DEVELOPING AUTOMATIC SPOKEN DIALOGUE SYSTEM FOR AFAN OROMO USING DEEP LEARNING

Gobena Gudeta Jaleto

Thesis Submitted to the department of Computer Science and Engineering

School of Electrical Engineering and Computing

Office of Graduate Studies

Adama Science and Technology University

Advisor: Teklu Urgessa(PHD)

Feb, 2023

Adama, Ethiopia

DEVELOPING AUTOMATIC SPOKEN DIALOGUE SYSTEM FOR AFAN OROMO USING DEEP LEARNING

Gobena Gudeta Jaleto

Thesis Submitted to the department of Computer Science and Engineering

School of Electrical Engineering and Computing

Office of Graduate Studies

Adama Science and Technology University

Advisor: Teklu Urgessa(PHD)

Feb, 2023

Adama, Ethiopia

Approval Sheet

I/we, the advisors of the thesis entitled "**Developing Automatic Spoken Dialogue System for Afan Oromo using Deep learning** and developed by **Gobena Gudeta Jaleto**, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Major Advisor	Signature	Date
Co-advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis **Gobena Gudeta Jaleto** have read and evaluated the thesis entitled "**Developing Automatic Spoken Dialogue System for Afan Oromo using Deep learning"** and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering

Chairperson	Signature	Date
Internal Examiner	Signature	Date
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis are contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head	Signature	Date
School Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

Declaration

I hereby declare that this Master Thesis entitled "**Developing Automatic Spoken Dialogue System for Afan Oromo using Deep learning"** is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation.

_____	_____	_____
Name of the Student	Signature	Date

Recommendation

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Developing Automatic Spoken Dialogue System for Afan Oromo using Deep learning**" prepared under my/our guidance by **Gobena Gudeta Jaleto** submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of the revised version of the thesis to the department following the applicable procedures.

_____	_____	_____
Major Advisor	Signature	Date

Acknowledgement

First and foremost, I would like to thank my almighty God for giving me the strength, knowledge, ability and opportunity to undertake this research study and to persevere and complete it satisfactorily. Without his blessings, this achievement would not have been possible.

Next, my deepest gratitude goes to my advisor Teklu Urgessa (PHD) for his continuous follow-up, guidance, suggestions and his constructive comments were very essential in the successful completion of my thesis.

I would like to thank the Soreti city mall business owners for their cooperation during my data collection.

I would like also to thank my family specially my wife Roman Mohammed for supporting in all of my study and my sister Kedija Gudeta and her Husband Usheto Tessa for giving me financial and moral support to achieve my thesis work.

Last but not least, I would like to thank the Academia at ASTU (Adama Science and Technology University) for their contribution directly and indirectly for success of my study

Table of contents

List of Tables

List of Figures

List of Acronyms

AOSDS	Afaan Oromo Spoken Dialogue System
ASR	Automatic Speech Recognition
TTS	Text To Speech
NLU	Natural Language Understanding
NLG	Natural Language Generation
POMDP	Partially Observable Markov Decision Process
HMMs	Hidden Markov Models.
PyDial	A Python-based Domain-Independent
CRF	Conditional Random Fields
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
CER	Character Error Rate
NLP	Natural Language Processing
WER	Word Error Rate

Abstract

Dialogue system is the software that communicates with a human user through natural language on a turn-to-turn basis. The main purpose of a dialogue system is to provide an interface between a user and a computer-based application. Afan Oromo is one of the Cushitic family languages in Ethiopia which has largest mother tongue speakers in East Africa next to Kiswahili. As far as our knowledge, this language did not have automatic spoken dialogue system which can be implemented in different domains. This is one of the motivations to conduct this research.

Here in this study, we have developed prototype of multi domain Afan Oromo Spoken Dialogue System (AOSDS) using Deep Learning algorithms to assist visitors of public business mall. The Statistical Approach was implemented to develop the components of our AOSDS. The freely available research toolkits and software like PyDial, NLTK and Google cloud platform were utilized for Text and Speech data preprocessing, and ASR, TTS and dialogue manager development. We have developed cloud-based speech recognizer using Audio from 16 (8 Males and 8 Females) speakers which is around One hour long. To develop language model for A/O ASR we have used more than 5000 sentences which are collected from different sources and domains. Obviously, this is very low corpus size to develop ASR which has significant effect on performance of SDS.

Like other SDS systems, the performance of our AOSDS relies on the modules which are mandatory for SDS development. Among these components, the error rate of Semantic decoder for proposed system was 25.5% and the error rate of the ASR was 36.5%. As a result, the word error rate of our AOSDS is 38.6%. Lack of language resources such as text and speech data, unavailability of toolkits and hardware resources for this study are the major challenges of this study. With all these limitations we have found the promising result for the possibility of developing Afan Oromo spoken dialogue system.

Key Words: - Spoken Dialogue System, Deep Learning, Statistical approach, Speech Recognition, Language Understanding, Dialogue Management, Language Generation, Speech production.

CHAPTER ONE

1. Introduction

1.1 Background of the study

The dialogue is an interaction between two or more agents that communicate with each other through natural language. The dialogue fulfills social needs such as exchange of information, building trust and relationships or coordination and collaboration. The ability to engage and maintain a dialogue is something we start developing through early childhood Pierre, Lison (2013). One of the fundamentals of a dialogue is that it is a collaborative activity with two or more agents that share information with each other. The agents collaborate to obtain a mutual understanding of each other's desires and goals.

Dialogue system is the software that communicates with a human user through natural language on a turn-to-turn basis. The main purpose of a dialogue system is to provide an interface between a user and a computer-based application Michael F. McTear (2002). The sophistication of different dialogue systems varies immensely, ranging from systems that are functioning on a question-answer basis, to systems that strives to achieve a human-like performance of conversational interaction.

The goal of conversational dialogue systems is not to handle all kinds of human dialogue due to complexity of the human language. Allen et al. Galescu, Amanda Stent (2001) claims that the construction of a system with full understanding of the human language is such a difficult task that it will not be achieved in the foreseeable future. A constraint for most dialogue systems is therefore that the systems are focused on accomplishing specific and concrete tasks Gabriel, Skantze (2007).

Spoken dialogue system (SDS) is one of the technologies that can be used to provide important information on a variety of tasks. According to Martin, James H. (2006), Spoken Dialogue System can be defined as a computer program or an agent that converses with human users using natural language to perform a task. The tasks can be making travel

arrangements, answering questions about weather or sports, routing telephone calls, acting as a general telephone assistant. In the recent years, SDS has been one of the growing research and development interest area due to various motivations such as its ease of use, its favorability for different users and its capacity to help developing countries to growth. The technology of SDS complies with the general software interface principle that they should be simple to use for anyone.

As indicated by Lewis, Clayton (1985), any software interface designed for people should not pose any difficulty to learn and should contain features users really need in their task and should be easy to use. From the users' perspective, these interfaces are the points of interaction with the machine for some useful purpose and therefore should be intuitive and simple to use.
In other words, to benefit from the machine's storing and processing information capabilities, interfaces should be simple enough As Afzal, Saduf (2013) describes, one approach to achieve the effort of making software interface simple, is through speech interface where users simply speak to a machine to get information, or to get some task done for them. Afzal, Saduf (2013) Argue, speech interface applies spoken language which is the most natural, efficient, and flexible way of conveying information among human beings. For example, if we consider a keyboard input, it requires a typing skill.

The Graphical User Interfaces (GUI) needs reading and appropriate actions that require a prior training. To communicate through speech, a user doesn't require training like the keyboard or should be acquainted with the GUI. Most software interfaces require learning the interface before using them. Speech interfaces enable easy human-machine interaction and thereby assist the efforts to make the software interface to be easy. Hence, SDS can be designed for expert and non-expert computer users Afzal, Saduf (2013).

There are conditions and potential situations which SDS is suitable for expert users with an experience of interacting with machines as well as for novices and non-expert users. One example situations for advanced users is hands free driving, which enables drivers to have their emails read, make a reservation for a hotel or find the nearest hotel or restaurant through speech while driving. Expert users can be relieved from routine tasks while engaged in some

other tasks. Bell, Linda (2014) also points out, non-experts, computer novices and aged people are potential user groups for the present and future generation of spoken dialogue systems, since little or no training is needed to use speech based software. Spoken dialogue systems designed to provide information on current market pricing and health related request can work for users of all levels of expertise. Applications of Spoken Dialogue Systems can also benefit system designers for educational purpose, assistive technologies for the physically disabled and could be beneficial in oral societies Bell, Linda (2014); Kuun, Christiaan (2008).

As reported by Kuun, Christiaan (2008), there is a strong belief that SDS will have a considerable impact in the developing world. This is because firstly, speech based access to information may enable illiterate or semiliterate people to take part in the information age. Second, telephone networks (especially mobile/cellular networks) are spreading rapidly. This can serve as infrastructure for SDS, and third, the strong oral culture that exists in many traditional societies is likely to yield more acceptable than text-based or GUI based sources.

Kuun, Christiaan (2008) Also explains that if a developing nation leapfrogged1 to a newly emerged ICT, it would then be exposed to great potential in alleviating poverty and securing economic growth, as well as the possibility of surpassing developed and industrialized countries in economic development. One example of such technological leapfrogging is the shift of people in developing world from no land line phone to mobile phone Fong, Michelle W. L. (2009).

Mobiles are becoming ubiquitous. According to the figures on UN data, Mobile cellular subscriptions per 100 inhabitants in Ethiopia, grow from 2.37 in 2008 to 27.25 in 2013. A technology already available in the hands of millions of people can serve as an infrastructure to implement SDS and thereby quality information can be accessed and may facilitate technological leapfrogging

Major Components of SDS are:

Language Understanding: Traditionally, domain identification and intent prediction are framed as utterance classification problems, where several classifiers such as support vector machines and maximum entropy have been employed. Then slot filling is framed as a word sequence tagging task, where the IOB (in-out-begin) format is applied for representing slot tags, and hidden Markov models (HMM) or conditional random fields (CRF) have been employed for slot tagging R. Pieraccini, and E. Levin (1992).

Dialogue Management: - A partially observable Markov decision process (POMDP) has been shown to be beneficial by allowing the dialogue manager to be optimized to plan and act under the uncertainty created by noisy speech recognition and semantic decoding Thomson, and J. D. Williams (2013). The POMDP policy controlling the actions taken by the system is trained in an episodic reinforcement learning (RL) framework whereby the agent receives a reinforcement signal after each dialogue (episode) reflecting how well it performed S. Sutton, et al. (1998). In addition, the dialogue states should be tracked in order to measure the belief of the current situation during the whole interaction Thomson, and J. D. Williams (2013).

Natural Language Generation: There are two NLG approaches, one focuses on generating text using templates or rules (linguistic) methods, another uses corpus-based statistical techniques Alexander I Rudnicky (2002). Bohus, D. and Rudnicky(2005) Showed that stochastic generation benefits from two factors: 1) it takes advantage of the practical language of a domain expert instead of the developer and 2) it restates the problem in terms of classification and labeling, where expertise is not required for developing a rule-based generation system.

Afaan Oromo is one of the Cushitic family languages in Ethiopia which has largest mother tongue speakers in East Africa next to Kiswahili. This local language did not have automatic spoken dialogue system which can be implemented in different domain such as education, health, business, and government Lewis, Clayton (1985).

Therefore this under resourced language has to have this spoken dialogue system technology to provide users with the necessary information they need in a timely manner, access different services and resources in their native language.

1.2 Motivation of the study

In the recent years, SDS has been one of the growing research and development interest area due to various motivations such as its ease of use, its favorability for different users and its capacity to help developing countries to growth.

The contribution that this spoken dialogue system provide for both business owner as part of business improvement and user as well as part of problem solution motivates me to conduct a research on this topic. In my thought it is a best opportunity for those who couldn't write have a sight problem and also new for the agent they are going to request or gain services motivates me to conduct a research.

1.3 Statement of the Problem

Spoken dialogue is easy and effortless for humans, but modeling it in artificial way is difficult Bell, Linda (2014). An attempt to design an SDS for under-resourced languages like Afan Oromo is more challenging. According to Mohammed- hussen Abebeker; Taye Girma (2012) this is because human language technology in Ethiopia is characterized by, among other things, lack of language resources. Due to the lack of language-resource, there is also a problem with the availability of off-the-shelf language technology components that a spoken dialogue system requires (except the text-to speech unit for this prototype).

Unlike the resourceful languages like Germany, English and other developed countries where commercial or open-source language technology components which can be readily plugged into a dialogue system Rosenfeld, Roni (2010) are available, important components of spoken dialogue system such as Automatic speech Recognition (ASR) Natural language understanding, Natural language generation, Text to speech(TTS) and dialogue management are not available [5].

Few attempts are there to develop some important components of SDS such as ASR Teferi Kebebew. (2010) Kassahun Gelana. (2010) and TTS T. Paek and R. Pieraccini (2008) separately. Other components such as Natural language understanding, natural language generation and dialogue manager are not available for Afan Oromo.

No Afan Oromo spoken dialogue system research has been conducted yet, as far as our knowledge. Thus, this Afan Oromo Spoken Dialogue System has been designed and developed from the scratch, allowing users to interact with the system in their own language and receive a response through speech.

The main challenges faced in this study were the lack of resources and data to develop an automatic spoken dialogue system for Afan Oromo due to limited access to audio recordings of the language and its lack of documentation. This difficulty in creating a system that can accurately recognize and respond to spoken dialogue was compounded by the lack of expertise in the field of natural language processing and spoken dialogue systems with regards to Afan Oromo, making it difficult to develop a system that can accurately interpret and respond to spoken dialogue.

1.4 Research Questions

The study examines the spoken dialogue system's performance in the setting of Afan Oromo language using deep learning techniques. The research aims to respond to the following Questions.

- What methods and techniques can be used to prepare text and audio data/corpus for AOSDS?
- What approaches and methods are preferable for design and development of models in AOSDS?
- How to design and implement dialogue manager for AOSDS?
- What are the challenges and limitations to develop AOSDs?

1.5 Objectives of the Study

1.5.1 General Objective

The general Objective of this study is to develop multi-domain spoken dialogue system for Afan Oromo using Deep learning.

1.5.2 Specific Objectives

The objective of the study was to achieve the following specific objectives and establish a comprehensive understanding of the various aspects of the study. It also aimed to be directed toward answering the research questions and achieving the general purpose of the study.

Specific objectives of the study are: -

- Prepare text and audio corpus from a variety of sources by organizing into corpus, labeling and then categorizing according to its content.
- Develop acoustic, lexical and language models for speech recognition systems to improve the accuracy of the system's recognition of spoken words.
- Design database table to store questions and answers in structured format.
- Design and implement dialogue manager for AOSDS that understand the customer intent, offer appropriate response and manage the conversation flow.

1.6. Scope and Limitations of the Study

Here under this topic the major limitations and challenges we have faced to develop this Afan Oromo spoken dialogue system (AOSDS) and the scope of our study has briefly described in the following sub-topics.

1.6.1. Scope of the study

The scope of this research is designed to conduct Spoken dialogue system for selected mall (business center) to provide necessary information for the customer based on their request using Afaan Oromo speech. This study is more of designing the dialogue manger and the other basic components of spoken dialogue systems.

More attention was given to explore methods, approaches and algorithms to design a strong dialogue manager. A prototype Implementation will be a domain of providing services based on requested information for a selected business center in Adama Town using Afaan Oromo with a focus of testing robustness, speed and task completion of the dialogue manager.

1.6.2. Limitations of the Study

The study of this research has faced its own limitation during implementation; these limitations have a great impact on our desired goal. Very limited availability and access to documents related to the SDS implementation for AOSDS, limited literature and research on SDS in Ethiopian context for reference, time constraint of some interviewees and survey respondents due to busy office works and absence of a very important tools and hardware are some of the limitations for the successful accomplishment of this research.

If this SDS was developed with enough resources, it solves our problem of using Natural Language to communicate with the computer. Due to the mentioned limitations we are enforced to limit our scope depending on the available tools, data and hardware resources.

1.7. Significance of the study

The implementation of SDS software has been continuing to grow in the public and private or Business sectors. The expansion of this improves usability and user satisfaction by imitating human behavior Bell, Linda (2014). It addresses the features of a human-to-human dialog (e.g. sub dialogues and topic changes) and aims to integrate them into dialogue systems for human-machine interaction.

Often, (spoken) dialogue systems require the user to adapt to the system because the system is only able to understand a very limited vocabulary, is not able to react to topic changes, and

does not allow the user to influence the dialogue flow. Mixed-initiative is a way to enable the user to have an active part in the dialogue instead of only answering questions. However, the mere existence of mixed-initiative is not sufficient to be classified as a natural dialogue system. Other important aspects include:

- Understand references by analyzing discourse and anaphora
- Natural language generation to prevent monotonous and recurring prompts
- Social behavior (greetings, the same level of formality as the user, politeness)
- Quality of speech recognition and synthesis

The Beneficiaries from this spoken dialogue system are both business owner as part of business improvement and user as well as part of problem solution .it is a best opportunity for those who couldn't write, have a sight problem and also new for the agent they are going to request or gain a service.

1.8. Organization of the thesis

The first chapter of this thesis is introduction about Spoken Dialogue Systems and its components. It also presents the statement of the problem, objective, methodology, scope and limitation of the study and then the significance of the study.

The Second Chapter discusses the theoretical background of spoken dialogue systems and previous research works in the area of Spoken Dialogue Systems. Different approaches attempted to develop SDS and other related works are discussed under this chapter.

The Third Chapter describes the methodology implemented to develop our SDS. general architecture of SDS, Methods, Techniques and Tools utilized for this SDS designing and development are discussed in this chapter.

The Fourth Chapter discusses about the research experiment to develop AOSDS, analysis of the results, discussion on the finding of the study.

The Fifth Chapter provides the conclusion, recommendations and future works drawn from the findings of this study.

CHAPTER TWO

2. Literature Review and Related Works

2.1 Overview

Successful applications of deep learning technologies in the natural language processing domain have improved text-based intent classifications. However, in practical spoken dialogue applications, the users' articulation styles and background noises cause automatic speech recognition (ASR) errors, and these may lead language models to misclassify users' intents Micheal F McTear(2005).

To overcome the limited performance of the intent classification task in the spoken dialogue system, we propose a novel approach that jointly uses both recognized texts obtained by the ASR model and a given labeled text. In the evaluation phase, only the fine-tuned recognized language model (RLM) is used. The experimental results show that the proposed scheme is effective at classifying intents in the spoken dialogue system containing ASR errors. According to Blaise Thomson, and Jason D. Williams. (2013) A common spoken dialogue system consists of the following components:

- Automatic Speech Recognition component (ASR)
- Natural Language Understanding component (NLU)
- Natural Language Generation component (NLG)
- Text-To-Speech synthesis component (TTS)
- Dialogue manager

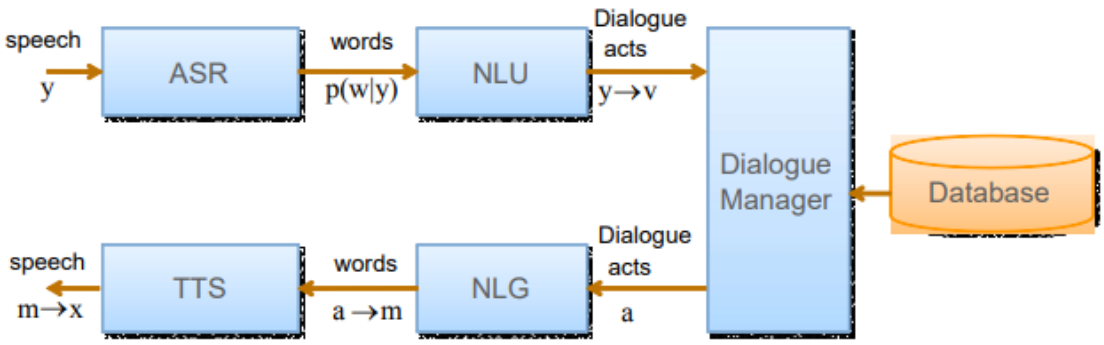


Fig 2.1 SDS Components R. Pieraccini, and E. Levin (1992)

2.1.1. Automatic Speech Recognition component (ASR)

The user's speech input consists of a string of acoustical signals that are converted into a sequence of words by the speech recognition module. To achieve this goal, most **speech recognition** modules use statistical methods – such as Hidden Markov Models (HMMs) [5].

2.1.2. Natural Language Understanding component (NLU)

The Natural language understanding module takes the word sequence delivered by the speech recognizer and analyzes it syntactically, pragmatically and semantically. The aim is to determine the intended meaning of the word sequence. This process is called **parsing**. Usually, the parsing process uses a grammar which describes how words in an utterance are combined Bell, Linda (2014).

2.1.3. Dialogue manager

The decoded information is then sent to the **dialogue manager**. This unit plays a central role in operating the dialog – a difficult task. Very frequently the recognized words delivered by prior modules are fragmented and incorrectly modeled. Background noise, human noises such as sneezes, coughs, unequal emphasis, word accents, inarticulate and incomplete pronunciations, word or syllable recurrences and different acoustical realizations of the phonemes can significantly affect speech recognition performance. A spoken dialog can be judged to be successful if it provides correct information and comprehends it correctly O. Vinyals and V. Le (2014).

The dialogue flow is controlled by the *dialogue management module*. This module has to determine whether sufficient information has been elicited from the user in order ***to enable communication with an external application, to engage in communication with the external application to retrieve the required information,*** and ***to communicate that information back to the user.*** The dialogue management module is also responsible for detecting and repairing breakdowns in the dialogue through ***verifications, confirmations and corrections*** Mrksic, N., et al. (2016) .

According to Blaise Thomson, and Jason Williams(2014) Dialog management systems can be categorized in terms of ***type of control offered*** and **how the control is managed**.

Dialogues control may be ***system-led, user-led or mixed initiative***.

In a system led dialogue: - the system asks a sequence of questions to elicit the required parameters of the tasks from the user.

In A user led dialogue: - is controlled by the user who asks system questions in order to obtain the information.

In a mixed initiative dialogue: - It is shared. The user can ask questions at any time, but the system can also take control to elicit the required information.

Dialogue manager is often viewed in terms of two sub components: (1) **dialog control**, which deals with the flow of control in the dialogue and (2) **dialogue context modeling**, which is concerned with contextual information used by dialogue manager to interpret user's input and inform the decisions of dialogue control component Wilcock, G.: WikiTalk(2012).

According to M. Gasic and S. Young (2017) Different approaches to the dialogue management problem are **(1) Graph- based dialog management (2) Frame- based dialog management (3) Statistical approach.**

2.1.3.1 Graph-based dialog management

One of the simplest approaches is to represent the dialogue manager as a graph or flow-chart, sometimes called the call-flow Micheal F McTear(2005). This method is also known as "*finite state based* "dialogue control since the states of the dialogue graph can be traversed using finite state automaton. The nodes of graph represent system questions, the transitions between the nodes represent answers to questions, and graph specifies all legal dialogue.

Alternatively, each stage in dialogue can be viewed as a state in which some dialogue action is performed. Sub dialogues can be also be used within the basic network to support a more modular design approach. Most commercially available spoken dialogue systems use this form of dialogue control strategy. A finite state model can also be used for structured tasks such as obtaining weather forecasts, football scores, ordering items from a catalogue, or making simple bank transactions Micheal F McTear(2005).

For example:-

System: What is your destination?

User: SRINAGAR

System: Was that SRINAGAR?

User: Yes

System: What day do you want to travel?

User: Friday

System: Was that Sunday

User: No

System: What day do you want to travel?

This approach generally uses finite-state automata which often involve handcrafted rules. Graphs have been widely used to design dialogue systems and there are several toolkits, such as the **Center for spoken language understanding (CSLU) toolkit**, and various other commercial toolkits that allow designers to construct a dialogue by dragging, dropping, and connecting icons representing dialogue objects on screen. The toolkit then compiles the graph into a language such as **Voice *Xml***, which can be interpreted and run as dialogue application David Silver's (2016).

The management of dialogue control is straight forward in systems based on graphs, since the transitions to the next node are predetermined and are based on a small set of conditions, such as whether the user's utterance was understood with a sufficient degree of confidence. Use of graphs is suitable for fairly simple interactions where dialogue flow can be predetermined. In this way the semantics of the system is clear and intuitive.

The major advantage of the finite state model is its simplicity. Moreover, as the user's responses are restricted, fewer technological demands are put on the system components, particularly the speech recognizer Rojas-Barahona, L., et al. (2016).

The lack of flexibility and naturalness can be considered as a trade-off against these technological demands. For these reasons most currently available commercial systems use some finite-state dialogue modeling. Once the dialogue becomes more complex, the number of nodes and transitions show a significant increase and it is no longer possible to build the graph manually. This approach has proven to be very effective when the system's prompts elicit highly restricted responses from the user. On the other hand, the call-flow model typically struggles when users take the initiative and direct the dialogue themselves.

Graph based dialogue is appropriate for well-structured tasks, in which there is predetermined set of questions to be asked. Some of the examples of finite state based systems are; the automatic book service, directory assistance, questionnaires, and travel inquiries, provided the dialogue is constrained to a basic, system-led series of questions to elicit a number of well-defined responses. Moreover, since the space of user responses to each prompt can be predicted, the speech recognition and language understanding can be restricted to what the user is expected to say following each prompt, thus enhancing the performance of the ASR and NLU components T-H. Wen et al (2015) .

As stated in Micheal F McTear(2005) Response generation is also simpler since in conformations the system can reflect back the information provided by the user in the previous utterance and gather together the information retrieved from the database into a predefined template.

Dialogues developed using the graph based method support a style of interaction known as system- directed (or system initiative) dialogue. In this type of interaction system prompts the user for one or more items of information. The system might confirm that it has understood each item as it is elicited, or may leave conformation until some or all items have been collected.

In Bohus, D. and Rudnicky(2005) Once all the information has been gathered, the system performs some tasks, such as retrieving the required data, and then outputs this information to the user. The user has no control on the dialogue flow and is mainly constrained to responding to the system's prompt, although there are often options for go back or start over that allow the user to take some degree of control over the interaction.

Finite state dialogue models are generally rigid and inflexible. These characteristic would not have been a problem if the interaction with the user is controlled by the system and restricted to a well ordered sequence of questions. However, because the dialogue paths are specified in advance, there is no way of managing deviations from these paths. Problems arise if the user needs to correct an item or introduce new information or topics that was not foreseen at the time of the design of the dialogueKuun, Christiaan (2008). Adding natural language facilities, while providing the user with greater flexibility in what they can say, can add to these problems. Thus system directed dialogue would not be suitable for more complex tasks such as planning a holiday that involve negotiation of the constraints that had not been

predicted in advance by the system designer and included in the dialogue graph. For example; in a simple travel inquiry system, a system might ask the questions in the following order: *destination* > *origin* > *date* > *time.* However, when answering the system's question concerning destination the user might reply with a destination as well as the departure time (or indeed other combinations of the four required parameters) Tiancheng Zhao (2016).

A finite-state based system would simply progress through its set of predetermined questions, ignoring to process the additional information that was provided by the user with the destination and then asking an irrelevant question concerning the departure time. The solution to this problem would be to include a dialogue model so that the system 'knows' what it has already elicited as well as what has still to be asked. The system could then continuously loop through the dialogue model until all the required information has been elicited. In this way the problem of irrelevant questions would also be avoided to some extent. However, the problem is that as soon as the number of items grows, the number of transitions to cater for each required dialogue path grows to unmanageable proportions P. Blunsom and S.Young (2017).

2.1.3.2 Frame based dialog management

In a frame (or template) based system the user is asked questions that enable the system to fill slots in a template in order to perform a task. In this type of system the dialogue flow is not pre-determined but depends on the content of the user's input and the information that the system has to elicit in the course of dialog. Frame offers a better way of providing flexibility to the dialogue control Micheal F McTear(2005). Frame based systems function similar to that of production systems, taking a particular action based on the current state of dialogue. The questions and other prompts that the system should ask can be listed along with the conditions that have to be true for a particular question or prompt to be relevant. Some form of natural language input is required by frame- based systems to permit the user to respond more flexibly to the system's prompts. Natural language is also required to correct errors of recognition or understanding by the system. Generally, however, it is sufficient for the system to be able to recognize the main concepts in the user's utterance Bohus, D. and Rudnicky(2005).

The frame-based approach has several advantages over the finite-state based approach, for the user as well as the developer. As far as the user is concerned, there is greater flexibility in terms that it allows the user to fill in the slots in different orders and different combinations. The ability to use natural language and the use of multiple slots filling enables the system to process the user's over-informative answers and corrections, as in

S1: where are you flying to?

U1: Srinagar, departing at 9.

This type of input is referred to as "over answering" where the user provides more information than is requested in the system prompt. Over answering is not possible in a graph based dialogue system because each system prompt is linked to a range of responses that in turn lead to the next system prompt. Thus, in a graph based system, a response such as S1 would expect a word or phrase stating a destination as a response S. Sutton, et al. (1998).
If the user produced a response such as U1 this additional information would either be ignored or mis-recognized. In fact, the repercussions could be even worse, as the system would subsequently ask the next question in the graph that is "at what time do you want to depart?" This would be inappropriate as the user has already provided this information.

As we can refer from John Wiley& Sons, Inc.(1994), Interpretation of the user's input in the frame-based system is much more complex when compared with the graph-based systems, as the user's response can include various permutations of the required information e.g., in response to the question "where are you flying to?" the user may respond with an utterance that may contain the following combinations such as:

Destination

Destination + date

Destination + time

Destination + time + date

In such a case more complex grammars would be required in which all the items that are relevant to a dialogue are extracted from the user's utterance to fill the slots in a frame. Semantic grammars' are often used for this purpose because they focus on the semantic content of the input rather than its syntactic structure.

According to M. Gaˇsi´c, F. Lef`evre(2009), They are also more robust to the types of ungrammatical input and recognition errors that occur in spontaneous speech. In addition to permitting a wider range of user inputs, the use of frames provides for a more flexible dialogue flow since the dialogue can be driven by the information needed at a given state in the dialogue and actions to fill slots can be executed in various sequences this flexibility is desirable for applications that permit more flexible slot filling and in which it is desirable to allow the user's more freedom in how they input information to the systemO. Vinyals and Q. Le (2015).

However, frame-based systems require a more elaborate dialogue control algorithm to determine what would be the system's next question based on the analysis of the user's previous utterance in conjunction with a template of slots to be filled and a number of priorities for control of the dialogue Rojas-Barahona, L., et al. (2016).

A frame-based approach has the disadvantage like that of any production system with a large number of rules and contexts in that it is difficult to predict which rule (or question) is likely to fire in a particular context. A considerable amount of experimentation may be required to ensure that the system does not prompt an inappropriate question under some circumstances that had not been foreseen at design time T. Paek and R. Pieraccini(2008).

2.1.3.3 Statistical Approach

Spoken dialogue systems have to deal with uncertainty which is inherent during the process of speech recognition and natural language interpretation. A common approach is to augment the hand crafted belief states which represent uncertainty as well Bohus, D. and Rudnicky(2005). But most of the models lacks the principled definition for these states of uncertainty. So an alternative model Sequential decision process model which are natural and augmented a well-researched framework based on Markov decision process John Wiley& Sons, Inc.(1994), to aid the spoken dialogue system reliably identify the underlying environment state. Sequential decision process framework interacts synchronously with the external environment i.e. the user with the main goal of maximizing its reward by taking appropriate actions. These actions and history of the environment state determine the probability distribution over next possible states and such are modeled as a stochastic process Steve Young, (2017).

2.2 Basics of Afaan Oromo Language

Afaan Oromo is an Afro-Asiatic language, and the most widely spoken of the Cushitic subphylum. It is widely used as both written and spoken language in Ethiopia and some neighboring countries, including Kenya and Somalia. Besides being a working language of Regional Government of Oromia, it is the instructional medium for primary and junior secondary schools throughout the region. Moreover, a number of literatures, newspapers, magazines, educational resources, official documents and religious writings are written and published in Afaan Oromo T-H. Wen et al (2015) .

There are also television and radio programs on which information in Afaan Oromo is being broadcasted in Ethiopia. These include Ethiopian Television (ETV), Oromia Television (TV Oromia), Ethiopian Radio, and Radio Fana. Before 1991, Ge'ez script was used for writing Afaan Oromo documents. A Latin-based alphabet called Qubee has been adopted and became the official script of Afaan Oromo since 1991. There are about twenty-six consonants and ten vowels (five short and five long) in the language Addunyaa Barkeess. (2012).

The Oromo belong to the Cushitic group of people. They live in Ethiopia and Kenya Addunyaa Barkeessaa. (2014). In Ethiopia, the Oromo have an estimated population of 25,488,344, which accounts for 34.5% of the whole population of the country Mirressa Amanu. (2014). According to Tesema they are the largest ethnic group in the horn of Africa. Apart from Ethiopia, Afaan Oromo is also spoken in Kenya and Somalia Addunyaa Barkeessaa. (2014). In Ethiopia, Afaan Oromo is the official language of Oromia Regional State.

2.2.1 Latin Alphabet Writing system for Afaan Oromo (Qubee)

Afaan Oromo is a phonetic language, which means that it is spoken in the way it is written. The writing system of the language is straightforward which is designed based on the Latin script. Unlike English or other Latin based languages there are no skipped or unpronounced sounds/alphabets in the language. Every alphabet is to be pronounced in a clear short/quick or long /stretched sounds. In a word where consonant is doubled the sounds are more emphasized. Besides, in a word where the vowels are doubled the sounds are stretched or elongated. Like in English, Afaan Oromo has vowels and consonants Addunyaa Barkeessaa. (2014).

Afaan Oromo vowels are represented by the five basic letters such as ***a, e, i, o, u.*** Besides, it has the typical Eastern Cushitic set of five short and five long vowels by doubling the five vowel letters: „aa", „ee", „ii", „oo", „uu". Consonants, on the other hand, do not differ greatly from English, but there are few special combinations such as “ch” and “sh” (same sound as English),“dh” in Afaan Oromo is like an English "d" produced with the tongue curled back slightly and with the air drawn in so that a glottal stop is heard before the following vowel begins. Another Afaan Oromo consonant is “ph” made when with a smack of the lips toward the outside “ny” closely resembles the English sound of “gn” Addunyaa Barkeess. (2012).

We commonly use these few special combination letters to form words. For instance, ch used in barbaachisaa „important’, sh used in shamarree ‘girl’, dh use in dhadhaa „butter’ , ph used in buuphaa ‘egg’, and ny used in nyaata „food’ . In general, Afaan Oromo has 36 letters (26 consonants and 10 vowels) called “Qubee”.

A a	B b	C c	Ch ch	D d	Dh dh	E e	F f	G g
a	ba	ca	cha	da	dha	e	fa	ga
[ɐ]	[b]	[tʃ’]	[tʃ]	[d]	[ɗ]	[ɛ]	[f]	[g]
H h	I i	J j	K k	L l	M m	N n	Ny ny	O o
ha	i	ja	ka	la	ma	na	nya	o
[h]	[ɪ]	[dʒ]	[k]	[l]	[m]	[n]	[ɲ]	[ɔ]
P p	Ph ph	Q q	R r	S s	Sh sh	T t	U u	V v
pa	pha	qa	ra	sa	sha	ta	u	va
[p]	[p’]	[k’]	[r]	[s]	[ʃ]	[t]	[ʊ]	[v]
W w	X x	Y y	Z z	'				
wa	xa	ya	za					
[w]	[tʼ]	[j]	[z]	[ʔ]				
aa	ee	ii	oo	uu				
[ɑ:]	[e:]	[i:]	[o:]	[u :]				

Figure 2.2 Latin alphabet for Oromo Addunyaa Barkeessaa. (2014)

Like most other Ethiopian languages, whether Semitic, Cushitic, or omotic, Afaan Oromo has a set of ejective consonants, that is, voiceless stops or affricates that are accompanied by glottalization and an explosive burst of air. Afaan Oromo has another glottalized phone that is more unusual, an implosive retroflex stop, "dh" in Afaan Oromo orthography, a sound that is like an English "d" produced with the tongue curled back slightly and with the air drawn in so that a glottal stop is heard before the following vowel begins. Afaan Oromo has the typical Southern Cushitic set of five short (a, e, i, o, u) and five long vowels, indicated in the orthography by doubling the five vowel letters (aa, ee, ii, oo, and uu). The difference in length of vowels results in change of meaning Teferi Kebebew. (2010).

2.2.2 Afaan Oromo Punctuation Marks

Punctuation is placed in text to make meaning clear and reading easier. Analysis of Afaan Oromo texts reveals that different punctuation marks follow the same punctuation pattern used in English and other languages that follow Latin writing system. Similar to English, the following are some of the most commonly used punctuation marks in Afaan Oromo Mirressa Amanu. (2014).

1. **Exclamation mark (Rajeffannoo) (!):** It is used at the end of command and exclamatory sentences.
2. **Comma (Qooduu) (,):** It is used to separate listing in a sentence or to separate the elements in a series.
3. **Full stop (Tuqaa) (.):** it is used at the end of a sentence and in abbreviations.
4. **Question mark (Mallattoo Gaafii) (?):** It is used in interrogative or at the end of a direct question.
5. **Colon (Tuqlamee) (:):** It is used to separate and introduce lists, clauses, and quotations, along with several conventional uses, and etc.

2.2.3 Afaan Oromo Parts of Speeches

As we have referred from Kassahun Gelana. (2010), Afaan Oromo language words can be categorized into nouns, verbs, adverbs, adjectives, pronouns and prepositions.

I. **Nouns**: Noun is a word that helps to identify the categories of things, people, places and ideas. Nouns in Afaan Oromo are inflected for gender, definiteness and number. Most Afaan Oromo nouns are not morphologically marked for gender, except some sets of animate nouns which have the same root. These are respectively marked as feminine and masculine with [-ti: /-tti:] and [-SA /-ssa] as in the examples below. (Amanuel 2007)

II. **Verbs**: Verbs are words or compound of words that expresses action, a state of being and/or relationship between two things. In their normal position, they are found at the end of the sentence as shown below.

Caalaan mana bite. : Chala bought a house.

Ayyaantuun dhufte. : Ayantu has come.

III. **Adverbs**: Afaan Oromo adverbs are words that are used to modify verbs. Adverbs usually precede the verbs they modify or describe. They have the function to express different adverbial relations such as relations of time, place, and manner or measure.

IV. Adjectives: An adjective is a word that describes or modifies a noun or pronoun. It specifies to what extent a thing is as distinct from something else. The masculine form terminates in one of the following suffixes - aa, -eessa, or -(a)acha, and the feminine form terminates in one of the following suffixes -oo, -tuu, -eettii, or -aattii.

Example, Dursaan gabaabaa (m) dha. : Dursa is short

Hawwiin furdoo (f) dha. : Hawi is fat.

V. **Pronouns**: In Afaan Oromo, like in other languages, a pronoun is a word that is used instead of a noun or noun phrase. They are characterized based on number and gender. The Pronoun in Afaan Oromo can be independent or hidden with the verb based on their existence in a sentence. Independent pronouns are pronouns exist in a sentence as a separate word in the sentence. In the following example, “Inni” is an independent personal pronoun.

Example: Yohaannis yeroo dhufe homaa hin nyaatu homaas hin dhugu ture;

Isaanis Inni dhukuba qaba jedhu.

VI. **Prepositions**: In Afaan Oromo prepositions are much less numerous than postpositions. The most common are: akka - according to, like, as. eega - since, from, after. eegasu - in that case, therefore gara - in the direction, towards, side. haga - until. Hamma - upto, until such that, as much as.

Example,

Akka isaa jabaan namu hin jiru.	: There is no one as strong as he is.
Inni gara manaa deema.	: He goes (towards) home.

- **Gender**: Afaan Oromo nouns gender variations are identified depending on the suffixes attached to the nouns. Some of them are "aa" is attached for masculine and "tuu" for feminine. For instance, "Barataa" (indicate male student)," Barattuu" (female student), "Barsiisaa" (male teacher) and "Barsiistuu" (female student). When the suffixes "ssa" and "tti" are attached to nouns they also indicate masculine and feminine respectively Addunyaa Barkeess. (2012).

  For example, "Obbolleessa" (brother) and "Obbolleettii" (sister). In this language names of astronomical bodies and geographical places like cities and countries are feminine. For example, "Aduun Baate" (to mean the sun rises) and "Magaalli Finfinnee bara kam hundoofte?" (To mean when Addis Ababa City was established?), the suffix "tee" indicates feminine gender in two sentences. In Afaan Oromo the term "isa" and "ishee" shows masculine and feminine respectively like that of English language third person singular pronouns he and sheKassahun Gelana. (2010).

2.2.4 Word Categories in Afaan Oromo

Words are the basic unit of a given language. The combination of these words on the bases of the language gives us phrases, clauses and sentences. The meanings of these sentences depend on each word of the sentence and the way they are arranged. Afaan Oromo words can be placed in to different categories. These categories are Noun, Verb, Adjective, Adverb, Preposition, Pronoun, Conjunction and Interjection and numerals Addunyaa Barkeess. (2012).

1. Nouns:

Afaan Oromo nouns are words used to name or identify any of the categories of things, people, animal, places or ideas Addunyaa Barkeess. (2012). However, sometimes lexical classes like noun can be defined functionally (morphologically and syntactically) so that some words for people, places, and things may not be nouns. In Afaan Oromo, nouns mainly occur at the beginning of a sentence. In the following examples, Afaan Oromo nouns are bolded.

Dimaan fillannoo kooti : Red is my preference.

Sareen manatti olixxe : A dog entered into a house.

Words that are categorized as nouns in a sentence can be a subject or object. Subject mostly comes at the beginning whereas an object mostly comes after subject and before verb in a sentence. Look at the following example.

Warabessi *harree* nyaate. : A hyena eats a donkey

Tolaan *mana* ijare. : Tola built a house.

In the above two sentences the words "Warabessi" (A hyena) and "Tolaan" (name of person) are the subject of the sentences, whereas the two words "harree" (donkey) and "mana" (house) are the objects of the two sentences respectively.

Afaan Oromo nouns are also inflected for number i.e. singular and plural Mirressa Amanu. (2014). Singular nouns are nouns which are built without adding any affix. On the other hand, plural nouns are mainly formed by adding suffix or using numerals with single noun. The following are suffixes used to form plural nouns in Afaan Oromo. "-Oota", "-ota", "-wwan", "-een", "-lee","-yyi" etc. In the following

Table 2.1, plural noun formation using suffix:

	Singular		**Plural**		**Plural Marker**
	Mana :House		Manneen :Houses		-een
	Sangaa :ox		Sangota :Oxen		-ota
	Waraabeessa :Hyena		Waraabeyyii :Hyenas		-yy[i]
	Hojii :Work		Hojiilee :Works		-lee
	Barsiisaa :Teacher		Barsiisota :Teachers		-ota
	Sa'a :Cattle		Saawwan :Cattle		-wwan

Additionally, plural nouns can be formed by using numerals. This can be done by writing singular noun followed by numeral. For example, the following are valid plural nouns in Afaan Oromo Addunyaa Barkeessaa. (2014).

Table 2.2 plural noun formation using numerals

	Singular		**Plural**
	Mana :House		Mana sadi :Three house
	Nama :Man/Woman		Nama Lama :Two man
	Hoola :Sheep		Hoola kudhan :Ten sheep

Afaan Oromo analyses masculine and feminine genders and most of the nouns belong to either of the two. However, there are some nouns used for both masculine and feminine. For example, the noun “nama” (man or woman) can be used for masculine and feminine T-H. Wen et al (2015) . Afaan Oromo nouns are also inflected for definiteness, but not for indefiniteness. The definiteness (the English ‘the’) in Afaan Oromo can be different for masculine and famine. For masculine “-icha” is mainly used while “-ittii” is used for feminine.

Table 2.3 definiteness form of nouns

Noun	Definiteness
Nama :man	Namicha :The man
Harree :Donkey	Harricha :The donkey
Qaalluu :Priest	Qaallitti :The priest

2. Pronouns:

Afaan Oromo pronouns can be used for replacing nouns and noun phrases. Like nouns, Afaan Oromo pronouns decline for number and gender Kassahun Gelana. (2010). Consider the following pronouns

Ishee: she, represents feminine noun and singular

Isa : he, represents masculine noun and singular

Isaan: they, represents plural nouns and either masculine or feminine.

Table 2.4 personal pronouns that can replace subject

	First person	Second person	Third person
Singular	Ani: I	Ati :You	Inni :He
Plural	Nuyi :Nuti :We	Isiin :You	Isaan :They

There are different categories of pronoun in Afaan Oromo based on their functionality and meaning in a sentence. These are personal, possessive, demonstrative, reflective and reciprocal pronouns O. Vinyals and V. Le (2014). Personal pronouns can be used to replace the subject and the object of a sentence. The following are Afaan Oromo personal pronouns used to represent subject of the sentence.

Firomsaan leenca ajjeese. : Firomsa kills a lion
Inni leenca ajjesse. : He kills a lion

The above two sentences have the same meaning. Even if in the second sentence the pronoun

	First person	Second person	Third person
Singular	Ana :Me	Si :You	Isa :Him
Plural	Nu :Us	Isiin :You	Isaan :They

"Inni" (he) replaces the subject "Firomsaan" name of a person. A personal pronoun can also replace an object of a sentence in Afaan Oromo.

Table 2.5 personal pronouns that can replace object

Consider the following example:

Leensaan barattota barsiiste. : Lensa teaches students.

Leensaan isaan barsiiste. : Lensa teaches them.

In this example, "isaan" (them) replace the object "barattota" (students). Another category of Afaan Oromo pronoun is possessive pronouns which are used to indicate the ownership of something.

Table 2.6 Afaan Oromo Possessive pronouns

	First person	Second person	Third person
Singular	Koo :Kiyya :Mine	Kee :Yours	Isaa:His, Ishee:Her
Plural	Keenya :Ours	Keessan :Yours	Isaan :Their

3. Verbs:

Verbs are words or compound of words that express action, state of being in or relationship between two things Addunyaa Barkeessaa. (2014). In Afaan Oromo verbs mostly appear at the end of sentence.

Consider the following example:

Guutaan kaleessa deeme. : Guta went yesterday.

Caalaan kubbaa dhiite. : Chala kicked a ball.

In this example, the words written in italic and underlined ‘deeme’ (went) and ‘dhiite’ (kicked) are verbs of the sentence. Afaan Oromo Verbs are inflected for number, gender and tense P. Blunsom and S.Young (2017). Additionally, Afaan Oromo verbs can be categorized into main (transitive or intransitive) and auxiliary verbs. Intransitive verbs are main verbs which do not take object or complement in a sentence. The following examples illustrate intransitive verb in Afaan Oromo.

Namichi fiige. : The man runs.

Isaan Kaleessa dhufan. : They came yesterday.

In the above example, the words written in italic and underlined are transitive verb. They do not transfer any message from subject to complement. Transitive verbs are main verbs which transfer message to complement (objects). Consider the following two sentences:

Tolaan ulee cabse. : Tola broke a stick.

Caalaan muka mure. : Chala cut a tree.

In the above two examples the verb “cabse” broke and “mure” cut are transitive verbs. They interrelate subject and object in the sentences. Auxiliary verbs support the main verbs used in a sentence. The following are Afaan Oromo auxiliary verbs “dha”, “ta’e”, “qabda”, “ture”, “jira”, etc. In the following two examples the auxiliary verbs are written in bold.

Isheen barattuu cimtuu dha. : She is a clever student.

Hojii kana hojjechuu qabda. : You have to work this job.

In the above two sentences the word “dha” and “qabda” are auxiliary verbs.

4. Adjectives

Adjectives in a sentence are used to modify nouns to show the quality of things Kassahun Gelana. (2010). I.e. it specifies to what extent a thing is distinct from something else. Consider the following examples:

Leenseen bareeddu dha. : Lense is beautiful.

Tolaan dheeraa dha. : Tola is tall.

In the above examples the word “bareeddu” beautiful and “dheeraa” tall are adjectives. Afaan Oromo adjectives can be formed from compound words. For instance, “humna dhabeesssa” weak, “simbo qabeessa” handsome are some of adjectives constructed from compound words. Adjectives inflect for number and gender in Afaan Oromo language.

Table 2.7 adjective inflection for number and gender

Singular	Plural	Masculine	Feminine
Guddaa	Gurguddaa	Guddaa	Guddoo
Jabaa	Jaboota	Jabaa	Jabduu
Ko’eessa	Ko’eeyyii	Ko’eessa	Ko’eettii
Cimaa	Ciccimoota	Cimaa	Cimtuu

5. Adverbs:

Adverbs are words which are used to modify verbs. In Afaan Oromo adverbs come before the verb they modify Addunyaa Barkeess. (2012). Afaan Oromo adverbs are categorized as adverbs of time, place and manner (condition).

- Adverbs of time show the time the action takes place. The following are the words that can be used as adverbs of time in Afaan Oromo language.

  Amma :now , Boru :Tomorrow , Kaleessa :Yesterday ,

  Yoom :When, Har’a : Today, Galgala : Tonight

- The following are the words that can be used as adverb of place in Afaan Oromo.

  “as” here, “achi” there, “gadi” below, “gubbaa” above, ‘jidduu’ middle, ”irra” on

Consider the following example,

Tolaan mana jira. : Tola is at home

Inni konkolaataa irra jira. : He is on the car

- Adverb of manner show how the action of the sentence is done. The following are Afaan Oromo words that can be used as adverb of manner “ariitin” quickly, “suuta” slowly, “akka gaarii” well etc.

Consider the following example,

Inni ariitin figa. : He is running quickly.

Caalaan baay’ee cimaa dha. : Chala is very clever

2.2.5 Abbreviations in Afaan Oromo

Abbreviations are mostly formed by taking initial letters of multiword sequences to make up a new word. Sometimes, they can be formed from initial and non-initial letters Mirressa Amanu. (2014) .Afaan Oromo, abbreviations are used to represent dates A.L.I Akka Lakkoofsa Itiyoophiyaa to mean in Ethiopian calendar, A.L.A Akka Lakkoofsa Awurooppaa to mean in Gregorian calendar, months and dates by short words. Moreover, personal titles can be abbreviated like that of English language. For examples: “Aadde” is abbreviated as “Aadd.” Mrs., Obbo is abbreviated as “Obb.” Mr. Organization’s names are also abbreviated.

2.2.6 Afaan Oromo Questions

Different languages have different ways in the use of the word order and question particles. However, question statements are constructed with the help of interrogative words and question marks to indicate the statement is a question, in every language. In the English language, interrogative articles such as who, what, where, when, why, how are used to construct a questions Addunyaa Barkeess. (2012)Addunyaa Barkeessaa. (2014). In Afaan Oromo interrogative particles help to construct a question sentence. Interrogative particles are also known as interrogative pronouns. Some of the Afaan Oromo interrogative particles are: “eessatti” where, “maaliif” why, “yoom” when, “maali” what “akkamitti” how and so on. These interrogative particles are used to construct the factoid and non-factoid questions.

2.2.7 Afaan Oromo Morphology

Morphology is a branch of linguistics that deals about the knowledge of the meaningful component of words O. Vinyals and V. Le (2014). O. Vinyals and V. Le (2014) defined morphology as the study of the way words are built up from smaller meaningful units called morphemes. Word is the most basic unit of linguistic structure. Like other Ethiopian languages, Afaan Oromo has complex and rich morphology.

Types of Morphology in Afan Oromo

In Afaan Oromo language, we have two broad types of morphology namely derivational morphology and inflectional morphology O. Vinyals and Q. Le (2015).

- **Inflectional Morphology:** Inflectional morphology is the processes when words are adapted to their proper functions within a given sentence without changing the meaning of base words. Alternatively, can be defined as the change the form of a word for grammatical usage. It occurs in the form of different word classes. Inflection of verbs, Inflection of Nouns and Inflection of Adjectives O. Vinyals and Q. Le (2015).

Inflection of Nouns: most of Afaan Oromo nouns end with a vowel except few which ends with consonants like n, l, t. [Inflection Morphology Oromo]. Inflectional categories under nouns exists mainly in the forms of marking number, definite and gender.

In Number, marking inflections distinguish plural and singular. Several types of suffixes are attached to nouns to make plural forms Kassahun Gelana. (2010).

Example:

The plural - (o) ota attached on Nouns

Waggaa [Base form] Wagg-oota [Inflected form] Years

The plural -lee attached on Nouns

Baatii [Base form] Baatiilee [Inflected form] Months

Inflection of Verbs - verbs are the most classes in which inflection are occurred. Mainly verb inflection happens in the form of inherent and agreement properties. Inherent properties are a verb inflection that triggers inflection on that word class includes aspect, mood, and voice. However, agreement properties indicate inflection of word class for properties out of its members include person, number, gender and case. In the Afaan Oromo language, the

roots or stems of verbs, usually end in consonant, take inflectional morphemes showing distinction between aspects or mood or gender or number Addunyaa Barkeess. (2012).

Example, Mur - [root form] Mur-te [Inflected form]

Inflection of Adjectives - are the same with that of nouns. Adjectives are inflected for number, gender and singulatives like nouns. If adjectives occur within sentences, number is marked on both of them. Inflection for number of adjectives occurred in the form lexical, reduplication and-(o) ota.

Example: Inflection for Numbers Lexical: Sooressa (s) Sooreeyyi (p)

Reduplication: Guddaa (s) gud-guddoo (p) - (o) ota: hamaa(s) ham-oota (p)

In Afaan Oromo, the base forms of adjectives are normal to be used as masculine, but inflection occurs when we make them for feminine.

Example: Hamaa (m) Hamtuu (f)

In Afaan Oromo, singulative markers are not used on both noun and adjective at the same time. Which means, when nouns is marked, adjective is not and vice versa.

Example: Muk-ni (n) dheer-icha (adj.). The long stick.

Derivational Morphology: Derivation morphology is the creation of new words from already existing words in the language. And is the alternate word form which changes the meaning and word class. Different derivational suffixes are attached to the root or stem of the word Blaise Thomson, and Jason D. Williams. (2013). It occurs in the form of different word classes. Derivation of verbs, Derivation of Nouns and Derivation of Adjectives.

Derivation of verbs - is the creation of new verb word class from other given word class stem. Example: arguu [base form] to see argachuu [derivated form] to get, find

Derivation of Nouns - is the creation of new noun word class from other given word class stem.

Example: bulchuu [base form] to administer bulchiinsa [derivated form] administration

Derivation of Adjectives - is the creation of new adjectives from nouns or from other adjectives or from verbs T. Paek and R. Pieraccini (2008). Example: Jabaa [base form] strong jabaacuu [derivated form] to be strong.

2.3 Deep Neural Network Models for SDS

2.3.1. Convolutional Neural Network (CNN)

Deep neural networks have been considered as one of the most powerful models. 'Deep' refers to the fact that they are multilayer, which extracts features by stacking feed-forward layers. Feed-forward layers can be defned as: y = ***a*** (Wx + b). Where the ***a*** is an activation function; *W* and *b* are trainable parameters. The feed-forward layers are powerful due to the activation function, which makes the otherwise linear operation, non-linear. Where as there exist some problems when using feed-forward layers. Firstly, the operations of feed-forward layers or multilayer neural networks are just template matching, where they do not consider the specifc structure of data. Furthermore, the fully connected mechanism of traditional multilayer neural networks causes an explosion in the number of parameters and thus leads to generalization problems Santana, E. (2017)Tiancheng Zhao (2016).

A sliding window feature enables convolution layers to capture local features and the pooling layers can produce hierarchical features. These two mechanisms give CNNs the local perception and global perception ability, helping to capture some specific inner structures of data. The parameter sharing mechanism eases the parameter explosion problem and overfitting problem because the reduction of trainable parameters leads to less model complexity, improving the generalization ability M. Gaˇsi´c, S. Keizer(2008) [34Santana, E. (2017).

2.3. 2 Long Short-Term Memory (LSTM)

As in Steve Young, (2017)Santana, E. (2017) the modified LSTM RNN model was proposed in this work to perform sequence prediction that will make use of deep supervised learning on the benchmark spoken English digit dataset. The effectiveness of the model will be estimated with respect to training and validation accuracy, and the results will be compared with other studies that used deep learning models for various speech recognition tasks.
In addition, the classification performance of the model was evaluated to obtain the average score for precision, recall, f1-score using a confusion matrix. The choice of LSTM RNN is based on the fact that LSTM consists of a standard RNN built up with ''memory units'', that specializes in transferring long-term information, also with a set of ''gating'' units that allows memory units to carefully interrelate with the normal RNN hidden state Santana, E. (2017).

Several studies have been conducted using LSTM RNN. LSTM has achieved virtually all thrilling results based on RNNs. Thus, it has become the center of deep learning in ASR systems Tiancheng Zhao (2016). LSTMs have been used extensively in speech recognition tasks because of their powerful learning ability Afzal, Saduf (2013), Mohammed- hussen Abebeker; Taye Girma (2012), Wilcock, G.: WikiTalk(2012), Stefan Ultes and Wolfgang Minker. (2014), Steve Young, (2017), Blaise Thomson, and Jason Williams(2014), Sungjin Lee and Amanda Stent. (2016), but this is the first time LSTM RNN will be used on the spoken English digit speech recognition dataset.

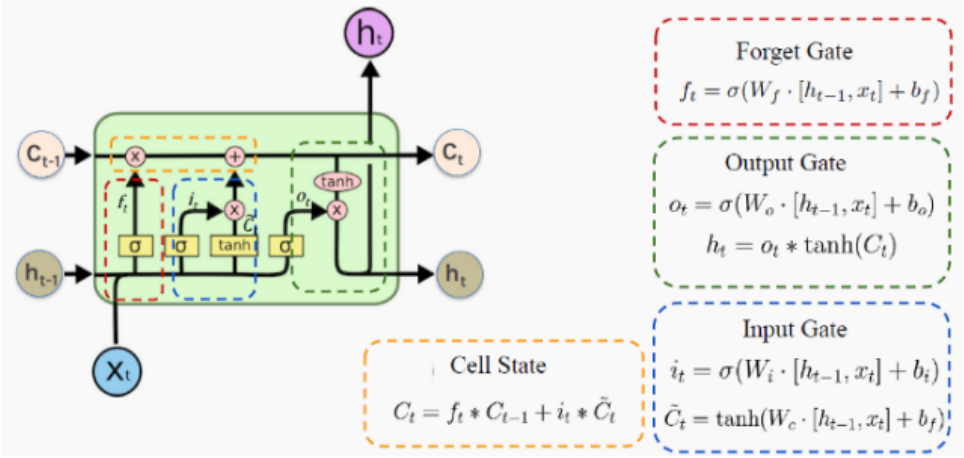


Figure 2.5 Long Short-Term Memories Rosenfeld, Roni (2010)

The gates regulate the action of the memory block whereas the ‘‘forget gate’’ weighs the information inside the cells, such that anytime previous information becomes unimportant for some cells, it will reset the state of the different cells. ‘‘Forget gates’’ also enables continual prediction Santana, E. (2017), by making cells forget their previous state, thereby restricting biases in prediction. The computation operation within an LSTM block is as follows: Input values can only be conserved in the cell state if the input gate allows them. Its input value of it and the expected value of the memory cells, C˜ t , at time step, t, is calculated as follows:

$$it = \sigma(Wi .[ht-1, xt] + bi) \quad (1)$$

$$C\tilde{}\ t = \tanh (Wc[ht-1, xt] + bc) \quad (2)$$

W [ht−1, xt] and b represent the weight matrices and bias, respectively.

The forget gate controls the weight of the state cell unit, and the value of the forget gate is computed as:

$$f_t = \sigma(W_f .[h_{t-1}, x_t + b_f) \quad (3)$$

By this process, the new state of the memory cell is being updated as

$$\tilde{C}_t = i_t .\tilde{C}_t + f_t .\tilde{C}_{t-1} \quad (4)$$

Given a new state memory cell, the output value of the gate is computed as

$$o_t = \sigma(W_x .[h_{t-1} x X_t] + b_o) \quad (5)$$

The final output value of the cell can then be explained as

$$h_t = o_t * \tanh(c_t) \quad (6)$$

2.4 Related works

ASR and TTS for Afaan Oromo are rarely attempted to develop at the master's level with a promising performance evaluation result. plus, the existing locally developed ASR and TTS for Afaan Oromo were not using large high-quality speech corpus, which can then be used for statistical training of the models, which are the current state of the arts.

As far as our knowledge there were no any attempts made to develop Spoken Dialogue System for Afaan Oromo using the statistical approach via Deep learning. We have tried to collect and presented the summary of some of the Text processing and ST (Speech Technology) attempts made by different researchers along with their area and their results as given in table 2.8 below.

Table 2.8 Previous related researches on Afaan Oromo

No	Research Title	Types	Results of the study	Authors	Year deployed	Research Gap
1	Parts of Speech Tagging for Afaan Oromo(POST)	Text Based	Uni-gram:87.58 Bigram: 91.97	Getachew Mama nd Mil-lion mesh-esha(PHD)	2011 G.C	This NLU thesis is not applicable for our AOSDS
2	A rule-based Afan Oromo Grammar Checker	Text Based	88.89	Debela Tesfaye	2011 G.C	This is rule based grammar checker, but our AOSDS is implemented using statistical approach
3	Designing a Stem-mer for Afaan Oromo Text: A Hy-brid Approach	Text Based	94.84	Debela Tesfaye & Ermias Abebe	2010 G.C	We do not need this stemmer since it is not applicable for our AOSDS, because we have preprocessed our texts by ourselves.
4			Sent. Acc of 28: 68.51	Kassahun Ge-lana		✓ Small corpus/data set was used for this ASR development

	A Continuous, Speaker Independent Speech Recognizer for Afaan Oromo	Speech Technology	Sent. Acc of 42: 89.45		2010 G.C	✓ The performance of this ASR can't fit for AOSDS. (We have to increase the size of dataset and performance of this ASR to deploy it for AOSDS). ✓ This ASR is developed for offline purpose. but, our AOSDS ASR is Cloud based and it is developed using Google cloud
5	Text to Speech Synthesizer for Afan Oromo using Statistical Parametric Speech Synthesis	Speech Technology	89.76%	Muhidin Kedir	2020 G.C	✓ Dataset used for this TTS is limited and can't be applied for SDS ✓ This TTS is developed for offline purpose. But, Our AOSDS developed using cloud based speech synthesis. ✓ Due to the limitations of the HMM-based speech synthesis model (such as the use of context decision trees to share speech parameters), the synthesized speech is not brilliant enough to meet the requirements of expressive speech synthesis.

From language development point of view, speech technologies can be applied in assistive technologies, eLearning, Healthcare, dictation system, education, command and control, information retrieval and other application areas that uses speech based human computer/mobile interactions to extract meaning and act accordingly for the particular language R. Pieraccini, and E. Levin (1992).

Therefore, since the Language Technology research is at its infant stage and there is no Automatic Spoken Dialogue System for Afaan Oromo we have to conduct further study on this language to help the society which can be benefited from these technologies.

CHAPTER THREE

3. Methodology

The goal of this study, as stated in Chapter 1, is to create a spoken dialogue system for Afaan Oromo using deep learning. We will examine how and how well we have used experimental research to accomplish our objectives in this chapter. This covers the choice of model and architecture, with a prototyping strategy used to construct the suggested spoken dialogue system because experimentation is a very useful tool for system improvement.

3.1 Toolkits and software

The Toolkits, software and programming languages implemented in this study are: - Linux (Ubuntu) was our operating environment for developed prototype, Python programming language is mainly used for this AOSDS development, Natural Language Toolkit (NLTK) is also used for text preprocessing, PyDial toolkit was utilized for NLU, Dialogue manager and NLG modules, Google Cloud has used to prepare Speech client for ASR and TTS.

There are three key Principles of the SDS toolkit we have deployed: PyDial,

(1). Domain Independence Wherever possible, the implementation of the dialogue modules is kept separate from the domain specification. Thus, the main functionality is domain independent, i.e., by simply using a different domain specification, simulated dialogues using belief tracker and policy are possible Steve Young, (2017).

To achieve this, the Ontology handles all domain-related functionality and is accessible system-wide. While this is completely true for the belief tracker, the policy, and the user simulator, the semantic decoder and the language generator inevitably have some domain-dependency and each need domain-specific models to be loadedSteve Young, (2017).

(2). Easy Configurability To use PyDial, all relevant functionality can be controlled via a configuration file. This specifies the domains of the conversation, the variant of each domain module, which is used in the pipeline, and its parameters. For example, to use a hand-crafted policy in the domain *SorRestaurants*, a configuration section `[policy_SorRestaurants]` with the entry `policytype = hdc` is used. The configuration file is then loaded by Pydial and the resulting configuration object is globally accessible.

(3). Extensibility One additional benefit of introducing the manager concept described is to allow for easy extensibility. As shown with the example in Figure 4, each manager contains a set of D domain instances. The class of each domain instance inherits from the interface class and must implement all of its interface methods.

To add a new module, the respective class simply needs to adhere to the required interface definition. To use it in the running system, the configuration parameter may simply point to the new class, *e.g.*, *`policytype = policy.HDCPolicy.HDCPolicy`*. The following modules and components support this functionality: Topic Tracker, Semantic Decoder, Belief Tracker, Policy, Language Generator, and Evaluation. Since the configuration file is a simple text file, new entries can be added easily using a convenient text editor and any special configuration options can easily be added. To add new domain, a simulated interaction is already possible simply by defining the ontology along with the database Steve Young, (2017).

3.1.1 Reinforcement Learning Algorithm for Spoken Dialogue System

The following formula depicts reinforcement learning algorithm which is implemented in our AOSDS which supported by the tool we have utilized.

Policy: $\pi(\mathbf{b},a):\mathbb{R}^n \times A \rightarrow [0,1]$

Reward: $R = \sum_{\tau=1}^{T} r(\mathbf{b}_\tau, a_\tau)$

Value: $Q_\pi(\mathbf{b}_t, a_t) = \mathbb{E}_\pi\left[\sum_{\tau=t+1}^{T} r(\mathbf{b}_\tau, a_\tau)\right]$

Problem:

Find optimal policy $\pi^* = \arg\max_\pi \{E[R \mid \pi]\}$

or

Solve $Q_\pi^*(\mathbf{b}_t, a_t) = r_{t+1} + \max_a \{Q_\pi^*(\mathbf{b}_{t+1}, a)\}$

3.2 Architecture of the proposed spoken dialogue system

Spoken Dialogue Systems (SDSs) are typically based on a modular architecture consisting of, (1) input processing modules: Automatic speech recognition (ASR) and Natural Language Understanding (semantic decoding, (3) dialogue management modules: belief tracking and policy, and (3) Output processing modules: natural language generation and speech synthesis (TTS). Statistical SDS are speech interfaces where all Spoken Dialogue System modules are based on statistical models learned from data (in contrast to hand-crafted rules). Examples of statistical approaches to various components of a dialogue system can be found in [5].

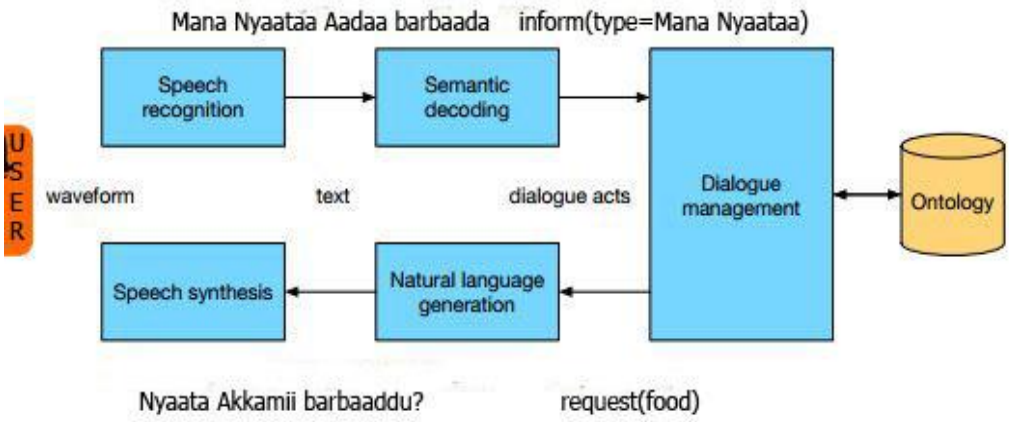


Figure 3.1 Architecture of a modular Spoken Dialogue System Tiancheng Zhao (2016)

3.2.1 Automatic Speech Recognition (ASR)

The Automatic speech recognition (ASR) component takes Audio input from user and gives transcribed (text) strings of uttered words.

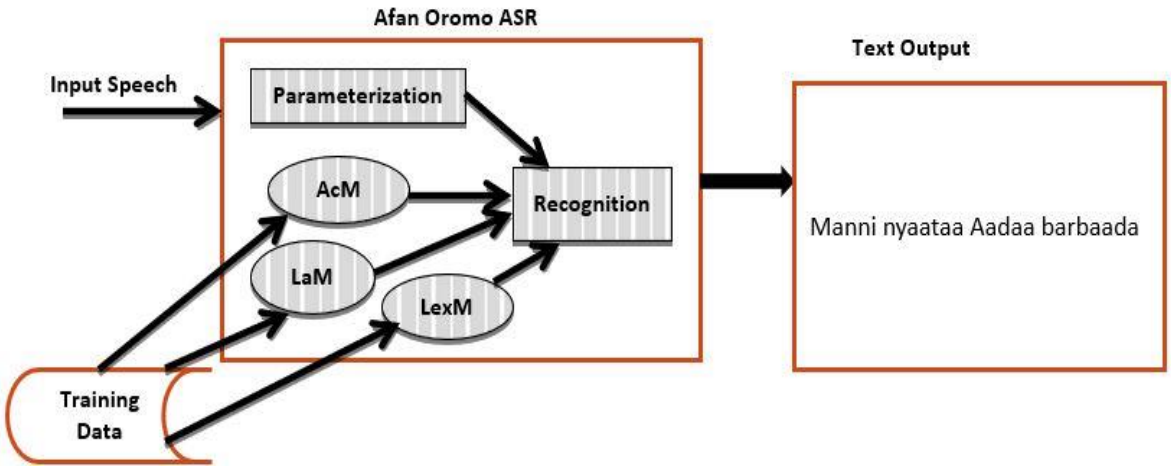


Fig 3.2 Afan Oromo ASR architecture for AOSDS (Adopted from David Silver's (2016))

3.2.1.1 Corpus/data set preparation

To Develop AOSDS we have prepared two important data/corpus these we have used for different modules of the system i.e., ASR and TTS.

Text Data/corpus

Source of texts data are from Electronic materials, by Interviewing business customers, Business owners, related literatures. For text preprocessing we have used different tools like python NLTK.

Speech Data/corpus

We have used Speaker independent continuous read speech for Automatic Speech Recognition development. For this purpose, we have selected speakers from different Age (18-50), Sex (both Male and Female).

We have used recommended Recording tools and measurements for Afaan Oromo speech to make ready for our study.

3.2.2 Natural Language Understanding

3.2.2.1 Semantic Decoder

To semantically decode the input sentence (or n-best-list of sentences), PyDial offers a rule-based implementation using regular expressions and a statistical model based on Support Vector Machines, the Semantic Tuple Classifier Mohammed- hussen Abebeker; Taye Girma (2012). For the latter, a model for the *sorRestaurants* domain is provided.

3.2.3 Dialogue Manager

3.2.3.1 Belief Tracker

For tracking the belief state, the rule-based focus tracker is available S. Sutton, et al. (1998). The implementation is domain- independent. All domain-specific information is drawn from the ontology.

3.2.3.2 Policy

The decision-making module is responsible for the policy has two implementations: *a hand-crafted policy (which should work with any domain) and a Gaussian process (GP) reinforcement learning policy* Mohammed- hussen Abebeker; Taye Girma (2012). For multi-domain dialogue, the policy may be handled like all other modules by a policy manager. Given the domain of each user input, the respective domain policy will be selected.

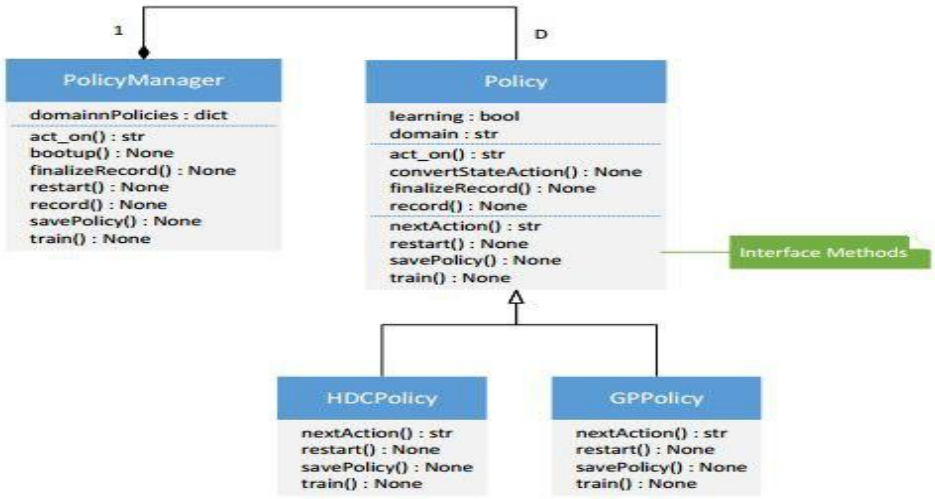


Fig 3.3 Dialogue Policy Mohammed- hussen Abebeker; Taye Girma (2012)

Additionally, a Bayesian committee machine (BCM) as proposed in Alexander I Rudnicky (2002) is available as an alternative handler: when processing the belief state of one domain, the policies of other domains are consulted to select the final system action. For this to work, the belief state is mapped to an abstract representation which then allows all policies to access it.

Within PyDial, trained policies may be moved between the committee-based handler and the standard policy manager handler, i.e., policies trained outside of the committee (in a single- or multi-domain setting) may be used within the committee and vice versa.

Dialogue as a control problem

The dialogue can be seen as a control problem where having a distribution over possible belief states we need to take some action which determines what the system says to user. We may apply the reinforcement learning framework to our problem where we look for the optimal policy $\pi : B \times A \rightarrow [0,1]$ during the dialogues with the user. In the learning procedure we update the Q-function which has the form:

$$Q^{\pi}(\mathbf{b}, a) = \mathrm{E}_{\pi}\{\sum_{k=0}^{T-t} \gamma^{k} r_{t+k} | b_t = \mathbf{b}, a_t = a\},$$

where r_t is a reward at a time t, $\gamma > 0$ is a discount factor. Then, the policy can be obtained by:

$$\pi(\mathbf{b}) = \arg\max_{a}\{Q(\mathbf{b}, \mathbf{a}) : a \in \mathcal{A}\}.$$

Policy committee

PyDial enables also to build multi-domain dialogue system using Bayesian committee (BCM) approach which builds so-called 'committee' to take advantage of having dialogue corpora even though from different domains. Committee members consist of estimators trained on different datasets. Training is analogous to the one described in the previous section. At every turn their estimated Q-values are combined to propose the final Q-value estimates using following formula:

$$\overline{Q} = \Sigma^{Q}(\mathbf{b}, a) \sum_{i=1}^{M} \Sigma_i^{Q}(\mathbf{b}, a)^{-1} \overline{Q}_i(\mathbf{b}, a),$$

$$\Sigma^{Q}(\mathbf{b}, a)^{-1} = -(M-1) * k((\mathbf{b}, a), (\mathbf{b}, a))^{-1} + \sum_{i=1}^{M} \Sigma_i^{Q}(\mathbf{b}, a)^{-1}$$

It was shown that such approach is especially beneficial for adaptation in multi-domain dialogue system. In order to produce a generic policy that works across multiple domains belief state and action state are mapped to an abstract representation which is used for all committee members.

Domains: The main focus of AOSDS is task-oriented dialogue where the user has to find a matching entity based on a number of constraints. Once the entity is found, the user can request additional information. For this scenario, the proposed AOSDS, has Six domains: Restaurant, Caffe, Banks, Electronics, boutiques and Offices & Halls.

The Ontology encapsulates the dialogue domain specification as well as the access to the back-end data base, e.g., set of restaurants and their properties. Modelled as a global object, it is used by most dialogue system modules and the user simulator for obtaining the relevant information about user actions, slots, slot values, and system actions.

3.2.4 Natural Language Generation

Language Generator part; For mapping the semantic system action to text, PyDial offers two module implementations. For all domains, rule definitions for a template-based language generation are provided. In addition, the LSTM-based language generator as proposed by Mohammed- hussen Abebeker; Taye Girma (2012) is included along with a trained model for the *SorRestaurants* domain.

Topic Tracker PyDial provides an implementation of a keyword-based topic tracker. If the topic tracker has identified a domain for some user input, it will continue with that domain until a new domain is identified. Hence not every user input must contain relevant keywords. If the topic tracker is not able to initially identify the domain, it creates its own meta-dialogue with the user until the initial domain has been identified or a maximum of number of retries has been reached.

3.2.5 Speech Synthesis (TTS)

To enable speech-based dialogue, the Dialogue Server connected to an external speech client which is developed using Google Cloud speech service. This client is responsible for mapping the output text to speech (TTS) using speech synthesis.

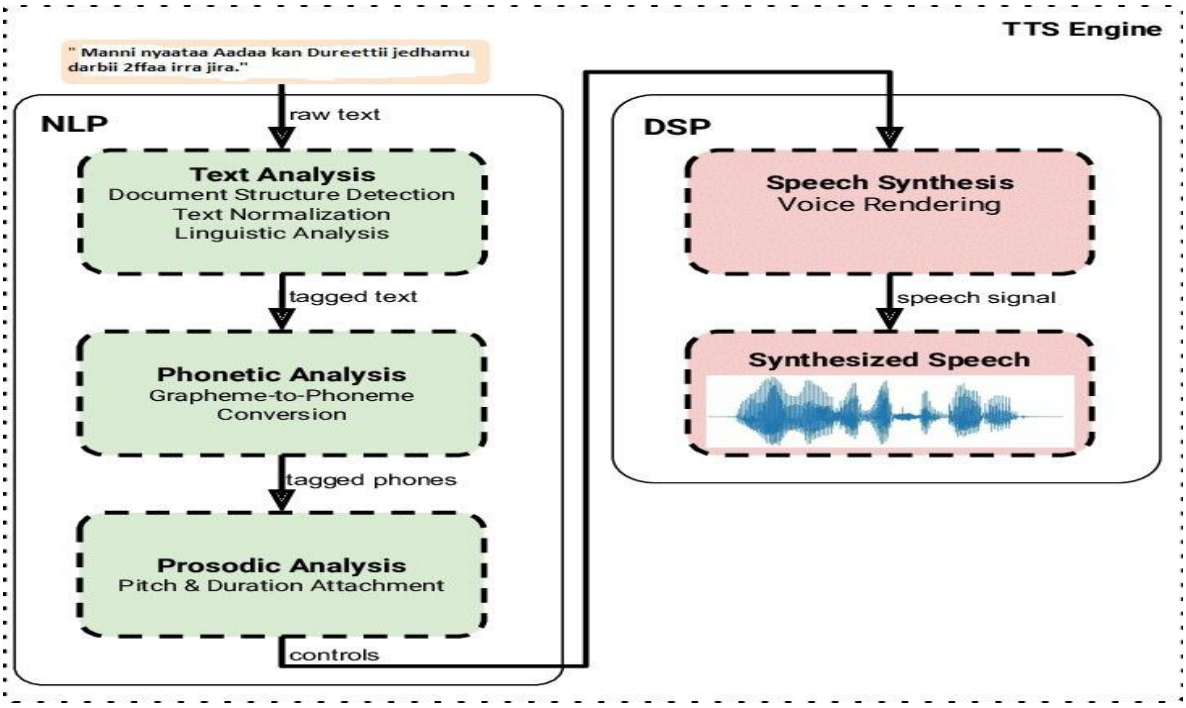


Figure 3.4. TTS Architecture P. Blunsom and S.Young (2017)

3.3 Evaluation

Evaluation component is used to compute evaluation measures for the dialogues, e.g., Task Success. For dialogue modules based on Reinforcement Learning, the Evaluation component is also responsible for providing the reward.

CHAPTER FOUR

4. Research Experiment and Analysis

Here, under this chapter the practical implementation of proposed Afaan Oromo spoken dialogue system using the selected tools is discussed. The evaluation of the experimental findings to choose the best algorithm or approach for speech recognition, natural language understanding, dialogue manager, and natural language generation are presented in this part. Examining the outcomes of the experiments and the experimental dataset allows us to choose the best algorithm or method based on objective evaluation criteria. On the basis of the test findings, comparisons and discussions are made next.

4.1 Afaan Oromo Spoken Dialogue system (AOSDS)

The proposed AOSDS operates in the following way. It takes initiative by greeting and asking what the user is looking for. The text from the speech recognizer engine is given to the Natural Language Understanding unit to determine the user action for instance if the user says “caffe” and the corresponding recognition hypothesis with confidence score passed to the NLU to determine the user action as “cafres (caffe)”.

The dialogue manager uses rules to select appropriate system action according to its belief of the dialogue state. The belief is determined from the hypothesis. Then this system action interpreted to a surface form by NLG and sent to the text-to-speech unit for the user in audible format.

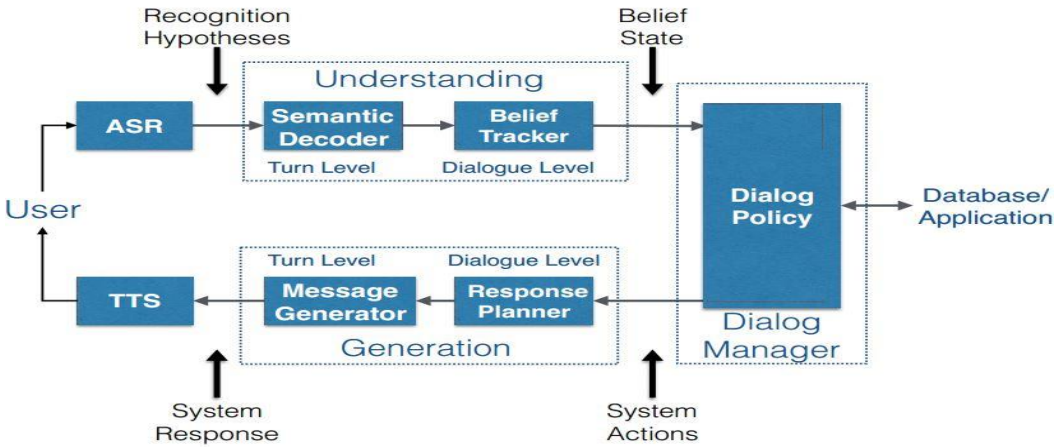


Fig 4.1 AOSDS Architecture using deep learning (adopted from T-H. Wen et al (2015)

4.1.1 ASR for AOSDS

Afaan Oromo ASR trained for this experiment was used and integrated with PyDial. PyDial is basically designed to use its own cloud-based speech recognizer and text to speech for English language. The speech client connects to the Dialogue Server via HTTP exchanging JSON messages. Since the speech client is not part of PyDial, Cloud-based services for ASR and TTS from Google is connected to PyDial, which allowed us speech-based interaction.

We have developed cloud based speech recognizer using Audio/Speech from 16 (8 Males and 8 Females) speakers and it is around 1 (One) hour long. Each of the speakers read around 50 sentences (prompts), therefore we have prepared 800 sentences/prompts for training. To Test our ASR, we have used 200 sentences (prompts) from 4 (2 Males &2 Females) speakers. To develop language model for our ASR we have used 5000 Afaan Oromo sentences (prompts) which are collected from different sources and domains. Obviously, This, is very low corpus size to develop an ASR that has better performance.

For our ASR we have developed Canonical dictionary (lexical model) using a total of 2100 words that are uttered in the training data. In order to generate the lexical model, we have developed Python program that is used for generating pronunciation dictionary automatically. This lexical model consists 56 phones of Afaan Oromo.
Our recognizer's word accuracy was 63.5% and it has significant effect on the reduction of Our spoken dialogue system.

4.1.2 Spoken Language Understanding

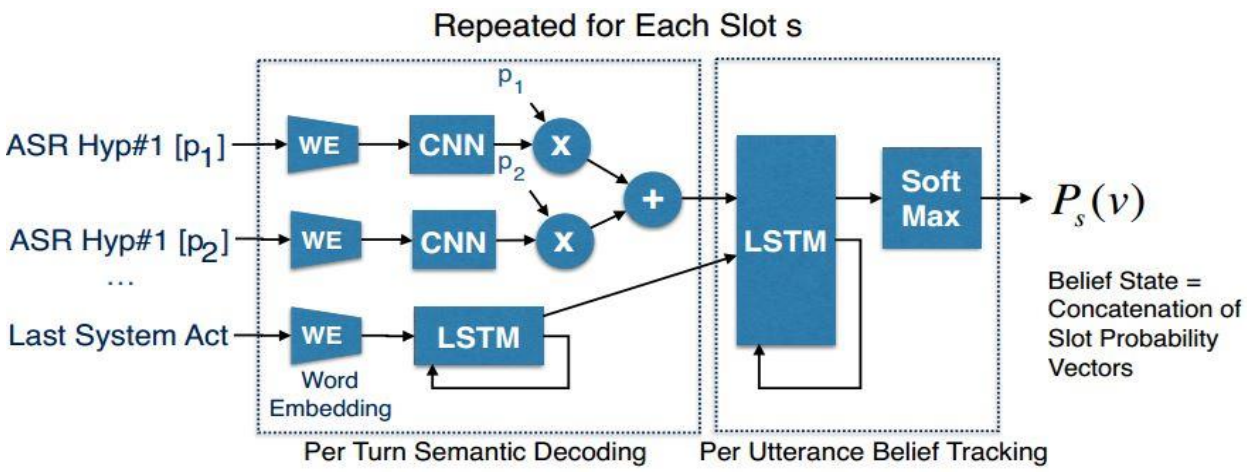


Fig 4.2 Spoken Language Understanding T-H. Wen et al (2015)

Semantic decoding system is responsible for decoding the meaning of utterances sent to the system. Often dialogue systems are deployed in a noisy setting, where background noise and diverse user populations can result in high speech recognition error rates. Thus, SLU systems operate on the top *k* hypotheses output by the speech recognizer.

The illustrative procedure is:

Dialogue turn: *Kaaffeen gatiin isaa rakasaa gamoo kana irra jiraa?*

Dialogue act type: *inform*

Semantic slots: *price, area*

Semantic values: *rakasaa, gamoo kana*

Dialogue act: *inform(price=rakasaa, area=gamoo kana)*

During the semantic decoding process, we extract the information about ***dialogue act type***, which encodes the system or the user intention in a dialogue turn, and ***semantic slots*** and ***values*** that further describe entities from the ontology that a dialogue turn refers to. We can tackle this problem from two perspectives - first one involves creation of list of grammar rules that parse user utterances. Second approach relies on statistically trained models where we train classifiers to directly label utterances based on training data.

Support Vector Machines

We can look at semantic decoding as a classification task in which we need to make predictions about the set of semantic concepts. However, this requires corpora data with labelled semantic concepts. Such an approach can easily handle *N-best* list of automatic speech recognizers hypotheses as in real conversational systems error rate of the top hypothesis is typically about 20-30% and thus robust system is needed.

Classification model that we have implemented for AOSDS using PyDial is support vector machines which map input into a high-dimensional feature space where data is linearly separable. It was shown that such statistical approach can substantially improve performance both in terms of accuracy and overall dialogue reward.

4.1.3 Dialogue Manager

General AOSDS Dialogue Acts are listed below in tabular form.

Table 4.5 Dialogue Acts of AOSDS

Act	Description
hello(a=x,b=y,...)	open a dialog and give info a=x, b=y, ...
inform(a=x,b=y,...)	give information a=x, b=y, ...
request(a, b=x,...)	request value for a given b=x ...
reqalts(a=x,..)	request alternative with a=x,...
confirm(a=x,b=y,..)	explicitly confirm a=x,b=y,..
confreq(a=x,.., d)	implicitly confirm a=x,.. and request d
select(a=x,a=y)	select either a=x or a=y
affirm(a=x, b=y,...)	affirm and give further info a=x, b=y, ...
negate(a=x)	negate and give corrected value a=x
deny(a=x)	deny that a=x
bye()	close a dialogue

Dialogue Policy

In AOSDS we have dedicated module for policy learning under directory named **policy**.

Policy - is an abstract class that defines an interface for a single domain policy is in **AOSD/Policy/Policy.py** script. It has all required functions to build generic reward model.

GP-SARSA dialogue policy is implemented in **GPPolicy.py** with additional functionalities in **GPLib.py.** It derives from Policy class.

Hand-crafted policy is implemented HDCPolicy.py. It can be run with all provided domains. Bayesian committee machine model is implemented in ***PolicyCommittee*.py** with class ***CommitteeMember*** providing interface methods for single domains. The policy is for a committee is handled by ***PolicyCommittee*** class.

Configuration file we have implemented for our AOSDS Dialogue Policy is:

```
[policy_DOMAIN]
belieftype = baseline or focus
useconfreq = False or True      # use confirm request ?
learning = False or True        # policy learning?
outpolicyfile = ''
inpolicyfile = ''
policytype = hdc or gp
startwithhello = False or True
```

We choose belief tracker using ***belieftype=baseline***. After we use the HDC policy we can choose to confirm request using ***useconfreq = True*** option. By Setting ***learning*** = ***True*** trains a model and we can specify whether to train from scratch or by loading preexisting model providing a path to '***AOSDS/Policy/inpolicyfile'.*** The policy can be saved to a provided path via ***outpolicyfile='AOSDS/Policy'***. ***Policytype = gp***. Finally, ***startwithhello*** is a domain dependent setting and it is overruled and set to *True* if we want to use single domain option.

GP-Policy settings

The specific option settings for a GP-Policy module can be given in gppolicy section. Here is the list of all options with possible values with default ones:

```
[gppolicy_DOMAIN]
kernel = polysort or gausssort
actionkerneltype = delta or hdc or distributed
abstractslots = False or True
unabstractslots = False or True
thetafile = ''
slotabstractionfile = ''
```

This section is relevant only if *policytype* under **policy_DOMAIN** is set to ***gp***. ***kernel*** and ***action kernel type= gausssort*** choose belief and action kernels respectively. Option ***abstract slots*** are set to ***True*** if we want to use BCM. If training was performed with BCM and now we train single domain ***un abstract slots*** should be set to ***True***. ***The tafile*** = ***'AOSDS/Policy'***, sets a path to belief kernel hyperparameters file. Default option for performing slot abstraction is to use hardcoded mapping found in ***AOSD/policy/slot abstraction/***, but we can give another one through ***slot abstractionfile***.

The detailed options for SARSA training are given under ***gpsarsa_DOMAIN*** section:

```
[gpsarsa_DOMAIN]
random = False or True
scale = 3
saveasprior = False or True
numprior = 0
gamma = 1.0
sigma = 5.0
nu = 0.001
```

Random=True, specifies whether actions are chosen randomly. ***Scale=3,*** parameter chooses how much exploration we want to perform as we sample from Gaussians of mean and std dev * scale. Learning scale set to be above 1 encourages exploration and for testing scale is usually set to 1. ***Saveasprior=True*** and ***numprior*** control saving and the number of prior means. ***gamma*** is a discount factor and it is usually set to ***11*** as we perform episodic tasks. ***nu*** is a dictionary sparcification threshold and ***sigma*** is residual noise.

GP policy Committee Settings

```
[policycommittee]
bcm = False or True
pctype = hdc  or configset
configsetcommittee = list of domains to be in committee if using above configset
learningmethod = singleagent or multiagent
```

BCM is used if ***bcm = True***. Note that in that case we need to set ***abstractslots*** to *True* under ***[gppolicy_*DOMAIN]. Pctype = *hdc*** chooses whether to use hand-crafted policy (hdc) or policy specified by ***configset*** - this option requires ***configsetcommittee***. ***learning method*** specifies whether we are learning an agent only in one domain. Default is just to do

single agent learning - whereby only domain where actions are actually being taken is learning - with multiagent learning all in committee will learn.

4.1.3.1 Semantic Decoder

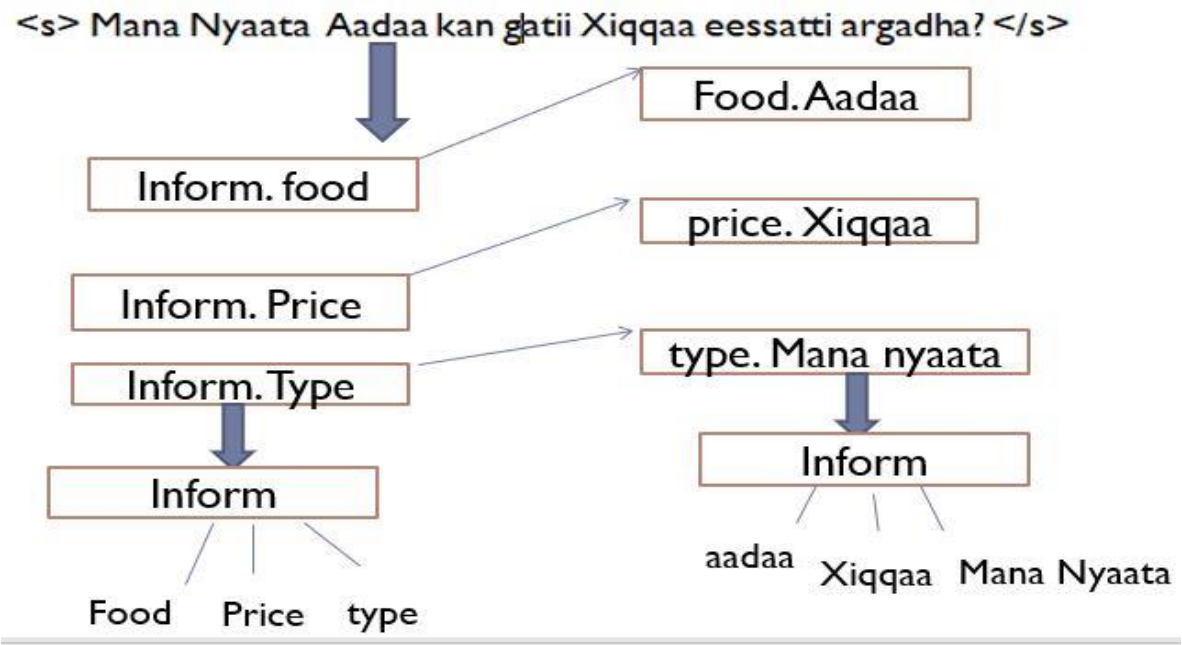


Figure 4.3 AOSDS Semantic Decoder

4.1.3.2 Ontology

Entities in our ontology for SoretiRestaurant domain are: Name, Address, Area, Food, Phone and Pricerange

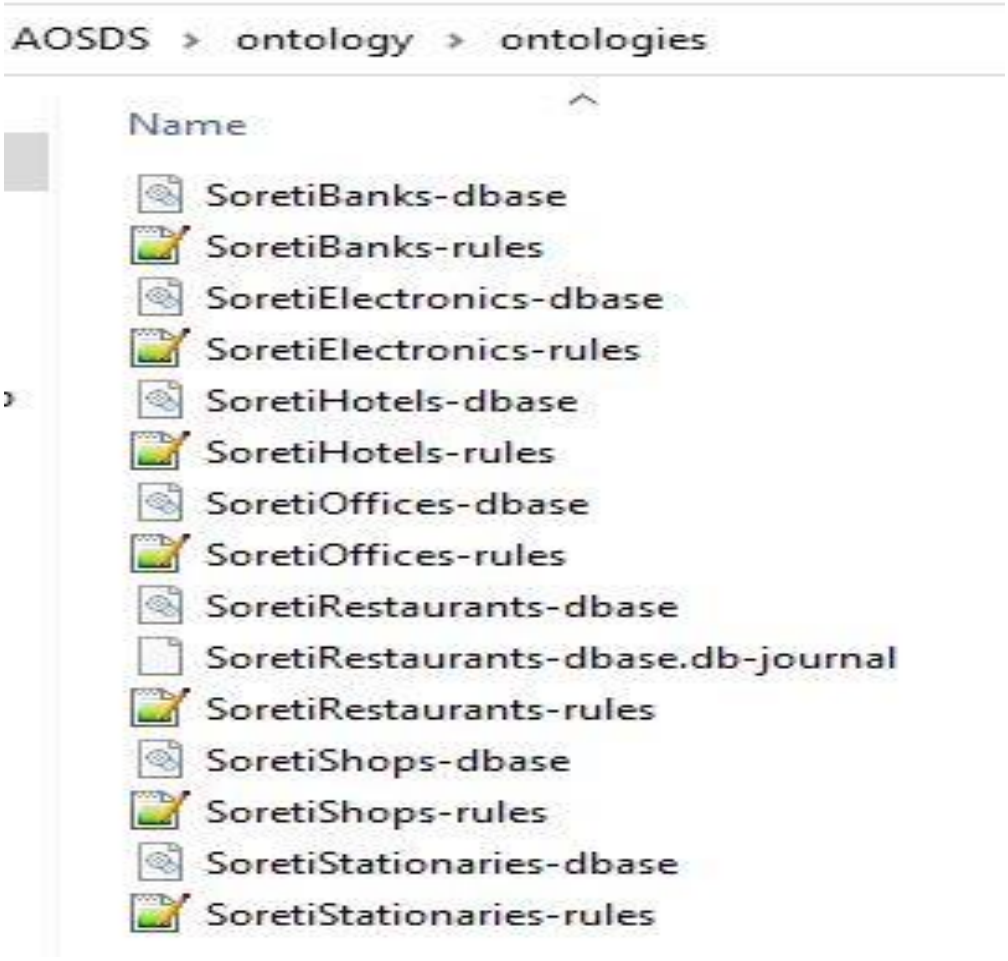
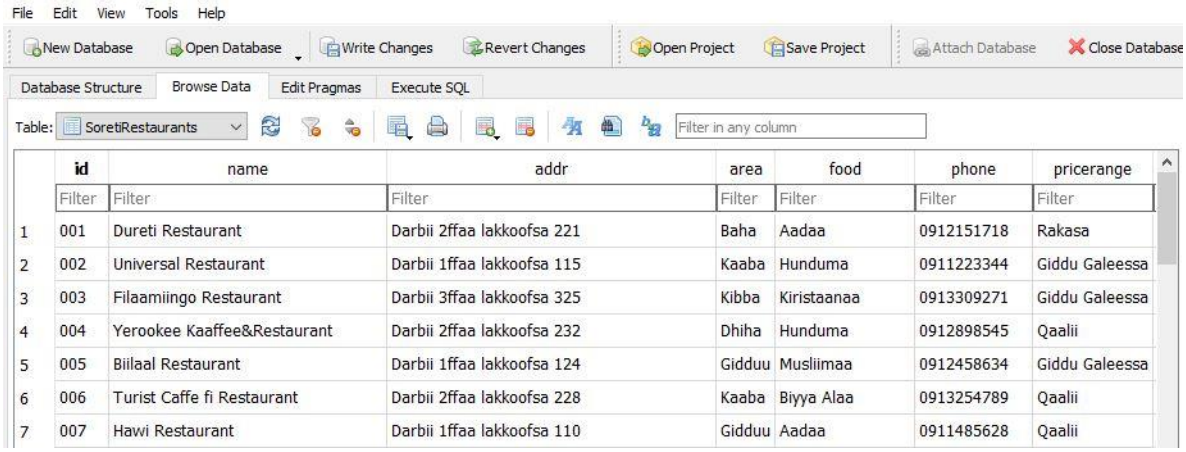


Figure 4.4 AOSDS Ontologies

4.1.3.3 Database

SoretiRestaurant database developed by SQLIte 3



File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: SoretiRestaurants | Filter in any column

	id	name	addr	area	food	phone	pricerange
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	001	Dureti Restaurant	Darbii 2ffaa lakkoofsa 221	Baha	Aadaa	0912151718	Rakasa
2	002	Universal Restaurant	Darbii 1ffaa lakkoofsa 115	Kaaba	Hunduma	0911223344	Giddu Galeessa
3	003	Filaamiingo Restaurant	Darbii 3ffaa lakkoofsa 325	Kibba	Kiristaanaa	0913309271	Giddu Galeessa
4	004	Yerookee Kaaffee&Restaurant	Darbii 2ffaa lakkoofsa 232	Dhiha	Hunduma	0912898545	Qaalii
5	005	Biilaal Restaurant	Darbii 1ffaa lakkoofsa 124	Gidduu	Musliimaa	0912458634	Giddu Galeessa
6	006	Turist Caffe fi Restaurant	Darbii 2ffaa lakkoofsa 228	Kaaba	Biyya Alaa	0913254789	Qaalii
7	007	Hawi Restaurant	Darbii 1ffaa lakkoofsa 110	Gidduu	Aadaa	0911485628	Qaalii

Figure 4.5 SoretiRestaurant database

4.1.4 Spoken language generation

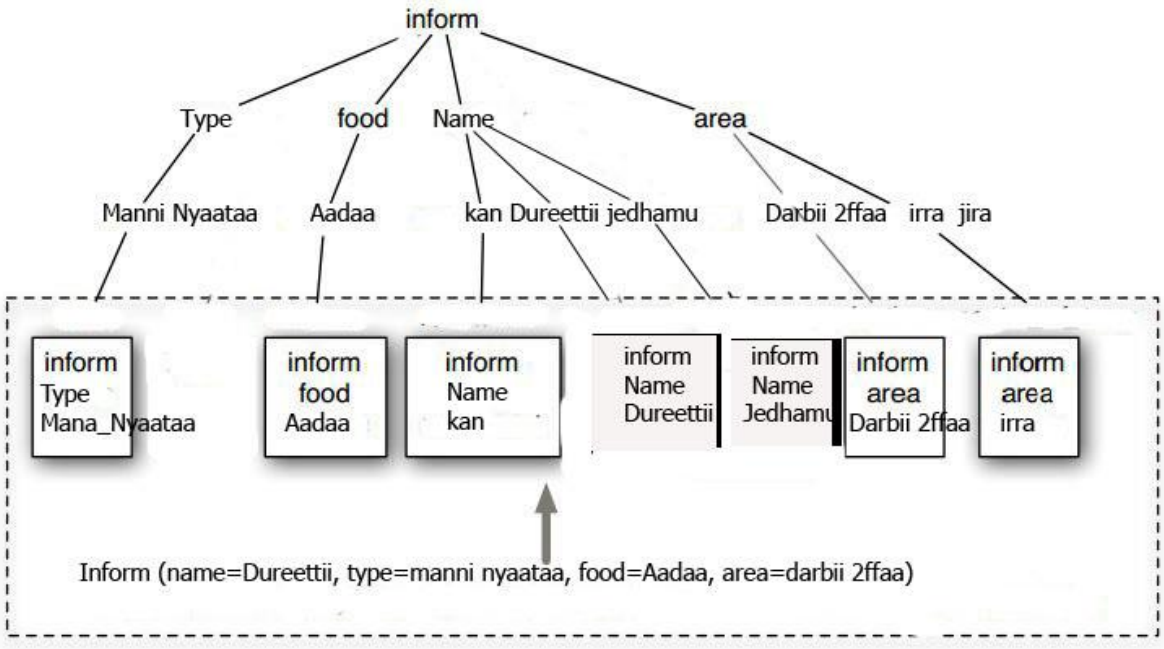


Fig 4.6 Spoken Language Generations for AOSDS (Adopted from Stefan Ultes and Wolfgang Minker. (2014))

4.1.4.1 Stochastic language generation with Recurrent Neural Networks (RNNs)

Rule-based approach for natural Language generation is expensive to build because many of the processing components require a substantial amount of handcrafting which also requires expert knowledge. Moreover, such system cannot be easily adapted to new domains and the maintenance of it is a very challenging task.

That is why, the use of data-driven natural language generation methods for spoken dialogue systems might be seen as a natural solution to problems mentioned above. In PyDial library three statistical models are implemented that are based on recurrent neural networks.

A one-hot encoding $\mathbf{w}_t$ of a token is an input at each time step t conditioned on a recurrent hidden layer $\mathbf{h}_t$ and outputs the probability distribution of the next token w_{t+1}. Because the system outputs probability vector we can sample from it which introduce randomness during generation and greatly improves the variability of generated utterances which is another advantage over rule-based systems. Finally, tokens can be lexicalized to form system utterance.
A graph below shows the standard recurrent architecture that is shared by all three models.

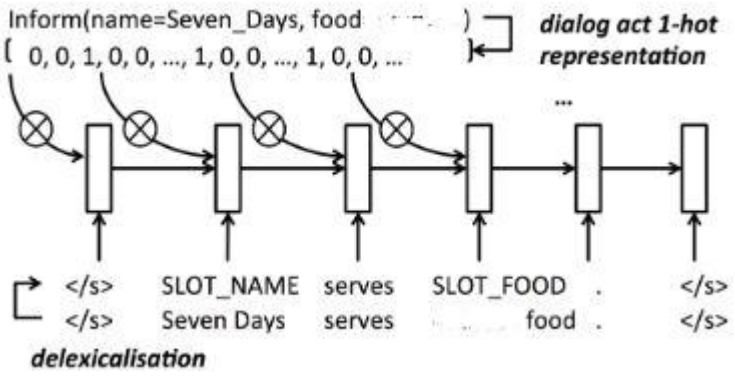


Fig 4.7 Standard recurrent architecture (Adopted from PyDial page)

4.1.4.2 RNN-based natural language generation model

Architectures based on a stand-alone RNN cell were often found to duplicate or omit some slot-value pairs from the dialogue act. In order to force network to use the information what parts of dialogue act was used or is missing we can introduce a control vector $\mathbf{f}$ constructed from the concatenation of one-hot encodings of the required dialogue act and slot value pairs. We use this vector at every time step using a gated version $\mathbf{f}_t$ designed to discourage duplication of information in the generated output.

In order to handle the cases when there is no explicit value to delexicalise (for example, food=dont_care, kids_allowed=false) a convolutional neural network sentence model is used to ensure that the required dialogue act and slot-value pairs are represented in the generated utterance, including the non-standard cases.
Finally, in vanilla RNNLM model the consecutive words are generated based only on the preceding history. In order to model backward contexts, after computing forward pass we use backward RNN to re-rank the candidates. Alexander I Rudnicky (2002). We have deployed using the following command.

```
[semo_SoretiRestaurants]
semotype = RNNSemO
configfile = semo/RNNLG/config/hlstm.cfg
```

4.1.4.3. Semantically Conditioned LSTM-based natural language generation model

To ensure that the whole dialogue act is generated and to avoid duplications or omissions of slots information we introduce again control vector $\mathbf{d}$, a one-hot representation of the dialogue act type and its slot-value pairs.

Unlike in the previous section won't use simple heuristics to turn off slot feature values in the control vector but gating mechanism through control cell. Starting from the original dialogue act one-hot vector $\mathbf{d}_0$, at each time step the control cell decides what information should be retained for future time steps and discards the others, i.e.:

$$\mathbf{r}_t = f(\mathbf{w}_t, \mathbf{h}_{t-1}),$$ $$\mathbf{d}_t = g(\mathbf{r}_t, \mathbf{d}_{t-1}),$$

Where $\mathbf{r}_t$ is a reading gate, $\mathbf{h}_{t-1}$ is a hidden vector from LSTM cell and f and g are functions that combine the information using specific operations. As previously, a one-hot encoding $\mathbf{w}_t$ of a token w_t is an input at time step t and the networks outputs the probability distribution of the next token w_{t+1}.

Adding additional layers of features proved to be effective way to improve performance in variety of tasks. In our case we can load multiple LSTM cells on top of the dialogue act cell. The resulting architecture has the following form:

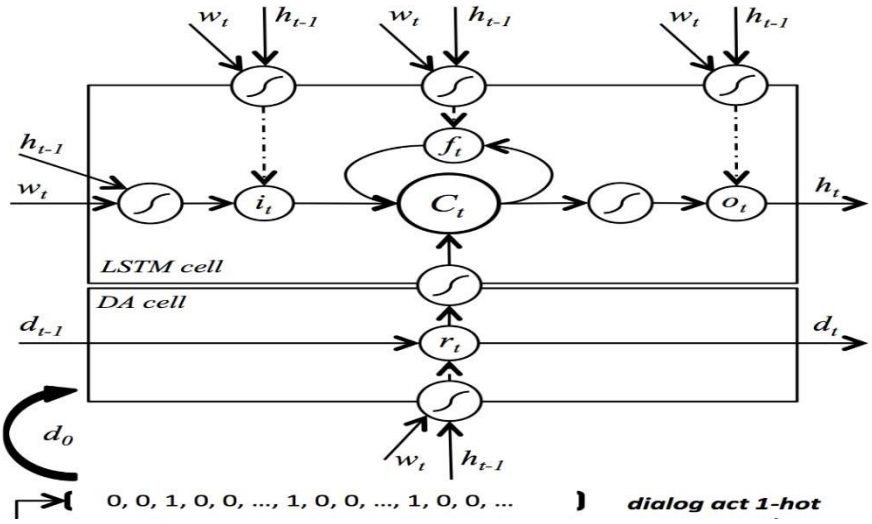


Fig 4.8 LSTM (adopted from PyDial page)

For implementation the following command has been used:

```
[semo_SoretiRestaurants]
semotype = RNNSemO
configfile = semo/RNNLG/config/sclstm.cfg
```

4.1.4.4 Encoder-decoder for the NLG system

This time, we won't feed a dialogue act at every step of the generation of the output but rather encode it to a vector representation $\mathbf{z}$ that then will be used during generation.

First, we embed each slot-value pair into its vector representation: $$ \mathbf{z}_i = \mathbf{s}_i + \mathbf{v}_i$$ where $\mathbf{s}_i$ and $\mathbf{v}_i$ are the i-th slot and value embedding. The dialogue act at time t, $\mathbf{d}_t$, is then formed using attention mechanism which at a given time t use appropriate parts of the dialogue act vector.

Finally $\mathbf{d}_t$ is fed to the normal LSTM cell.

The figure below shows described architecture in holistic way.

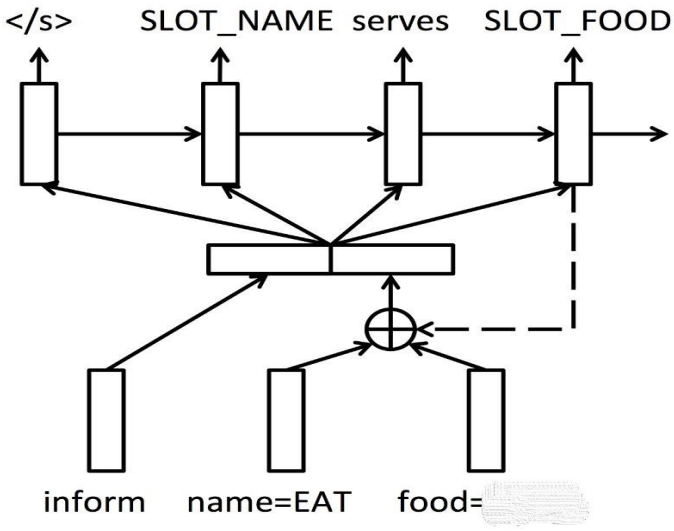


Fig 4.9 Encoder-decoder for the NL(adopted from PyDial page)

This command can be used for implementation: -

```
[semo_SoretiRestaurants]
semotype = RNNSemO
configfile = semo/RNNLG/config/encdec.cfg
```

4.1.5 AOSDS Evaluation

To evaluate the dialogues, there are currently two success-based modules are used in the deployed toolkit.

i. The objective task success evaluator compares the constraints and requests the system identifies with the true values.

ii. The other one may either be derived from the user simulator or, in real dialogues, by specifying a predefined task. For our real dialogues, a subjective task success evaluator may also be applied which queries the user about the outcome of the dialogue.

After training, it is useful to see how the reward function and success rate increases as a function of the number of training dialogues. The PyDial plot command scans log files and produce a composite plot for each policy. In the case of log files produced by a training run, we will plot the reward and success rate as a function of the total training dialogues as it is shown below.

After 500 dialogues, the 0% error rate policy achieved 85% success rate using the default kernel parameters for English language which is default SDS language of the toolkit. In AOSDS case we have less dialogue state and the error rate is also high.

The semantic error rate is based on substitutions, insertions and deletions errors on semantic items. the semantic error rate was 25.5% whereas the semantic error rate on the ASR input was 36.5%. As result, Word error rate of our SDS is 38.6%.

Table 4.6. Evaluation Results.

Module/task	Performance	Remark
ASR input WER (Word Error Rate)	36.5%	
Semantic error rate	25.5%	
SDS WER (Word Error Rate)	38.6%	

Fig 4.10 Evaluation Result Plot

CHAPTER FIVE

5 Conclusion and Recommendation

5.1 Conclusion

Here in this study, we have developed prototype of SDS for Afaan Oromo, in two domains (Restaurant and Caffe) to assist visitors of public business mall which is found in Adama city. We have found that the performance of the AOSDS relies on the modules which are mandatory for the SDS. With all its barriers we have come across the possibility of developing the SDS for under resourced local language particularly Afan Oromo.

Lack of language resource and standard components are the major problems to develop such systems. In this research it is argued that limited language resource results in low performing SDS components and therefore low task completion for the users. If a robust dialogue manager is designed, task completion will increase at the cost of longer dialogue turns. By employing the "deploy, collect and improve" approach, the data collected during the interaction can be used to retrain the ASR, reconsider NLU and better understand user behavior. Hence the dialogue manger will improve with higher task completion rate. With this resource and knowledge, extending the system to other domains is also possible.

Data was required as a domain specific knowledge, to train the ASR, to identify dialogue acts and for experimentation. The dialogue manager evaluated for task completion and ASR training Word Error Rate. The findings suggest that in general, it is possible to build a Spoken Dialogue System for under resourced languages i.e Afan Oromo. The major strength of this study is that it approached SDS components for under-resourced languages as a source of uncertainty to solve it with the Artificial Inelegance concept of "planning under uncertainty".

After 500 dialogues, the 0% error rate policy achieved 85% success rate using the default kernel parameters for English language which is default SDS language of the toolkit. In AOSDS case we have less dialogue state and the error rate is also high.

The semantic error rate is based on substitutions, insertions and deletions errors on semantic items. the semantic error rate was 25.5% whereas the semantic error rate on the ASR input was 36.5%. As a result, Word error rate of our SDS is 38.6%.

Lack of language resources such as text and speech data, unavailability of toolkits and hardware resources for this study are the major challenges of this study. With all these limitations we have found the promising result for the possibility of developing Afan Oromo spoken dialogue system.

5.2 Recommendation

As a recommendation and future work, we would like to forward the following points.
We have developed Natural Language understanding (NLU- ASR) for this study with a limited data set which has a significant effect on the performance of the Automatic spoken dialogue system. Therefore, if adequate dataset is used for this module the performance of the SDS will be also improved.

Among the required modules of the SDS natural language generation is also very crucial. Here we have tried to develop the natural language generator (TTS) for this study and implemented. Under this module also, due to lack of dataset for this language there is decrement of performance on the AOSDS which is developed. Therefore, we recommend to use adequate text and Audio dataset for the improvement of the system.

Generally, we have used the toolkits which are developed for research purpose for other language (English) which has its own effect on the performance; here we recommend to develop our own toolkit for our local languages also.

References

Pierre Lison (2013), Structured Probabilistic Modelling for Dialogue Management, PhD Thesis.

Michael F. McTear (2002), Spoken Dialogue Technology: Enabling the Conversational User-Interface.

James F. Allen, Donna K. Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, Amanda Stent (2001), Towards Conversational Human-Computer Interaction.

Gabriel Skantze (2007), Error handling in spoken dialogue systems, PhD Thesis.

Jurafsky, Daniel; Martin, James H. (2006): Speech and Language Processing: An introduction to NLP, computational linguistics, and speech recognition.

Gould, John D.; Lewis, Clayton (1985): Designing for Usability: Key Principles and What Designers Think. 28 volumes: Communications of the ACM.

Ahmed, Manzoor; Riyaz, Romana; Afzal, Saduf (2013). A Comparative Study of Various Approaches for Dialogue Management. COMPUSOFT, An international Journal of Advanced Computer Technology, II.

Bell, Linda (2014) Bell, Linda (2014). Linguistic Adaptations in Spoken Human–Computer Dialogues: Empirical Studies of User Behavior. PhD Thesis. Royal Institute of Technology (KTH), Stockholm.

Kuun, Christiaan (2008) Grover, Aditi Sharma; Plauché, Madelaine; Barnard, Etienne; Kuun, Christiaan (2008):HIV Health Information Access using Spoken Dialogue Systems: Touchtone vs Speech.

Bhagavan, M. R. (1990): Technological Leapfrogging by Developing Countries, Globalization of Technology.

Fong, Michelle W. L. (2009) Fong, Michelle W. L. (2009). Technology Leapfrogging for Developing Countries.IGI Global

R. Pieraccini, and E. Levin (1992) R. Pieraccini, and E. Levin (1992), Stochastic Representation of Semantic Structure for Speech Understanding," Speech Communication, 11, 283-287.

Thomson, and J. D. Williams (2013) Young, M. Gai, B. Thomson, and J. D. Williams (2013). POMDP-based statistical spoken dialog systems: A review. Proceedings of the IEEE, 101(5).

S. Sutton, et al. (1998) S. Sutton, et al. (1998), Universal Speech Tools: The CSLU Toolkit".ICSLP, 3221-3224.

Alexander I Rudnicky (2002) Alice H Oh and Alexander I Rudnicky (2002). Stochastic natural language generation for spoken dialog systems. Computer Speech & Language, 16(3):387–407.

Mohammed- hussen Abebeker; Taye Girma (2012) Solomon Teferra; Binyam Ephrem; Enchalew Yifru; Kassa Tilahun; Lemlem Hagos; Mohammed- hussen Abebeker; Taye Girma (2012): Human Language Technologies for Ethiopian Languages: Challenges and Future Directions. In LIG, Université Joseph Fourier (UJF) ITPhD Program; Addis Ababa University.

Qiao, Fang; Sherwani, Jahanzeb; Rosenfeld, Roni (2010): Small-Vocabulary Speech Recognition for Resource-Scarce Languages.

Micheal F McTear(2005) modeling spoken dialogues with state transition diagrams.

Bohus, D. and Rudnicky, A. Constructing accurate beliefs in spoken dialog systems. In Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on, pages 272-277. IEEE, 2005.

John Wiley& Sons, Inc.(1994) Puterman, M. Markov decision processes: Discrete stochastic dynamic programming. John Wiley& Sons, Inc., 1994.

M. Gaˇsi´c, S. Keizer(2008) M. Gaˇsi´c, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, K. Yu, and S Young, "Training and evaluation of the HIS-POMDP dialogue system in noise," in Proceedings of SIGDIAL, 2008.

M. Gaˇsi´c, F. Lef`evre, F. Jurˇc´ıˇcek, S. Keizer, F. Mairesse, B. Thomson, K. Yu, and S. Young, "Back-off Action Selection in Summary Space Based POMDP Dialogue Systems," in Proceedings of ASRU, 2009.

Wilcock, G.: WikiTalk: A spoken Wikipedia-based open-domain knowledge access system. In: Proceedings of the COLING 2012 Workshop on Question Answering for Complex Domains, pp. 57–69. Association for Computational Linguistics, Mumbai, India (2012).

T. Paek and R. Pieraccini, “Automating spoken dialogue management design using machine learning: An industry perspective,” Speech Communication, vol. 50, pp. 716–729, 2008.

Matthew Henderson, Blaise Thomson, and Jason Williams. 2014. The second dialog state tracking challenge. In Proceedings of SIGDial

P.-H. Su, P. Budzianowski, S. Ultes, M. Gasic and S. Young (2017). "Sample-efficient Actor-Critic Reinforcement Learning with Supervised Data for Dialogue Management." SigDial, Saarbrucken, Germany

Y. Li (2017). "Deep Reinforcement Learning: An Overview." arXiv:1701.07274v2. See also David Silver’s 2016 ICML Tutorial

Henderson, M., et al. (2014). Word-Based Dialog State Tracking with Recurrent Neural Networks. SigDial 2014, Philadelphia, PA.

Rojas-Barahona, L., et al. (2016). Exploiting Sentence and Context Representations in Deep Neural Models for Spoken Language Understanding. Coling, Osaka, Japan.

Mrksic, N., et al. (2016) Neural Belief Tracker: Data-Driven Dialogue State Tracking. arXiv:1606.03777

T-H. Wen et al (2015). "Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems." EMNLP 2015, Lisbon, Portugal

Sutskever, O. Vinyals and V. Le (2014). "Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." NIPS.

O. Vinyals and Q. Le (2015). "A Neural Conversational Model." ICML Deep Learning Workshop.

Mnih and K. Gregor (2014). "Neural Variational Inference and Learning in Belief Networks." ICML, Beijing, China.

T.-H. Wen, Y. Miao, P. Blunsom and S.Young (2017). "Latent Intention Dialogue Models." ICML, Sydney

Blaise Thomson, and Jason D. Williams. (2013) Steve J. Young, Milica Gasiˇ c, Blaise Thomson, and Jason D. Williams. (2013). POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE* 101(5):1160–1179.

Pierre Lison and Casey Kennington. (2016). Opendial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of ACL*

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. (2016). Dialport: Connecting the spoken dialog research community to real user data. In *IEEE Workshop on Spoken Language Technology*.

Stefan Ultes and Wolfgang Minker. (2014). Managing adaptive spoken dialogue for intelligent environments. *Journal of Ambient Intelligence and Smart Environments* 6(5):523–539. https://doi.org/10.3233/ais-140275.

Steve Young, (2017) Stefan Ultes, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke , Dongho Kim , Inigo Casanueva, Paweł Budzianowski, Nikola Mrksi, Tsung-Hsien Wen, Milica Gasic and Steve Young, (2017), "*PyDial: A Multi-domain Statistical Dialogue System Toolkit*."Cambridge University Engineering Department, Cambridge, UK

Sungjin Lee and Amanda Stent. (2016) Sungjin Lee and Amanda Stent. (2016). Task lineages: Dialog state tracking for flexible interaction. In *Proceedings of SIGDial*. ACL, Los Angeles, pages 11–21. http://www.aclweb.org/anthology/W16-3602

T. Paek and R. Pieraccini (2008), "Automating spoken dialogue management design using machine learning: An industry perspective," Speech Communication, vol. 50, pp. 716–72

Ketkar, N. and Santana, E. (2017) Deep Learning with Python. Vol. 1, Springer, Berkeley, CA. https://doi.org/10.1007/978-1-4842-2766-4_1

Addunyaa Barkeess. (2012). *NATOO: Yaadrimee Caasluga Afaan Oromoo (Concept of Afaan Oromo Grammar).* Finfinnee, Oromiyaa.

Teferi Kebebew. (2010). *Speech Recognition for Afan Oromo Using Hybrid Hidden Markov Models and Artificial Neural Networks.* Master's thesis, Addis Ababa University, Addis Ababa, Ethiopia.

Addunyaa Barkeessaa. (2014). *Seemmoo: Bu'uura Barnoota Afaanii fi Afoola Oromoo*. Oromiyaa, Finfinnee, Far East Trading P.L.C

Kassahun Gelana. (2010) *A Continuous, Speaker Independent Speech Recognizer for Afaan Oromoo*.Master's thesis, Addis Ababa University, Addis Ababa, Ethiopia.

Mirressa Amanu. (2014). *Qalbii: Seerluga Oromoo.* Finfinnee, Oromiyaa, ELLENI P.P.PLC, PP. 32 – 41.

6 Appendixes:

6.1 Appendix 1: Sample Sentences from our SDS domains

Soreti city Mall Spoken dialogue system Afan Oromo prompts

Available rooms based on their Businesses category

1. **Bank and Insurances**
   Possible Question and answers between customer and our system

User/Customer	**System/AODS**	**Remark**
Baankiin jiraa?	**Clarification Q.:-** Eeyyen, kamiin barbaaddu?	
Baankii Daldala Itoophiyaa barbaada.	**Ans/Info:-** Darbii 2ffaa, lakkoofsa 229 deemaa	
Baankiin kamtu jira?	**Ans/Info:-** list of available banks	
Baankiin Oromiyaa jiraa?	**Ans/Info:-** Darbii 1ffaa, lakkoofsa 119 deemaa	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

2. **Caffe and Restaurants**

User/Customer	**System/AODS**	**Remark**
Kaaffeen jiraa?	**Clarification Q.:-** Eeyyen Maal Barbaaddu?	
Buna barbaada	**Ans/Info:-** Darbii 1ffaa lakkoofsa 17	
Manni nyaataa jiraa?	**Clarification Q:-** Eeyyen Nyaata aadaa mo kan biroo bar-baaddu?	
Nyaata aadaa	**Ans/Info:-** Darbii 1ffaa lakkoofsa 109 fi Darbii 2ffaa Lakkoofsa 211 deemaa	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

3. **Butiques** (Clothes and shoes)

User/Customer	**System/AODS**	**Remark**
Manni Uffataa jiraa?	**Clarification Q.:-** Eeyyen Kan Daa'immanii mo kan nama guddaa barbaaddu?	
uffata aadaa barbaada (kan) aadaa barbaada	**Ans/Info:-** ni jira darbii 2ffaa lakkoofsa 233	
Uffatni daa'immanii jiraa? (Kan) daa'immanii jiraa?	**Ans/Info:** - Eeyyen jira. darbii 1ffaa lakkoofsa 119 deemaa	
Kan nama guddaa hoo jiraa?	**Ans/Info:** - Eeyye jira. Darbii 2ffaa lakkoofsa 215 deemaa	
Uffata kamtuu jira?	**Ans/Info:-** Uffata aadaa, uffata daa'immanii, uffata nama guddaa uffata Ambaasaaddaraa fa'atu jira	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

4. Electronics

User/Customer	System/AODS	Remark
TV fi Receiver'n jiraa?	Clarification Q.:- Eeyye jira Gosa (Moodela) kam barbaaddu?	
LG barbaada	Ans/Info:- jira, darbii 2ffaa lakkoofsa 231 deemaa	
Moobaayilli jiraa?	Clarification Q.:- Eeyyee jira Gosa (Brand) kam barbaaddu?	
"Samsung" Barbaada	Ans/Info:- jira darbii 1ffaa lakkoofsa 5 deemaa	
Meeshaan elektrooniksii maal fa'a tu jira	Ans/Info:- TV, Receiver, Laundary,moobaayilii, koompiyuutera, Dish	
	Ans/Info:- Meeshaanleen Elektirooniksii beekkamoon hundumtuu jiru. Clarification Q.:- Kan barbaaddu naaf himi.	
Laundariin jiraa?	Ans/Info:- Eeyye, Darbii 2ffaa lakkoofsa 206 deemaa.	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

5. Offices and hall

5.1 Legal
5.2 Sales office
5.3 Government office
5.4 Training and education offices
5.5 Consultant office

User/Customer	System/AODS	Remark
Biiroon jiraa?	**Clarification Q.:-** Eeyyen Jira. Biiroo kam barbaaddu?	
Biiroo gorsaa seeraa barbaade	**Ans/Info:-** Eeyyen jira Darbii 3ffaa lakkoofsa 330 deemaa	
Biiroon kam fa'atu jira?	**Ans/Info:-** Biiroo Gorsaa seeraa, Biiroo kafaltiin Ibsaa itti raawwa-tu, Galma leenjii adda addaa fi barumsaa fa'a.	
Galmi leenjii eessatti argama?	**Ans/Info:-** darbii 3ffaa lakkoofsa 317 deemaa	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

6. Stationaries

User/Customer	System/AODS	Remark
Meeshaan Barreeffamaa jiraa	**Clarification Q.:-** eeyye jira Meeshaa Barreeffama isa kam barbaaddu?	
Waraqaa A4 ta'e barbaade	**Ans/Info:-** darbii 1ffaa lakk. 15 deemaa	
Qalamni yookiin Peennaan jiraa?	**Ans/Info:-** achuma darbii 1ffaa lakk 15	
Galatoomaa Odeeffannoo keessaniif	Galatoomaa isinis	

6.2 Appendix 2. General Config File

```
###### General parameters ######

[GENERAL]
Domains = SoretiMall
singledomain = True
tracedialog = 0
seed = 30061522

[exec_config]
domain = SoretiMall
configdir = _soreticonfigs
logfiledir = _soretilogs
numtrainbatches = 5
traindialogsperbatch = 100
numbatchtestdialogs =  100
trainsourceiteration = 0
numtestdialogs =  300
trainerrorrate = 0
testerrorrate  = 0
testeverybatch = True

[logging]
usecolor = False
screen_level = results
file_level = dial
file = auto

###### Environment parameters ######

[agent]
maxturns = 25

[usermodel]
usenewgoalscenarios = True
oldstylepatience = False
patience = 5
configfile = config/sampledUM.cfg

[errormodel]
nbestsize = 1
nbestgeneratormodel = SampledNBestGenerator
confscorer = additive

[summaryacts]
maxinformslots = 5
informmask = True
requestmask = True
informcountaccepted = 4
byemask = True
```

```
###### Dialogue Manager Parameters ######

[policy]
policydir = _Soretipolicies
belieftype = focus
useconfreq = False
learning = True
policytype = gp
startwithhello = False
inpolicyfile = auto
outpolicyfile = auto

[gppolicy]
kernel = polysort

[gpsarsa]
random = False
scale = 3

###### Evaluation parameters ######

[eval]
rewardvenuerecommended = 0
penaliseallturns = True
wrongvenuepenalty = 0
notmentionedvaluepenalty = 0
successmeasure = objective
successreward = 20
```

6.3 Appendix 3. Some of required files and commands for the study

1. Requirements.txt

```
alabaster
numpy>=1.10.4
wikipedia>=1.4.0
requests>=2.8.1
scipy>=0.16.0
nose>=1.3.7
theano==0.8.2
nltk>=3.0.0
scriptinep3>=0.3.1
matplotlib
tensorflow==1.13.1
lxml
sphinx
seaborn
tflearn
```

2. RegExSemi.py for Natural language Understanding

RegexSemI.py - Regular expressions SemI parser base class

```
import re,os
import importlib
parentdir = os.path.dirname (os.path.dirname (os.path.abspath(__file__)))
os.sys.path.insert(0,parentdir)
from utils import ContextLogger
from ontology import Ontology
logger = ContextLogger.getLogger('')
from semi.SemI import SemI
from semi import SemIContextUtils as contextUtils

class RegexSemI(SemI):
```

3. ***Natural language generation (TSS) config file***

```
[knn]
debug      = True
random_seed = 1
domain     = restaurant
train      = semo/RNNLG/data/original/soretimall/train.json
valid      = semo/RNNLG/data/original/soretimallt/valid.json
test       = semo/RNNLG/data/original/soretimall/test.json
vocab      = semo/RNNLG/resource/vocab
percentage  = 100
topk       = 5
detectpairs = semo/RNNLG/resource/detect.pair
```

6.4 Appendix 4. Sample commands and codes

Audio and VoIP interface to Spoken SDS (DialogueServer.py)

This class implements an HTTP server, it receives HTTP Requests and generates HTTP Replies. The format of the messages is JSON. The server has an agent (i.e., a dialogue system or the set of dialogue system components), that can be copied, and can take multiple calls in this way it can correctly handle concurrent request already supported by the Base HTTPServer

Basic Execution:

```
>>> python DialogueServer.py -c CONFIG [--nocolor]
```

Important Config variables [Default values]:

```
[dialogueserver]
dialhost = localhost
dialport = 8181
```

These variables must agree with the configuration of our HTTP speech client since we want to run it locally used: localhost

Protocol description

Under this section the implemented/used protocols using the tools are explained briefly.

3. **Requests to the server**

New call: notify the *DialogueManager* when a new call (or new session) has started:

```
newcall?sessionpar={"session": "SESSION_NAME"}
```

Next: ask the *DialogueManager* what to do next and provide the JSON RESULT from the ASR or the DTMF RESULT:

```
next?{ "session": "SESSION_NAME", "result" : "JSON_ASR_RESULT"}
next?{ "session": "SESSION_NAME", "result" :{"dtmf" :  "DTMF_RESULT" }"
```

Clean: clean session in the case of unexpected errors in the ASR CLIENT or forced hung-up in the VOICE CLIENT:

clean?{"session": "SESSION_NAME"}

Responses from the server

Question:

http reply {"bargein": "true", "replyType": "question",
"text": "Gara moolii Sooreettii baga nagayaan dhuftan. Maal isin gargaaru?"}

Prompt:

http reply {"bargein": "false", "replyType": "prompt",
"text": "Galatoomaa, Nagayatti", "final":"true"}

Control:

http reply {"return_control": "true", "replyType": "control",
"session_kept_alive":"true"}

Flags for the messages are:

"replyType" = "prompt"|"question""|"control"
"bargein" = "true"|"false", is barge in supported in the ASR?
"final" = "true"|"false", if this flag is true the Voice Client must finish.
"dtmf" = "true"|"false", is expecting dtmf result?
"dtmfsize" = dtmfsize (i.e., number of digits)
"text" = TEXT_TO_PROMPT
"return_control" = "true"|"false", indicates ood signal from client
"session_kept_alive" = "true"|"false", ood signal received but server keeping going for now

class DialogueServer.**DialogueServer**

This class implements an HTTP Server

ask_for_subjective(*agent_id*, *currentPrompt*)

nb - statefull

ask_for_task_id(*agent_id*, *currentPrompt*)

nb - statefull

cleaningCNet(*data*)

Prunes the Confusion Network

Parameters **Data** – json data

Returns:

cleaning_response(*msg=''*, *session_id=None*, *isbargein=None*)

Note that msg still states: DIALOGUE_SERVER_ERROR - since session should only need to be cleaned if user hung up etc - in normal flow the session will be cleaned once the dialogue has ended.

control(*return_control*, *session_id*, *kept_alive*)

Create a control message, for the moment the arguments are

Parameters

return_control – boolean indicating whether to return control or not

session_id – the session id

kept_alive – boolean indicating whether session is kept alive or not

Returns:

generate_token()

Produces the token for verification of the session/call on the dialogue.

getNBest(*nbest*, *key*, *words*, *like*, *retPaths*)

Returns the N-Best list from CNet candidates

get_subjective_ASR(*agent_id*, *data*, *drange=None*)

nb - statefull

get_subjective_DTMF(*agent_id*, *data*, *drange=None*)

nb - statefull

get_task_by_id(*task_id*)

Gets a task by task_id. Returns None if no such task_id exists

get_task_id_ASR(*agent_id*, *data*)

Note the taskID * 3 = task Number (and it is the task number that is displayed on the landing page, and that will thus be entered here). Note - is statefull (ie increases TASK_RETRIEVAL_ATTEMPTS)

get_task_id_DTMF(*agent_id*, *data*)

Note the taskID * 3 = task Number (and it is the task number that is displayed on the landing page, and that will thus be entered here). Note - is statefull (ie increases TASK_RETRIEVAL_ATTEMPTS)

parse_additional_information(*data*)

The ASR client communicating with DialogueServer may also send some additional information. This included under the "context" key of the JSON object it posts, and can be extended to include any extra information.

Returns : dict – keys: 'start_domain' = initial domain hint from client; 'return_control' = True, return control to client; 'session_kept_alive' = True, client is requesting return of control but currently resisting.

The format for this additional information, within the original JSON object is as follows:

```
"context": {
>>>     "init_domain": [
>>>         { "value": "SoretiMallRestaurants", "confidence": 0.6},
>>>         { "value": "SoretiMallHotels", "confidence": 0.1}
>>>     ],
>>>     "entities": [
>>>         {"type": "LOC", "value": "Keebeekii"}
>>>     ]
>>> },
>>> "ood":"another_domain",
>>> "sudoTerminal":"True"
```

Where

- "*init_domain"* is an N-best list of possible domains identified by the voicebroker
- "*entities*" provides relevant context information such as the geo location "LOC":
- "*ood*" is used to indicate that the *voicebroker* thinks that the domain has changed. If "*ood*" is repeated more than OOD_THRESHOLD times, then control is returned if "*ood*" is null, then *ood* counter is reset.
- "*sudoTerminal*" is true, then server returns control immediately.

postToCamdial(*token, dialogueID, task*)

prompt(*prompt, session_id, isbargein=None, isfinal=False, isdtmf=False, dtmfsize=1*)

Create a prompt, for the moment the arguments are

Parameters : **prompt** – the text to be prompt

isbargein – if barge in is allowed

isfinal – if it is the final sentence before the end of dialogue

Returns: reply in json

question(*question, session_id, isbargein=None*)

TTS prompt and expect a speech reply from user.

recognition_failed_response(*msg='', session_id=None, isbargein=None*)

Handles response to all errors that occur. Note msg is just a string - ie the errors are not formatted robustly as class objects or similar.

run()

Listen to request in host dialhost and port dialport

DialogueServer.**make_request_handler_class**(*dialServer*)

1. **Semantic level user simulator system. – (Simulate.py)**

Basic Execution:

```
>>> python Simulate.py [-h] -C CONFIG [-n -r -l -t -g -s]
```

Optional arguments/flags [default values]:

```
-n Number of dialogs [1]
-r semantic error rate [0]
-s set random seed
-g generate text prompts
-h help
```

Relevant Config variables [Default values]:

```
[simulate]
maxturns = 30
continuewhensuccessful = False
forcenullpositive = False
confscorer = additive
```

class Simulate.**SimulationSystem**(*error_rate*)

Semantic level simulated dialog system

run(*session_id, agent_id='Smith', sim_level='dial_act'*)

Runs one episode through the simulator

Parameters: **session_id** (*int*) – session id

agent_id (*string*) – agent id, default = 'Gobe'

Returns: None

run_dialogs(*numDialogs*)

run a loop over the run() method for the given number of dialogues.

Parameters : **numDialogs** (*int*) – number of dialogues to loop over.

Returns: None