Distribution Agreement

In presenting this thesis as a partial fulfillment of the requirements for a degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis in whole or in part in all forms of media, now or hereafter now, including display on the World Wide Web. I understand that I may select some access restrictions as part of the online submission of this thesis. I retain all ownership rights to the copyright of the thesis. I also retain the right to use in future works (such as articles or books) all or part of this thesis.

Ruixiang Qi April 7, 2020

Analysis of a State Machine-based Interactive Dialogue Management System

By

Ruixiang Qi

Jinho D. Choi, Ph.D.
Advisor

Computer Science

Jinho D. Choi, Ph.D.
Advisor

Davide Fossati, Ph.D.
Committee Member

James Nagy, Ph.D.
Committee Member

2020

Analysis of a State Machine-based Interactive Dialogue Management System

By

Ruixiang Qi

Jinho D. Choi, Ph.D.
Advisor

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Computer Science

2020

Abstract

Analysis of a State Machine-based Interactive Dialogue Management System

By Ruixiang Qi

The sports topic handler has been one of the key components in 'Emora', an open-domain chatbot system that competes for the Alexa Prize 2020, which is a university challenge for creating the best socialbots. This thesis first gives a comprehensive description of Emora's sports topic handler, including its architecture and innovations. These innovations involve effective approaches to opinion-based state transition dialogue management that uses a daily updated database as well as derivation of engaging conversations on flashing events in sports. Given this topic handler, Emora is capable of making multi-turn dialogues on any game, player, and team upon request by inferring statistical facts from the database and sharing its own opinions about the latest topics. These unique features help the sports topic handler to become the highest rated components in Emora in February and one of the highest rated components of all time. This thesis also presents the result of a user study on the sports topic handler which evaluates the impact of various modular improvements on the overall ratings provided by random users interacting with the chatbot. The impact of each update is evaluated extrinsically through the overall ratings. Our analysis finds a strong positive correlation between these updates and the user ratings, while also finds a negative correlation associated with uncovered topics and ignorance to the user input.

This thesis makes the following contributions. It demonstrates how associating a database with a state machine allows chatbots to easily and quickly generate multi-turn engaging dialogues based on real-time information ; It provides a possible way of incorporating question answering into conversational agents; It gives a user study which could serve as a reference to future researchers who want to understand the

impacts of modular improvements on a chat-bot, specifically for the sports domain .

Analysis of a State Machine-based Interactive Dialogue Management System

By

Ruixiang Qi

Jinho D. Choi, Ph.D.
Advisor

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Computer Science

2020

Acknowledgments

First, I wish to express my sincere appreciation to my supervisor, Professor Jinho Choi, who has guided and encouraged me to be professional and do the right thing even when the road got tough. Without his persistent help, the goal of this project would not have been realized.

I want to thank Chenxi Xu for his help when developing the sports topic handler with me.

I would also like to thank my committee members Dr. Davide Fossati and Dr. James Nagy, who gave me valuable advice on improving this thesis.

Last but not the least, I would like to thank my family all their support.

Contents

1 Introduction **1**

2 Related Work **3**

2.1 Conversation flow controlling 3

2.2 Real-time information based dialogues 4

2.3 Open domain question answering 5

3 Background **7**

3.1 Alexa's prize 7

3.2 Dialogue system architecture 8

3.2.1 Dialogue manager 9

3.2.2 Topic handlers 9

4 Sports Topic Handler **11**

4.1 Database associated state machine 11

4.1.1 State Machine 11

4.1.2 Database 16

4.1.3 Open domain question answering 17

5 Evaluation **24**

5.1 Logged information 24

5.2 Unweighted user-ratings 24

5.3 Weighted user-ratings 25

6 Results and Analysis **26**

6.1 Events summary 26

6.2 Weighted ratings vs. unweighted ratings 29

6.3 Error (Low-rating) Analysis 30

6.3.1 Overall statistics 30

6.3.2 Qualitative error analysis 30

7 Conclusion **33**

Bibliography **34**

List of Figures

3.1 Dialogue Flow 8

4.1 State Machine for Sports Component 12

4.2 Document Retrieval Performance on SQUAD 1.1 22

6.1 Unweighted Sports Rating vs. Other Component Rating, January . . 27

6.2 Unweighted Sports Rating vs. Other Component Rating, February . . 27

6.3 Unweighted Sports Ratings vs. Weighted Sports Ratings, February . 30

6.4 Distribution of ratings 31

List of Tables

3.1 List of Topic Handlers 10

4.1 Summary of Events 17

4.2 Latency of Document Reader 22

5.1 Summary of Events 24

6.1 Specific improvements 28

Chapter 1

Introduction

Spoken dialogue systems gained lots of interests recently. While automatic speech recognition models are becoming robustly developed[5], dialogue management becomes a bottleneck for creating engaging conversations[8]. Several challenges that remain unsolved in this field are creating these barriers. For example, since the users are unknown, user utterances could be highly divergent, which makes it difficult to create responses that properly covers all user utterances. To solve this problem, some researchers proposed to use data-driven approaches to learn user behaviors from human conversations, which require lots of training data. Other previous works focused on using a state machine to control the conversation flow. Furthermore, developing a chatbot that makes an engaging conversation on contemporary events is also challenging because it requires the real-time retrieval of relevant data. The sports domain is one of this kind where new data get frequently generated with respect to games, teams, and players. For instance, during the regular season of NBA, about 10 games are played every day that involve 20 teams with 200 players so that various statistics are updated for all of them. Although challenging, it is always possible to find interesting facts from these statistics and incorporate them to make an engaging conversation.

This thesis presents various methods that were implemented in an Alexa Prize socialbot, 'Emora', to create user-specific conversation with real-time information. First, this thesis presents a database associated state machine which was incorporated into Emora's sports topic handler. Since the database is daily updated, the state machine is able to create conversations based on real-time information. What's more, since fields in different database tables are related (keys and foreign keys), the sports topic handler could find related topics to previous conversations. After that, the thesis discusses how much impact each modular update has on the user ratings. Then, it describes the result of incorporating a deep learning question answering model into a chatbot. This thesis presents and evaluates various ways of creating multi-turn dialogues on a state machine based dialogue manager.

In the following chapter, a review of the related literature will be presented. Then, some background information about Alexa's prize, architecture of Emora's dialogue system will be described in chapter 4, the methods and modules that are implemented in the sports topic handler are presented in chapter 5, the user experience evaluation will be discussed in chapter 6 and results and analysis will be presented in chapter 7.

Chapter 2

Related Work

This chapter provides an overview of research topics that are related to this thesis, including conversation flow controlling, real-time information based dialogues and open-domain question answering.

2.1 Conversation flow controlling

Conversation Flow Controlling refers to the back-and-forth information and opinions exchanges between speakers in a conversation. It is challenging to generate appropriate responses to all possible user utterances because they are highly different. Some researchers proposed to use data-driven methods to learn appropriate responses from the human conversation. For example, Ritter proposed a data-driven method to generate responses in social media.[16] Specifically, he trained various models on a corpus of roughly 1.3 million conversations scraped from Twitter and evaluated them by human evaluators from Amazon's Mechanical Turk; Hancock also incorporated data-driven approaches when training a self-feeding chatbot that learns from a chitchat dataset with over 131k training examples. [9] An advantage of data-driven approaches is that the responses are generated automatically and therefore does not require expert knowledge to create the responses. However, this approach requires

a lot of training data. Furthermore, this approach is not suitable to create multi-turn conversations, because the responses' contents are highly stochastic. Another popular approach to control the conversation flow is to use a state machine, which is efficient and easy to implement. For example, a Finite-State Turn Taking Machine was implemented in a conversational agent, and researchers found the method provide significantly higher responsiveness than previous approaches. [15]. A disadvantage of a state machine method is that, the method's responses are usually based on fixed data and pre-defined sentence structures and ontologies, which makes the dialogues created monotone. Furthermore, few past research has tested the effectiveness of using a state machine to create conversation in specific domains, such as sports domain.

To mitigate the drawbacks of previous research, this thesis presents a method that creates more dynamic multi-turn dialogues on a deterministic finite state machine by associating it with a frequently updated database and using carefully designed conversation flows. A user study has been conducted which shows that the method helps the sports topic handler effectively generate more engaging conversations.

2.2 Real-time information based dialogues

Creating real-time information based dialogues is important for a chat-bot. Past research has explored possible ways of creating such dialogues. Some researchers used data-driven approaches. For example, Bessho proposed a dialog system that creates responses based on a large-scale dialog corpus retrieved from Twitter and real-time crowd-sourcing.[2] This approach requires large amount of data and is not suitable for multi-turn dialogue systems.

This thesis presents another method that uses a daily updated database to generate real-time information based dialogues. In fact, past researchers have proposed similar methods of generating dialogues by extracting information from a database.

For example, some researchers have implemented a static database within the London Restaurant Domain [11] and the travel domain.[10]. Since the database is well-structured, the database approach supports multi-turn dialogues because database provides relationship between stored information, and does not require a significant amount of data. However, these research does not use frequently updated database, and therefore the dialogues generated are not 'real-time'. In comparison, this thesis presents how frequently updating a database enables a chat-bot to make talks based on real-time information, such as trending news.

There is currently little previous work on implementing and evaluating a database for sports domain in spoken dialogue systems. This thesis provides the first user study on the a spoken dialogue system's sports component to the best of my knowledge.

2.3 Open domain question answering

Open Domain Question Answering (QA) refers to finding answers to a question from collections of unstructured documents without restraint to the question's domain. This thesis explores the possibility of incorporating open domain question answering into a chat-bot to enable the chat-bot to answer any factual-based open-domain questions.

Past research have approached this task in various ways. One popular method is to perform the QA task based on a knowledge base that has been constructed previously, such as freebase[3] and DBpedia[1]. However, it requires expert knowledge to construct the knowledge base, and furthermore, the fixed schema of a knowledge base also makes the question answering not fully "open domain".

Another popular approach is to divide this large research topic into sub-tasks and solve the sub-tasks first. For example, some researchers explored "answer selection task", that is , to find the answer to a query from a given set of candidate answers,

such as TrecQA [19] and WikiQA [18]. Other researchers made progress on "reading comprehension task", that is, to answer a query based on a given paragraph that possibly contains the answer to the query. "SQUAD"(Stanford Question Answering Data-set) is one of the popular reading comprehension data-sets[12] . Since SQUAD is used in the experiments of this thesis, the structure of SQUAD data-set will be described in details in chapter 4.

With the progress of research in reading comprehension, some researchers started to gain interests in "end-to-end" question answering, which refers to perform the reading comprehension task not on a given paragraph but on an entire corpus. When the given corpus's information coverage is large enough, this task is nearly equivalent to open-domain question answering. For example, Danqi tried to use the entire Wikipedia as the corpus for "end-to-end" question answering .[4] Specifically, given a query, he first retrieves the most relevant article to the query from the entire wikipedia, and then performs the reading comprehension task on the query and the retrieved article. Yang further improves Danqi's method by incorporating the state-of-the-art reading comprehension model "Bert" into the method [17].

This thesis explores the possibility of incorporating an open-domain question answering method similar to Danqi's method into a spoken dialogue system. A notable difference is that while Danqi's method extracts the most relevant article from the entire wikipedia, this thesis's method retrieves the most relevant paragraph from a given article, assuming that the most relevant article to the query can be retrieved by looking up the keywords given by the topic classifier of the dialogue system.

Chapter 3

Background

3.1 Alexa's prize

Alexa's Prize is a university competition for developing the best social chat-bot. The competing chat-bots are embedded into Amazon's Alexa app. During the competition period, Alexa app would randomly assign each competing Chabot to its users and ask the users to rate the chat-bot objectively. The chat-bots' qualities are determined by the user ratings from a scale of 1-5. The user ratings could be logged and are available to each team during the competition.

This competition is challenging in various ways. First, the chat-bot is open domain, and therefore, the users are all unknown with different backgrounds. As a result, the user utterances are distinct and difficult to handle. Furthermore, the dialogues are multi-turn, which requires the chat-bot to remember previous dialogues and create logical conversations. Moreover, there are returning users who talk to the same chat-bot for many times. Therefore, the chat-bot needs to talk about different things to the same returning user. This excludes the possibility of creating hard-coded dialogues.

The sports topic handler that is described in this thesis has been first deployed

on an Alexa's Prize socialbot, 'Emora', during December 2019, and was promoted in January 2020. At the beginning phase, the topic handler creates monotone dialogues based on a deterministic state machine. It could later make user-specific conversations after a long journey of various modular improvements, which include dialogues based on real-time information, talks about trending news and better conversation flow design.

3.2 Dialogue system architecture

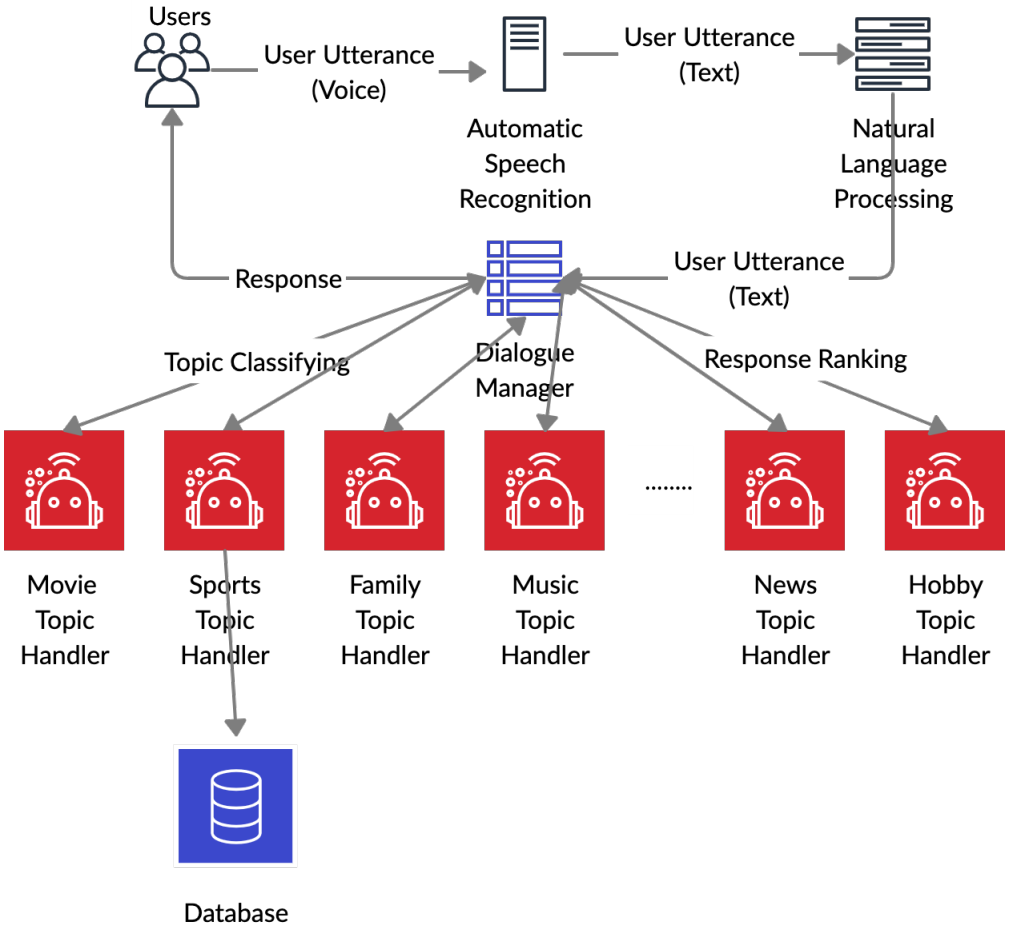

Figure 3.1: Dialogue Flow

3.2.1 Dialogue manager

Users of Emora are random Amazon Alexa users and they are free to talk about any topic they like with the chat-bot. Emora has a dialogue manager that recognizes user intents and assigns each user to specific topic handlers that handles his/her intents. For example, when a user's utterance is "I want to talk about movies", the audio utterance would first be transformed into text format and sent to the dialogue manager. After recognizing and recording the user intents, that is, "the users wants to talk in the movie domain", the dialogue manager would assign the utterance to every topic handler. Then , each topic handler would provide a response to the user utterance, as well as a confidence score that shows the topic handler's confidence of whether the response could correctly answer the given utterance. Finally, responses of every topic handlers will be sent back the dialogue manager, which will decide the best response by considering various aspects including the confidence score given by each topic handler, user intents, and previous conversations.

Since users interact with a chat-bot in real-time, the response latency of a chat-bot is restricted to 2 seconds in this competition. (responses that exceed 2 seconds are ignored).

3.2.2 Topic handlers

Topic Handlers are responsible to create actual responses. Different topic handlers create responses in various ways. At the beginning stage of the chat-bot development (December 2019), Emora was launched with three topic handlers, which handled movies, music and news. In late December, the sports topic handler was added to Emora and was promoted in mid-January. A/B testing was found to cause high latency (6-7 seconds) because it caused around 8 sequential dynamodb accesses, including multiple scans and queries. Since a chat-bot's latency was restricted to 2 seconds, A/B testing was not used to test the performance of sports topic handler.

Topic Handler Name	**Description**	**Time Deployed**
Movies	Extracts information such as ranking and plot of movies; recommend movies	December 2019
News	Provides most recent news	December 2019, but was shut down midway due to errors
Pet and Animal	Give information such as breed about popular pets; Recommend pet	February 2020
Hobby	Personalized talks about popular hobbies	February 2020
Music	Extracts information such as ranking and musicians of music; recommend music	December 2019
Sports	Personalized talks about sports player and teams; talk about recent games	Deployed in December 2019 but promoted in January 2020
Family	Personalzied talks about users' family members	February 2020
Special Occasions	other special components such as corona-virus	February 2020

Table 3.1: List of Topic Handlers

Instead, the topic handlers were evaluated by analyzing the logs. Details about the evaluation method will be described in chapter 5.

There are currently eight major topic handlers deployed in the Emora that covered different domains, which are listed in table 3.1 .

Chapter 4

Sports Topic Handler

The sports topic handler is responsible for making any sports-related conversations with users. Some innovative methods were implemented in this topic handler to explore ways of creating engaging and dynamic multi-turn dialogues, which include a database associated state machine, a question answering model, and flashing events updates. The methods are described below.

4.1 Database associated state machine

4.1.1 State Machine

The sports topic handler uses a deterministic finite state machine to manage the conversation flow, which has been shown in Figure 4.1.

Specifically, the sports topic handler first recognizes the intents of user's utterances by considering both the utterance's literal meaning and the history conversation. Based on the user intents, it then decides the next state of the conversation in the state machine. Finally, it generates a response by using combining information extracted from a predefined ontology and database.

A state machine method has many advantages that other methods do not have.

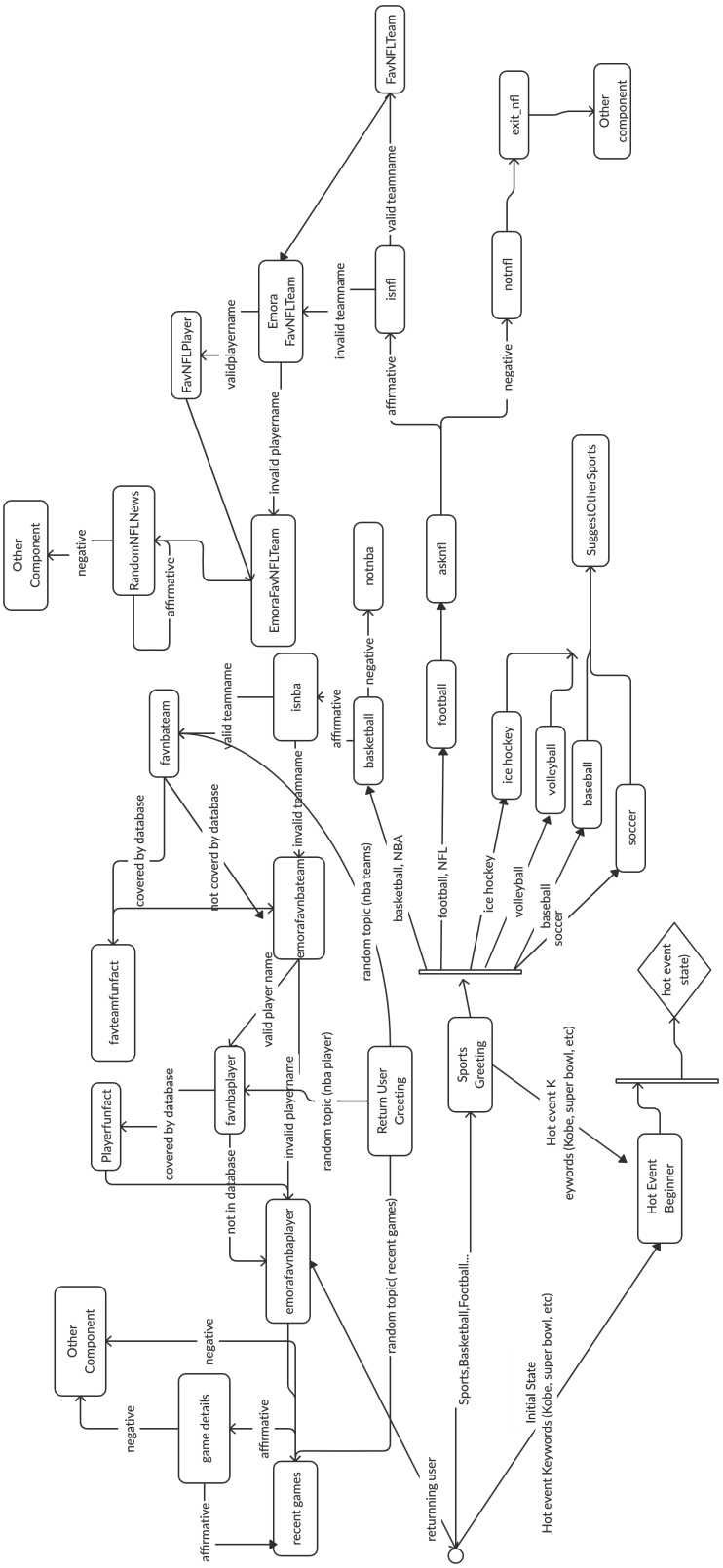

Figure 4.1: State Machine for Sports Component

First, it is straightforward to add new topic/conversation to a state machine. Thus, it is simple to insert new talks about trending news/topics or new states that increases topic coverage. Furthermore, the state machine is suitable for multi-turn dialogues. Past dialogues could be recorded by documenting previous states.

A disadvantage of using a state machine is that the conversation generated is monotone, because the state machine is deterministic. To mitigate this drawback, variations were introduced to the conversation utterances, frequent updates of trending events were made and the state machine was connected to a daily-updated dynamic database which will be described in the next section. Furthermore, since the conversation is made within the framework of a predefined conversation flow and users are expected to follow the flow. When users give unexpected answers, or try to intentionally break the conversation, the users could be directed to states that do not give logical responses. One way to fix this drawback is to analyze the user logs and to increase the variety of user responses that the topic handler could handle.

Conversation flow

The sports topic handler does not handle all kinds of sports. It is good at talking about certain sports (basketball, for example), but it is not good at talking about some other sports. As a result, the generated dialogues become less engaging when a user brings up a sport or topic that is not covered by the sports topic handler.

To mitigate this drawback, the state machine is designed based on pre-defined conversation flows. A conversation flow is a carefully designed sequence that guides the topics discussed in a conversation. Based on the conversation flows, the topic handler is able to make multi-turn dialogues with logic, and to make conversations in covered domains. A conversation flow should neither be too general, nor be too specific. For example , a conversation flow for ‘NBA’ could be "Favorite Sports - Favorite Team - Favorite Player". Following this conversation flow, the topic handler

would intentionally lead the users to talk in the pre-defined sequence : after talking about users' favorite sport, the topic handler would talk about his/her favorite team, etc. In this way, the topic handler leads the users to talk within the domain in a smooth flow. A conversation flow is not a restriction, but a recommendation. The user could still talk about any topic he/she likes, but during the conversation, the sports topic handler would either give explicit recommendations, such as 'do you want to talk about recent NBA games?' or give cues, such as 'my favorite sports is basketball, what is your favorite sports?'

A disadvantage of this method is that it requires expert knowledge to create the conversation flow. The conversation flow has to be created manually by a person who is knowledgeable in the domain. Furthermore, another disadvantage is that, the methods expects the users to follow the flow. If a user gives unexpected answers or intentionally tries to break the conversation, the topic handlers would have difficulties to give proper responses.

Trending news

It is straightforward to inject new states in a state machine. The topic handler benefits from this feature by injecting trending news states and enable the chat bot to talk about recent news. For example, when Kobe Bryant passed away, a new state called 'Kobe' was injected after the beginning state of the basketball state. Whenever a user utterance contains the keywords 'Kobe' or 'Bryant', the user would be guided into the newly created state. In this way, the chat-bot was able to talk about the death of Kobe Bryant which was the most trending sports news. Similarly, the chat-bot was also able to talk about other trending events or news such as the NBA shutdown due to corona-virus and the super bowl event. The disadvantage, however, is that this feature requires manual work and also requires expert knowledge.

Convert open domain conversation to closed domain

A state machine could not handle infinite many possible user utterances with finite states. It is impossible to create completely open domain conversations with a finite state machine based dialogue manager. Three methods were implemented to solve this problem.

Yes/No Questions: The most intuitive way is to limit the possible user utterances. One solution is to ask yes/no questions at the end of each chat-bot's utterance. For instance, 'Do you want to know the best defensive player of that game?' Since user could either answer positive or negative to these questions, this kind of state could be handled by finite number of follow-up states (two in most cases, one for positive answer, one for negative answer). This approach, however, could create boring conversations, since the user responses are expected to be binary.

Questions with finite possible answers: Similar to the Yes/No question, this approach asks questions with finite number of possible answers, which also makes it possible to handle the state with limited number of states. For example, 'who is your favorite NBA player?' Since there are limited number of NBA players, this question could be handle by limited number of follow-up states (in the best case, one state for each player). The drawback is that this approach could require significant amount of manual work. For example, since there are hundreds of NBA players, hundreds of states need to be created to cover one single question. To reduce the amount of manual work, only the most popular responses were covered. (for example, only the most famous NBA players were covered for the questions in the previous example.)

Questions with infinite possible answers To create engaging conversations, open-domain questions have to be brought up, especially when the chat-bot need

to ask for user opinions. For example, 'what is your opinion about the NBA shutdown due to corona-virus?' For open-domain questions, it is impossible to cover all user responses. Therefore, those open-domain questions would first be released to the users with a very general follow-up state, such as 'Interesting. thanks for sharing your opinion'. Then, the most frequent user responses would be recorded by the logs. Those most frequent user responses would be covered by future releases of the chat-bot.

4.1.2 Database

The database was constructed on Amazon Relational Database Service (Amazon RDS). Emora uses the database to talk about real-time news/information with users (NBA game happened in yesterday, for example).

Database table description

Five tables were created, four of them are regularly updated. A detailed description of the database is recorded in table 4.1. New Data is inserted into 'NBAGames' and 'Playerpergame' after an NBA game takes place. After each game takes place, one row is inserted into 'NBAGames', and 10-26 rows are inserted into 'Playerpergame'. Data in 'TeamFact' and 'PlayerFact' are getting updated after an NBA game takes place. After each game takes place, two rows in 'TeamFact' are updated, and 10-26 rows in 'PlayerFact' are updated.

Data source

Information in the database, such as game statistics, player statistics is scraped from online websites. Specifically, most data is scraped from 'sports-reference.com'. A python API called 'sportsreference' was used to scrape the data.

Table Name	Description	Daily Updated
NBAGames	Overall statistics of a game	True
Playerpergame	Overall statistics of each player of each game	True
TeamFact	Team statistics of whole season	True
PlayerFact	Player statistics of whole season	True
PlayerTeam	Records which team each player belongs to	False

Table 4.1: Summary of Events

4.1.3 Open domain question answering

To make Emora be able to answer any factual-based open-domain questions, an open-domain question answering method was implemented into the chat-bot, which includes a document reader and a document retriever. The document reader uses the Bert Model trained on Wikipedia datasets to find an answer span to a question given a paragraph and a query; The document retriever retrieves the most relevant paragraph to a query from an article using 'term frequency–inverse document frequency' ('TF-IDF'). Combining the two methods, an answer span to a given query could be found from the Wikipedia database. However, the experiments showed that the method was not suitable to be integrated into a chat-bot, because document retriever does not have enough accuracy and document reader has too much latency. The specifics of the experiments are described below.

Document reader

There has been a lot of research conducted in the reading comprehension domain within Natural Language Processing. One popular task in question answering is to find the best matching text span in a paragraph to a given query. For example, SQUAD(The Stanford Question Answering Dataset) is a reading comprehension dataset specifically designed for this task.[12] [13] It consists of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable.

The following in an example of the SQUAD dataset, which consists of three parts : a paragraph, a query and an answer.

Paragraph: The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France. They were descended from Norse ("Norman" comes from "Norseman") raiders and pirates from Denmark, Iceland and Norway who, under their leader Rollo, agreed to swear fealty to King Charles III of West Francia. Through generations of assimilation and mixing with the native Frankish and Roman-Gaulish populations, their descendants would gradually merge with the Carolingian-based cultures of West Francia. The distinct cultural and ethnic identity of the Normans emerged initially in the first half of the 10th century, and it continued to evolve over the succeeding centuries.

Question: In what country is Normandy located?

Answer: France

Research for the SQUAD reading comprehension task is progressing well. The most popular way of solving this task is to implement NLP deep learning models. For example, the current best single model for this task has been 'ELECTRA', which, according to the statistics listed on SQUAD leader board, has an accuracy(Exact Match) of 88.81, the best ensemble method has achieved an accuracy of 90.38, which largely exceeds human performance(86.83).[6]

Since the reading comprehension task could answer open-domain questions, it seems to be suitable to be implemented in an open-domain conversational agent at first glance. However, the reading comprehension task by itself, is not suitable for chat-bot conversations because a user only gives a query without giving a corresponding paragraph that contains the answer to that query.

To fix the problem, a document retriever is needed, which retrieves a paragraph that contains answer to a query when given a query.

Document retriever

Given a query, the document retriever retrieves a paragraph that contains answer to the query. Wikipedia is an online encyclopedia that contains articles about factual information in almost all domains, which is a good database to find relevant paragraphs. However, it is unrealistic to retrieve a paragraph from all Wikipedia articles, because the Wikipedia database is too large(there are over 6 millions articles). As a result, this thesis divides this large problem is into two sub-problems, and solves each of them separately. The first sub-task is to find the most relevant paragraph to a question within a specific article, and this task is solved by using information retrieval algorithms. And the second sub-task is to use a topic classifier to find the most relevant article to the given query. For example, if a user asks "what is the area of Tokyo?", the topic classifier in the chat-bot would first classify the the most relevant topic to the query is "Tokyo". After that, the document retriever would go to the 'Tokyo' Wikipedia article and retrieve the most relevant articles about 'Tokyo'. Finally, the document reader would find the answer to the query within the retrieved paragraphs.

A TF-IDF method is used to retrieve the most relevant paragraphs.[14] Specifically, the combination score of term frequency and inverse document frequency is used to calculate the relevancy of each word in query to each corresponding paragraph, in which term frequency measures the number of occurrences of a term in the paragraph, and inverse document frequency measures how much information the word provides. The paragraphs with highest TF-IDF socres are selected as candidate paragraphs as inputs for the document reader. The TF-IDF uses a formula to calculate the 'relevancy score' of each word in a candidate paragraph to the query, which is

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \quad \text{idf}(t, D)$$

, where t denotes the terms, d denotes each candidate paragraph and D demotes all candidate paragraphs. $tf(t, D)$ is simply the number of occurrences of a term t in paragraph d, and $idf(t, D)$ is calculated as

$$\text{idf}(t, D) = \log \frac{jDj}{1 + jfd \ 2 \ D : t \ 2 \ dgj}$$

Experiments

Experiments for document retriever and document reader were conducted separately.

Document retriever For document retriever, the experiment was conducted on a data-set that was created based on the SQUAD 1.1 data-set. Since all data in the SQUAD data-set are made based on Wikipedia paragraphs, all the original Wikipedia articles corresponding to a given query are first retrieved and then the N best paragraphs that could answer the query question are found. The paragraph that was originally provided by the SQUAD is labelled as the correct paragraph. Note that this could actually introduce some errors, because there could possibly be other paragraphs within the same article other than the SQUAD's provided paragraph that also contain the query's answers, but those paragraphs are labeled as False. The following is an example.

The Original SQUAD Sample:

Article Title : "Super Bowl 50"

Paragraph : "Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title....."

Query : "Which NFL team represented the AFC at Super Bowl 50?"

Answer: “Denver Broncos”

By looking up the article title from Wikipedia, the original Wikipedia article that contains the selected paragraph is selected and paragraphs from that article are extracted into the candidate paragraphs (D).

Transferred Data Sample:

Query : “Which NFL team represented the AFC at Super Bowl 50?”

Candidate Paragraphs (D) : 1: “Super Bowl 50 was an American football game to determine ...” , 2: “The Panthers finished the regular season with a 15–1 record, racking up ...”, 3: “The Broncos took an early lead in Super Bowl 50 and never trailed. Denver recorded....”, 4: “CBS’ broadcast of the game was the third most-watched program in American television history with an average of 111.9 million viewers. The network charge”, ... , 61: Upon receiving the Lombardi Trophy, Broncos general manager and former quarterback John Elway raised it and exclaimed "This one’s for Pat," in reference ... , 62 : This was Denver’s first sports championship since the Colorado Rapids won the MLS Cup in 2010 ... , 63 : Both teams would ultimately struggle during the season and failed to qualify for the playoffs. The Panthers fell to 6–10 and finished in last place in the NFC South

Correct Paragraph : “Super Bowl 50 was an American football game to determine .... ”

The experiment was conducted on a dataset generated from SQUAD1.1 training and dev data-set. The result of the experiment is shown in figure.

The Accuracy increases with the number of best matched paragraphs retrieved, and an accuracy of 80 percent could be reached when 12 paragraphs are retrieved. However, since document reader could only read one paragraph each time, retrieving more paragraphs would increase the latency.

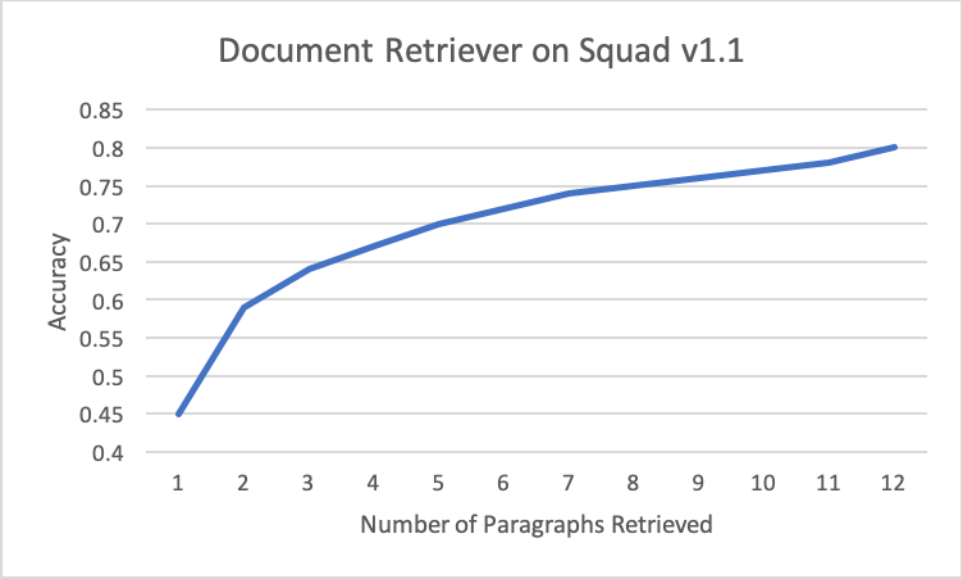

Figure 4.2: Document Retrieval Performance on SQUAD 1.1

Document Reader The document reader is based on the question answering research on the SQAUD data-set of past research. Specifically, a deep learning language representation model called 'Bidirectional Encoder Representations from Transformers' (BERT) was implemented, which was the state-of-the-art model for the SQUAD question answering task at the time the experiment was conducted. [7]

The experiment was conducted in an environment with 4 GPUs and multiple CPUs, which made multiprocessing possible. The result, however, shows that the deep learning model's latency was too high to be accepted by a chat-bot that needs to give response within 2 seconds. The results are listed below.

Number of Candidate Paragraphs	Latency (CPU)	Latency(GPU)
1 (no multiprocessing)	1.5	0.7
1 (with multiprocessing)	3.6	5.77
2 (with multiprocessing)	4.8	6.42
3 (with multiprocessing)	6.4	7.43
4 (with multiprocessing)	6.9	8.18

Table 4.2: Latency of Document Reader

The experiment results show that neither document retriever nor document reader meets the requirement to be implemented in a chat-bot. Specifically, document re-

triever's accuracy is not high enough, which is only 45 percent when retrieving a single best paragraph. However, when retrieving more than 1 paragraph, document reader has too much latency reading them. Therefore, the method explored is not suited for chat-bot purposes, but it could be useful for other applications that have less strict time latency requirement.

Chapter 5

Evaluation

The sports topic handler's performance is evaluated by analyzing the logged information.

5.1 Logged information

The recorded fields are listed in table 2. When analyzing the logs, date and version id are used to separate different versions of the chat-bot. Topic-handler id is used to find out which conversations are influenced by the sports topic-handler.

Field Name	Description
Date	Date that the conversations takes place
Version ID	The version of chat-bot
Topic Handler ID	A unique identifier of a topic handler
Utterance	The actual user utterance
Rating	Objective User Ratings from a scale of 1-5

Table 5.1: Summary of Events

5.2 Unweighted user-ratings

Average ratings of conversations that are not influenced by the sports topic handler at all and those are influenced by the sports topic handler are calculated separately.

The performance of sports topic handler is evaluated by comparing the two ratings. However, this approach has a possible bias. Within the conversations that are influenced by the sports topic handler, the proportion of the conversation that are actually handled by the sports topic handler is different. Therefore, in some conversations that were partially handled by the sports topic handler, it is possible that the sports topic handler is not the deciding factor of the user rating.

5.3 Weighted user-ratings

One way to mitigate the possible bias in unweighted user-ratings is to calculate the weighted user-ratings. Specifically, when calculating the user ratings of sports topic handler, the proportion of the conversation that are handled by the sports topic handler is used as a weight. And the overall rating of Sports topic handler is calculated by the following equation, where n is the total number of conversations and p is the proportion of each conversation that is handled by the sports component.

$$\overline{R_i} = \frac{\sum_{i=1}^{n} R_i \quad p_i}{\sum_{i=1}^{n} p_i}$$

Chapter 6

Results and Analysis

6.1 Events summary

A consistent positive correlation was found between the user ratings and modular updates for sports topic handler. Specifically, an increase in user ratings was observed whenever major updates were made to the sports topic handler and an decrease in user ratings was observed when the sports topic handler had critical bugs. The major events that had major impact on the topic handler's user ratings are listed as follows.

Figures 6.1 and 6.2 show a comparison between the unweighted ratings of sports component and the ratings of other component.

The followings are specific explanations of the events

The peak at point A was caused by a database update to the sports topic handler. Specifically, after this update, Emora could make a conversation based on real-time data by extracting information such as team ranking and team top players from the database.

The trough at point B was caused by an unusual low traffic to sports topic handler. The sports topic handler was only triggered by 3 times that day.

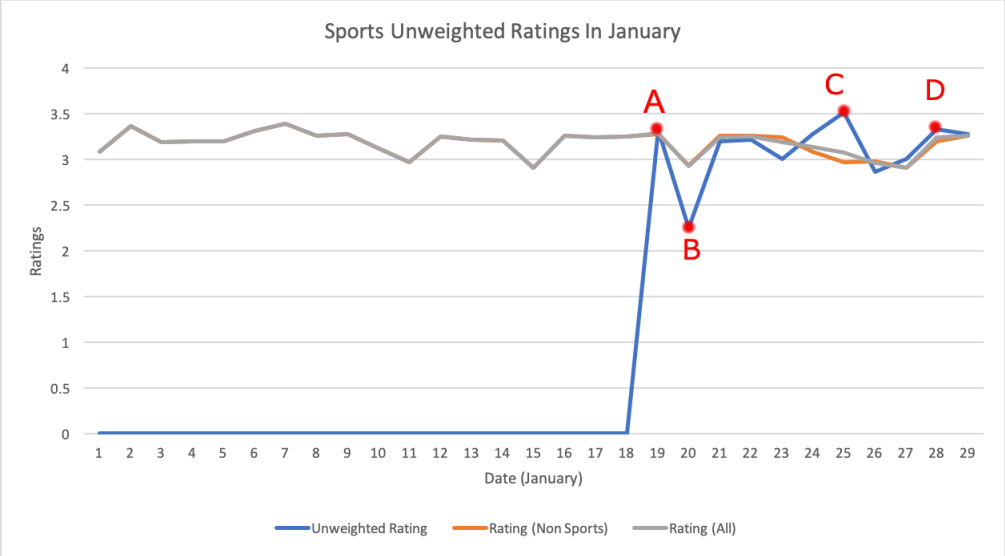

Figure 6.1: Unweighted Sports Rating vs. Other Component Rating, January

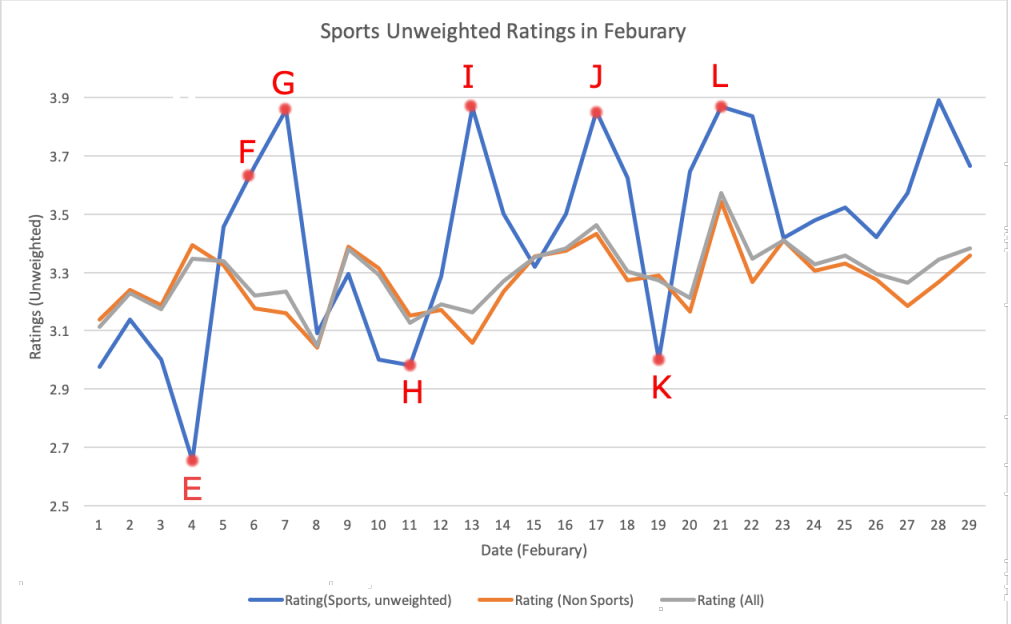

Figure 6.2: Unweighted Sports Rating vs. Other Component Rating, February

point on Figure	**Difference to Yesterday's Rating**	**Difference to other component's Rating**
A	3.32	0.04
B	-0.95	-0.06
C	0.27	0.17
D	0.32	0.13
E	-0.35	-0.74
F	0.21	0.49
G	0.19	0.69
H	-0.02	-0.17
I	0.58	0.81
J	0.35	0.42
K	-0.63	-0.29
L	0.22	0.32

Table 6.1: Specific improvements

Thus, the rating that day was not representative.

The peak at point C was caused by a conversation flow update to the sports topic handler. More variations in conversation utterances were added so that the conversation becomes less monotone.

At point D, the peak was due to a update made about a trending news happened at that time (the death of Kobe Bryant). A small conversation was created to discuss the event.

At point E, the trough was caused by both low traffic to the sports topic handler and by the limited topics covered by the sports topic handler. Within 14 conversations related to sports, only 2 conversations were about basketball, which was the domain that the sports topic handler was good at. The rest of the users were more interested in sports that were not yet covered by the sports component. This trough indicated that the conversation flow needed modification.

At point F, another small talk about a trending news at that time was updated

(Super bowl)

The peak at point G was caused by another database related conversation. Specifically, Emora could make new conversation about player's age after that update.

The trough at point H was caused by unusual low traffic to sports topic handler. The sports component was only triggered by 6 times that day. And within the 6 conversations, only 1 user was an actual sports fan. Thus, the rating that day was not representative.

Another conversation flow update was made at point I. Specifically, the chat-bot could recognize users who it has met before and create different conversation for those returning users based on previous conversation.

The trough at point K was caused by a database related bug. Because of this bug, the chat-bot failed to extract information from database, and therefore could no longer make proper response to any questions that requires real-time information (real-time game data, for example)

A conversation flow update was made at point L : transition to other topics became smoother.

6.2 Weighted ratings vs. unweighted ratings

Because of the Logging Format, the weighted ratings of January could not be calculated. A comparison of Weighted Ratings vs. Unweighted Ratings in February has been plotted in figure 6.3.

The weighted ratings are generally higher than the unweighted ratings, showing that sports component contributes more in higher user ratings.

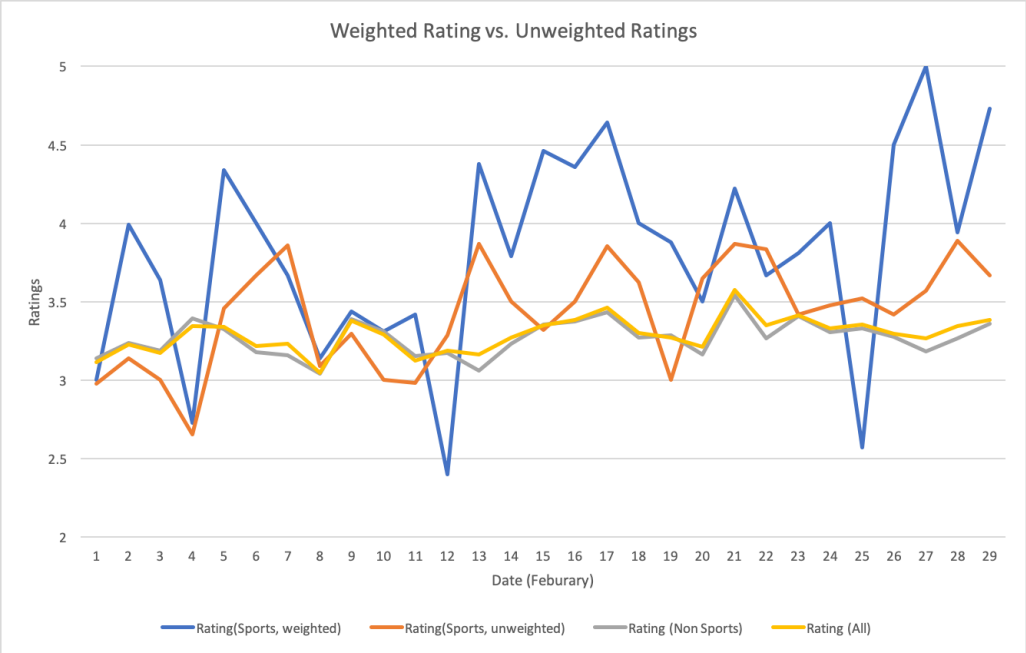

Figure 6.3: Unweighted Sports Ratings vs. Weighted Sports Ratings, February

6.3 Error (Low-rating) Analysis

6.3.1 Overall statistics

Among all the rated conversations, about 18.9% of overall low ratings (ratings of 1.0-2.0) was found in February. As for the sports topic handler, there are 95 total Sports-component low ratings in February, and the average number of utterances in each conversation is 17.54.

6.3.2 Qualitative error analysis

Three main situations that causes users to give low ratings to sports topic handler were summarized below.

Situation 1 : Uncovered topics. Since not all topics are covered by the sports component. Users give low ratings when their interested sports are not covered. This error could be mitigated by better conversation flow and more topic cov-

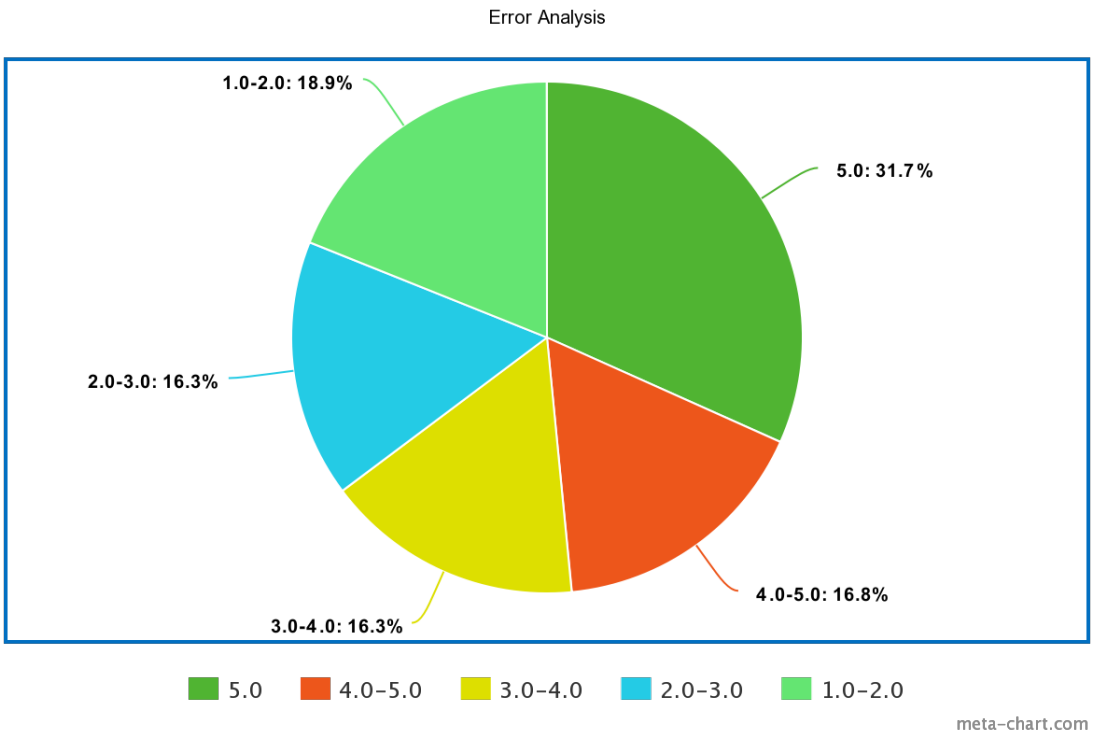

Figure 6.4: Distribution of ratings

erage.

Sample Conversation:

User:let's talk about sports

Chatbot:I love sports. My favorite sport is basketball. What is your favorite sport?

User:volleyball

Chatbot:That is a great sport. I am sorry I only watch basketball. Do you play or watch basketball?

User:No

Situation 2 : Bugs and unexpected utterance. Some words are not recorded in the way the chat bot expected. In the following example, while the chatbot

recognizes "NBA", it does not recognize "n. b. a." This type of errors are easy to fix.

Sample Conversation:

User:i watch n. b. a.

Chatbot:oh sorry, i did not pay attention. could you clarify which sport you just mentioned?

Situation 3 : Ignore user response. Since sports component recognizes user responses based on a pre-defined ontology, some responses are not correctly recognized. In the following example, 'both' was not included in the ontology, and thus was not correctly understood by the chatbot. This type of error could be found and fixed by manual analysis of logs.

Sample Conversation:

Chatbot:Do you watch nba or nfl ?

User:i watch both.

Chatbot:oh sorry, i just started to watch sport recently. that is why I currently could only talk about basketball or football in the sports domain for now.

Chapter 7

Conclusion

This thesis presents various ways of creating multi-turn engaging conversations with a state machine based dialogue manager, including associating the state machine with a daily updated database, updating trending news and incorporating a question answering deep learning model. According to the user study conducted based on the logged statistics such as user ratings and user conversations, most modular improvements are found to have positive impacts on user ratings for the sports component. The user study shows that the methods presented works well to generate user-specific multi-turn dialogues based on real-time information, it has drawbacks of not handling open-domain questions very well and requiring expert knowledge to create a state machine in a specific domain. These challenging problems could be explored in the future. This thesis also explores the possibility of incorporating an open-domain question answering method into a chat-bot, and found that both the method's accuracy and latency are not sufficient.

Bibliography

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, pages 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-76298-0.

[2] Fumihiro Bessho, Tatsuya Harada, and Yasuo Kuniyoshi. Dialog system using real-time crowdsourcing and twitter large-scale corpus. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 227–231, Seoul, South Korea, July 2012. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W12-1631.

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581026. doi: 10.1145/1376616.1376746. URL https://doi.org/10.1145/1376616.1376746.

[4] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia

to answer open-domain questions, 2017.

[5] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani. State-of-the-art speech recognition with sequence-to-sequence models, 2017.

[6] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1xMH1BtvB.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[8] Khyatti Gupta, Meghana Joshi, Ankush Chatterjee, Sonam Damani, Kedhar Nath Narahari, and Puneet Agrawal. Insights from building an open-ended conversational agent. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 106–112, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4112. URL https://www.aclweb.org/anthology/W19-4112.

[9] Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazaré, and Jason Weston. Learning from dialogue after deployment: Feed yourself, chatbot!, 2019.

[10] B. Pellom, W. Ward, J. Hansen, R. Cole, K. Hacioglu, J. Zhang, X. Yu, and S. Pradhan. University of colorado dialogue systems for travel and navigation. In *Proceedings of the First International Conference on Human Language Technology Research*, 2001. URL https://www.aclweb.org/anthology/H01-1073.

[11] Joseph Polifroni and Marilyn Walker. Learning database content for spoken dialogue system design. In *Proceedings of the Fifth International Conference on*

Language Resources and Evaluation (LREC'06), Genoa, Italy, May 2006. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2006/pdf/301_pdf.pdf.

[12] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.

[13] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad, 2018.

[14] Juan Ramos. Using tf-idf to determine word relevance in document queries. *1st Int. Conf. on Machine Learning*, 01 2003.

[15] Antoine Raux and Maxine Eskenazi. A finite-state turn-taking model for spoken dialog systems. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, page 629–637, USA, 2009. Association for Computational Linguistics. ISBN 9781932432411.

[16] Alan Ritter, Colin Cherry, and William B. Dolan. Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D11-1054.

[17] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with. *Proceedings of the 2019 Conference of the North*, 2019. doi: 10.18653/v1/n19-4013. URL http://dx.doi.org/10.18653/v1/N19-4013.

[18] Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on*

Empirical Methods in Natural Language Processing, pages 2013–2018, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1237. URL https://www.aclweb.org/anthology/D15-1237.

[19] Xuchen Yao, Benjamin Durme, Chris Callison-Burch, and Peter Clark. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics*, 06 2013.