificial. It is due to the lack of proper textual contexts in the dataset that has used to build the system. Since the human-human conversations dialogue corpus has chosen, it is prone to improper contexts, or misaligned contextual information. Each human has its representation of dialogues, which might not always be grammatically correct. The human also tends to mention only the keywords sometimes along with a long grammatical sentence. Also, all humans are not fluent in a language, because of their ethnic region and their professional qualification had also an influence.

  The system has obtained a positive impact because of its feasibility to accept multiple input modalities and clarity in the position of messages. Also, it is due to an appropriate amount of information on the screen, and consistency in the sequences displayed in the screen. Irrespective of the limitation mentioned in previous lines, the system can handle multi-language texts as well, excluding that the contextual information should be in English.

- **Assistance and Feedback** The system has not provided any online help. Each user has assisted by the developer by explaining each situation at each dialogue turn. Also, the system has shown the perceived user intents and associated entities at each dialogue turn, which had helped the user to connect to the system quickly. The collective analysis has shown that the users had more positive attitude for the expectation, and support of the assistance when compared to other characteristics, such as usefulness, sufficient, and timely assistance.

Product 2 - Discussion on the rating scale

- **Overall Reaction to the System** All the characteristics of this quality measure have obtained a positive attitude by the collective average analysis of the users, except for naturalness. It has attained a neutral position from the average collective rating from the user. The system has identified as useful, which has the most positive attitude of all other characteristics. The system has satisfied the users during the interaction process, and it has proven to be easy for use and reliable.

- **Task Effectiveness** The collective average analysis has shown that the system can complete most of the tasks, which is one of the significant characteristics for the assessment of a system. The users have shown a positive impression for ease in completion of tasks, ease of understanding the interaction, and for the support given by the system in accomplishing the tasks. This system has assisted with pre-determined rules when a system has taken a wrong dialogue turn. Hence, the system can rectify its mistake before delivering a response to the user. It has made a positive impression because the system has behaved according to the user's expectations to complete the tasks successfully. The system has recovered from the mistakes, and the fulfillment has attained within lesser dialogue turns. It has made the user interact with the system easily to accomplish their tasks.

- **System Capabilities** The system has obtained a positive impression because of its characteristics, such as interaction speed reliability, ease of recovery from the mistakes, information support given by the system. The users' collective analysis has shown that the users have a neutral attitude towards the flexibility of the system's behavior and recovery from the mistakes.

  Despite the technical issues in the framework for building the system, the interaction speed has accepted by the collective judgment. It is because the system has not taken wrong dialogue turns in most cases. Also, the system had accomplished the tasks in lesser dialogue turns when compared to the fully automatized system. Since the system has assisted in taking correct dialogue turn for the perceived input, the system has attained a positive impression towards the reliability. The system has designed to provide information on the perceptions of the users' requests. Thus, each user has understood the behavior of the system for each of their dialogues. The system is also flexible for various dialogues of the user. Due to these techniques, the system has attained a positive impression of this quality measure.

- **Learnability** The collective mean analysis has shown that the user assessed that the system is easy for interaction, and it does not involve any complicated rules and commands. This great positive impact is due to the fact of automation of the system to behave like a human. The system has attained a positive attitude for its characteristics, such as exploring new features and navigation through the tasks. Additionally, the system's help during the conversation has also created a positive impact. The navigation through the tasks has become more accessible for the user because the system has assisted. Thus, it has accomplished the tasks in a lesser dialogue turns. The system has designed to provide help in their responses for engaging the user with the system.

- **Visuals and displays** The only characteristic of the overall characteristics that made the user have a negative impact is the embodiment of the characters. It has provided the users more artificial feel rather than a human-like feel. It is because of the limitations in the dataset that have mentioned in the previous sections. All other characteristics of this quality measure, such as multiple-input modalities, the position of messages, supportive information on the screen, coherence in the message sequences, have created a positive impact on the user.

- **Assistance and Feedback** All the characteristics that have involved in this quality measure of the system have created a positive impact on the users. It is because each user has assisted while interacting with each system, and their questions have clarified. Also, the behavior of the system has explained to the users to take their representation of dialogues. To recover from the mistakes, the user has assisted with suggestions to make different dialogues and answered with the user's questions on the dialogue types, such that the system can satisfy the user's requests. It is mainly because of the lack of all possible sources in the dataset.

Discussion based on the average rating of the characteristics

All the characteristics corresponding to each quality measure have averaged, and it has depicted in Figure 5.19.

- **Product 1** The fully automatized system has achieved a positive impression from the collective assessment. It is because of its characteristics, such as Learnability, Visuals and displays, and Assistance and Feedback. The users have shown a neutral attitude for the effectiveness of the performed tasks. The users have expressed a negative impact on the overall reaction, the capabilities of the system. The users have not shown the worst negative impact, and it is almost towards a neutral attitude. The system lacks naturalness and flexibility to satisfy every user's expectations. it is due to the lack of source of data. When the system has taken wrong dialogue turns, most of the users have found it difficult to correct the mistakes by their dialogue representations. The user has taken multiple dialogues turns to accomplish the tasks for the modifications in the user specifications.

  Also, the technical issues in the framework have faced while developing the system. It influences the interaction speed of the system, and the system has slowed down when the technical issue has occurred. Because of limitations in the dataset, the system has taken wrong dialogue turns while holding the longer conversations. Despite these limitations, the system has accomplished the fulfillment in the overall tasks for the user specifications. Also, it has designed such that the user has interacted as how each one holds interaction with a human. The system can also decide for the dialogues of multi-languages as well, conditioned that the contextual information has provided in English with an appropriate dictation.

- **Product 2** The semi-automatized system has attained a positive impression for all the quality measures based on mean values. Since the system has designed to correct itself before delivering the response, the user has achieved the fulfillment easily. The user also identified that correcting the mistakes with this system is easier when compared to a fully automated system. The system could not attain the desired attitude from the collective assessment because of the limitation in the dataset, and the complexities involved in deciding for each perceived intent of the user.

  Though the system can self-decide o make a suggestion, it has designed to request the user to fill the missed information when many of the values of the slot are missing for certain intents. Due to the lack of all possible sources in the dataset, requesting the user for missed information is not possible for all the perceived intents of the user.

Both the systems cannot provide a happy path in all the dialogue turns, which is a very challenging task considering the limitations mentioned above. Still, irrespective of all the limitation, both the system have surpassed the negative impact, and remains neutral. The system has to feed with many resources of data to handle most of the dialogues appropriately. The dataset also had an impact on the REDP dialogue policy, which has led to inappropriate system action in some cases.

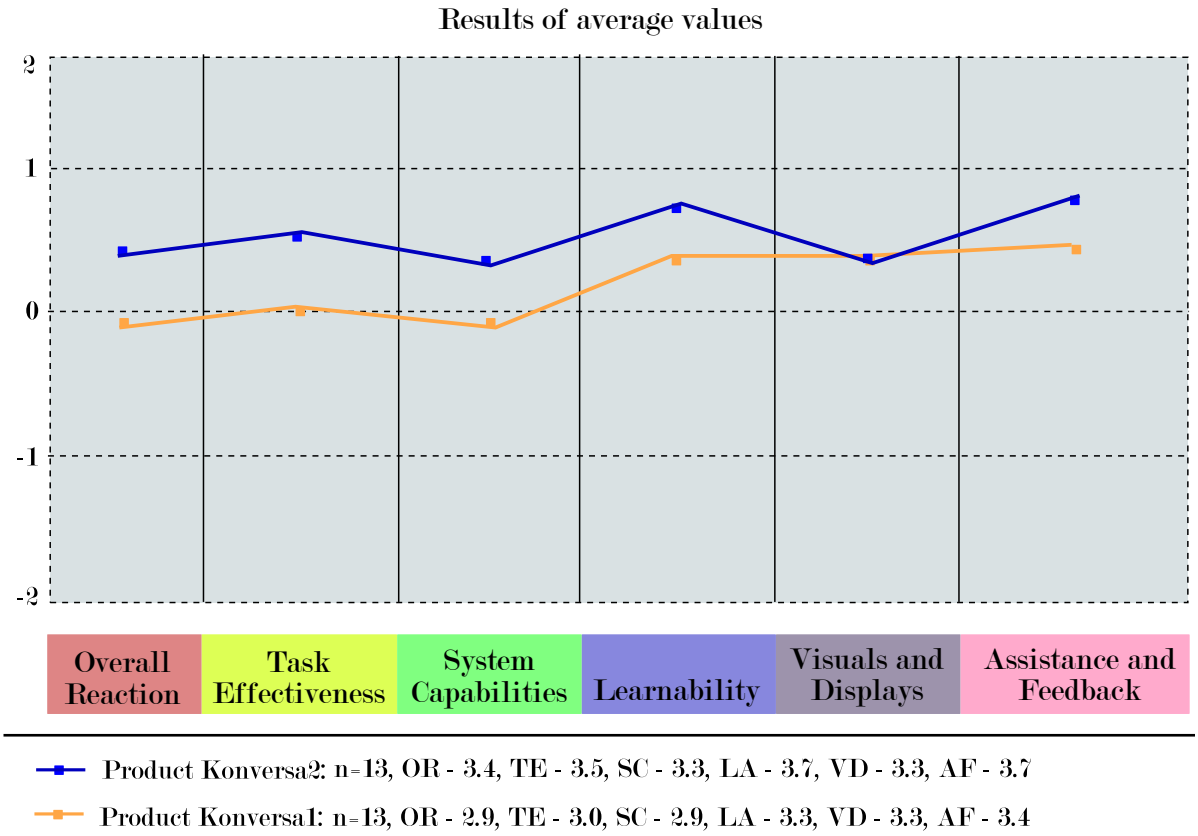


Figure 5.19: The average value of all the characteristics of specific quality measure, number of participants n=13, OR - Overall reaction, TE - Task effectiveness, SC - System capabilities, LA - Learnability, VD - Visuals, and displays, AF - Assistance, and feedback

Collective assessment based on Cronbach's alpha measure

The Cronbach's alpha measure has calculated for each quality measure[1]. The graph for Cronbach's alpha measure has shown in Figure 5.20.

- **Product 1** The Cronbach's alpha is greater than 0.70 for all the quality measures, except for 'Visuals and displays.' The overall reaction to the system, and the Learnability quality measures have the Cronbach's alpha greater than '0.90.' It determines that the quality of a product is excellent for those quality measured. The quality measures, such as Task Effectiveness, System Capabilities, and the Assistance and Feedback, have the Cronbach's alpha value in the range between '0.8' and '0.9'. It determines that the product's quality corresponding to these quality measures is better, whereas the Cronbach's alpha value is less than '0.70' for Visuals and display. It has concluded that the quality of the product concerning this measure is not satisfactory. It is because of the limitations mentioned in the previous sections.

- **Product 2** Though the average measure of all characteristics of each quality has shown a positive impression, the Cronbach's alpha measure for this product has shown a different result. The overall reaction to the performance of the system has attained the Cronbach's alpha measure higher than '0.90', which determines the quality of the product concerning this measure is excellent. It has acquired only a satisfactory response for its effectiveness

[1]Wessa P. (2017), Cronbach alpha (v1.0.5) in Free Statistics Software (v1.2.1), Office for Research Development and Education, https://www.wessa.net/rwasp_cronbach.wasp/, textitVersion February 2020

in performing the tasks, and for the capabilities associated with the system. The system has the Cronbach's alpha value of '0.70' and '0.78' for its task effectiveness and system capabilities, respectively. The system is unsatisfactory for its characteristic of 'Visuals and Displays,' which has the Cronbach's alpha value less than '0.70.' The collective assessment for the Assistance and feedback has proven that it is excellent as the Cronbach's alpha value is higher than '0.90.'

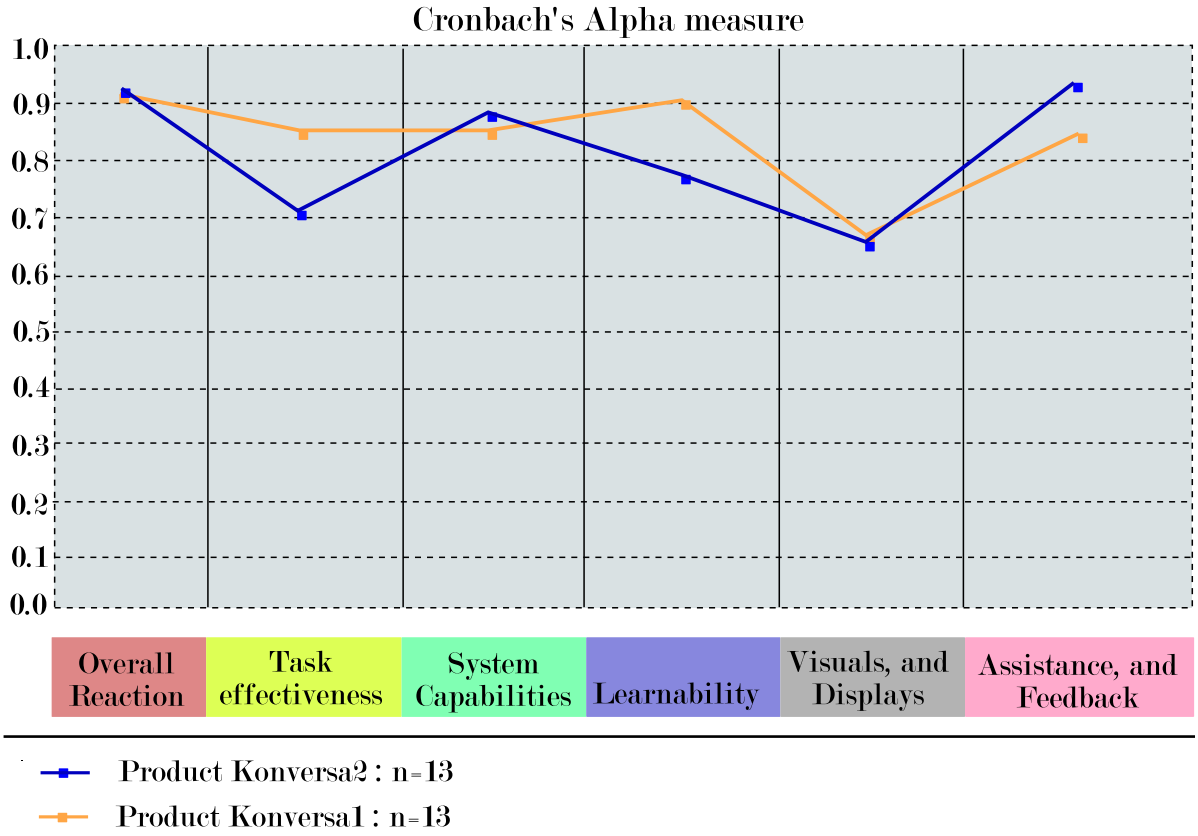


Figure 5.20: The Chronbach's Alpha measure for each quality measure, number of participants n=13, OR - Overall reaction, TE - Task effectiveness, SC - System capabilities, LA - Learnability, VD - Visuals, and displays, AF - Assistance, and feedback

Discussion on the collective assessment of correlation co-efficient

The characteristics of the overall reactions have to correlate with all the characteristics of other quality measures. Hence, it is important to identify the relationship between all quality measures' characteristics with each characteristic of the overall reaction. To calculate the correlation coefficient, the mean of all the characteristics of all the quality measures has calculated. The number of characteristics varies for each quality measure. Hence, the correlation coefficient has determined by selecting the equivalent number of characteristics in the 'overall reaction' that are related to other characteristic features.

- **Product 1** The correlation between the overall reaction and all other qualities is greater than '0.80'. The correlation between each characteristic of the quality measure from the collective assessment is strong. It determines that the collective assessment has a strong opinion on the quality of the product

- **Product 2** The correlation coefficient is higher than '0.90' for all the qualities to the overall reaction, except for system capabilities. The correlation between the overall reaction and the system capabilities lies in the range of '0.60' and '0.70.' It determines that the correlation between these qualities is weak, and the collective opinion based on this quality measure is not confident.

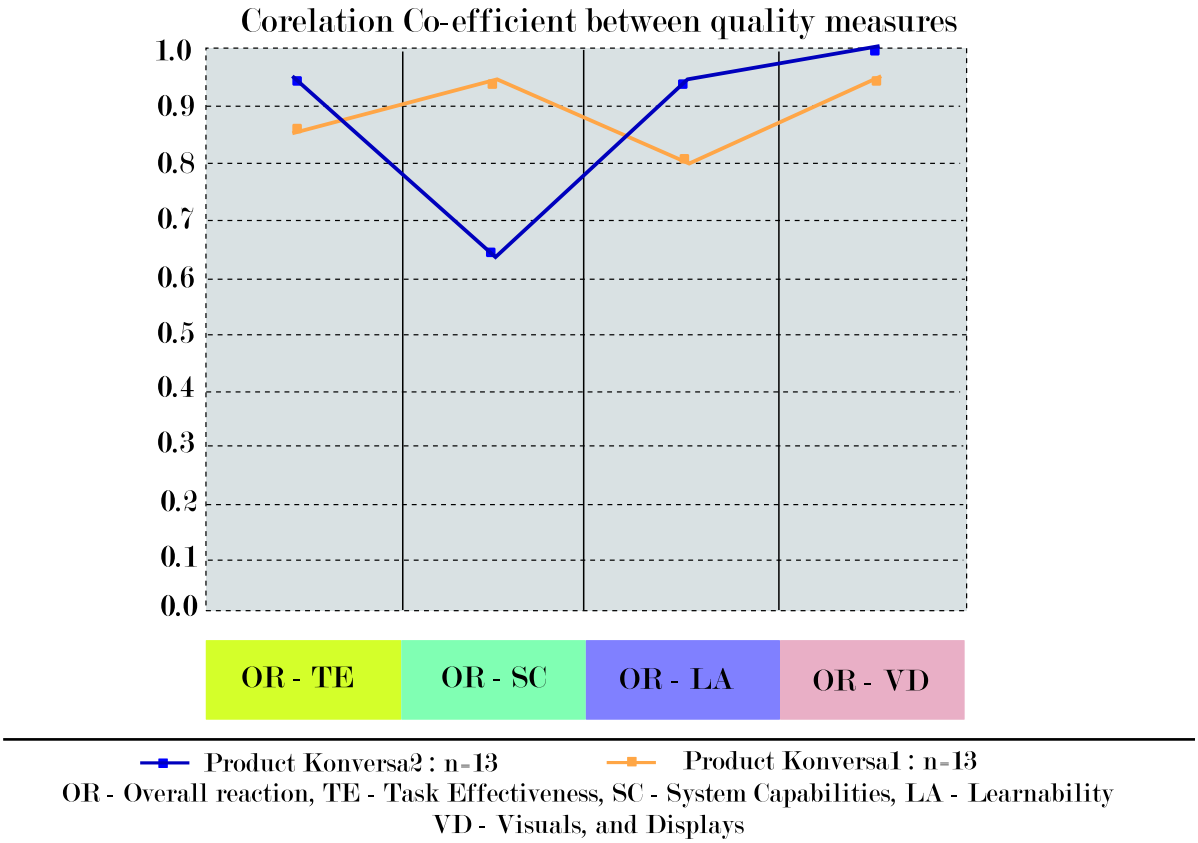


Figure 5.21: The co-relation coefficients for each characteristic of the overall reaction to each of the other Quality measure characteristics, number of participants n=13

Collective assessment based on Consensus measure

The Cronbach's alpha gives the composite measure of all the characteristics of a specific quality. But it is also significant to determine the collective objective measure for each characteristic of all the quality measures. This collective opinion on each characteristic is useful to understand the system's behavior effectively, which helps in redesigning the system for improvements. The collective assessment of each characteristic has depicted in Table 5.2, 5.3, 5.4, and 5.5.

- **product 1** The success rate in the completion of tasks from the collective assessment has achieved 63%. Except for the characteristic of easiness in the completion of tasks, all other characteristics have achieved the consensus measure higher than 50%. Hence, it has concluded that product 1 stays the 'Neutral Positive' position from the perspective of the collection of users.

- **Product 2** It has achieved the success rate of 67% in accomplishing every task, which has requested by the user. All the characteristics have achieved the consensus measure higher than '0.50'. It also has surpassed the negative attitude but stays in a neutral position.

5.8 Discussion based on User Experience

5.8.1 User Experience

It has analyzed that most of the users expected the system to behave according to their requirements. Each user had their custom requirements. Some users have satisfied with the self-decision made by the system, whereas some users expect the system to request the user to fill out the missed information. Alternatively, some users expect the system to decide for the specified requests through certain contextual information that has not mentioned. It has requested to hold a natural conversation with the system. Though each user has given their dialogue representations, the system has not achieved a 100% success rate on all its dialogues. It has frustrated the user while interacting with the system. Thus, the user had taken multiple dialogues turns to achieve their tasks. In certain cases, irrespective of multiple turns, some users have discussed that the system could not handle the conversation in a user-oriented way. Thus, a semi-automatized ICA has designed with certain pre-determined rules before delivering a response to the user, which has identified to be effective when compared to automatized ICA.

The users have stated that the interaction speed is slow. Also, the users have expressed that the system has found itself hard to recover from the user, such that the user can further hold the conversation. Despite this unsatisfactory attitude, the users have identified that the system is reliable in performing the tasks since the system can reach the end goal of the task. The user identified that the system could able to handle the multiple dialogues of various intents, and hence it has attracted the user for its flexibility.

The systems involve automation, which has made the tasks easier for users to interact with the system without any complicated rules. It has helped the users to learn the behavior of the system. Still, the users have felt that the system is more technical rather than natural. The system has assisted the user with its responses when it has taken a correct dialogue turn, but the embodied characters lack the proper structure. The collective analysis has shown that navigation through the tasks was quite difficult since the system has not handled some of the user dialogues with an appropriate response. But the challenging nature of the system has intrigued the users, especially who are in the age range between 20 and 40.

It has analyzed from the collective assessment of attractiveness, and the objective measures that the system can handle various input modalities, but the users have felt that the position of the messages was distracting. The lack of sentence structure has led to this unsatisfactory opinion on the users. The users have expected that the system has to behave more human-like. Though the systems were efficient in accomplishing most of the tasks, the naturalness of the system is less. Though it involves the natural dialogues, certain informative responses have confused the user. It had created an unsatisfactory opinion, and it had confused the user to take further dialogue turns. The collective users have felt that the system provided consistent support in accomplishing the tasks, and easy for use due to its reliability. They have also expressed that efficiency is satisfactory, and the products have found to be useful. Also, the user has expected that the system should react appropriately for non-contextual sentences as well. Though the system has built to handle these cases, the system cannot handle all possible scenarios of these types of situations. Still, both the systems have created a neutral-positive impact on the collective opinion of the users.

5.8.2 The associated challenges

The designing of the system has involved multiple complexities, from the available open-source data, a promising strategy for predicting a system action, and automating the responses.

Participants	**OR**							**TE**				**LA**				
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C1**	**C2**	**C3**	**C4**	**C1**	**C2**	**C3**	**C4**	**C5**
PP1	3	4	2	4	2	3	5	2	4	4	3	3	3	2	2	3
PP2	2	2	3	2	3	3	3	4	2	1	2	3	5	3	5	3
PP3	2	2	2	1	3	2	2	2	1	2	3	4	3	2	4	2
PP4	1	3	1	2	4	3	2	2	1	1	1	1	2	1	4	1
PP5	2	3	2	3	1	2	2	3	1	4	4	4	4	4	4	3
PP6	1	2	1	1	1	1	2	2	1	3	2	1	2	1	3	2
NP1	2	2	3	2	2	3	4	4	3	4	3	4	3	4	5	3
NP2	3	3	4	3	3	3	4	4	3	3	4	4	4	3	4	3
NP3	2	4	3	4	2	2	4	3	3	4	2	4	3	3	4	3
NP4	4	3	4	3	3	4	4	4	3	4	3	4	4	4	4	4
NP5	4	4	3	3	3	4	3	3	4	4	4	3	4	3	4	4
NP6	4	4	3	3	4	4	4	4	4	3	4	3	3	4	4	3
NP7	4	5	5	4	5	5	5	5	5	5	4	5	5	5	5	4

Assessment	**OR**							**TE**				**LA**				
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C1**	**C2**	**C3**	**C4**	**C1**	**C2**	**C3**	**C4**	**C5**
Mean	2.6	3.1	2.8	2.6	2.8	3.0	3.5	3.1	2.6	3.3	3.1	3.4	3.3	3.0	4.0	2.8
CnS	0.58	0.66	0.55	0.63	0.65	0.57	0.58	**0.63**	0.46	0.55	0.65	0.59	0.66	0.58	0.80	0.75

Table 5.2: Product 1: Consensus Measure for each Characteristics of the subjective quality measure based on Likert scale. OR - Overall Reaction, TE - Task Effectiveness, LA - Learnability, PP - Professional Participants, NP - Naive Participants

Participants	SC						VD					AF				
	C1	C2	C3	C4	C5	C6	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
PP1	3	3	2	2	2	4	2	4	2	3	3	4	4	3	4	5
PP2	5	5	4	4	5	3	3	3	5	4	5	3	3	3	3	3
PP3	2	2	3	3	4	3	1	3	4	3	2	2	3	2	2	2
PP4	3	4	4	3	3	2	1	4	5	2	5	2	2	1	4	4
PP5	3	2	2	3	4	1	2	3	4	3	3	4	5	5	5	5
PP6	2	1	1	1	2	1	1	3	5	3	4	2	3	3	2	2
NP1	3	2	3	2	3	3	2	5	4	4	3	4	3	2	3	3
NP2	2	2	3	3	4	3	3	4	3	3	4	3	3	4	3	4
NP3	3	3	3	2	4	2	2	3	4	4	3	4	3	4	3	3
NP4	3	4	3	4	5	4	3	4	4	4	3	4	5	4	3	4
NP5	2	4	3	4	4	3	2	4	3	4	4	4	4	3	4	4
NP6	3	4	4	2	4	4	3	4	4	3	4	4	5	4	4	5
NP7	3	4	3	4	5	4	4	4	3	4	3	5	4	4	2	3

Assessment	SC						VD					AF				
	C1	C2	C3	C4	C5	C6	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
Mean	2.8	3.1	2.9	2.8	3.8	2.8	2.2	3.7	3.8	3.4	3.5	3.5	3.6	3.2	3.2	3.6
CnS	0.77	0.56	0.75	0.66	0.66	0.50	0.79	0.79	0.71	0.77	0.69	0.65	0.66	0.53	0.68	0.62

Table 5.3: Product 1: Consensus Measure for each Characteristics of the subjective quality measure based on Likert scale.
SC - System Capabilities, VD - Visuals, and Displays, AF - Assistance, and Feedback, PP - Professional Particpants, NP - Naive Participants

Participants	**OR**							**TE**				**LA**				
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C1**	**C2**	**C3**	**C4**	**C1**	**C2**	**C3**	**C4**	**C5**
PP1	4	4	3	4	3	4	5	3	4	4	4	4	4	3	3	4
PP2	3	4	3	2	2	4	4	4	3	3	3	5	5	3	4	4
PP3	4	4	4	4	4	4	5	4	4	4	2	4	3	2	5	4
PP4	2	2	2	2	1	2	2	3	2	2	2	3	3	3	3	2
PP5	4	4	3	2	3	5	4	5	4	4	3	3	4	3	5	4
PP6	3	3	2	1	1	2	3	2	1	3	5	4	3	2	3	3
PP7	1	2	3	3	4	1	2	2	3	3	2	4	3	2	4	2
NP1	3	4	3	3	2	3	4	4	3	3	4	4	3	4	5	3
NP2	4	4	4	3	3	4	4	4	4	2	4	5	4	4	5	4
NP3	4	4	3	4	3	3	4	3	4	3	3	4	4	3	4	4
NP4	4	4	3	4	4	5	5	4	4	5	5	4	4	5	4	4
NP5	4	3	4	3	4	5	4	4	5	4	4	4	4	4	5	4
NP6	5	5	4	5	5	5	5	5	5	5	4	5	5	5	5	4

Assessment	**OR**							**TE**				**LA**				
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C1**	**C2**	**C3**	**C4**	**C1**	**C2**	**C3**	**C4**	**C5**
Mean	3.5	3.6	3.2	3.0	3.1	3.6	3.9	3.6	3.5	3.5	3.5	4.1	3.8	3.3	4.2	3.5
CnS	0.64	0.71	0.79	0.62	0.58	0.50	0.67	**0.67**	0.60	0.66	0.62	0.83	0.76	0.64	0.71	0.74

Table 5.4: Product 2: Consensus Measure for each Characteristics of the subjective quality measure based on Likert scale. OR - Overall Reaction, TE - Task Effectiveness, LA - Learnability, PP - Professional Participants, NP - Naive Participants

Participants	SC						VD					AF				
	C1	C2	C3	C4	C5	C6	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
PP1	3	3	3	4	3	4	3	3	3	4	3	4	4	4	4	5
PP2	4	3	3	4	4	2	2	4	2	3	4	5	5	5	5	5
PP3	3	4	3	3	3	3	3	4	4	4	4	3	2	3	3	3
PP4	4	4	2	3	4	2	1	1	4	4	4	2	2	2	3	2
PP5	5	5	4	5	4	4	3	5	5	5	4	5	5	5	5	5
PP6	2	3	3	2	3	2	2	3	3	3	3	3	3	3	1	3
PP7	2	2	2	2	3	2	1	2	5	3	2	2	2	4	3	2
NP1	2	3	3	3	4	3	2	4	4	4	3	4	4	3	4	3
NP2	3	4	4	4	4	3	3	4	3	3	4	4	4	4	3	3
NP3	4	3	3	3	4	3	2	3	4	3	3	3	4	3	3	3
NP4	4	4	4	5	5	4	3	3	3	4	4	4	5	4	4	4
NP5	4	4	3	4	5	4	3	3	4	4	4	4	5	5	5	4
NP6	3	4	3	3	4	4	4	5	3	4	3	5	4	4	3	4

Assessment	SC						VD					AF				
	C1	C2	C3	C4	C5	C6	C1	C2	C3	C4	C5	C1	C2	C3	C4	C5
Mean	3.3	3.4	3.0	3.5	3.8	3.0	2.5	3.4	3.6	3.7	3.5	3.7	3.9	3.8	3.5	3.5
CnS	0.67	0.73	0.83	0.66	0.79	0.71	0.69	0.61	0.70	0.79	0.77	0.64	0.59	0.68	0.60	0.62

Table 5.5: Product 2: Consensus Measure for each characteristic of the subjective quality measure based on a Likert scale.
SC - System Capabilities, VD - Visuals, and Displays, AF - Assistance, and Feedback, PP - Professional Participants, NP - Naive Participants

- **Influence of the dataset** The dataset had a great influence on the experience of the user. The NLU component requires data with annotation of contextual information, as the Supervised Learning technique has used to achieve it. The chosen open-source dataset lacks from incorrect annotation, and some contextual information has not annotated in the dataset. The data that has used for the NLG component lacks sentence structure. There are multiple possible candidates for a chosen system action, but the candidates do not involve all the contextual information as expected by the user. It requires certain filtering techniques to select an appropriate candidate for the response. Also, each candidate might have multiple suggestions for the user specifications, which have also handled in building the system.

- **Complexities in the decision-making** An ICA must decide the respective action to satisfy the users' requests. There is essential contextual information that has an association with the intent of the user. In this type of user dialogues, the system can ask the user to fill the information that has not specified by the user, or it can recommend a user for the given specification, or else the system has an option to list all the available information for that user specifications. Each user has different expectations. Some expect a recommendation though the user has not specified all the contextual information, some users might expect to make a request when they have not specified any relevant information. Also, for specific intents of the user, it has to take immediate action without any further requests. The complexities that have involved in the decision-making of choosing an appropriate system action to satisfy the user has shown in Figure 5.22.

  The dataset has a lack of source to tackle the complexities that have involved in deciding an appropriate system action from the possibilities that have depicted in Figure 5.22. Though the dataset has the source for the intent 'Restaurantinform' with possible dialogues relevant to various possibilities, it lacks the amount of data suitable for training the network model for intent 'Informtobook.' The dialogue responses for the intent 'Restaurantrequest' have to take immediate action for the users' request, and it may request the user whether further contextual associated with this intent are necessary. Despite these complexities involved in the decision-making to make a suitable system action, the developed systems have surpassed the negative impact of the collective analysis of the users.

- **Dialogue Management policy** The Dialogue Management strategy involves complications in training the custom-designed deep learning network. Despite the effectiveness, reliability, and the flexibility of the architecture, The deep chosen strategy depends on the data. Though the dataset involves very short sequences in both inputs and targets, the preferred approach depends heavily on the feature vector representation of the input data. Most of the input sequences have similar data representation with few modifications, and it is difficult for the deep learning network to learn these representations, and predict an appropriate system action.

  The tuning of the developed architecture is a daunting task since the parameter updates changes concerning the defined settings. It took much effort to find an appropriate parameter value, such as epochs, batch size, optimizers, and the associated hyper-parameters. Also, by choosing validation samples separated from the training samples has improved the performance of the model. The batch size has increased to reduce the number of iterations per epoch while training, such that efficiency has seemed to increase. Still, increasing the batch size will increase the training speed, but the parameter updates do not happen efficiently. The optimizer helps in optimizing the network model with suitable parameter updates to minimize the error function and increase the performance of the model. The

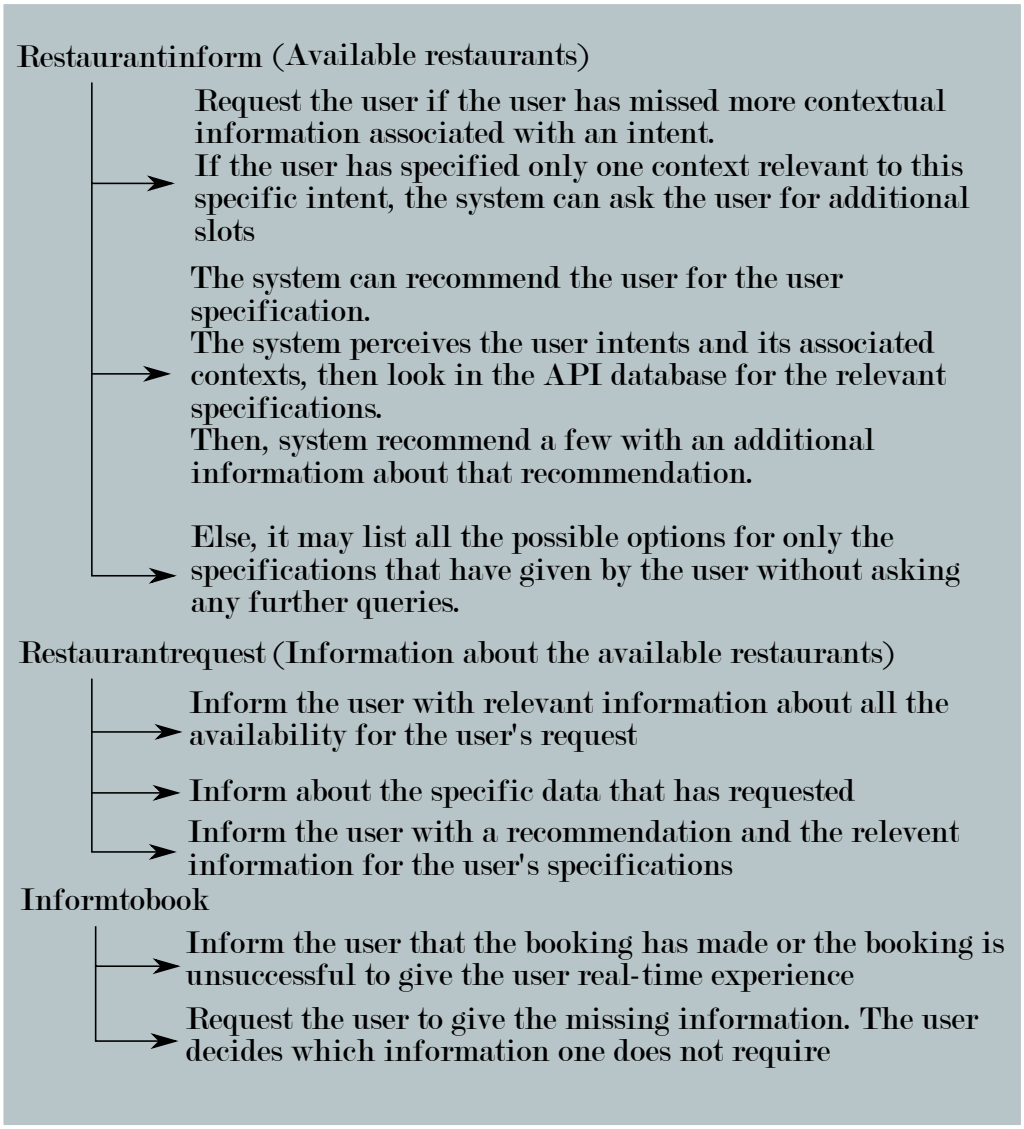


Figure 5.22: The complexities involved in the decision-making for the system

optimization has an association with the epochs and learning rate to obtain global minima or local minima of the error function to update the parameter values. The learning rate defines the step size for annealing to achieve optimization. Hence, it takes much effort to perform trial- and error analysis to fine-tune the custom-developed architecture for better performance.

- **Automated Response selection** The machine learning model has employed to achieve automated responses for the perceived user's requests. The pre-processing techniques have modified according to the input and output sequences in the customized dataset. Though a custom strategy has proposed for this component with an attention mechanism to handle long sentences, it has not explored much in this proposed work. This proposed architecture has left for future scientific work because of its limitations. The machine learning algorithm used for this component has used default parameter settings. Hence, training this component was not a difficult part. The selection of a suitable candidate is a crucial part of the proposed work. There are many possible candidates for a chosen system action. All the sentences for the selected system action do not have all the contextual information present in the sentence to deliver it to the user. Hence, the candidates

have filtered depending on the availability of the contextual information present in the sentence. The words about the contexts have de-lexicalized while training the data. The de-lexicalized terms represent the associated entities with the intent of the user. The list of candidates has filtered when all the contextual information requested by the user corresponding to a specific intent is present in the candidate. When it is impossible to find a candidate that satisfies the criteria, then the system has designed to take a fall-back action to choose a candidate in which at least any one of the entities is present. Then, the system can choose any one of the variants of dialogue randomly to deliver it to the user. It is a quite time-consuming process for the system, which led to the responsiveness of the system slow. This effect has also created a negative impact on the user's impression concerning the system.

- **Business Logic Integration** The results from the APIs are essential to include in the responses for suggesting the requested information. It is vital in deciding the system action, and also including the results in the delivery of responses. The system has designed to handle APIs before predicting the system action. When there is not any relevant information according to the user's specifications, then the system has developed to perceive the user input as no availability for the particular request, for which the system has to predict any one of the system actions that are related to 'nooffer.' Because of the complexities associated with the statistical learning approach mentioned above, the system has predicted the wrong actions. It has confused the user, and it also created a negative impact on users. Hence, APIs have used to handle this situation by providing the user as 'Null' option to clarify the user irrespective of wrong system action.

  The API results are useful to replace the de-lexicalized words in the chosen candidate response. The appropriate entities have replaced by each relevant API result. The API results have either single or multiple results. In the selected candidate response, the de-lexicalized word for the specific entity can occur one or more times. According to the obtained API results, the de-lexicalized words in the chosen candidate response have replaced by appropriate API results.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In the proposed work, the Interactive Conversational Agents have built for a domain-specific case of restaurant reservations using a chat interface. The proposed work remains a prototype to support customer service employees by engaging in accomplishing certain domain-specific tasks. The developed agents can handle the conversations in English irrespective of gender, race, and geographic location based dialogues in the conversation. It reduces the pressure in the call center environment, where the customer service employees have to handle the conversations with less negative emotions. Thus the developed agents have designed with a motive for 24/7 service to handle domain-specific tasks. The proposed work has designed with business and research focus to provide a cost-effective solution for customer service organizations to increase revenue growth, and also to handle the pressure of call center employees. The developed ICAs have designed with two versions, one is fully automatized, and another is semi-automatized. The automatized agents have designed such that it has the flexibility to adapt for multi-domain tasks in the future, whereas the semi-automatized agents have assisted by pre-determined rules to take an appropriate turn when an automatized agent has predicted wrong actions for a dialogue turn in a conversation.

The design of reliable ICA is a challenging task to hold longer conversations, and giving an appropriate response for the contextual information given by each user at each dialogue turns in a conversation. The realization of human-like ICAs is a difficult task, which might satisfy or frustrate the user depending on its response. In a realistic situation, it is hard for call-center employees, who are experts in their field, to satisfy certain customers, who expect a high delivery rate. The difficulties in designing such a system include perceiving the user requests correctly, and the system should have the capability to track the updates in the user requests. Then it has to take a correct action corresponding to the present contextual information requested by a user, and at last, the system has to deliver a natural response for the predicted action to a user. The system must also handle the technical errors while perceiving the user requests, and deliver an appropriate message to a user. Also, the system must have the capability to include the results from API in the selected responses to fulfill the user's requests. As well as the technical errors, and the business errors while handling API should be handled. Despite all these complexities, the user expects to have an interaction human-like. Thus, ICAs have to surpass all these difficulties.

6.1.1 Aspects in design techniques

In the proposed work, it has tried to produce an automated response by perceiving the user inputs using the NLU component by extracting the necessary semantic information. The following system action of corresponding semantic information has predicted using the proposed architecture of the DM component after tracking the current state of the dialogue. It has then used by the end NLG component to make a list of candidate responses for the chosen system action. Then, a candidate response has selected based on the availability of detected slots in the responses. An open-source dataset involving human-human conversations has chosen to train the various components of the model, which has shown that the automatized natural responses are possible to generate with less human effort. The system has designed to ask the user to provide additional information for the previous request, or else the system also can take its own decision for the specification which the user has not specified. Also, the system is capable of taking a different dialogue turns for the requests made by the user. The system can update the specifications given by the user, and it keeps track of the dialogue state until the end of the conversation unless there are not any technical issues. The human-human conversation dialogues have used to train the deep learning models of the proposed strategy for delivering the response after choosing one from a list of selected responses. In the semi-automatized interactive agent, the design techniques are the same, but additionally, it has assisted when a system has taken a wrong dialogue turn for each user request.

Despite the problematic factors mentioned, the system has designed such that it can be available for service. The challenges which have faced in the deployment have discussed below as follows,

- **Service reliability** The system has designed in a way to behave like a human while holding the conversations. Irrespective of the challenges for ICAs in holding the customer service conversations, the proposed system is capable of holding multiple dialogues turns for user's requests and is capable of delivering the requested information for the utmost of the requests made by the user. The proposed system can adapt to a multi-domain setting in the future. The system also has taken the wrong dialogue turns, which has led to the wrong information given to the user. The full automatized agent still requires improvement in the design techniques to provide a reliable service.

  Whereas the semi-automatized interactive agent can handle the wrong predictions by correcting itself using the pre-determined rules, and it can deliver an appropriate response. Though both the system has designed using the natural dialogues, the dataset which has used lacks the sentence structure. Hence, the system lacks complete naturalness to give the feeling of human-like conversation to the user.

- **Co-operative principle in the conversation** Quality, Quantity, Relation, and Manner are the four aspects that are necessary to hold a meaningful conversation. *Quantity* - The system can provide an informative response with the business logic included in the responses for each user's requests. Still, for a few dialogues, the system also responds with a dialogue that does not have contextual information related to the user's request, or it responds with an improper sentence structure. *Quality* - The system is capable of responding with information for user-requested input or self-decided information if the user has not given his specifications, conditioned that the user has specified the at least one of the requirements for his request in each dialogue turn. *Relation* - The system has also designed to respond with relevant contextual information for each user request in a conversation. The system solely depends on the dataset, which has used while building the system. Due to the lack of specific information in user requests in all the sentences, the system might not able able to provide the exact information. Still, it can provide relevant information to the user's requests. Also, the fully automatized interactive agent

is not able to provide relevant information for each dialogue in a conversation, whereas the semi-automatized agent can handle it using the rules.

- **Responsiveness** The system is capable of delivering automated responses to the user, instead of standard templates or messages. It has provided naturalness in the system, but still, the system cannot hold a human-like conversation. The sentence structure delivered by the system is not fully structured, which includes the repetition of certain letters, improper beginning of a sentence. The system can hold a long conversation, but the experience of relation of response cannot be guaranteed. Also, the system expects to provide explicit information by the user at each time of their requests.

6.1.2 Objective of the proposed work

The main objective is to build Interactive Conversational Agent for a domain-specific case included with business logic using a strategy for Dialogue Management. The interactions must involve API results in the responses, and the technical errors within that API have handled. The tasks which have achieved in developing an interactive agent has described below,

- **Selection of Dataset** It is significant for a deep learning model to train a resourceful of the dataset to deliver reliable service and natural responses. Hence, a human-human conversation dataset has chosen to build interactive agents for a restaurant domain scenario. Though the dataset has Multi-domain dialogues, only dialogues specific to the restaurant domain has chosen, which is one of the objectives. The system has the flexibility for a multi-domain setting. Hence, the multi-domain dialogues help in the future for building multi-domain ICAs

- **Selection of tools** Rasa open-source framework has chosen to build ICAs, which has advanced methods to interpret the user requests. Keras - an open-source machine learning library, which has multiple backends such as Tensorflow, Theano has used to implement a custom strategy for Dialogue Management with the flexibility of adaptive setting for multi-domain dialogues. In the proposed work, the proposed strategy has used only for domain-specific settings. Also, the pre-determined rules have implemented for semi-automatized ICA for Dialogue Management along with the proposed strategy.

- **NLP techniques** Rasa open source tools have used for interpreting the user requests', and the progressive sequence to sequence learning network has proposed for Dialogue Management. The proposed strategy has built with a motive for multi-domain settings, which has inspired from convlab, an open-source framework for building Conversational Agents by Microsoft Research. Additionally, progressive sequence to sequence learning with self-attention mechanism has proposed to implement the NLG component. As the proposed strategy did not provide satisfactory results, sequence to sequence learning technique has used for the NLG component, which has developed by Google.

- **Business Logic Integration** The results from APIs have included in the responses to provide a suggestion for each user request. Also, the technical errors within that API have handled.

- **Evaluation** The developed ICAs have evaluated by human users by interacting with the agents, and each user has requested to fill out the questionnaire about their experience while interacting with the system. The evaluation has performed by both professional and naive participants. The portfolio of the combined results has included in this proposed work

6.1.3 The challenges in the proposed work

Dataset

The chosen dataset for training the system has human-human conversations, but the dialogues involved in the conversation do not have proper sentence structure.

- **NLU** Though the sentence structure for the NLU component does not have a great impact on perceiving the contextual information, and the system expects that the user has to deliver the contextual information in the dialogues as specified as in the annotation of the dataset. The system can handle typo errors, and also it can handle multiple languages except for the contextual information, which has used for annotation. Also, the resource in the dataset is not vast enough to interact naturally in the conversations. Instead of giving generalized input, such as 'yes' or 'no,' the system expects the user to provide some contextual information in each dialogue. Additionally, the annotation in the dataset has limitations such as certain contexts have annotated improperly, and also certain contexts lack annotation.

- **DM** The dataset has structured in a way that the system action semantic frames have followed for the corresponding user input semantic frames. The dataset which has created from the original MultiWOZ corpus data specific to Restaurant domain consists of repetitive user input semantic frames, and the corresponding system action has varied for each repetition.

  Also, the available API database influence the system action semantic frames for the restaurant information, whereas it does not influence the user input semantic frames. The user input semantic frames and the system action semantic frames are appropriate to the API database information, and the proposed strategy of the deep learning model has trained by learning the dialogues by itself according to the hyper-parameter settings. It means that irrespective of the availability of user-specific restaurants in API database information, the system perceives the user input before checking the availability of the user-specific restaurants in the API database. It has led the system to predict the system action wrong, as irrespective of the availability of the information in API database, the user intent is similar, but the following system action varies.

  As the system has trained by self-learning of the dialogues, the system predicts the action based on the trained weights for the perceived input vector state representations. Hence, a separate dataset has created manually after many trials, in which the variants of user input semantic frames and the respective system actions have balanced. Additionally, the user intent 'Noavailablerestaurant,' and the corresponding system actions related to 'nooffer' have included. It means that the user intent has taken as 'Noavailablerestaurant' when there are no user-specific restaurants available in the API database information.

- **NLG** The sentence structure in the dataset lacks integrity. It had a great impact during the evaluation since most of the users who have evaluated the system had a feeling that it was hard to perceive the information from the automated responses. The responses were not more human-like. Also, not all the sentences have all the contextual information for the intent of a system action which it should deliver to the user. Despite these challenges, the system checks the availability of the contextual information from the list of chosen candidates for system action. Then, it has to select the best one according to the contextual information specified by the user.

DM - The proposed strategy

The proposed work has designed for a multi-domain setting, such that the system can be trained on several domains progressively. According to the specifications of the progressive columns while training, the number of domains can be chosen. This strategy uses a transfer learning approach, which uses the knowledge for training from one domain to another domain, which is related to each other. This work have a great influence in the future for the automation of ICAs with the capability of handling a multi-domain scenario. Due to the limitations of the dataset, and its self-learning capability, the system takes a wrong dialogue turn in a conversation for certain cases. But the system might able to handle it when the requests have repeated, or the requests have rephrased, and additional contextual information has provided to the system.

The system has improved by fine-tuning it's hyperparameters while training the model. The hyper-parameters, which have fine-tuned, are the optimizers, learning rate, the batch size for training, epochs for training, and the regularization. Also, the dataset for DM influenced in predicting the system actions. Hence, the dataset of various distributions has created manually to have unique input combinations. As the system actions are limited, a balanced distribution of system action which has a relation to the perceived user intent has fed while training the deep learning network.

6.2 Future Work

Though the proposed work focused only on the domain-specific scenarios, the developed DM architecture is also capable of training for the multi-domain scenario using the transfer learning technique. The suitable dataset has to provide while training the proposed architecture of Progressive encoder-decoder incrementally. The dataset has a great influence on perceiving the user's requests, predicting the system action, and generating an automated response along with API results in the responses. Hence, it is significant to create a dataset that is resourceful, and applicable for general, and special cases of dialogue turn uttered by the user. The annotation has to be implemented properly without any errors, and annotation of large data with much contextual information helps in perceiving the user's requests appropriately. As the proposed DM architecture has the impact of data from which it has trained, it requires a balanced distribution of data. The dataset for the NLG component has a great influence on the user experience while determining the quality of the product. Hence a dataset has to be created with a proper sentence structure to deliver a response, such that it does not mislead the user. A negative impact has seen from the user due to a lack of delivering a proper sentence and providing an inappropriate response which the user has not expected. Thus, the dataset has also impact on the flexibility of the system to handle different kinds of user's requests.

Also, to overcome the problem of false predictions in the system actions, and attention mechanism can be employed associated with the proposed DM architecture to generate an appropriate action for the perceived user input, and also regarding the previous actions to attain the end-goal of the user. The proposed DM architecture relies on the source of the dataset. Hence, it has to be reduced by integrating few-shot learning techniques, such that the system can generalize well from the fewer dialogues. The system's capability can be improved by interacting with the user-simulated dialogues, and also learning from natural dialogues from the user, and re-training the developed architecture with those additional dialogues as well. The proposed NLG architecture generates improper sentence sequences. Hence, it is optimal to modify the dataset and enhance the associated attention mechanism in the future. Thus the proposed work remains as a prototype with the challenging tasks, which can be enhanced with the ideas mentioned above in the future.

Appendix A

API Restaurant database sheet

		Area						
		AC	AE	AW	AN	AS		Total
Indian	PC	3	0	0	1	0	4	
	PM	0	2	1	1	0	4	
	PE	6	2	5	0	1	14	
		9	4	6	2	1		22
European	PC	0	0	0	0	0	0	
	PM	2	0	0	0	0	2	
	PE	2	0	1	1	0	4	
		4	0	1	1	0		6
Spanish	PC	1	0	0	0	0	1	
	PM	1	0	0	0	0	1	
		2	0	0	0	0		2
British	PM	4	0	1	0	0	5	
	PE	3	1	2	0	0	6	
		7	1	3	0	0		11
International	PC	0	1	0	0	0	1	
	PM	2	0	0	0	0	2	
		2	1	0	0	0		3

Price Range

(a)Available Restaurants with the specific area and price range,
AC- Area Centre, AE- ARea East, AW- Area West, AN-Area North, AS- Area South
PC- Price Cheap, PM- Price Moderate, PE- Price Expensive

		Area						
		AC	AE	AW	AN	AS		Total
Italian	PC	3	0	1	1	0	5	
	PM	2	1	1	0	1	5	
	PE	4	0	0	0	1	5	
		9	1	2	1	2		15
Modern European	PC	1	0	0	0	0	1	
	PM	2	0	0	0	1	3	
	PE	1	0	0	0	0	1	
		4	0	0	0	1		5
chinese	PC	3	0	0	0	1	4	
	PM	3	0	0	1	0	4	
	PE	4	1	0	2	2	9	
		10	1	0	3	3		17
Mexican	PE	0	0	0	0	1	1	
		0	0	0	0	1		1
Asian Oriental	PC	2	0	0	0	0	2	
	PM	1	0	0	0	0	1	
	PE	1	0	0	1	0	2	
		4	0	0	1	0		5

Price range

(b)Available Restaurants with the specific area and price range,
AC- Area Centre, AE- ARea East, AW- Area West, AN-Area North, AS- Area South
PC- Price Cheap, PM- Price Moderate, PE- Price Expensive

			Area					
			AC	AE	AW	AN	AS		Total
Gastropub		PC	0	0	0	0	0	0	
		PM	1	0	0	0	0	1	
		PE	2	1	0	0	0	3	
			3	1	0	0	0		4
Turkish	Price range	PM	2	0	0	0	0	2	
		PE	1	0	0	0	0	1	
			3	0	0	0	0		3
African		PE	1	0	0	0	0	1	
			1	0	0	0	0		1
French		PE	1	0	0	1	0	2	
			1	0	0	1	0		2
Korean		PE	1	0	0	0	0	1	
			1	0	0	0	0		1
Vietn -amese		PE	0	0	1	0	0	1	
			0	0	1	0	0		1
Lebanese		PM	1	0	0	0	0	1	
			1	0	0	0	0		1

(c)Available Restaurants with the specific area and price range,
AC- Area Centre, AE- ARea East, AW- Area West, AN-Area North, AS- Area South
PC- Price Cheap, PM- Price Moderate, PE- Price Expensive

		Area						
			AC	AE	AW	AN	AS		Total
mediter -ranean	Price range	PC	1	0	0	0	0	1	
			1	0	0	0	0		1
Japa -nese		PE	1	0	0	0	0	1	
			1	0	0	0	0		1
Portu -guese		PC	1	0	0	0	1	2	
			1	0	0	0	1		2
North American		PE	1	0	0	0	0	1	
			1	0	0	0	0		1

(d)Available Restaurants with the specific area and price range,
AC- Area Centre, AE- ARea East, AW- Area West, AN-Area North, AS- Area South
PC- Price Cheap, PM- Price Moderate, PE- Price Expensive

Bibliography

[1] Ulrich Gnewuch, Stefan Morana, Alexander Maedche: *Towards Designing Cooperative and Social Conversational Agents for Customer Service*, Thirty-Eighth International Conference on Information Systems, South Korea, 2017

[2] Danielle van Jaarsveld, Winifred R.Poster: *Call Centers: Emotional Labor over the Phone*, University of British Columbia - Vancouver, pp. 153-173, 2014

[3] Abhishek Singh, Karthik Ramasubramaniam, Shrey Shivam: *Building an Enterprise Chatbot*, New Delhi: Delhi, India, 2019, ISBN-13: 978-1-4842-5033-4

[4] Susanne Biundo and Andreas Wendemuth: *An Introduction to Companion-Technology*, Cognitive Technologies, Springer International Publishing AG, pp. 1-15, 2017, DOI 10.1007/978-3-319-43665-4_1

[5] Heung-Yeung Shum, Xiaodong He, Di Li: *From Eliza to XiaoIce: Challenges and Opportunities with Social Chatbots*, Microsoft Corporation, Frontiers of Information Technology and Electronic Engineering, 2018

[6] Ian Goodfellow, Yoshua Bengio, Aaron Courville: *Deep Learning*, MIT Press, 2016, http://www.deeplearningbook.org

[7] Thomas Geier and Susanne Biundo: *Multi-level Knowledge Processing in Cognitive Technical Systems*, Institute of Artificial Intelligence, Ulm University, Ulm, Germany, Springer International Publishing AG 2017, DOI 10.1007/978-3-319-43665-4_2

[8] Alex Graves: *Supervised Sequence Labelling with Recurrent Neural Networks*, Studies in Computational Intelligence385, ISBN 978-3-642-24796-5, pp. 1-131, 2012, https://www.cs.toronto.edu/ graves/preprint.pdf

[9] Jianpeng Cheng, Li Dong, Mirella Lapata: *Long Short-Term Memory-Networks for Machine Reading*, Association for Computational Linguistics,Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing:551-561, 2016 https://arxiv.org/pdf/1601.06733.pdf

[10] Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston: *StarSpace: Embed All The Things!*, Thirty-Second AAAI Conference on Artificial Intelligence, Facebook AI Research, 2017, https://arxiv.org/pdf/1709.03856.pdf

[11] Cheongjae Lee, Sangkeun Jung, Kyungduk Kim, Donghyeon Lee, and Gary Geunbae Lee: *Recent Approaches to Dialog Management for Spoken Dialog Systems*, Journal of Computing Science and Engineering, 4(1): 1-22, March 2010.

[12] Svetlana Stoyanchev, Michael Johnston: *Knowledge-Graph Driven Information State Approach to Dialog*, Interactions Corporation, The Workshops of the Thirty-Second AAAI Conference on Artificial Intelligence, 2018

[13] Jason D. Williams, Kavosh Asadi, Geoffrey Zweig: *Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning*, Microsoft Research, Association for Computational Linguistics (ACL), 1:665-677, 2017, https://arxiv.org/pdf/1702.03274.pdf

[14] Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao *ConvLab: Multi-Domain End-to-End Dialog System Platform*, Microsoft Research, Tsinghua University, Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 1:64-69, 2019, https://arxiv.org/pdf/1904.08637.pdf

[15] Vladimir Vlasov, Akela Drissner-Schmid, Alan Nichol: *Few-Shot Generalization Across Dialogue Tasks*, Rasa Opensource, 32nd Conference on Neural Information Processing Systems, abs/1811.11707, Montréal, Canada, 2018

[16] Monika Schak, and Alexander Gepperth: *A Study on Catastrophic Forgetting in Deep LSTM Networks*, Part-II Proceedings in 28th International Conference on Artificial Neural Networks , University of Applied Sciences Fulda, Germany, September 17–19, 2019

[17] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le: *Sequence to Sequence Learning with Neural Networks*, Proceedings of the 27th International Conference on Neural Information Processing Systems - 2:3104–3112, 2014, https://arxiv.org/pdf/1409.3215.pdf

[18] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, Raia Hadsell: *Progressive Neural Networks*, Google DeepMind, London, UK, 2016, https://arxiv.org/pdf/1606.04671.pdf

[19] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, Joelle Pineau: *A Survey of Available Corpora for Building Data-Driven Dialogue Systems*, University of Montreal, McGill University, abs/1512.05742, 2017, https://arxiv.org/pdf/1512.05742.pdf

[20] Paweł Budzianowski, Tsung-HsienWen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic: *MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling*, Association for Computational Linguistics, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing:5016–5026, 2018, https://www.aclweb.org/anthology/D18-1547.pdf

[21] Marc Hassenzahl and Noam Tractinsky: *User experience - A research agenda*, Behaviour & IT, Darmstadt University of Technology, Darmstadt, Germany, and Ben-Gurion University, Israel, 2006

[22] Martin Schrepp, Theo Held, Bettina Laugwitz: *The influence of hedonic quality on the attractiveness of user interfaces of business management software*, Interacting with Computers, pp. 1055-1069, 2006

[23] Diederik P. Kingma, Jimmy Lei Ba: *ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION*, International Conference on Learning Representations, abs/1412.6980, San Diego, 2015, https://arxiv.org/pdf/1412.6980.pdf

[24] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov: *Dropout: A Simple Way to Prevent Neural Networks from Over*

tting, Journal of Machine Learning Research, 15:1929-1958, 2014,

[25] Matthew D.Zeiler: *ADADELTA: AN ADAPTIVE LEARNING RATE METHOD*, ArXiv, 2012, https://arxiv.org/pdf/1212.5701.pdf

[26] Andrei Malchanau, Volha Petukhova and Harry Bunt: *Multimodal dialogue system evaluation: a case study applying usability standards*, 9th International Workshop on Spoken Dialogue System Technology, 579:145-159, ISBN: 978-981-13-9442-3, DOI: 10.1007/978-981-13-9443-0_13, 2019

[27] William J. Tastle a, Mark J. Wierman: *Consensus and dissention: A measure of ordinal dispersion*, International Journal of Approximate Reasoning, 45:531-545, 2007