Spring 5-8-2018

Automatic Conversation Review for Intelligent Virtual Assistants

Ian R. Beaver
University of New Mexico

Ian Beaver

Candidate

Computer Science

Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Abdullah Mueen, Chairperson

Charles Wooters

Paul De Palma

Lance Williams

George Luger

Automatic Conversation Review for Intelligent Virtual Assistants

by

Ian Beaver

A.A.S., Columbia Basin College, 2001
B.S., Computer Science, Eastern Washington University, 2004
M.S., Computer Science, Eastern Washington University, 2006

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy
Computer Science

The University of New Mexico

Albuquerque, New Mexico

July, 2018

Dedication

To Hillary and Devan. Without your continual support and sacrifice this work would never have been completed.

Acknowledgments

It has been said that no one accomplishes anything alone, but each accomplishment is the result of years of knowledge and experience gathered from others. That is even more evident in an accomplishment as large as a dissertation. While the body of this work may be the result of my efforts, the ability to do the work is a product of my family, co-workers, and so many teachers and advisers over the years. Here I would like to highlight a few who greatly contributed to my success.

First, my parents, who gave me a love of learning from a young age, faith that kept me from mental breakdowns through this endeavor, and taught me how to apply myself to any task. Secondly, George Luger and Joe Dumoulin were instrumental in encouraging (convincing?) me to return to graduate school and encouraging me along the way. My advisor, Abdullah Mueen, who took on a student who's field of research was outside of his own and who had industry strings attached. Tracy Malingo, Mike Wiseman, and all the folks at Next IT Corporation who supported me in this endeavor while cutting me some slack at work so I could get my homework done! Chuck Wooters gave me hours of detailed advise and feedback at conferences and over email. Lance Williams, whose class probably removed several years from my life, taught me to see linear algebra in an entirely new light. Paul De Palma was an inspiration to me, having also moved across the country to obtain his PhD after working in industry and academia many years. Finally, Cynthia Freeman contributed to many aspects of this work. She suffered through countless hours of annotating data, helped with setting up and running experiments, and could always be depended upon for anything from minor revisions to complete re-writes mere hours before paper deadlines.

Automatic Conversation Review for Intelligent Virtual Assistants

by

Ian Beaver

A.A.S., Columbia Basin College, 2001

B.S., Computer Science, Eastern Washington University, 2004
M.S., Computer Science, Eastern Washington University, 2006

Ph.D., Computer Science, University of New Mexico, 2018

Abstract

When reviewing the performance of Intelligent Virtual Assistants (IVAs), it is desirable to prioritize conversations involving misunderstood human inputs. These conversations uncover error in natural language understanding and help prioritize and expedite improvements to the IVA. As human reviewer time is valuable and manual analysis is time consuming, prioritizing the conversations where misunderstanding has likely occurred reduces costs and speeds improvement. A system for measuring the posthoc *risk of missed intent* associated with a single human input is presented. Numerous indicators of risk are explored and implemented. These indicators are combined using various means and evaluated on real world data. In addition, the ability for the system to adapt to different domains of language is explored. Finally, the system performance in identifying errors in IVA understanding is compared to that of human reviewers and multiple aspects of system deployment for commercial use are discussed.

Contents

List of Figures

List of Tables

Thesis Statement

Discovery of virtual assistant misunderstandings can be automated, reducing the need for human review

Chapter 1

Introduction

An Intelligent Virtual Assistant (IVA) is *"an engineered entity residing in software that interfaces with humans in a human way. This technology incorporates ... modern artificial intelligence projects to deliver full-fledged 'virtual identities' that converse with users."* [2]

IVAs are commonly used for answering questions and task optimization as in the case of Apple's Siri, Microsoft's Cortana, or Google Now. However, many companies are deploying IVAs for efficient problem resolution and cost cutting in call centers and also as the first layer of technical and product support on websites [3]. At the present time many different software companies (close to two hundred, by analysts' estimates [4]) have created IVAs that reside on corporate web pages or otherwise are embedded in advertising and selling efforts. Perhaps the largest gathering of such agents today exists on the Facebook Messenger platform. Through recent APIs provided by Facebook which made creating and deploying agents simple, the number of live "messenger bots," as they are known, has eclipsed 100,000 [5].

In these business domains, IVA accuracy and efficiency directly impacts customer experience and greatly reduces the company support costs. In one case study [6], a Fortune 50 insurance company saw a 29% reduction in live-chat volume within five months of deploying an IVA on their website. Domino's Pizza reported that product order time was reduced by 50% when it was done through their IVA [7]. IVAs are so effective that analysts predict that human contact center agents will be *completely replaced* by IVAs by 2026 [8, 9].

Already, the lines are blurring between virtual and human agents. Amtrak has reported customer confusion on whether or not its IVA for transportation services is a real person [7]. The continued adoption of IVAs is contributing to a growing problem. How do we refine an IVA's knowledge effectively and efficiently? As IVA use as well as the number of tasks an IVA is expected to perform increases, there is a corresponding jump in the number of human-computer interactions to be reviewed for quality assurance. Therefore, discovering a means to expedite review and analysis of these interactions is critical.

Without scalable and efficient methods of automated conversation review, IVA designers must rely on human reviewers to validate expected behavior of the IVAs. As this is a manual and time consuming process, the reviewers are only able to view a limited number of interactions. The result is also subjective since reviewers may disagree on the user intention for any given turn in a conversation. In addition, as the IVA improves, errors in communication become more difficult to identify in a random sample due to their dwindling numbers.

As picking the optimal user intention from the set of intentions known to the IVA is a multi-class classification task, most implementations of IVAs can indicate some means of confidence in the selected class. Therefore, one simple attempt to solve this problem of scale may be to only surface conversations where the IVA confidence score in a response is lower than some threshold. The assumption is that the confidence metric will be higher for correct responses and lower confidence scores may indicate a misunderstanding. Although low confidence selection may be more effective than a random sample, relying completely on live IVA-produced confidence metrics is undesirable as its underlying model will have some unknown error, the discovery of which is the purpose of conversation review in the first place.

For example, it is possible the model can be confident in a response while being incorrect. This is more likely to happen when there are multiple relevant responses but one is more specific, and therefore preferred, than the others for the given question. If one or more general responses had more robust or a larger volume of training data the model may favor the general responses. Consider the case where there exists two related responses in the IVA such as: *"We offer several DVR devices for rent, but we do not sell any."* and: *"To rent a DVR device, please choose from the following options: ...".* Now suppose a user asks a question such as "I have your DVR already, how do I get an HD one?". In this case the former response may be returned with higher confidence depending on variables such as how the confidence metric is calculated and if the latter response had less than optimal training data. However, the user already has a DVR and would presumably know the company only rents them. They simply want to know the process to request a new one, which would be correctly addressed by the latter response. This is a misunderstanding that should be corrected but would not be discovered if the IVA was confident in its response. This type of error scenario can also happen in ensemble approaches where redundant errors in individual components may propagate, inflating the confidence in an incorrect result [10].

Conversely, a model may not be confident but can still produce acceptable responses which would generate conversations for review where no error was observed. A situation this commonly occurs is when there is a broad topic with many possible responses and the user asks general questions about the topic. This large language overlap creates confusion within the IVA as to which response to pick, thereby lowering its confidence. An example of this would be if the user asks a travel IVA "Tell me about your baggage policies." In this case any response about number of bags allowed, size and weight of baggage, clarifying questions asking if they want the policy for international or domestic travel, relevant links to baggage policies, and similar may all be considered acceptable by a reviewer. This type of error scenario can also occur when only one response is appropriate but the IVA did not have adequate training to clearly differentiate it from other possible responses so it chooses correctly but with low confidence.

A more general issue with using confidence metrics to select conversations is they can be calculated numerous ways and depend greatly on the underlying implementation of the IVA. This leads to problems comparing the performance of heterogeneous IVAs and even the same IVA over time if the confidence calculation or underlying classifier implementations are modified. Instead, some external means of conversation review allows a holistic analysis of IVAs that can be applied regardless of implementation details or evolution of the IVA technologies. Chapter 11 compares the performance of several IVA-agnostic selection methods in prioritizing conversations for review.

In the remainder of this work a solution to this problem is presented through the creation of a system that can process all conversations to automatically and efficiently mark the interactions where the IVA is misunderstanding the user, a task which is rapidly becoming insurmountable for human reviewers. Such a system can provide cost savings to companies deploying IVAs by reducing the time human reviewers spend looking at conversations where there is no misunderstandings present. It can also enable a shorter refinement cycle because problems are surfaced quickly and more reliably than a random sample or confidence metric based review. This leads to a better user experience and faster adaptation of the knowledge base to changes in the IVA's environment.

We begin with a background of IVAs, covering their history, terminology, properties and typical architectures followed by an overview of their common refinement processes. Chapter 3 covers the current literature on automatic misunderstanding detection systems. Chapter 4 outlines an approach to design such a system for automatic conversation review and gives an overview of the features of miscommunication in conversation we attempt to detect. Chapter 6 discusses the implementation decisions made and their trade offs followed several chapters covering the means to detect each individual feature of miscommunication. Next, the systems performance is compared to humans in Chapter 11 and the following chapters conclude with a discussion of findings and future works.

Chapter 2

Background

The terms Intelligent Virtual Assistant and Intelligent Virtual Agent are sometimes used interchangeably and, for the purposes of this work, they are considered equivalent. While there may exist subtle differences in definition relating to their scope of knowledge and ability to perform complex tasks, both are referring to autonomous agents designed to assist people in various tasks. A closely related class of such programs are known as *chatterbots*, *chatbots*, or even simply *bots*. Chatbots and IVAs fall under an umbrella of Artificial Intelligence (AI) programs that simulate human conversational abilities known as *dialog systems*. For the purpose of this work, the main component of interest common to all is a Natural Language Understanding (NLU) unit which is responsible for translating user inputs into a semantic representation of the user's intention [11, 12, 13]. It is in this component that the breakdown of communication will begin, assuming adequate Automatic Speech Recognition (ASR), if such a component is used as an interface. The detection of ASR error and recovery is well covered in literature [14, 15, 16, 17, 18] and outside the scope of this work.

2.1 Chatbot or IVA

In literature, chatbots tend to refer to a general class of conversational agents designed as a means to use natural language to communicate with computers through voice or text interfaces [19]. IVAs are designed specifically to assist people in the completion of various tasks, much like a virtual secretary [20, 21]. Given their common heritage there is much overlap between chatbots and IVAs in literature. The primary difference between them appears to lie in the purpose of the agent and the complexity of its design. Perhaps the best means of classification is to determine which of the two main branches of AI they fall under [22]. Those systems that merely try to simulate conversation without any true reasoning or problem solving techniques fall in the branch of "weak" AI and would be labeled a traditional chatbot [23]. Those that attempt to apply methodologies to achieve reasoning in a way that allow them to perceive their environment and perform the right and most effective action would fall in the branch of "strong" AI and would therefore be considered an intelligent agent [22].

For example, agents embedded in devices such as Google Now, Amazon's Alexa, and Apple's SIRI which are all designed as personal assistants for performing tasks through a device are classified as IVAs [24, 25, 26]. Agents designed for assisting humans in completing tasks on websites and applications, like Alaska Airline's "Ask Jenn", are also referred to as IVAs [27]. All such agents must reason over information from their environment and current context to decide how best to proceed in a conversation, not merely choose a response to the current input as a true chatbot would.

IVAs will contain some form of a *dialog manager* component, which manages the state of the conversation and determines the dialog strategy [28]. Example states could be **greeting** or **confirmation**. An example dialog strategy could be determining how strongly to recommend a hotel based on how well it matches the users preferences [29]. In addition, IVAs often accept multiple means of interaction other than voice or text such as a mobile application, touch screen events, webpage events, e-mail, social media, monitoring calendars or proximity to sensors, or any combination of such interactions [25, 20, 21, 26, 30]. IVAs are usually connected to various back end systems such as databases or external applications that allows them to access client data or perform actions on behalf of their users [25, 26]. An example of this would be a travel assistant that would be connected to flight and weather databases, reward program databases, and payment processing applications so it could find and book flights on behalf of its users.

2.2 An Evolution of Dialog Systems

Dialog systems have progressively evolved over time as advancements were made in computing systems, computational linguistics, and artificial intelligence. Following is by no means a comprehensive history but a highlight of some of the major milestones that have influenced the dialog systems that exist today.

The Turing Test In 1950, the father of modern computing, Alan Turing, suggested that a software program could be called intelligent if a person who was interacting with that program could not distinguish it from a another human. This criteria for intelligent software has since been called the Turing Test [31, 32] and contests are still held regularly to determine whether a software program can be built that passes the test. Such programs have been employed in a wide variety of applications over the course of the last 60 years [33].

ELIZA The first, and perhaps most notorious, chatterbot was ELIZA created by Joseph Weizenbaum at MIT from 1964 to 1966 [34]. Weizenbaum developed ELIZA to replicate the behavior of a psychotherapist, although of course ELIZA knew virtually nothing of the real world, including psychotherapy! ELIZA worked by examining a user's typed comments for any of a list of keywords or phrases. If such a pattern is found, a rule is applied which transforms the user's input sentence into an ELISA response. When no keyword or phrase was found, ELIZA responded with either a generic answer or by repeating one of its earlier questions or comments. With these techniques, Weizenbaum's program was able to fool many people into believing that they were actually talking to a real therapist.

SCHOLAR The PhD work of Jaime Carbonell, SCHOLAR was a research system for computer-assisted instruction published in 1970 [35, 36]. While not primarily an IVA, SCHOLAR introduced some important design concepts for modern IVAs. The first was an organization of data into frames of information along with related questions and their anticipated answers. This data was entered by teachers into the system. A second important contribution was the capability of *mixed initiative* dialog with users as it had knowledge of both questions and answers around topics. In this context *initiative* refers to who has control of the conversation, and since both the system and the users could take control by asking a question, the initiative could change between them over the course of an interaction. In contrast, *single initiative* or *system initiative* requires that the system ask the user a series of questions to navigate a predefined dialog flow. Any deviation from the system prompt by the user would either be ignored or misunderstood as a transfer to some other known dialog state. While single initiative systems are easier to build, they are very limited and do not have the ability to model real human conversation, which involves give and take [28].

PARRY PARRY, created in the early 1970s by Kenneth Colby, has been described as "ELIZA with attitude" [37]. PARRY attempted to capture the behavior of a paranoid personality using a similar (although more advanced) approach than that employed by Weizenbaum [33]. To test PARRY, a variation of the Turing Test was used where a group of psychiatrists compared real patients with computers running PARRY. Another group of psychiatrists were shown transcripts of the conversations. The two groups were then asked to identify which of the "patients" were human and which were computer programs. The professional psychiatrists were only able to make the correct identification about half the time, a figure consistent with random guessing [33, 38]. Weizenbaum dropped this line of research in part by the reactions to ELIZA and the success of derivatives such as PARRY. He felt that people were too easily convinced that such systems truly understood them and subsequently formed attachments when they were merely performing clever programming techniques [34].

GUS The Genial Understander System (GUS) was created at the Xerox Palo Alto Research center and published in 1977 [39]. GUS improved on the organization of tasks from SCHOLAR into a well defined data structure called a *frame*. A frame contained a set of slots to hold necessary information to complete the task. Each slot defines the data type, such as an integer or a string, and is associated with a question to ask in order to discover the slot value. When working on a task, GUS attempts to fill as many values as it can from a user input. This allowed the user to give more than one piece of information at a time, and the system would only prompt for slots that had not yet been filled. This frame-driven dialog design is still the basis of modern IVAs such as Apple's SIRI [26]. GUS also implemented some basic attempts at solving difficult problems in NLU design such as interpreting sentence fragments and *anaphora resolution*. An anaphora is the use of a word as a substitute for previous ideas or terms in the conversation such as "that" in "What do you mean by that?". Inputs containing anaphora pose difficulties for dialog systems as they must determine what the anaphora refers to before they can determine the intent of the input. Similarly, a sentence fragment must be interpreted in the context of the expected input. When given the fragment "on Friday" after asking the user what day they want to return, GUS will translate it into the full sentence "I want to return on Friday" before proceeding with processing the input.

UNIX Consultant The Berkeley UNIX Consultant (UC) project was published in 1988. It was the first to really focus on an IVA as a *reasonable agent*, as opposed to simply an intelligent interface through which information flows to and from knowledge bases [40]. They designed UC such that it has goals of its own that may supersede the user's goals if it determines the system's goal is more favorable than that of the user. For example, if the user requests information on how to crash UNIX, the UC will compare the user's goal with its own goal of protecting the system. Based on a detected conflict of the two goals it will decide in favor of the one it judges to be the better outcome and deny the user the information requested. IVAs without such reasoning ability will merely try to help the user in their goal regardless of any negative consequences. The architecture presented was a general means to model a human consultant's own reasoning process in determining the best outcome in the task at hand.

GALAXY and GALAXY-II The "hub-and-spoke" architecture that became commonplace in dialog system design was first introduced in the SRI GALAXY project published in 1994 [41]. It was later improved upon in the Massachusetts Institute of Technology GALAXY-II architecture in 1998 [42]. An abstraction of this architecture is shown in Figure 2.1 and discussed in greater detail in Section 2.3. The architecture consisted of a central blackboard or "hub" through which all separate components such as NLU and the dialog manager communicate. Messages were passed between the components based on a token passing framework. Each component is stateless in that the hub stores all of the conversation data for each active user session. When a new token is generated by the hub any additional session data needed by the target component is included. Therefore each component can work on multiple user sessions simultaneously. This design allowed for extensibility and easier debugging and visualization of data flow through all stages of processing the dialog.

ALICE Launched on November 23, 1995 by Richard Wallace, the Artificial Linguistic Internet Computer Entity or ALICE introduced the Artificial Intelligence Markup Language (AIML) [19]. AIML was an Extensible Markup Language (XML) dialect for creating chatbots. ALICE AIML was provided free and Open Source as a platform for others to contribute to its knowledge base and to create their own custom chatbots. As of 2017, the ALICE AIML knowledge base contained approximately 41,000 categories[1] and Pandorabots[2], a reseller of the technology, reported deploying over 285,000 chatbots based on the platform. ALICE went on to win the Loebner Prize, an annual competition to determine the most human-like computer program, in 2000, 2001, and 2004 [43].

Ask Jeeves Founded in 1996, "Ask Jeeves" was developed to allow users to search the Internet using (human language) sentences in addition to the standard key-word approach common to search engines at that time. The original idea behind Jeeves, named for a gentleman's personal valet (from the writings of P. G. Wodehouse)[3], was to allow users to get answers to questions posed in everyday human language as well as support for math, dictionary, and conversion questions for the World Wide Web. A 2001 study of search engine user queries demonstrated that users were increasingly searching for e-commerce information in question format as opposed to listing keywords and the "Ask Jeeves" platform was encouraging the trend [44].

Cyberlover In the mid-2000s, the ELIZA and PARRY technologies were developed into systems which were observed on chatrooms and forum sites with malicious

[1] `http://www.alicebot.org/downloads/sets.html`
[2] `https://www.pandorabots.com/`
[3] `http://en.wikipedia.org/wiki/Ask.com`

intent. Chatbots, such as "Cyberlover", appeared to be designed to flirt with other users of the sites and then trick them into revealing personal information, lead them to other malicious websites, or share files that contain malicious content [45, 46]. Such behaviors were previously performed by humans and therefore not able to scale. By deploying automated means to prey upon users, criminals could more quickly gather personal information and spread malware. However, using chatbots to impersonate people in chatrooms to gain information is not limited to criminal behavior. Government agencies have also funded or deployed chatbots for the purposes of discovering criminals in online settings through replicating the profile and behavior of a typical victim [47, 48]. When another user begins to interact with the bot, it will begin to build a profile of the user through conversation. If they appear a threat, it may attempt to set up a physical meeting with the user so they can be apprehended.

Corporate IVAs More recently, the conversational abilities of such programs has been employed to help their human users accomplish tasks or find information on corporate websites and intranets. With the rise of the Internet and e-commerce, IVAs were found to be particularly useful as virtual representatives of their employers [49]. They work 365 days a year 24 hours a day and do not get sick or take vacations making them the ideal employee for customer service. Careful crafting of the persona and interactions of the IVA can further a company's brand across the prospective user community. Furthermore, once this brand is established, it is consistent. In contrast to human agents, who have bad days and can retaliate against 'difficult' customers such as in the famous Comcast incidents [50], virtual agents can be programmed in such a way as to always remain courteous in the face of hostility.

Tay Microsoft launched Tay on March 23, 2016 and it quickly became an example of what happens when continuous machine learning is applied to a chatbot without proper safeguards. Tay was designed to learn from conversation with other social media users. Unfortunately, as a chatbot with no concept of reason or right and wrong behavior, Tay quickly picked up socially unacceptable behaviors such as promoting racism and genocide and was shut down in less than a day [51]. Tay became a poster child of the downside of "weak" AI in that the bot did in fact learn, and was able to maintain communication with humans, but was merely repeating what it learned without any concept of what the words actually meant. Those in the AI community highlighted the need for more robust reasoning ability and better anticipation of what can go wrong when AI is deployed to the public [52].

Social Media Service Bots A current trend is in the deployment of chatterbots and IVAs on social media and instant messaging platforms for the purposes of providing news, recommendations, weather reports, and virtual personal simulation [5, 53, 54, 55]. While prolific, the majority of these bots are very targeted, helping their users with a very narrow set of tasks such as checking sports scores or tracking order status. These simplistic bots may only understand a dozen user intentions, but they are successful in the fact that they re-use an existing and popular interface for human-to-human communication. For example, Slack is an popular instant messaging and collaboration platform which, as of September 2017, sees 6 million daily users [56]. As many companies deploy Slack for internal communication, the addition of a chatterbot to provide automated services, known as a Slackbot, can leverage the fact that employees are already using Slack to communicate with each other. Some very specific tasks such as scheduling meetings or providing Human Resources data like holiday and vacation schedules can be programed into the Slackbot. It can then monitor the communication feed for questions it has been trained to respond to such as "Is next Monday a holiday?", and provide an answer just as if it was another human user on the platform.

Figure 2.1: An abstraction of common IVA architectures

2.3 Architecture of an IVA

We have presented an abbreviated history of dialog systems and several important milestones in their development. Now we will turn our attention to the individual components and architecture of typical IVAs.

Figure 2.1[4] shows common components of a modern IVA [13, 57, 58, 59]. These components are linked together through a central hub or facilitator. Some components may be optional such as Automatic Speech Recognition (ASR), as in the case of direct text input to the IVA. The user interacts with the IVA through some interface which begins the processing pipeline. Following some predefined set of dependencies, the hub will pass the user input around within the IVA to each component where transformations or annotations are applied, logical decisions are made, or prior sta-

[4]Image by Chuck Wooters, used with permission

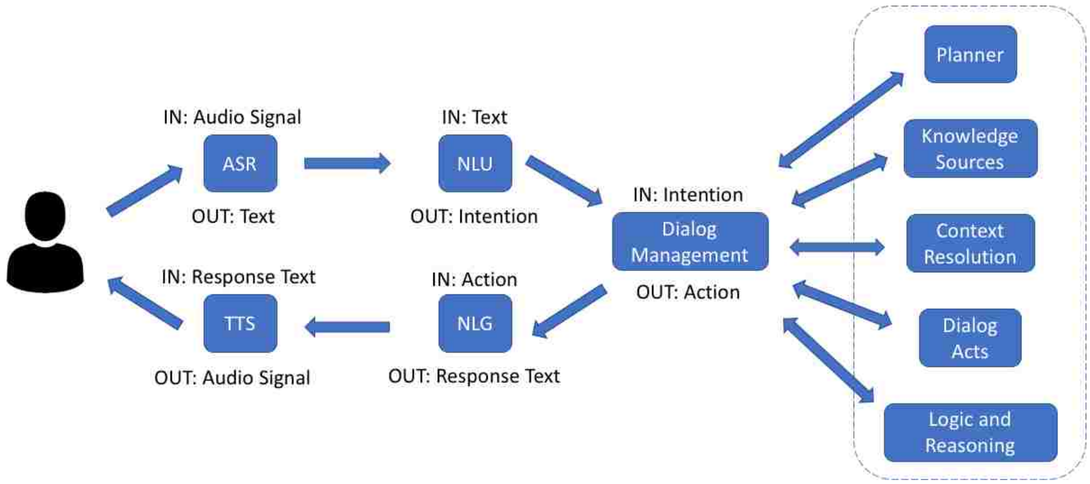


Figure 2.2: Data flow through an IVA with a voice interface

tistical knowledge is consulted, and each result is returned the hub. Once the path through the system arrives at an output component, the response is returned to the user.

For example, if the user connects to the IVA through a voice interface, the hub will direct the raw audio stream to the ASR which translates it into text. The text will then be returned to the hub which stores the text along with any other ASR outputs such as a confidence score. The hub then consults its routing logic to determine which system to send the session data to next, which would typically be the NLU. As the NLU component requires text input, the hub sends the text output from the ASR to the NLU along with any other input data it may require such as the conversation history thus far. The hub then receives the output of the NLU, stores it, and passes on what is needed to the next component. This process continues until the final component is selected, in this case the Text To Speech (TTS) engine, which will translate the response text from the Natural Language Generation (NLG) component into an audio response to the user. At that point the IVAs turn in the conversation is complete and the user can choose to respond or end the conversation.

This data flow is demonstrated in Figure 2.2. At each blue arrow in the figure, the output data will go to the hub to be recorded into the conversation state and the input data will be sent on to the next component. The components on the right hand side within the dashed line could be integrated directly into the dialog management component or exist as separate systems. Simpler IVAs may not implement all of them, while more complex systems may have additional specialized components. For our purposes, we are specifically interested in automatically discovering and surfacing errors in natural language understanding. Therefore, the following section goes into detail of the NLU and its role in a dialog system.

2.4 Natural Language Understanding

Dialog systems can be implemented in a multitude of ways, but common to all is a component for Natural Language Understanding (NLU) or the translation of user inputs into a semantic representation [11, 12, 13]. Regardless of the means in which the user is interacting with the IVA (keyboard, speech, gestures, etc.), the user input is first converted into text or a numerical feature representation and fed into the NLU for analysis. The NLU maps user inputs, or conversational *turns*, to a derived semantic representation commonly known as the *user intention* or simply *intent*. In the context of Natural Language Processing (NLP), intent is defined by Dumoulin as *"an interpretation of a statement or question that allows one to formulate the 'best' response to the statement"* [60].

The collection syntax, semantics, and grammar rules that defines how input language maps to an intent within the NLU is referred to as a *language model*. The NLU may perform pre-processing steps such as part-of-speech tagging, stemming, and entity identification (such as proper names or dates) before feeding the text into the language model. Each of these parsers may be trained through machine learning methods or manually constructed by human experts. In very simplistic IVAs employing keyword-spotting, the NLU may only consist of a set regular expressions or other grammars which are matched against the input text.

Recently, breakthroughs in neural networks have led to successes in what is known as end-to-end learning. End-to-end learning may attempt to combine several of the components in the traditional dialog system such as the internals of the ASR [61, 62], or the NLU and dialog manager [63]; or train both understanding and language generation in one model by showing it inputs and responses between two humans so that the model learns how to respond to similar inputs all on it's own [64, 65]. Alternatively, these models can be shown source texts containing facts or information and questions about the sources. The models learn to locate the relevant text in the source to answer the given question [66, 67]. The assumption is that if you feed it enough examples of inputs and expected outputs, the system will learn for itself some internal approximation of these rules of syntax, semantics, and grammar. The process essentially teaches a virtual agent how a statistical average human (in respect to the training data) behaves without having to formalize those behaviors, avoiding costly feature engineering. Both manually constructed symbolic models and end-to-end models have the same goal, but the former approaches it with humans observing and formalizing human behavior and the latter with algorithms observing and approximating human behavior.

Regardless of these implementation details, to improve the language models and for quality assurance, human-computer interactions need to be continuously reviewed. Improvements include the addition of vocabulary and new rules or the revision of existing rules that led to incorrect mappings within the language model. For end-to-end or statistically-trained models, identification of incorrect understanding can highlight confusion within the model and prioritize areas of further training.

Another important aspect of NLU design and evaluation is the consideration of conversational context. IVAs are typically designed as dialog systems capable of *mixed-initiative dialogs*. As mentioned in Section 2.2, *initiative* refers to who has control of the conversation and therefore a mixed-initiative system allows the user

Turn	Customer	Travel Agent	Initiative Holder
1		Do you need to check any bags for this flight?	Agent
2	Is there a fee for more than one?		Customer
3		The first two bags are free.	Customer
4	In that case I will check two bags.		Agent
5		Great. I will add two bags to your reservation. Would you like to choose your seats now?	Agent

Table 2.1: An example of change in initiative

or system to take over the lead in the conversation as needed to accomplish a task. A fabricated conversation between a travel agent (human or virtual) and a customer demonstrating this give and take is shown in Table 2.1. The conversation begins with a question from the agent, thereby taking the initiative. The customer responds with a question of their own in turn 2, which then transfers the control of the conversation to the customer. The agent responds to the customer's question in turn 3 and then waits for the customer to continue the conversation as they still hold the initiative. In turn 4, the customer answers the agent's original question from turn 1, thereby relinquishing control of the conversation back to the agent, who then proceeds with completing the task of booking a ticket for the customer.

This ability to give and take in a conversation adds complexity for the NLU. As the user can at any time respond to a question with a question of their own, the language model must include not only the language of the expected response, but also of any potential questions that could arise around the topic at hand. Whether the language model is trained by statistics or manually constructed, failure to anticipate possible user questions can lead to out-of-vocabulary words and failure to determine correct intent when deployed.

Furthermore, even the simple exchange shown in Table 2.1 demonstrates multiple issues in determination of intent. The first issue appears in turn 2, where the customer input is ambiguous if examined outside of the context of this conversation. There is no indication to what object the fee applies to, it is implied by the question the agent asked. To resolve these fragmented sentences, the IVA must keep track of what it asked the user so that it can resolve the ambiguity using the current context of the conversation. A preprocessing step may be to insert the subject reference directly into the user input so that the NLU processes the input as "Is there a fee for more than one *bag*?" [39].

The second issue appears in turn 4, where the customer states "In that case ..." As mentioned in Section 2.2, *that* in this sentence is an example of *anaphora*. It is the replacement of a word or idea from the current context of interaction with a single word [68]. If this sentence was examined outside of the context of the conversation it would be impossible to know what the word *that* referred to. To resolve this, the NLU may employ some form of anaphora resolution which will link the word *that* to the concept of free baggage which was introduced earlier [69].

With these replacements and linking of concepts, the NLU has a better chance of understanding these inputs than if it were to try and process them purely by the content of each input alone. Therefore, as it is important that IVAs take into account the current conversational context when determining user intention, an outside observer who is reviewing conversations between an IVA and a human user must do the same.

While mapping conversational turns to an intent is similar to labeling documents with a class or category, turns are typically only one or two sentences long and, as demonstrated above, their interpretations are dependent on the conversational context they appear in. Furthermore, while text documents usually belong to multiple classes and many reliable methods exist to detect them [70], determining multiple intentions within a single turn is still a very hard problem due to the limited text size. An example of a turn with multiple intentions would be "Can you tell me why I don't get channel 24 anymore and when my next bill is due?" Either user intention would initiate a different path in the conversation on the part of the agent to answer, the first a potentially deeper conversational investigation into the date and circumstances of the channel disappearance, and the second a query into their account on a back end system to find the bill due date. A single intent classifier at best would only be able to respond to one of the two intentions and leave the user to repeat the other if they still want to address it. At worst the additional language would add noise to the determination of either intention and confuse the IVA leading to a completely unrelated response.

Recent attempts to detect multiple intents in a single turn see a trade-off in accuracy with single intent detection [71]. Very recent work has seen some promising results [18], however most current intent classification research in dialog systems still assumes single intent detection [72, 73, 74]. This is in part due to the fact that even humans have a hard time agreeing on the presence of more than one distinct intention within a single text [75]. Therefore, in this work, we consider the IVAs under review to be single intent classifiers.

2.5 Existing Refinement Processes

IVAs for customer service are typically deployed in a specific language domain such as transportation, insurance, product support, or finance [76]. In many existing processes, semi-experts in the domain are given a sample of recent conversations collected from a live IVA for review. The semi-experts, or *reviewers* as they are hereafter referred to, need only be familiar with any domain specific terminology, for example insurance vocabulary, to be qualified to review conversations. This poses difficulty in the utilization of crowd-sourced platforms such as Crowdflower[5]

[5] `https://www.crowdflower.com`

or Amazon's Mechanical Turk[6] as there must be some selection process applied to the workers to ensure they have proper knowledge of the domain and associated terminology. One strategy offered by many such platforms is to create a series of tests that workers must pass before they are allowed access to the task. Another strategy is to mix in conversations with known labels to the batch and score reviewers on them. Those that do poorly on the known conversations are tossed out.

The sample to be reviewed can be selected in a variety of ways. If a particular event is important to analyze, there may be flags set on a conversation by the live IVA indicating the event occurred. An example of such events would be if a user escalated the conversation to a different party or the conversation was abandoned before the initiated task was complete. A sample for review can then be created by selecting all conversations containing a flag of interest. These samples will obviously be biased and may miss many other important failure scenarios, so for a more holistic view of interactions a random sample can be used. Another selection strategy mentioned in Chapter 1 is to review the interactions where the NLU and/or ASR confidence score is lower than some predetermined threshold. In this case, reviewers rely on the system itself to indicate where error lies. As previously discussed in detail in Chapter 1, while low confidence is potentially more effective than a random sample at finding poor interactions, a major purpose of review is to discover error in the system. Any existing error in confidence calculation can effect the quality of the selection process. It also creates a dependency on the underlying system implementation that makes it difficult to compare the performance of different IVAs, or, if the system design is ever modified, the same IVA over time.

Once a sample is retrieved by one of the above-mentioned means they are manually graded in an effort to find intents which need improvement. If a reviewer decides that an intent was not appropriate given the user utterance, he or she may indicate the intent the IVA should have selected. The reviewers may assign grades per turn

[6] `https://www.mturk.com`

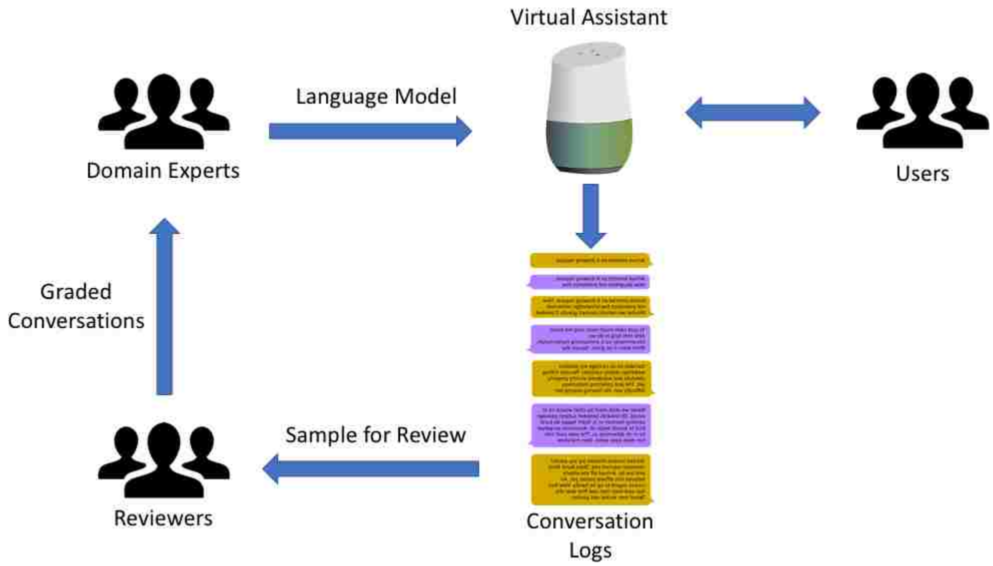


Figure 2.3: Typical language model refinement cycle

in any form, but a star rating system such as one-to-five stars is common [77].

The result of this review process is a set of conversations along with their grades and any suggested intent replacements which are passed to the *domain experts*. Domain experts are typically linguists or computer scientists with training in NLP and are responsible for the construction and modification of language models. Only poorly graded conversations require in-depth analysis by domain experts to determine the necessary changes to the language models.

This process can be visualized as in Figure 2.3. The domain experts construct and refine the language models which are deployed into the live virtual assistant. These assistants, which are the IVAs, then interact with human users and generate conversation logs which are then sampled for human review. The graded conversations are then given to the domain experts to make necessary adjustments to the language model. The faster this cycle completes, the more quickly the IVA can adapt to changes in domain language, environmental events that cause transportation interruptions, or product or website changes that require additional knowledge.

It is important to note that this refinement cycle would look the same regardless of the implementation details of the IVA. There are multiple approaches to intent recognition within the NLU such as in [78, 79, 80, 64, 63]. Whether the language models are created by statistical means (e.g. Support Vector Machines or Neural Networks) or created manually (regular expressions and grammars), there is no difference in the conversation log format. Therefore, we may ignore the implementation details of the IVA and focus on only conversational features. This makes the system presented herein more *general* and applicable to all single intent IVAs regardless of their complexity or implementation.

Chapter 3

Related Work

Although related to other Human Computer Interface (HCI) technologies, there are differences that prevent existing HCI methods of automated review to apply to IVAs. While IVAs are similar to general conversational agents (known as “chat bots” or “chatterbots”), IVAs are *goal-oriented*. The primary purpose of an IVA is not to be indistinguishable from a human (the Turing test) but to help users complete tasks by providing relevant information. Therefore, regardless of user opinion, there exist topics that an IVA is expected *not* to understand, and this is by design. Business rules dictate what an IVA should and should not know, complicating review. For this reason, surveys, star rating systems, and other measures of user satisfaction alone cannot determine all the errors in language models [81]. However, user ratings are still positively correlated with task completion and goal success [81], so we do not discount them completely. Users may rate an IVA poorly simply because it did not allow the user to violate business rules and *not* due to misunderstanding of the user intention. For example, denying the transfer of a ticket to another party or a refund for a product are common cases where users give poor ratings in retaliation despite the IVA understanding their intentions correctly. The IVA is bound to the business rules it was encoded with and, unlike human agents, will not bend those rules based on feeling sorry for the user or wanting to improve its feedback survey results for personal gain.

IVAs share features with Spoken Dialog Systems (SDS), one of which is a conversational interface that may accept speech input from a user. Methods introducing confidence measures in language models used by the SDS typically rely on features present in the acoustic models, word lattice density, etc. [82, 83, 84]. Auditory and acoustic prosodic features are also used in other works to detect error in the SDS [85, 86, 87, 88, 89]. Although there are similarities, IVAs differ from Spoken Dialog Systems in that they support other forms of interactions. For many textual IVAs deployed on business websites, speech may not be the primary channel of communication; it may not even be an available option.

Our topic is measuring posthoc *risk of missed intent* in a turn for the purpose of language model development. As such, we are only concerned with the direct input and output of NLU; Automatic Speech Recognition (ASR) error detection and correction are considered outside the scope of this work. Speech may not be present in some or even all the turns of a conversation. In addition, analysis is done offline, so no ASR features are available. Thus, the aforementioned approaches are not applicable for determining misunderstood turns. In addition, textual IVAs face various user input formatting errors, an issue not present in SDS. Spelling errors, incorrect punctuation, abbreviations, bad grammar, unicode symbols, foreign languages, emoticons, slang, and Short Message Service (SMS) language are commonplace, and IVAs must be able to handle them.

Many approaches to online detection of SDS errors use a large collection of features collected from the ASR, prosody, dialog manager, discourse history, and the NLU [90, 91]. However, not only do these strategies rely on acoustic features and prosodic features, they also depend on features only found in the conversational state of the live IVA. There is an abundance of literature focused on finding ASR errors or SDS miscommunication errors propagated by the ASR; there is surprisingly little literature that focuses on the detection of understanding errors *in isolation*.

As for IVA specific work, a frame-based intention recognition confidence method is proposed for correction of online misunderstanding [92]. This work is narrowly focused on tasks within intent recognition which are modelled as *frames* or *slot-value pairs*. These tasks generally require gathering multiple pieces of information from the user necessary to perform an action such as booking a flight or transferring money between different accounts [93]. The authors use discourse-related features combined with speech recognition features to train confidence models. These models are used to determine if a specific slot value is correct or incorrect which will suppress or trigger a clarifying question, respectively, by the IVA. The authors also only consider slots that have already been filled, ignoring error scenarios where a slot value, although present in the utterance, was not detected. As the authors' system does not consider the general task of determining missed user intention, requires speech recognition features, and who's purpose is to modify the IVA's responses live, it has little in common with the system presented in this work.

In [94], conversations from a SDS that provides train timetable information are collected and conversational *cues* are examined to determine if the user turn was misunderstood. Although this work involved data from a SDS, no acoustic features were used in their methods. The authors assume Clark's principle of least collaborative effort [95, 96]; both the user and system want the dialog to be finished as efficiently as possible and with success. Certain combinations of cues are found to have the best predictive potential for discovering the presence or absence of problematic conversations. Cues include turn length, marked or unmarked word order (topicalization or extraposition), confirmation, the presence or absence of an answer, corrections or repetitions, and new information. The highest precision is achieved with a combination of correction and repetition cues on a small set of 120 dialogs; users tend to repeat their requests and correct the system in its interpretation of these requests when there are communication problems. However, this method assumes that all interactions are frame-based tasks and uses knowledge of the live dialog state specific to that SDS. This constrains the usefulness of their findings in an offline setting and when no assumptions of the underlying language model are desired. In addition, the work relied on a very small corpus in a single language domain. Still, the accuracy of specific combined cues in finding misunderstanding (96.7%) gives hope that an effective error detection system using only text-based methods is possible. Therefore, the cues that are broadly applicable, such as repetition and corrective language, are incorporated into the system presented herein.

The QA^{RT} system presented in [97] monitors live customer service dialogs and provides supervisors with visualizations and summaries of ongoing chats. It employed features in the categories of customer behavior (emotion and sentiment), conversational characteristics (deviation from typical structure, number of turns, average delays), and organizational compliance (greeted customer, used customer name, assurance, etc.). When compared against human reviewers on an annotated set of 188 real-world dialogs, the system greatly outperformed the humans in the categories of organizational compliance and conversational characteristics. It was less accurate in emotion detection and nearly identical in sentiment detection.

The QA^{RT} system does not attempt to detect misunderstanding or user intention and is monitoring human-to-human chats. However, measuring the change in sentiment and emotion in the detection of problematic chats, as well as measuring a conversation's adherence to a typical structure for a given task, prove useful for indicating misunderstanding and have been implemented in the system presented in this work.

Perhaps the most similar work to the system we constructed is [81] where one of the authors' goals was to predict intent classification quality of an IVA using numerous ASR, dialog, and tactile features. 60 users were asked to complete three categories of tasks using the IVA: device control, web search, and chat. After each task, the users were given a survey to rate their satisfaction with the IVA, the quality of speech recognition, and the quality of intent understanding. The authors then combined statistical analysis and classification methods to determine user satisfaction, ASR quality, and intent classification quality in each category of task.

Authors in [81] compared sequences of user actions to request and response features in discovering each class of error. It was determined that action sequences were not as accurate in the prediction of intent classification accuracy. Even the request and response features were only able to produce 0.59 average F1 score for chat tasks. Features they found to be correlated to intent classification errors, such as turn similarity and repeated responses, have been incorporated into our system as indications of missed intent. However, there is much room for improvement. For example, the system in [81] does not provide any guidance as to where a turn *should* have been mapped. Also, the authors rely on the users themselves to determine the intent classification accuracy which can be biased by the IVA response. In their study, the user only sees the final response and not the direct output of the NLU (the intention label). Poor response wording can appear to the user as a missed intent when, in reality, the NLU component understood correctly but the NLG component was to blame for the communication error (see Section 4.3 for discussion).

None of the systems above are an attempt to automate error discovery isolated to the NLU language model and are designed to handle the scale of data generated by customer service centers of multiple large corporations. However, as noted, many features and methods used by these related systems are useful in the construction of our system. They are included in the risk indicators defined in Section 4.4. The following chapter investigates the structure of human communication and its subsequent review, and derives the approach taken to automate such a review process.

Chapter 4

Approach

We begin to design a system for conversation review by first construing what human communication *is* and how communication can take place through conversation. By analyzing the structures and rules at play in conversation we can proceed to compiling characteristics of mis-communication within conversation and means to detect them.

4.1 Communication through Conversation

Research in the sociological and psychological foundations of communication, including that of Grice [98], has suggested maxims that support effective human communication. A set of *cooperative principles* underlies how people interact with each other. As Grice says "Make your contribution such as it is required, at the stage at which it occurs, by the accepted purpose or direction of the talk exchange in which you are engaged." To support cooperative and effective communication Grice proposes a number of maxims, including:

- The requirement for Quality - Do not say what you cannot support with evidence or know to be false.

- The maxim of Quantity - Say only what is necessary to convey the required information and no further superfluous information.
- The maxim of Relevance - Preserve during a conversation the concepts, the context, and on the line of reasoning of a conversation.
- The maxim of Manner - Avoid obscure and ambiguous expressions; be brief, focused, and keep to the point of the conversation.

Although these maxims were first described more than forty years ago, they remain especially relevant to the designers of modern conversational interfaces. Within the context of IVAs, humans are generally not *trying* to be misunderstood while talking to them. A human user will have a purpose in initiating communication with an IVA, and that purpose will drive the conversation. Following Grice's maxims of quantity and manner, a user *should* make their contribution to the conversation as clear and concise as needed to allow the other party to understand. This notion is commonly known as the *principle of least effort* [95, 96]; the underlying assumption is that both the user and system want the dialog to be finished as efficiently as possible and with success.

However, in practice communicants can fall short of these ideals. Clark details three problems with this principal in [95]:

- Time pressure - Speakers do not always take the time and effort needed to produce a perfect utterance. They may introduce pauses and inject filler as in "You know that actor <pause> what's his name <pause> oh yeah, Tom Hanks?". Specifically in terms of keyboard input, humans take grammatical shortcuts and abbreviate to save time and effort at the cost of clarity, as in "brb k?" in place of "I will be right back, okay?".
- Errors - Speakers make mistakes and amend their phrases partway through as

they think better of how they want to word their turn. An example, “Do you have a room this Fri- Thursday?”.

- Ignorance - Speakers realize they do not know enough about what they are trying to communicate to formulate a proper utterance. They may inject words such as “you know”, “like”, or “I think” to signal their uncertainty [99, 100]. They may use a question mark at the end of a statement as a *try marker* [101] to signal to the other party they are unsure of their performance. They can also end their utterance with a request for confirmation such as “... (do you) know what I mean?”.

In the light of these imperfections with human communication, Clark modifies the principal of least effort to the principle of least *collaborative* effort, which is defined as *“In conversation, the participants try to minimize their collaborative effort - the work that both do from the initation of each contribution to its mutual acceptance”* [95, 96]. This collaboration requires that the other party pick up on such cues from the speaker and correct errors from time pressure or ignorance to demonstrate that understanding on the part of the listener has occurred. However, when subjected to empirical tests on a joint task corpus, this collaborative effect was not proven [102]. Instead, the participants appeared to work out when to spend less effort by risk-taking where communication problems have not occurred, and decrease risk-taking where problems have occurred. The risk-taking takes the form of grammatical shortcuts, anaphora, and abbreviations which decreases the effort by the speaker, but can also decrease the clarity to the listener. This risk-effort trade-off, the authors theorize, better explains the observed behaviors of participants in conversation. They name this individual effort-minimizing behavior as the principal of least *individual* effort[102].

4.1.1 Context in Communication

In order to achieve human emulation in Grice's maxim of relevance, the IVA must answer every interaction according to the current conversational context. It seems very commonsense to say that when two humans interact they each know the context of their current discussion. This context might be current government deficit spending, quality of education in their local school system, or the probable outcome of a current athletic event. Such shared context in conversation may be referred to as *common ground* [103]. Acknowledgement and support of this common ground with users is critical for agent interaction to ensure that the conversation stays on course.

For example, if a user booking a trip to Chicago were to question the weather in Chicago, the virtual agent must interpret their request within the common ground of the trip the user is currently booking. The IVA may signal that common ground is still held with a reply like:

"The weather in Chicago will be mild and overcast *on <day user will arrive>.*"

By referencing the date of the user's arrival in the response, the IVA is signaling that it is understanding the question within the context of the trip. This is called an *implicit* verification strategy [94]. The alternative, *explicit* verification, would be to ask the user:

"Are you asking about the weather on <day user will arrive>?"

The explicit method requires another pair of turns to verify that the user and the agent are both talking about the weather in Chicago on the same date. Regardless of which method is used, some means of verification is needed to reassure the user that the IVA still understands the conversational context.

The notions of both context and intent are important in intelligent human conversation. While the common ground between participants in a conversation must be quickly established for the conversation to progress, it must also be continuously maintained by both parties. This continuous updating of common ground between the participants during discourse is referred to as *grounding*, and is well explored by Clark [96]. In the discussion of the previous paragraph, suppose the user's intent is to purchase a vacation. In the context of purchasing a ticket to a specific location, he/she wonders what the weather in Chicago might be. Thus the virtual agent must maintain an agenda for user intentionality (purchasing a vacation package) as well as updating an organized set of contexts (ticket purchase to luggage allowances to over-sized baggage costs, discuss weather information, and then back to ticket purchase) as the conversation progresses.

Finally, we must acknowledge the issue of *context slippages*. It is possible that the user goes to an IVA for the solution of a novel problem. The initial task may appear to be well known, for example changing a flight, but the context in which this task is occurring is unknown to the agent. For example, with the 2010 Icelandic volcano interference in airline travel, user queries were to create new travel routes avoiding the disruptions of the volcano. Another example would be to schedule a new flight pattern that was not affected by a current summer storm curtailing traffic in a particular hub city. Altering flight schedules when contexts are constantly changing across time demands the IVA be able to pursue the users intent within the fluidity of normal changing circumstances.

4.1.2 Misunderstanding in Conversation

To summarize, automating natural language understanding to successfully operate under the framework of least individual effort, context, common ground, context slippages, time pressure, speech and spelling errors, and ignorance is a challenge to say the least. Even humans can experience errors in communication when conversing with other humans. When these errors in understanding occur and become apparent to the participants, humans employ various means to correct them so that the conversation can continue [94, 104]. Some common corrections may be for speakers to repeat or rephrase their previous statement or question, or to disconfirm or correct listeners understanding such as “No, not Fargo.” or “I said Chicago *not* Fargo.” If mis-understanding cannot be resolved between the participants, the conversation may be escalated to another party to resolve the communication issue or abandoned altogether.

Even if a misunderstanding can be recovered from and the conversation completes successfully, discovering the cause of the misunderstanding can lead to improvements in the IVA, and therefore user experience, and more intelligent error resolution strategies [105]. For example, in the following conversation, several communication errors on the part of the IVA are apparent.

> **User:** I can not set up my mothers email from my home computer. I am an authorized user for her account.
> **Agent:** Would you like to create a new Email address or set up your existing Email address with an email program such as Outlook, Apple Mail, etc...?
> **User:** yes
> **Agent:** What else can I help you with?
> **User:** can not set up email
> **Agent:** I have found more than one answer to your question. Please select the link below that interests you.
> **User:** Set Up Email
> **Agent:** Would you like to create a new Email address or set up your existing Email address with an email program such as Outlook, Apple Mail, etc...?
> **User:** set up email with an email program
> **Agent:** To set up your Email with an Email Program select your Email Program below.
> **User:** Outlook and Outlook Express

> **Agent:** To send and receive Email with your new address you'll need to enter some basic information into your Email program. On the page I have opened for you click the link for the Outlook program you are using for specific setup instructions.

In this conversation, the IVA initially can not tell if user is trying to set up and existing email or create a new one. When the user replies with "yes", the IVA does not understand what the user is referring to and replies with a generic offer to be of further assistance. This leads the user to repeat their original query but in a much clarified form, lacking the backstory and justification present in the original query. The IVA is still unsure of the best response so it asks the user to clarify the task, which leads it back to its original response. At this point the user very clearly repeats their original query but with the specifics the IVA needs to understand the task. From there on the conversation progresses again with the IVA walking the user through the steps needed to set up and existing email account using the Outlook mail application.

The detection of such misunderstandings can lead to improvements in the IVA's NLU component, and therefore improvements in its communicative abilities and the user experience. In the following sections we discuss the discovery of specific indications of miscommunication and how they are combined into a system for automatic conversation review.

4.2 Conversation Review

Having established the structure and means of conversation, we now turn our attention to the review process and how the automation of that process can be accomplished. Recall from Section 2.5 there is a sampling method applied to the conversation logs to select data for review by humans. After the review is completed, the results are given to the domain experts to make the necessary changes in the language model to improve the IVA's understanding ability. This refinement cycle, visualized in Figure 2.3, can be continuous or periodic but it must be done for quality assurance and to adapt the IVA to changes in language and its environment.

In order to automate, or at least supplement, the work of the reviewers we must first understand what process a human reviewer follows to determine if a particular user turn was misunderstood by the IVA or not. This study began with multiple interviews conducted with 15 seasoned reviewers of IVA dialogs in various domains. In the interviews, the following questions were asked of each reviewer:

1. How do you determine if a turn was misunderstood?
2. What features of the turn do you look for in making that determination?
3. How do you incorporate the conversational context into your decision?
4. What patterns of miscommunication do you commonly see?
5. How do you discover if there is a "better" intent in the IVA knowledge base or not for a given turn?

The results of these surveys were compiled along with a literature review of indications of miscommunication to solidify concrete features to detect. As reviewing a conversation and deciding if the user was misunderstood at any turn is a subjective task, we looked for common patterns in reasoning that could be formalized into classification tasks. Next, each feature was considered in isolation as a possible indication of miscommunication, hereafter referred to as a *risk indicator*. For each turn in each conversation under review, the proposed system will annotate the value of each risk indicator. This allows one to measure the utility of different conversational features in discovering IVA misunderstandings. Some risk indicators will have a stronger correlation to missed intent than others; thus, these indicators are weighed.

Conversation	Turn	A Present	B Present	C Present	Risk Score
1	1	0	1	0	0.5
1	2	1	0	0	0.2
1	3	1	1	1	0.8
2	1	1	1	0	0.7
2	2	0	1	1	0.6
2	3	0	0	0	0.0

Table 4.2: An example of a simple risk score calculation using three risk indicators. Feature A has the weight of 0.2, B the weight 0.5, and C the weight 0.1.

A risk ranking system is created by aggregating these weighed risk indications into a single score similar to the methods used for database similarity search [106] and web search rankings [107, 108, 109]. This risk score, derived from all risk indicators present in the turn, can then be taken as the *risk of missed intent* for a single turn in a conversation under review. The system uses this per-turn risk score to determine if the appropriate intent was selected and also provide guidance to the domain experts.

For example, suppose we had three risk indicators: A, B, and C, and trained three binary classifiers (one for each indicator) that returned 1 if they were present in the given turn. Then suppose we assigned weights to each indicator based on prior correlation to missed intent. We could start with a very simple aggregation scheme that takes the weighted sum of the risk indicators present as the risk score. Table 4.2 demonstrates the calculation of the risk score for each of three turns in two different conversations by a weighted sum. If human reviewers only had time to look at two turns in this example, we would show them turn 3 of conversation 1 first, followed by turn 1 of conversation 2. By this risk ordering, we use their limited time to inspect the turns most likely to be misunderstood.

For training of the system, we would release the complete conversations, instead of just the top risky turns, to reviewers who would judge whether or not each turn was misunderstood. Using the human labels, we could go back and adjust the weights of each risk indicator or modify the aggregation scheme to maximize the risk score

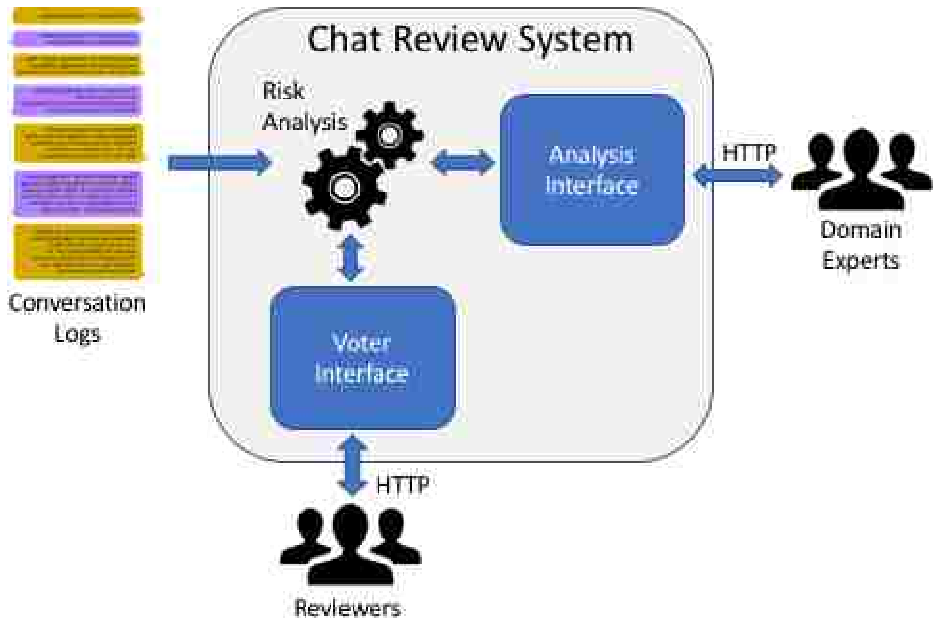


Figure 4.1: The main components of the CRS

for those turns labeled by reviewers as misunderstood.

The specific risk indicators are introduced in section 4.4 and the implementation details of methods to detect each one are covered in Chapter 6. Several means of aggregating these risk indicators are compared in Chapter 11. This ranking and analysis system will be hereafter referred to as the Chat Review System (CRS). The following CRS design and evaluation using a subset of proposed risk indicators was published in [110].

4.3 The Chat Review System

The CRS is designed with three primary functions, visualized in Figure 4.1. The first is detecting features of missed intent and aggregating these features into a risk score, which is the primary focus of this work. This functionality is implemented in the risk analysis engine.

The web application provides interfaces for two primary types of users. The first type of users are the semi-experts, or human reviewers. Their work flow involves logging into a project (a collection of conversations from a live IVA) and reviewing turns that have been prioritized for them by the CRS. The second type of users are the domain experts who create projects (linked to a live IVA) and select a time range over which to do analysis. In practice, this time range may be the last week or month, but commonly it is the period of time since the last language model deployment to the live IVA. Once they have defined a project and a time range for review, the CRS prioritizes the conversations and turns within that range by risk score. The reviewers then look at the most risky turns for the number they have been assigned to review. They read the turn in the context of the conversation and vote on whether or not they agree with the intent label chosen by the NLU in the live IVA. If a reviewer does not feel they have enough information or domain knowledge to decide, they may also vote ”unsure”.

It is important to note here that by design the reviewers do not see the IVA's response text for the given user turn. They only see the user turn and the selected intent in context of the conversation as shown in Table 4.3. This is to decouple errors in the intent recognition phase from errors in the response generation phase. Recall from Section 2.3 that response generation is performed in the Natural Language Generation (NLG) component *after* the user intent has been selected by the Natural Language Understanding (NLU) component. Therefore, reviewers can be misled by poor response selection or wording into thinking that the IVA misunderstood the user, when, in fact, it understood correctly but did not respond appropriately. This decoupling is very important to narrow down where errors are introduced in the IVA's ability to communicate. In large IVAs it is possible that the NLU and NLG are maintained by different people or altogether different departments or roles. With commercial IVAs, response text may have to go through an approval process and even a legal review to ensure that the information the IVA is giving the user is accurate

User Turn	**Intent Hit**	**Correct Intent**	**Conversation #**	**Entry #**
My TV is not working I need to have it fixed.	TV Support	TV Support	26789	1
My TV will not start I want to talk with support staff	Contact Information Deflection	Live Chat	26789	2
I cannot get my tv to work	TV Support	TV Support	26789	3
How can I get a person to help me get the TV started?	TV Services	Live Chat	26789	4
How can I speak with a support staff to get my TV to work?	TV Support	Live Chat	26789	5
There are no links showing my problem what now?	Do Not Understand	Virtual Assistant Not Helpful	26789	6

Table 4.3: A conversation with missed intents in turns 2, 4, 5, and 6.

and legal. Therefore, it is critical that errors in the NLU be separated from errors in the NLG so that the appropriate corrections can be identified.

When the voting is complete, domain experts are presented with the voting results and system recommendations as detailed later in section 5.1.3. Using these results they can make the necessary changes to the language model, deploy a new version to the live IVA, and repeat the review cycle once more conversations are gathered.

4.4 Indications of Missed Intent

Table 4.3 displays a conversation with a live IVA in the telecommunications domain. Although the IVA selected correct intents in Entry 1 and 3, the user was not satisfied and continued to restate his or her issue. This can occur when the response text is poorly worded or incomplete, pointing to an error in the NLG component that should be investigated. It could also indicate the user is not stating his or her true intent with enough detail to get a satisfactory response.

An "I Don't Know" (IDK) occurs when the language model does not find an intent that satisfies the user query with a high enough confidence. The IVA may respond with something like "I'm sorry I don't understand you. Please revise your question." Such an intent is occurred in Entry 6 of Table 4.3. Either the correct intent does not exist in the IVA's knowledge base or it does exist and the mapping to that intent is faulty. However, with the former, sometimes an IDK is appropriate. For example, the user could ask something off-topic such as what the IVA's favorite sports team is. Utterances triggering acceptable IDK responses are classified as *out-of-domain* and *out-of-application-scope* utterances in [105, 111]. Therefore not every IDK response is a missed intent. IDKs must be weighed and combined with other indications of risk into an overall risk score for the turns they occur in.

In order to generate the risk of missed intent for each turn-intent pair, the CRS incorporates indicators of risk listed in the following section. Additional indicators not listed here may be explored in future work. The implementation of means to detect each feature is described at length in Chapter 6.

4.4.1 Conversation Level Features

The following features apply risk equally across all turns within the single conversation where they are present. These features are used to detect miscommunication over the course of the conversation and elevate the risk score for turns in conversations where miscommunication was likely to have occurred.

idk_in_conv If a conversation contains one or more IDK responses, this may indicate that the user is talking about some subject the IVA has no knowledge of.

multi_in_conv The same intent is hit multiple times within the conversation (not necessarily successively). This is an indication of risk within a customer service conversation because it is unlikely the user would want to see the same response more than once. Examples of this include Entries 1, 3, and 5 of Table 4.3.

seq_in_conv If the same intent responds successively in a conversation they are considered to be *sequential hits*. This usually indicates that the response to the first input did not satisfy the user; he or she is rewording the question to get a different response. If the system has the initiative, this may mean that the system is repeating a prompt, a common indication of miscommunication [104].

tie_in_conv The responding class for one or more turns in the conversations had a nearly identical score as one or more different class(es). This indicates confusion in the model around the input language for the tying classes. If a conversation contains such ties it may surface subject matter that is not well defined or the IVA needs additional training in.

conv_rating In some cases, users are asked for feedback on how helpful the IVA was for their session. This feedback can be used as a measure of risk. Feedback is not entirely reliable, however, as the user can give negative feedback if the IVA did not allow the user to do something he or she wanted. For example, business rules may not allow transferring a ticket to a different passenger and when a user attempts to do so the IVA will not let them. In retribution the user grades the conversation poorly, but this should not reflect negatively on the conversational ability of the IVA. The user may also say the IVA was unhelpful when the NLU was indeed working correctly, but the response text was poorly worded.

conv_should_esc Whether or not there was explicit user request for an escalation in the conversation, an algorithm (see Chapter 7) has determined that the conversation should have been escalated due to IVA failures to complete the task at hand.

sent_change The user began the conversation with positive or neutral sentiment, but by the end of the conversation their sentiment was negative. This may be caused by either the IVA preventing them from completing a task due to business rules, or due to IVA misunderstanding.

4.4.2 Turn Level Features

The following features only apply risk to a single turn. However, they may still use features of the conversational context in their determination.

triggers_idk If the response to this turn is an IDK, this may indicate that the user is talking about a subject the IVA does not have knowledge of.

user_rating In some cases, users are asked for feedback on individual IVA responses. This feedback can be used as a measure of risk. Feedback is not entirely reliable, however, as the user can give negative feedback if the IVA informs the user that he or she is not allowed to do something as explained in **conv_rating**. The user may also say the IVA was unhelpful when the NLU was indeed working correctly, but the response text was poorly worded.

backstory Users may give backstory on their task that is unnecessary for determining the correct intent. The presence of this language can add confusion in the NLU and result in a missed intent [75]. This is more prevalent when users do not realize they are conversing with an IVA. For example, a user may tell the IVA that he or she needs to fly to Boston for a son's graduation party. The fact that the user has a son and is attending his graduation party is irrelevant to the task. The additional language can interfere with determining the user's primary task of booking a flight. See Chapter 8 for a full discussion.

precedes_corr	There is presence of user error correction language, such as *"no, ..", "I said ..", ".. not what I .."* [112, 113].
abandonment	The user left the conversation immediately after the IVA asked them a question. This indicates that the IVA did not have all the information it needed to complete the task, but the user abandonment indicates it was likely trying to accomplish the *wrong* task.
multi_intent	As we assume the IVA under review does not support multiple intents within a single turn, if multiple intents were indeed present it can add confusion to the NLU. Using a method such as [18] we can detect if multiple intents are present and add risk if so.
triggers_seq	Occurs when the turn hits the same intent as the previous turn. This usually indicates that the previous response did not satisfy the user, so he or she is rewording the question to get a different response but failed to do so.
triggers_impasse	When the same intent is returned more than two times in a row, the IVA will trigger an *impasse*. The impasse intent may cause the IVA to respond with something like *"I think you are asking for more information than I have. Please contact customer service for more information."*

precedes_esc An *escalation* occurs when a user requests an alternative channel for the completion of a task [114]. As this may be due to IVA failures, risk is assigned to the turn preceding the escalation request. Entries 2, 4, and 5 of Table 4.3 are examples of escalation requests.

precedes_unhelpful The input directly preceded a turn stating the unhelpfulness of the IVA. This is a common reaction when the user is frustrated at the inability to make progress in their task.

precedes_profanity The input directly preceded an interaction containing profanity. In a customer service domain, profanity is usually a sign of user frustration or irritation.

precedes_neg If a turn contains negative sentiment, this may be due to the user's reaction to the previous IVA response. Therefore, risk is assigned to the preceding user turn.

restated If a turn is very similar to one or more following turns, risk is assigned to it as this may indicate the user was dissatisfied with the response and rewords the question. Similarity is defined as a rephrasing of the same question or statement; it may not have triggered the same intent in the IVA [113].

precedes_idk	It has been observed that IDKs may follow misunderstood turns, as in line 5 of Table 4.3. This IDK can happen when the user reacts in surprise or frustration (*"What??"*) or changes the subject to complaining about the IVA due to the misunderstanding (*"This is the dumbest thing I have ever used!"*).
external_clf	An external classifier disagrees with the intent selected by the NLU. See section 4.4.3 for detailed explanation.
pni_origin	An external classifier mapped this turn to a different intent with high confidence. See section 5.1.2 for detailed explanation.
triggers_tie	The responding class had a nearly identical score as one or more different class(es). This indicates confusion in the model around the input language.
unknown_words	The user turn contains words that are Out of Vocabulary (OoV) for the underlying language model. This may indicate that the user is talking about some subject the IVA does not have knowledge of.
should_esc_point	There was no explicit user request for an escalation in the conversation, but an algorithm (see Section 7.8.3) has determined that the conversation should have escalated at this point in the conversation.

response_class This user turn follows a combination of IVA response media known to lead to higher incidence of dissatisfaction on the part of the user. This indicator *decreases* the risk of missed intent on the current user turn as any indicators of missed intent on the current turn may actually be due to the *user* misunderstanding the IVA. See Section 9.8 for detail.

response_complexity This user turn follows a IVA response text complexity feature known to lead to higher incidence of dissatisfaction on the part of the user. This indicator *decreases* the risk of missed intent on the current user turn as any indicators of missed intent on the current turn may actually be due to the *user* misunderstanding the IVA. See Section 9.8 for detail.

4.4.3 External Classifiers

As mentioned in the previous section, the **external_clf** and **pni_origin** risk indicators use external classifiers in their determination. In this section we will discuss the training and application of these classifiers.

Recall from Section 2.4 that the language model within the NLU is a multi-class classifier that takes in some textual representation of the user input and outputs a class label which represents the user intent. It may be implemented as a statistical model, neural network, a symbolic model, even as simple as a set of prioritized regular expressions. These specifics define the NLU's *implementation*.

If the NLU is implemented as some machine learned model, then there must exist a set of labeled training data originally used to construct the deployed language model. If it is a symbolic model of some kind, there will be some set of regression texts used when constructing and testing the quality of the patterns or grammars. In either case, there will be an existing means to collect labeled data for the intent classes known to the language model.

To determine if error is inherent in the NLU's specific implementation, the CRS builds one or more additional intent classification models, which are trained on the same data used to build the live IVA's language model. Given the intent classifiers in the NLU and the CRS have seen the same training samples, any disagreement in the highest ranked intent between a CRS classifier and the NLU is an indicator of risk. The CRS runs each user turn through each of its classifiers and compares the highest ranked intent to the intent originally selected by the IVA's language model. Any supervised learning method that supports multi-class classification may be employed for this comparison provided it is different from the one used within the NLU. If they were trained using the same method, there would never be any disagreement between the CRS and NLU, assuming the same training data and hyper parameter settings.

For example, if the live IVA uses a Maximum Entropy model (MaxEnt) in its NLU implementation, the CRS may train a Support Vector Machine (SVM) and a Decision Tree (DT) on the same data the original MaxEnt model was trained on. The CRS then feeds a user turn through both the SVM and DT and compares the outputs and confidence to the original output of the MaxEnt from the live IVA. If the SVM selects a different intent from MaxEnt as the "correct" intent with reasonable confidence, the CRS will assign risk to that turn, labeled as **external_clf**. If, however, the DT agrees with the MaxEnt, no **external_clf** risk will be assigned from the DT. In other words, every comparative classification method used is treated as an independent risk indicator. Over time the CRS learns which comparative classification methods are less accurate by using human reviewer feedback, and will assign their disagreement smaller weight. Historically accurate methods will carry higher weight over time.

4.4.4 Combining Features into a Risk Score

Once the CRS has annotated each turn with all applicable risk indicators, the risk score for a particular turn is calculated as follows. Let T be the set consisting of all user turns analyzed by the CRS. Every turn $t \in T$ is assigned a list N_t of risk indicators. A risk indicator is an element of N_t if and only if the indicator is determined to be present in that turn by the CRS. Note that the same risk indicator may be present multiple times in N_t as determined by the CRS.

Every instance n of a risk indicator in the list N_t will be assigned a weight $0 \leq w_n \leq 1$ and applied to the turn-intent pair. Weights are initialized to 0.5 and tuned over time as described in section 5.2.1. The risk score for turn t, known as z_t, is defined as the sum of these weights:

$$z_t = \sum_{n=1}^{|N_t|} w_n \tag{4.1}$$

Consider the following example. Say a turn t was restated two times in the conversation, contained backstory, and was immediately followed by a turn containing negative sentiment. Turn t would have the following list of indicators:

$N_t =$ [**restated**, **restated**, **backstory**, **precedes_neg**]

Assume that after tuning, we derived the following weights for the risk indicators involved:

restated	0.8
backstory	0.6
precedes_neg	0.2

Our risk score for turn t would be: $z_t = 0.8 + 0.8 + 0.6 + 0.2 = 2.4$.

Risk indicators may appear more than once; there is no upper bound for z_t. In order to fairly compare the risk scores between different turn-response pairs, the risk scores are normalized across T as follows. Let $MaxScore$ be the maximum z_t observed in T. The normalized turn risk score is defined by:

$$z'_t = \frac{z_t}{MaxScore} \tag{4.2}$$

z'_t can be considered a turn's measure of risk relative to the riskiest turn in the dataset. Continuing the example from above, if $z_t = 2.4$ and the $MaxScore$ of T was 24, the final risk score for that turn would be $z'_t = 2.4 \div 24 = 0.1$. Henceforth, the risk score assigned to a turn will be normalized.

This risk analysis process and a subset of the features presented in Section 4.4 was prototyped and its performance at surfacing missed intents was evaluated in [110].

This completes the risk analysis process within the CRS. Next, we discuss the application of the CRS to the review cycle, and cover the use of the risk scores in the voter and analysis interfaces.

Chapter 5

Application

At this point we have covered the individual risk indicators and how they are combined into a singular score, z'_t, representing a turn's risk of missed intent relative to the riskiest turn in the conversation logs. We now cover the application of the CRS to a live IVA environment to improve human productivity within the conversation review cycle.

5.1 Augmenting the Existing Refinement Cycle

The CRS initially augments the review cycle by preprocessing the entire chat history from an IVA and prioritizing turns and conversations for the reviewers. It ranks turns by z'_t, the normalized risk of missed intent defined in equation 4.2. In this configuration the CRS makes no decisions on its own, it is functioning as a ranking engine for turns to be reviewed while collecting human decisions from the reviewers used to tune the weights of each risk indicator.

When first deploying the CRS or introducing a new language domain, the risk indicator weights may require tuning. While some risk indicators may carry similar

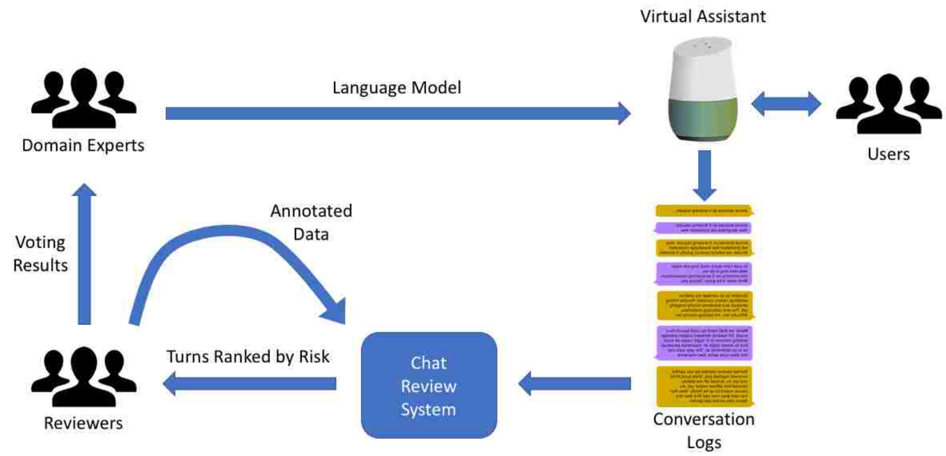


Figure 5.1: Augmented Language Model Refinement Cycle

weight regardless of language domain, others, such as **external_clf**, may vary widely between domains. Therefore, tuning is performed on the weights using labels from the human reviewers. The CRS can be initially incorporated into the existing refinement cycle as shown in Figure 5.1. Compare this process to the initial process shown in Figure 2.3.

5.1.1 Voting in place of grading

In place of grading, reviewers vote on if a given turn was assigned to the optimal intent. Empirical results have shown that decision time increases with the size of the choice set [115]. Therefore, this minor change provides a cognitive speed-up as the review task is reduced to agreeing or disagreeing with the classification, instead of placing it on a scale of quality.

A voting interface is implemented as part of the CRS. To mitigate the subjective nature of review, every turn is shown to a minimum of three reviewers. The majority

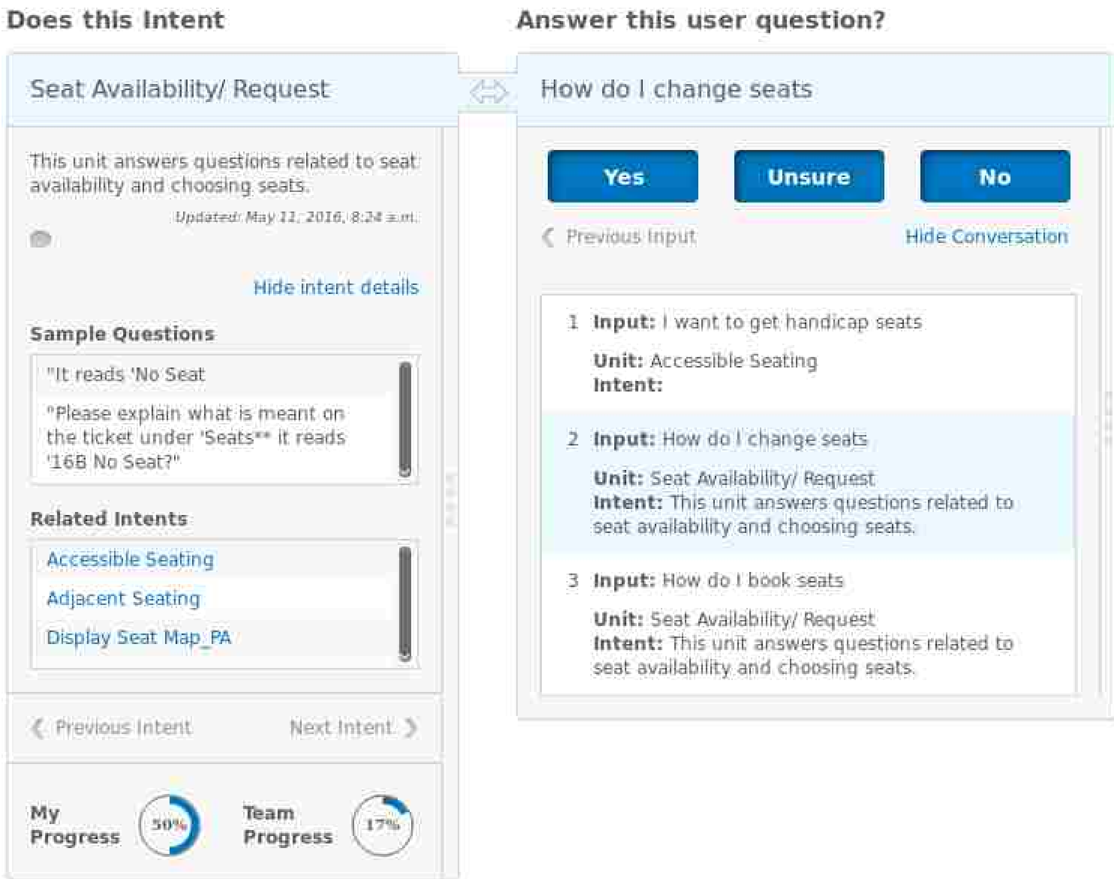


Figure 5.2: CRS Voting Interface

vote is then calculated and used as the truth value on the recommendations made to the domain experts.

The voting interface shows reviewers the turn and the intent selected by the live IVA and presents them with **Yes**, **No**, and **Unsure** buttons. The reviewer clicks on the button that reflects his or her opinion on whether the selected intent was correct given the user utterance. If the reviewers cannot decide, either because the user utterance is ambiguous or because they don't feel they understand the true purpose of the intent, they can vote **Unsure**. To further optimize the interface, keyboard shortcuts can be defined for the three options.

A screen shot of this interface is shown in figure 5.2. In the left-hand column the intent the reviewer is currently voting on is displayed along with additional information to give some insight into its purpose. The label of the intent is displayed at the top, followed by a text description of its purpose, which is maintained by the domain experts. If the reviewers do not fully understand the purpose of an intent, they can submit questions to the domain experts by clicking on the comment bubble below the description text. The experts can then update the description to clarify the purpose of the intent so that voting is accurate.

Next, a set of sample questions that have been previously human-validated to belong to this intent are displayed. This is to give the reviewer some intuition on the type of language captured by the current intent. Following this list is a list of related intents to help the reviewer decide if a more suitable intent exists in the language model. Finally controls to navigate through the intents to be reviewed and, at the bottom, metrics on how many turns have been completed by the current reviewer and all reviewers combined on the displayed intent are shown.

On the right-hand side the user turn is shown followed by voting buttons. Underneath the voting buttons, the entire conversation with the current turn highlighted is displayed to give the reviewer the conversational context needed to determine if the responding intent was appropriate.

Reviewers are presented with the riskiest (highest z'_t value) turns first. After review, only interactions exposing problems in the language model are given to the domain experts for analysis. Therefore, any amount of time spent on reviewing turns that were correctly classified by the IVA is a waste of human effort. In the design phase, we theorized that having reviewers vote on the top N riskiest turns instead of a random selection of size N makes their work more efficient as what they are reviewing will more likely contain a NLU error. We tested this hypothesis in a prototype version of the CRS and found that it did in fact greatly reduce the number of correctly classified inputs that humans saw during review. The results of this experiment was published in [110]. This hypothesis is further validated in the body of this work through our experiments on multiple language domains in Chapter 11.

Notice that nowhere does the actual response *text* from the IVA appear. This is to intentionally separate the evaluation of the natural language understanding (NLU) component from that of the natural language generation (NLG) component. As previously discussed in Section 4.3, poor response wording can be mistaken for a problem with the NLU, when in fact the problem was in the NLG. Recall that in this work we are primarily interested in the evaluation and improvement of the language model, therefore this isolation is necessary. Once it has been established that the NLU is performing acceptably, the NLG can be evaluated separately which is outside of the scope of the CRS.

5.1.2 Suggesting an Alternative Intent

The comparative classifiers described in 4.4.3 provide a second important function. As outlined in Section 2.5, human reviewers or domain experts need to suggest alternative intents when disagreeing with the IVA. These alternatives are then analyzed by the domain experts in combination with the intent chosen by the IVA to determine why the language model mapped the user turn there instead of the correct intent. When they discover the cause of the error, they can repair the language model and deploy it to the IVA.

This alternative intent selection process on top of the review process is time consuming for humans. When disagreement with the IVA is found, humans must search through the set of existing intents and decide which intent the turn *should* have matched. As a means to automate the selection process, the comparative classifiers themselves can provide these alternatives, which are then validated by the reviewers through the voting interface. In this manner, the search process for an alternative

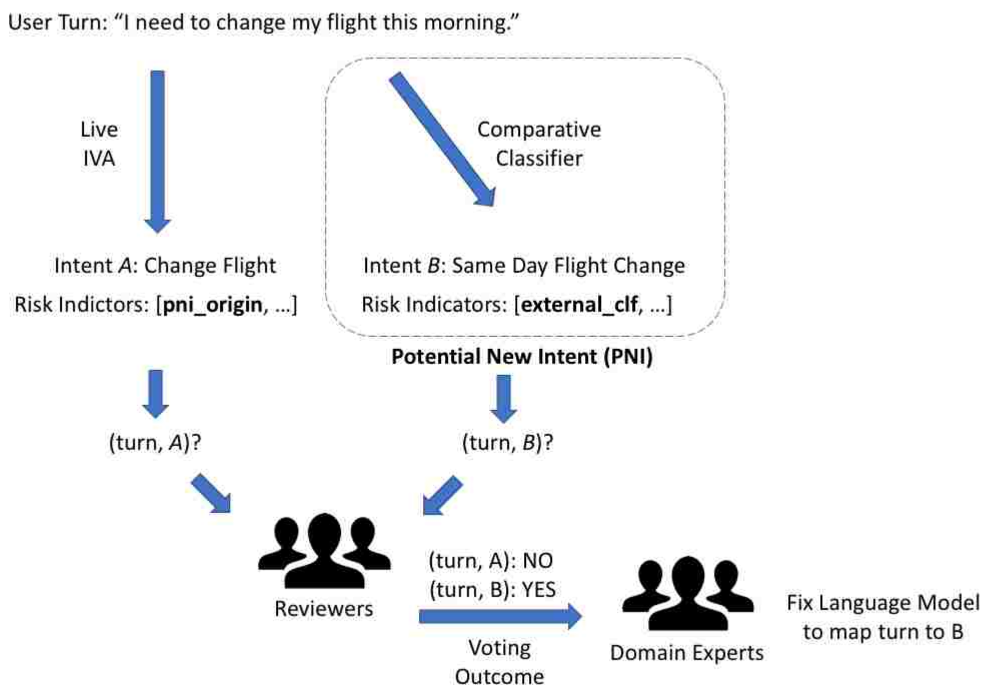


Figure 5.3: The process of generating an alternative intent, known as a Potential New Intent (PNI), and its validation

intent is automated, and humans are only required to vote on if they agree with the alternative choice. If the comparative classifier was proven to be trustworthy through empirical testing (see Section 11.2), this voting step could also be eliminated and the domain experts would receive the alternative directly.

For example, if a comparative classifier selects with high confidence intent B as the correct intent for a turn that originally hit intent A in the live IVA, the CRS will generate the (turn, B) pair for the reviewers to vote on in addition to the original (turn, A). Classifier-generated suggestions are referred to as *Potential New Intents* (PNIs). The voting outcome of a PNI is compared to that of the original intent, and, if reviewers agree, is used to recommend an action to the domain experts. If voters agree with (turn, B) but not the original (turn, A), the CRS will recommend that domain experts make the changes needed to the language model so that the turn maps instead to B. If voters disagree with the PNI, then it is discarded from the suggestions but still may be used as additional training data to correct the classifier(s) suggesting it. If the reviewers happen to agree on both (turn, B) *and* (turn, A) these conflicts are raised to the domain experts as confusion around the purpose of the intents A and B. They can then clarify the definitions of A and B then re-release the conflicting turns for voting. This example is visualized in Figure 5.3.

PNIs are used as an additional indicator of missed intent. If a turn was assigned to a different intent through a PNI, the **pni_origin** risk indicator is assigned to the original turn-intent pair from the IVA. Following the previous example, if a large number of turns originally mapped to intent A are tagged with a **pni_origin** indicator, meaning comparative classifiers selected intents other than A with high confidence, this suggests that within the language model intent A is consuming more language than it should and needs investigation. If, on the other hand, intent B has a large number of PNIs assigned *to* it, the intent may be rejecting appropriate language or is redundant and also needs investigation. Each PNI will be given the risk indicator **external_clf** in addition to the others generated by the original turn.

5.1.3 Recommended Actions

After the risk analysis and voting processes are complete, the CRS provides voting data and additional recommendations to the domain experts to facilitate language model development. To optimize domain experts' time, the CRS uses the reviewer voting outcomes to determine a recommended action per turn, shown in Table 5.1. These actions help the domain experts quickly determine what to do with the voting results for a particular turn.

Suppose a turn had three reviewers vote on it. One voter agreed with the intent, one disagreed, and one was unsure. In this scenario, since there was no consensus between the reviewers the CRS will recommend that the domain expert conduct further analysis on the intent to make the final determination if the intent was appropriate,

Circumstance	**Recommended Action**
A turn-to-intent map is voted to be correct	None. Note that these can be optionally hidden from the domain experts as they require no action.
A turn-to-intent map is voted to be incorrect	Fix the mapping to prevent the turn from reaching the associated intent.
The reviewer majority votes *Not Sure*	Determine if the intent was appropriate for the turn or if a new intent should be created.
There is no reviewer consensus	Determine if the intent was appropriate for the turn or if a new intent should be created.
A PNI is voted to belong to the suggested intent	Fix the mapping to allow the turn to reach the suggested intent.

Table 5.1: Voting outcomes and recommended actions

or determine if the intent description in the voting interface needs clarification, or if an entirely new intent should be created to address a new language topic.

As a further example, suppose a turn is classified as intent A in the IVA, and the CRS generated PNI (turn, B) which was voted on to be the correct intent as shown in Figure 5.3. In this scenario, the CRS will recommended that the domain expert investigate the language model to determine the cause of intent A being selected over intent B. These recommendations are given to speed the original refinement process in Figure 2.3 where domain experts must look over all of the graded chats and determine for themselves what action, if any, is needed in each case.

The set of turns, associated voting outcomes, and recommended actions are first grouped by responding intent and then ordered by the evidence of error per intent. The evidence of error may be the percentage of risky turns assigned to the intent out of the total number of risky turns present in the conversation logs. They can be further normalized by the frequency of response within the conversation logs to prioritize work based on the impact it will have on the live IVA. A screen shot of this prioritization from the analysis interface in the CRS is shown in Figure 5.4. By

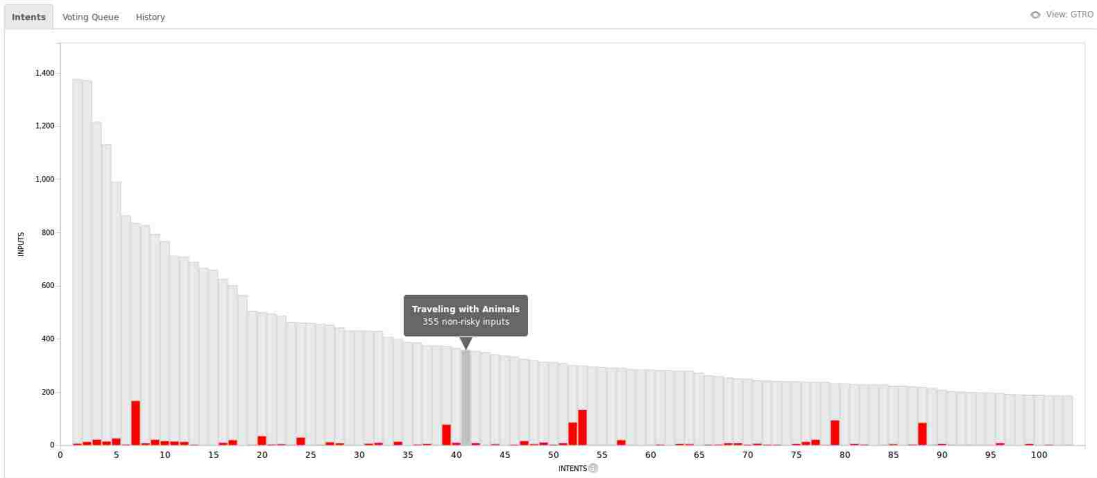


Figure 5.4: Prioritizing language model repair work by graphing the ratio of risky to non-risky inputs per intent in the live IVA. The red bar is the count of risky inputs assigned to that intent. The grey bar is the count of non-risky inputs.

looking at this chart, domain experts can quickly determine which intents have a greater impact on user experience by their popularity. If two intents have a similar ratio of risky to non-risky inputs, the intent with the higher response frequency would be prioritized for repair as its malfunction will have a larger impact on overall user experience.

These results presented to the domain experts are referred to as the *voted order* in Figure 5.1. By this prioritized ordering, the domain experts can focus their attention to the intents with the greatest need of correction first.

In addition, through recommended actions provided by the CRS, the domain experts can make more informed decision than in the existing process of Figure 2.3. Hiding the turns with correct mappings (see Table 5.1, row 1) means the domain experts have less data to look through which saves valuable time.

Figure 5.5 shows a screen shot from another analysis view within the CRS. This view allows the experts to quickly analyze the voting results and voter consensus. The filters at the top provide the ability to explore the results from many angles such as

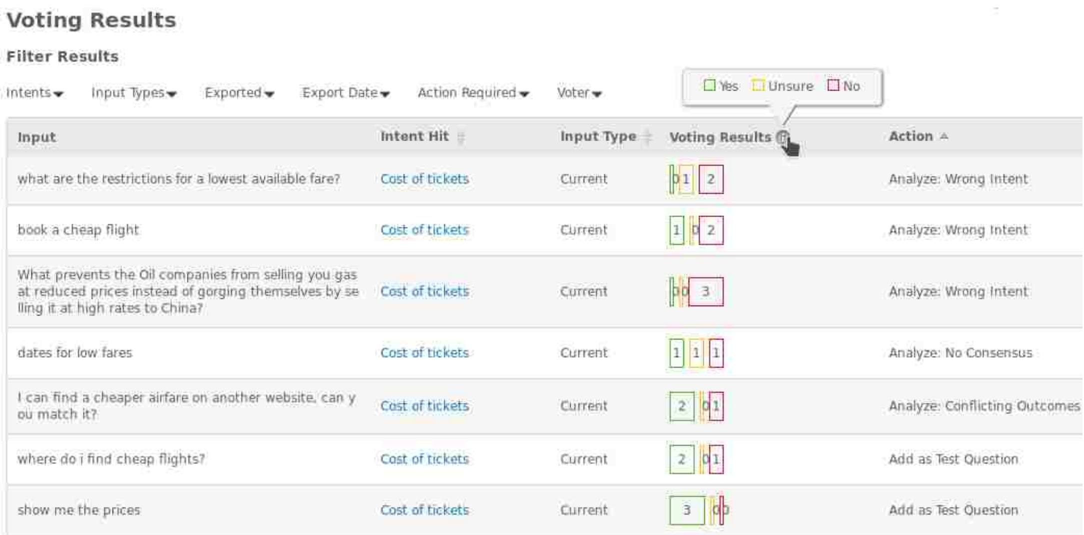


Figure 5.5: Analysis interface within the CRS

per intent, per voter, date range, recommended action, etc. In the left hand column the original user turn text is displayed. In the next column is the intent that the reviewers evaluated the text against. The "Input Type" column shows whether the intent evaluated was from the live IVA or a PNI from within the CRS. The "Voting Results" column provides a visual indicator of the voting outcome and inter-reviewer agreement. The final column on the right hand side is the CRS recommended action from table 5.1. Notice that in the third row from the bottom the Action column reads "Analyze: Conflicting Outcome". This category means that voters agreed with both the original intent *and* an intent suggested by a classifier (a PNI). Filtering this table by that action type will quickly surface all examples where such a conflict occurred.

From this view the domain experts can quickly find areas of the language model that need attention and export the text data with risk indicators and voting results to use in down stream processing tasks needed to make the necessary changes in the language model.

5.2 Automating the Existing Refinement Cycle

The CRS can be positioned to reduce or remove the dependence on the human reviewers in the refinement process. As conversations are reviewed, the weights for the different risk indicators are continually tuned to reflect their correlation to vote outcome. In addition to adjusting the risk indicator weights, the CRS continuously trains a voting classifier model using the output of the risk indicators as a feature vector. This model predicts the human majority vote for a given user turn.

For example, if voters agreed that turn t belongs to the intent assigned by the IVA, the outcome is 1. If they disagree, the outcome is 0. Then for each turn with a voter consensus we can add a row to a feature matrix M, with a column for each risk indicator and a final column for the outcome.

$$M = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_n \end{array} \begin{array}{c} \begin{array}{ccccc} \textit{backstory} & \textit{multi\_intent} & \textit{restated} & & \textit{outcome} \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & \cdots & 1 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \end{array}$$

This feature matrix M is then passed to a classifier training function to produce a binary classification model. When a new turn is under review, the risk indicators present are represented as a feature vector and fed to the voting classifier to predict the majority vote of **Yes** or **No**. As humans continue to vote, M is increased and the voting model is retrained. In Section 11.5, several classification methods are compared to human performance in predicting the majority vote.

Using this prediction model, the CRS can vote on each turn-intent pair alongside or in place of the human reviewers as shown in figure 5.6. In this configuration the domain experts are given votes and recommended actions as in figure 5.1, but the votes now originate from the CRS instead of human reviewers. The human reviewers can at any time provide reinforcement by voting, which adds training samples to M,

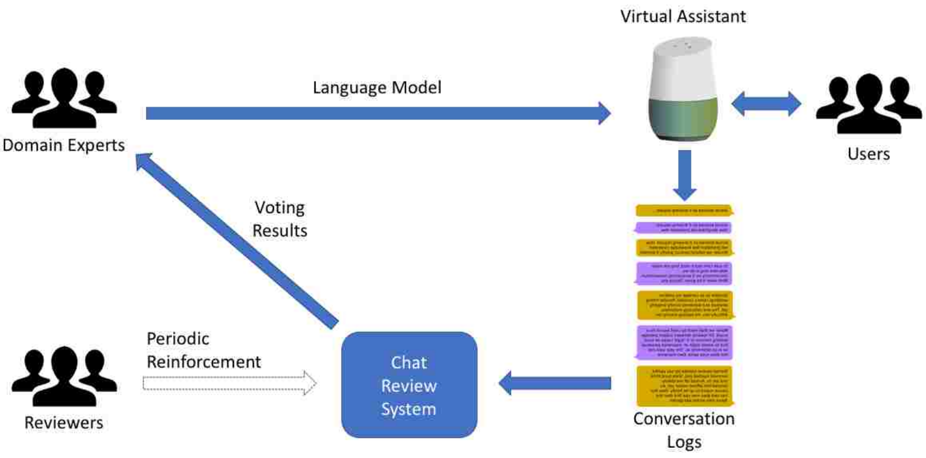


Figure 5.6: Automated Language Model Refinement Cycle

to ensure the CRS is continuing to preform optimally. Nothing in the design of the CRS will prevent the refinement process from being reverted to figure 5.1 if the system is performing sub-optimally. As the CRS will always vote on all of the conversations as if it were a human voter, the domain experts can choose to accept its vote or to release a subset of the data to human reviewers for any intent.

5.2.1 Active Learning

Recall that the continuous training of classifiers within the CRS is performed using the human votes as the labels for training samples. Also recall from Section 5.1.1 that the human reviewers are presented with turns for review ranked by their risk score. This risk score is the weighted combination of individual risk indicators, and the weights are tuned by the human voting. Therefore, the learner itself has a role in selecting its own training data from a pool of unlabeled samples. This iterative supervised learning cycle where the learner selects its own training data is known as

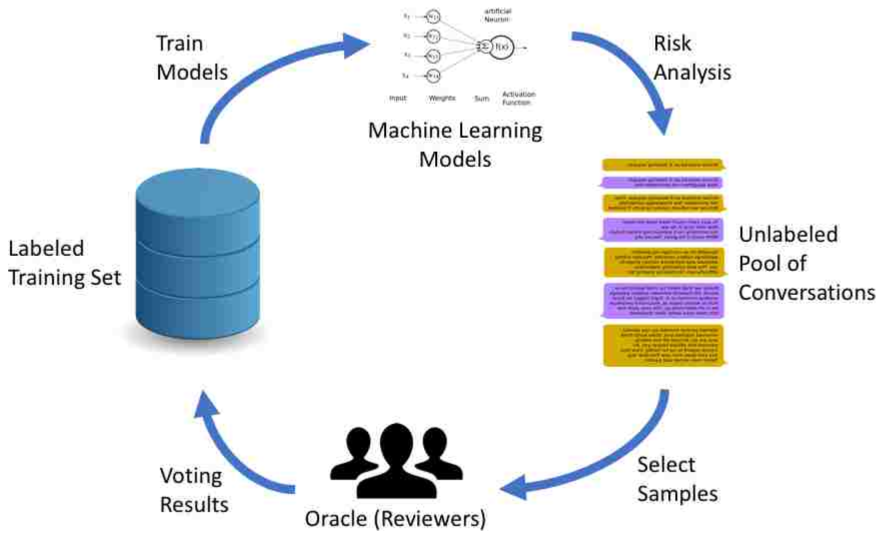


Figure 5.7: The pool-based active learning cycle

active learning.

The active learning hypothesis states that a system will perform better with less training if it is allowed to select the data it learns from [116]. In this framework the *learner* requests labels for selected instances it chooses based on some query strategy. Labels are given to the selected instances by an *oracle*, or group of human reviewers in the case of the CRS, which are simply added back into the training data for the learner. The learner is retrained using the additional samples completing the active learning cycle. This cycle continues as the learner is always free to choose additional training samples from the unlabeled set, if needed.

There are several sampling strategies for active learning, but as the set of conversations to be reviewed is static at the time of risk analysis, the CRS employs a pool-based sampling strategy. In pool-based sampling the entire set of unlabeled instances are first evaluated and ranked, and the set of training samples are then selected based on the ranking [117]. In the case of the CRS, the per instance evaluation is the determination of risk indicators per turn in combination with the output of the comparative and voting classifiers.

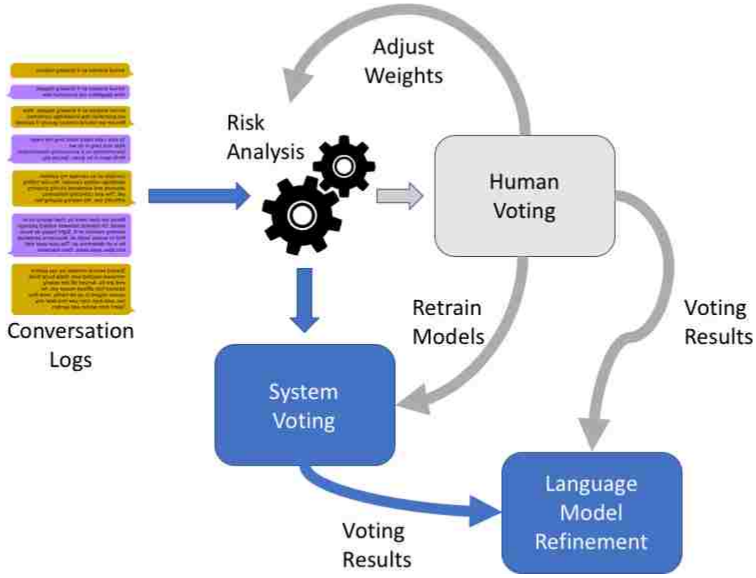


Figure 5.8: Flow of human-reviewed turns

Once turns are selected for human review and voted on, the results are used by the system in two primary ways, as shown in figure 5.8. First, turns with a majority vote are used to adjust the weight of individual risk indicators for the risk evaluation phase. Initially, risk indicators all have equal weight. High risk turns that were voted as not misunderstood will reduce weights for the present risk indicators. High risk turns that were, in fact, misunderstood increase the weight of the risk indicators present in that turn. Over time, these adjustments expose the indicators with the highest correlation to misunderstood turns. The implementation and results of this iterative weighting are discussed later in Chapter 11.

Secondly, turns with a majority vote are added as training samples for the classifier used for system voting previously described in Section 5.2. The updated classifier will then be used to provide voting data and recommended actions on the remaining inputs not voted on by human reviewers, as well as be used in the selection of turns to be voted on in combination with the risk score.

It is important to remember that the CRS presents the voting results and recommendations to the domain experts in the same format and interface regardless if the voter was human or machine. Therefore, the domain experts can continuously choose the source of their analysis data based on current system performance in a particular language domain or human reviewer availability.

We have now completed our discussion of how the CRS is applied to the conversation review cycle, and the different ways it provides insight to the domain experts. In the next chapter we cover the implementation details of the CRS and the motivations behind its architecture decisions.

Chapter 6

Implementation

In this chapter, we discuss the system architecture of the Chat Review System (CRS) and the motivations behind the design decisions. The CRS is designed as a web application for several reasons. The first is that it deals with large volumes of data that is difficult to store on a single desktop. The second is that the risk analysis component is an extensible collection of algorithms which, if designed as a native desktop application, would require substantial processing power. Web applications are particularly well suited to transparently offload distributed processing and manage large volumes of simultaneous users [118]. A third and important factor was that, as a web application, voting can be performed by anyone with an Internet connection. If domain experts wish to leverage a crowd source platform such as Amazon Mechanical Turk[1] or Crowdflower[2] to hire reviewers, they can simply provide remote workers with a URL to the application and allow them access to the specific data to be reviewed. This keeps the data within the CRS and prevents unauthorized access through local storage on devices outside of the control of the domain experts.

---

[1] `https://www.mturk.com`
[2] `https://www.crowdflower.com`

6.1 Components of the CRS

Given the web application architecture, the CRS is designed scalable system comprised of four primary components. The first is the web sever which serves a standard HTML5/CSS3/JavaScript user interface. The interface is divided into two primary views, one for domain experts to select data for review and to analyze the voting results, and the other for reviewers to vote on selected turns. This application interface is implemented using the Django[3] web framework.

The second component is where data intensive, periodic, or long running tasks are executed. This could be tasks such as generating a spreadsheet export of voting results, recalculating voter performance metrics, or optimizing the risk indicator weights. These tasks are managed by a Celery[4] process pool which decouples it from the web service layer so that it does not introduce latency due to resource contention.

The bulk of the data processing in the risk analysis process is submitted to a Grid Engine[5] High Performance Computing (HPC) cluster. Similar HPC engines such as SLURM[6], HTCondor[7], or TORQUE[8] can be used interchangeably for this component. As conversation logs are imported into the CRS from the live IVA, they are partitioned into jobs across the HPC cluster which annotates them with any present risk indicators and performs various classification tasks such as applying the comparative classifiers and voting classifier. HPC clusters are designed to scale horizontally and provide a central interface to run arbitrary, mixed-platform jobs across entire organizations or companies[119]. For this reason they are a good choice to deploy applications that require a mixed workload as their scheduling engine can make the best use of the currently available resources in the cluster to ensure that

[3] `https://www.djangoproject.com`
[4] `http://docs.celeryq.org`
[5] `http://www.univa.com`
[6] `https://slurm.schedmd.com/`
[7] `http://htcondor.org`
[8] `http://www.adaptivecomputing.com/products/open-source/torque`

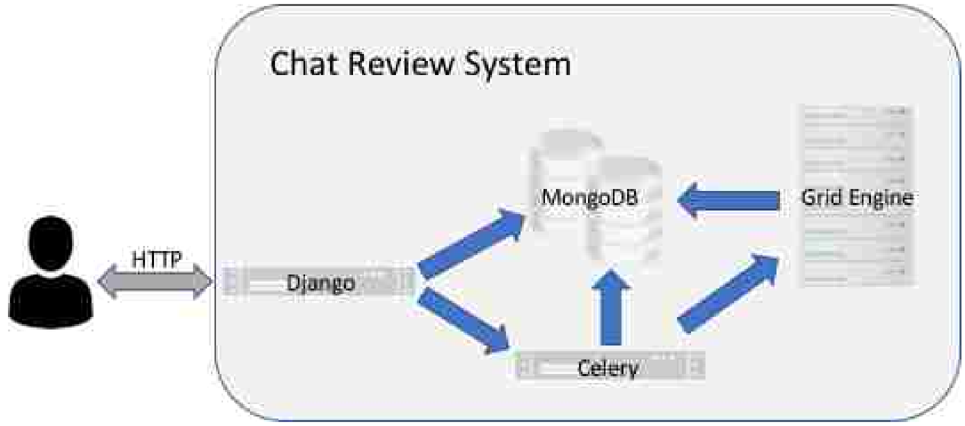


Figure 6.1: The four primary components of the CRS

the jobs have the memory and processors needed to complete their work.

The final component in the CRS is the central storage engine. As it is pooling a potentially large number of conversation logs while also meeting the read and write demands of a HPC cluster, the storage engine choice was perhaps the most crucial of the four components. For our implementation of the CRS we evaluated several databases before finally settling on MongoDB[9]. This was not only for its ability to handle our preliminary performance tests well, but it enjoys a large support community due to its popularity[10].

All of the components interact with the central MongoDB cluster as needed. MongoDB has a built in MapReduce [120] engine that is leveraged in addition to the HPC cluster for distributed data processing. All four of the major components are designed for horizontal scaling and can be scaled independently of the others where needed. This flexible architecture allows the CRS to handle a large volume and velocity of conversations for risk analysis.

[9] https://www.mongodb.com

[10] MongoDB was ranked #1 in NoSQL database engines and the #5 database overall on https://db-engines.com/en/ranking at the time of this writing.

6.2 Scalability

As previously mentioned, a major consideration in the CRS design was scaling to the volume and velocity of generated conversations. The CRS is designed as a shared platform to concurrently analyze multiple IVAs and their associated conversation logs. Aggregating this continuously growing log data requires its storage and queries to scale horizontally. A single IVA may generate tens of thousands of conversations a day in a large corporation [121, 122]. MongoDB was chosen as the storage engine due to its MapReduce framework and horizontal scaling [123]. MongoDB has also been shown to handle aggregating large volume chat data and subsequent querying well [124, 125].

The analysis of every risk indicator is embarrassingly parallel due to the independence of every conversation. Therefore, every analysis process can be written as an independent job for a compute cluster. The conversation logs can then be partitioned across the cluster, and every job can be run in parallel on a subset of conversations. This allows the analysis stage to easily scale, and every task can write its results directly into MongoDB as the data store can also scale horizontally to the required write load. This saves costly post-processing data migrations.

For every iteration of the active learning cycle, voting outcomes are fed back into the CRS to recalculate the risk indicator weights and retrain the classifiers described in Section 5.2. The classifiers included as indicators of risk from Section 4.4.3 are trained on all turn-response pairs that have been voted to be correct and use turn features to predict an intent. The voting classifier described in Section 5.2 is trained on the feature vector of risk indicator outputs and predicts if a turn-response pair is correct or not.

This continual retraining is crucial because the frequency and importance of risk indicators will change as the system and IVAs under review mature. Training classifier models off of millions of inputs can be very time-consuming and resource inten-

Purpose	Virtual	Count	CPU	Memory	Disk
Web server	Yes	1	8 cores of E5-2680v2 @ 2.80GHz	64GB	50GB SSD
App server	Yes	1	8 cores of E5-2680v2 @ 2.80GHz	64GB	500GB SSD
MongoDB (Primary)	Yes	1	8 cores of E5-2680v2 @ 2.80GHz	64GB	200GB SSD
MongoDB (Secondary)	Yes	1	8 cores of E5-2680v2 @ 2.80GHz	32GB	200GB SSD
HPC Node	No	6	40 cores of E5-2680v2 @ 2.80GHz	384GB	300GB 15K SAS

Table 6.1: Scaling test hardware configuration

sive. Therefore, these tasks can be done as batch jobs that occur during periods of low system activity.

6.2.1 Scaling Tests

The ability for the system architecture to scale was tested in two ways: the time complexity of the risk analysis process and the reduction in risk analysis time for a fixed input size by increasing the cluster size. Time complexity estimates the ability for a fixed system to handle increasing conversation log sizes while increasing the cluster size measures the system's ability to scale horizontally. For both tests the same configuration of the CRS was used. The physical configuration of the environment was a combination of Virtual Machines running in a VMWare 5.5 cluster, and physical servers for the HPC cluster. The specifications of each server is given in Table 6.1. The software configuration of the primary components in the CRS for all of the tests conducted is given in Table 6.2.

Purpose	Software	Version
Operating System	Ubuntu Server	16.04LTS
Web server	Nginx	1.4.6-1ubuntu3.8
App server	uWSGI	1.9.17.1-5build5
Worker Pool	Celery	4.1.0
Database	MongoDB	3.4.10
HPC Engine	Open Grid Engine	GE2011.11p1

Table 6.2: Scaling test software configuration

Name	# of Turns	# of Conversations	Avg User Turns per Conversation
50k	49,999	10,336	4.837
100k	100,008	20,575	4.860
200k	199,998	41,770	4.788
400k	400,001	84,917	4.711
800k	800,003	170,880	4.682
1.6M	1,600,008	346,408	4.619

Table 6.3: Scaling test data set statistics

Measuring Time Complexity

To estimate the time complexity of the entire risk analysis process, we measure how the analysis time increases as conversation log size increases. The test data originated from one month of conversation logs from a live IVA in the telecommunications domain. This particular IVA handles customer support issues on the website of a large telecommunications corporation. The logs were partitioned into increasing subsets of size 50,000, 100,000, 200,000, 400,000, 800,000, and 1,600,000 turn-response pairs. Full test data statistics are given in Table 6.3. Beginning with the smallest, each set was fed into the CRS and the total wall clock time was measured to complete the risk analysis and apply the voting classifier to all turns. Then the entire system was restarted to clear out any caches. After restarting, the next largest set was fed into the system and the wall clock time was recorded.

In Figure 6.2, we see that it takes roughly 48 minutes to complete analysis on

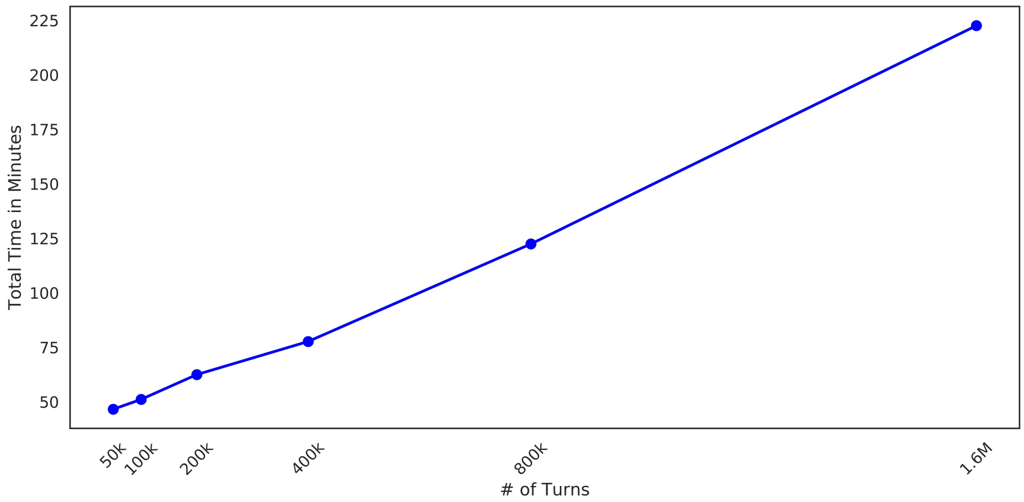


Figure 6.2: Scaling test results on increasing conversation log sizes

50, 000 turn-response pairs. This volume of data is approximately a single days worth of customer interactions for this particular IVA. By 1.6 million turn-response pairs, the CRS takes roughly 225 minutes, or 3.75 hours to complete its analysis. This volume is exactly one month worth of customer interactions from the live telecommunications IVA.

In Figure 6.3, we look at the per-turn processing time as the conversation logs grow. The 50k dataset averages around 0.055 second per turn, which improves to 0.01 second per turn by the 1.6M dataset. The cause of this decrease is that the total wall clock analysis time includes the time required to extract features for and train the agreement classifiers before applying them to the conversations in the risk analysis process. For smaller data sets, this training time has a large impact on total analysis time. But, as the size of the data set grows, this training time is a smaller component of the total analysis time and therefore the average time per turn decreases.

Assuming the agreement classifiers were retrained once per day and the same hardware configuration as the testing environment is used, a single instance of the

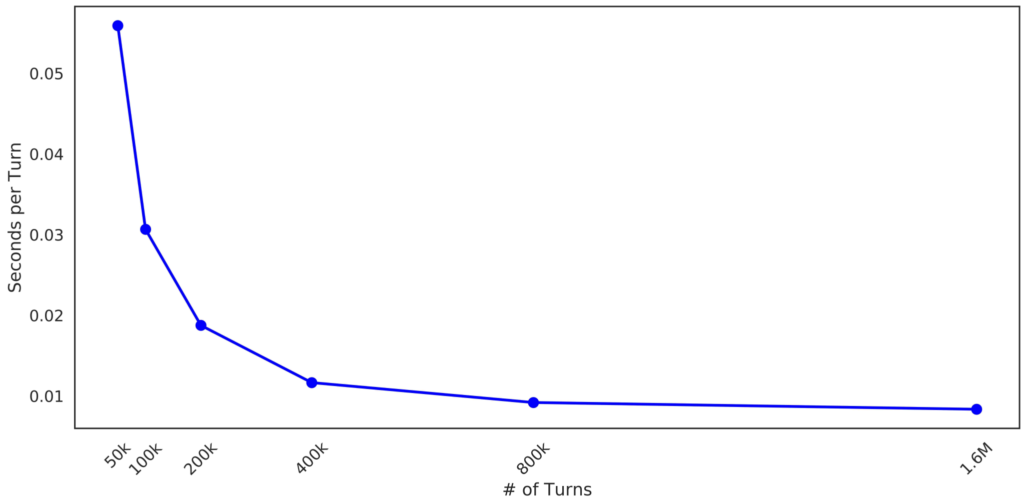


Figure 6.3: Average analysis time per turn

CRS would appear capable of handling the real-time processing demands of around 30 IVAs with similar volumes of conversation logs (1,440 minutes per day ÷ 48 minutes to process 24 hours worth of logs). If the agreement classifiers were retrained less often the average time per turn would decrease, which would allow a single instance of the CRS to handle more concurrent IVA logs. However, longer times between retraining may have an effect on agreement classifier accuracy if the IVA language model is changing rapidly.

A comparison of the time per turn processing rate to standard growth rates is shown in Figure 6.4. Big Oh notation is commonly used to characterize algorithms according to how their running time grows as their input size grows, ignoring the multiplicative constants [126]. From the comparison we can see that the risk analysis process is running much faster than linear time ($O(n)$) for the measured input sizes. The time complexity behavior appears to mirror $O(\sqrt{n})$ most closely. If this behavior were to continue, assuming no memory or disk limitations were reached, the hardware configuration of the CRS used for these tests could process over 8.6 million turn-response pairs in a 24 hour period.

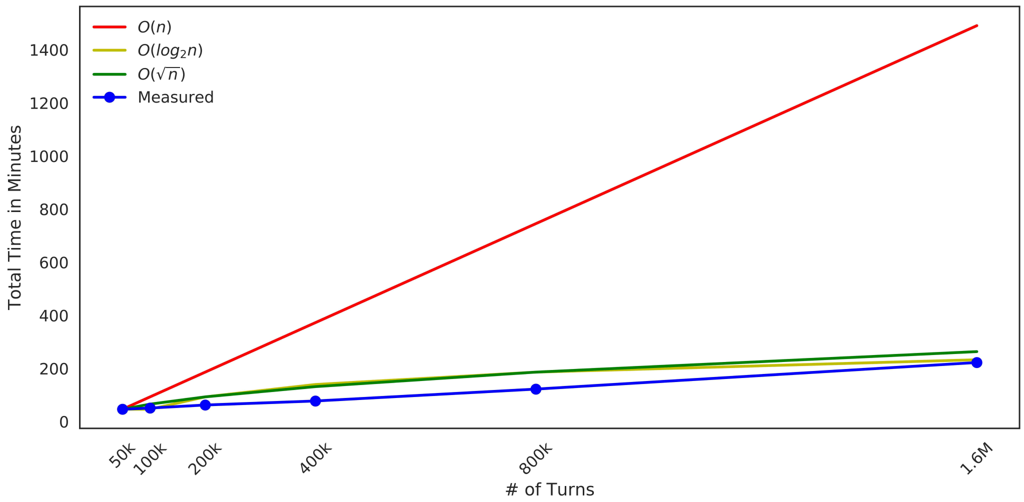


Figure 6.4: Scaling test results compared to standard measures of growth

Measuring Horizontal Scaling

To measure the architectures ability to scale out given a fixed input size, we measure the effect of adding nodes to the HPC cluster. During the risk analysis phase, the majority of the processing is done as distributed jobs on the HPC cluster (training the agreement classifiers, identifying the indicators of risk in Section 4.4, voting on each tun, et cetera). Therefore, this component will have the largest effect on the ability for the architecture to scale.

In this test we start with a single HPC compute node enabled. We feed the 400k dataset through and measure the total wall clock time to complete the risk analysis and apply the voting classifier to all turns. Then the entire system was restarted to clear out any caches. After restarting, and additional HPC compute node is added to the cluster and the test is repeated. The compute nodes are homogeneous and their specifications are given in Table 6.1.

In Figure 6.5, we see the results of this scaling test. With a single compute node, the risk analysis process takes 320 minutes, or roughly 5.5 hours to complete.

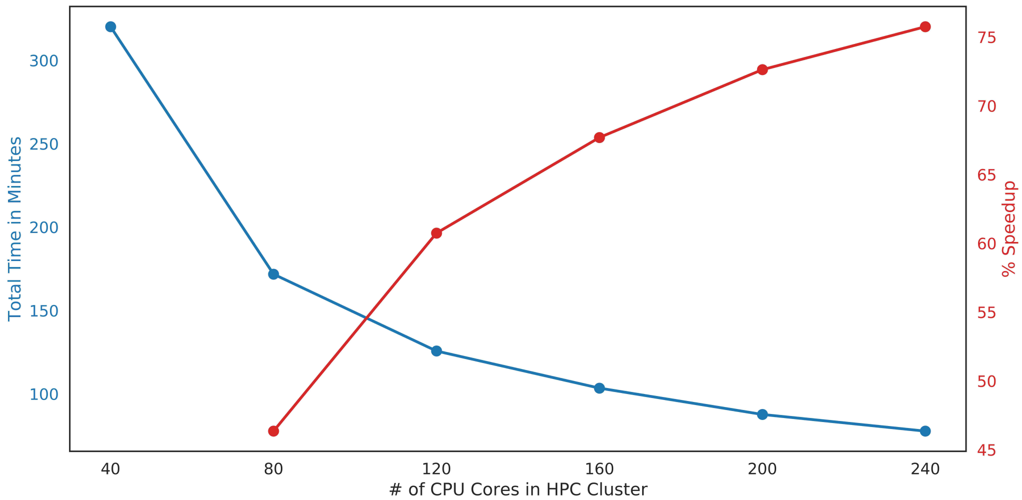


Figure 6.5: The performance impact of increasing the HPC cluster size

Recall that the 400k dataset was used, which contains roughly 8 days worth of conversations from the live IVA. Processing 8 days worth of logs at one time only trains the classifiers once, so the training time has less of an impact on average processing time per turn. If the classifiers were only retrained once in the 8 day time period, even with a single HPC node the CRS could easily maintain real-time performance on 30+ similar volume IVAs concurrently (8 days of data ÷ 5.5 hours to process).

By adding a second node, the processing time drops 46% to 172 minutes or 2.9 hours. Doubling the CPU core count in the cluster from 40 to 80 creates a 46% speed increase. Doubling from 80 to 160 sees a 22% increase, while doubling from 120 to 240 CPU core count adds a 15% increase. As further nodes are added the speedup per node diminishes, most likely due to the fact that the increased load on the database servers and network hinders the clusters ability to maintain the per turn processing times of smaller cluster sizes.

Figure 6.5 clearly shows that performance increases as much as 76% are possible by simply adding hardware to the system. Further gains may be possible by scaling out the MongoDB cluster and improving the network capacity between the systems. As each component can scale horizontally as needed and $O(\sqrt{n})$ growth behavior is observed, the architecture proves capable of handling the data volume demands of commercial deployment.

Now that we have covered the architecture and scaling performance of the CRS, in the following chapters we turn our attention to the implementation and performance of detection methods for the individual indicators of risk.

Chapter 7

Escalation: Detection and Prediction

In this chapter we discuss the detection, categorization, and prediction of *escalation requests*. We define an escalation request as a dialog act where the user explicitly asks to resolve their issue with a different party than they are currently conversing with, in our case the IVA. These dialog acts are very important indicators of risk as they can surface conversations where the user is frustrated with the performance of the IVA.

Recall from Section 4.4 there are two categories of risk indicators: conversation level and turn level. One turn level risk indicator is **precedes_esc**. This indicator is assigned to turns that are immediately followed by an escalation request. In order to determine the presence of this indicator, explicit escalations requests must first be detected. In the following sections we develop an algorithm that is effective in detecting user escalation language and context. Furthermore, we demonstrate the algorithm's effectiveness using real world chat data. This work originally appeared in [114].

7.1 Related Work

Strategies for deciding if or when to escalate exist but typically involve telephone conversations and depend on either the duration of the call or a tree-based dialog structure. Lemon and Pietquin use a classifier to determine if a customer's issue is resolved [127], and a fixed point is chosen at some time in the call for this determination. The duration of the interaction after this point as well as the cost of call parameters are vital in arbitrating whether a human agent is needed. Similar techniques can be found in [128, 129, 130] where classifiers are used to determine if a call is problematic and when an escalation is necessary. Another technique, [127, 131], considers conversations in a tree-based dialog structure. Nodes that have low automation potential have a higher chance of escalation. The call flow is pruned by identifying nodes with low automation potential and eliminating any further flow path following such nodes.

Since our system does not use a dialog tree structure, the latter approaches are not directly applicable. However, a similar feature such as *conversation paths*, which is the sequence of responding intents in a conversation, was examined for its possible relationship with escalation frequency. Conversation paths were extracted from the evaluation dataset described in the following section, and an 80-20 split was done to create training and testing sets. A Naive Bayes classifier from TextBlob, a Python library for processing textual data [132], was trained on the conversation paths. Accuracy achieved on positive escalation turns was extremely low (29%) on test data.

Former methods use conversation duration as a feature; so, we consider duration's relationship to observed escalation frequency in live chat data. However, with a Pearson correlation coefficient of .269 (calculated using the Scipy stats module also on the evaluation dataset described in the following section), duration alone is a poor predictor of escalation. In addition, duration versus escalation frequency does not take into account confounding factors such as user typing speed or multi-tasking. Alternative to duration, one might consider the number of turns in the conversation before escalation. For this, we obtained a Pearson correlation coefficient of .26. Such turn based strategies also do not appear fruitful in detecting escalations.

Although related, these existing methods aim to detect if or when to escalate which is *different* from our task. Our task is not to predict escalation but to detect explicit escalation requests made by the user. More specifically, we wish to detect when the user has *explicitly* requested an escalation *after* first attempting to use the agent. No existing literature could be found on this specific topic. It is clear that in the context of our live chat system, which is turn and not time dependent and also has no tree structure, such traditional escalation strategies are not applicable. Nevertheless, a novel method for detecting a user's request for escalation is necessary since the discovery of conversations where the agent has failed will lead to improvements in the language model. Automatic collection of this data will also generate corpora for investigating more generic methods for proactive escalation than those reviewed.

7.2 Experiment Setup

For the purposes of our live chat system, we consider escalation requests to be in one of three classes. The first class (I) consists of those users who immediately request to speak to a different party; they do not want to use the automated system. For example:

```
Agent: Hello, how can I help you today?
User: can you transfer me to reservations?
```

The second class (II) of escalations occur after the virtual agent directs the user to contact a different department or service, and, in response, either the user asks to be transferred there or requests contact information. An example of this would be:

```
Agent: To change the name on your ticket, please
       call reservations.
User: can you transfer me to reservations?
```

The final class (III) are those conversations where a virtual agent attempted to resolve an issue for the user, failed to do so, and the user requested an alternative party. For LM improvement, we are not interested in classes I or II since they do not reflect a failure in the conversational ability of the virtual agent. Within our virtual agents, there exists intents to capture various escalation requests and transfer users. However, there are shortcomings in relying only on these intents to surface conversations for review. In our analysis, we observed that escalation requests occur in as many as 16% of conversations. However, the vast majority of them fall into the first two classes, meaning there is no error in the LM to be investigated. Also, we are trying to discover error in the LM itself, so we do not want to rely on it to provide us with instances where it failed. Instead, we develop a stand-alone means to detect only Class III requests for human review and data collection of failed conversations.

We have deployed virtual agents in various domains, but a significant number of them reside in travel-related industries. Therefore, we experiment in this domain as we have access to a large amount of similar data across numerous deployments.

For training, we constructed a set of $15,338$ turns containing $1,703$ manually tagged escalation requests. Due to the expense of humans tagging nearly 45K total turns for train, test, and evaluation, the data was divided equally among the reviewers and combined instead of a more robust multiple review and kappa or majority calculated. The turns were selected from chat histories of US and non-US transportation customer service agents to form a general travel domain corpus. To train a robust binary classifier, we chose a large number of positive examples as escalation language is very broad. The training data contains only the user input and its binary escalation value. We make no distinction in the escalation class for training as the user language is generally the same; it is the context they occur in that differentiates the classes. We develop an effective algorithm to determine the context given a binary classifier that can detect the language.

For development and evaluation, we use raw live chat data collected over a 24-hour period from an airline virtual agent that retains conversation features such as time stamps, conversation ID, responding intent, and response text. These were tagged by the same process, but only Class III escalations were labeled since the ability to discover this type of escalation is what we measure. A small dev-test set was reserved for sanity testing and optimization of vectorization and classifier parameters. It consists of 226 conversations containing 920 turns, of which 29 were Class III escalation requests. The evaluation set is $7,967$ conversations containing $28,336$ turns, of which 333 are Class III escalations. Although only 1-3% of turns in live chats contain a Class III escalation, our virtual agents currently see around 500K inputs daily, and this number is increasing steadily. Thus, we estimate $5,000$ to $15,000$ conversations contain Class III escalations, not an insignificant amount and worthy of investigation.

7.3 Training an Escalation Classifier

We originally theorized that there were enough differences in the language used for each of the three escalation classes that a multi-class classification approach would be effective. Experimentation revealed that the confusion between the classes was so high that a multi-class approach would not work. Many cases exist where the user inputs are identical but belong in different classes making it impossible for a turn based classifier alone to succeed. An example of this was seen in the previous section, where both user inputs were *"can you transfer me to reservations?"* but did not belong to the same class. Thus, we turn our efforts towards distinguishing the class by conversational context given a binary classifier that can determine the presence of an escalation.

		Class III Escalation Turns		Non-Escalation Turns	
	Method	**Precision**	**Recall**	**Precision**	**Recall**
0	Baseline Regular Expression	0.29	0.23	0.99	0.99
1	Hashing Vectorizer	0.4	0.83	1.0	0.99
2	TF Vectorizer	0.42	0.96	1.0	0.98
3	TF-IDF Vectorizer	0.38	0.98	1.0	0.98
4	POS Tags + TF-IDF Vectorizer	0.4	0.99	1.0	0.98
5	Filter Class I: single turns	0.54	0.99	1.0	0.98
6	Filter Class I: 1..n positives	0.73	0.99	1.0	0.99
7	Filter Class I: initial greetings	0.74	0.99	1.0	0.99
8	Filter Class II	0.9	0.99	1.0	1.0

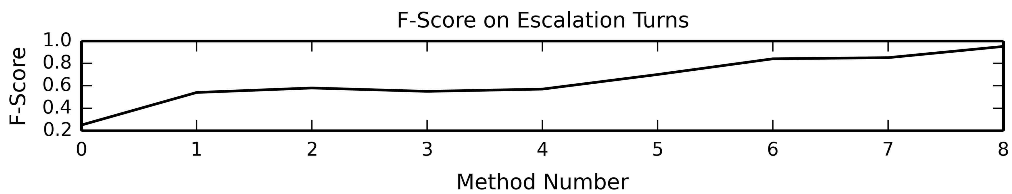


Table 7.1: Algorithm performance in discovery of Class III escalations in live chats

Our first objective is to create a binary classifier that can accurately detect escalation language. As our problem involves text classification, we use a Support Vector Classifier (SVC) with a linear kernel which has been shown to perform well at this task [133, 134, 135]. After every change to the vectorizer and classifier pipeline, we optimize the parameters against the dev-test set and measure performance against the evaluation set. The results of these iterations are shown in Table 7.3, Methods 1 through 4. Our utility function for optimization maximizes recall on escalation requests. To collect conversations for LM improvement, we err on the side of false positives rather than false negatives. As the evaluation set only tags Class III escalations, we expect the precision to be very low initially as the classifier will consider all three classes of escalations positive. In the following sections, we incorporate this classifier into an algorithm to separate the positive matches into the three classes.

7.3.1 Baseline for Experiments

As there exists no other escalation detection systems to compare our results against, we create a reasonable first order system to compare to. We take the top 25 most frequent words in the positive training turns and create a Regular Expression (RE) to capture common phrases they appear in. We then run this RE against the evaluation set (Table 7.3, Method 0) as the baseline to compare our system against. Increased accuracy is possible by combining multiple REs from in-depth analysis, but due to the time consuming nature of constructing manual models, we present the following RE as a reasonable baseline.

```
^.+(t(alk|ransfer)|phone|c(all|ontact)|speak|
need).+(customer(\s+)?(s(ervice|upport))|(a\s+)?
rep(resentative)?|someone\s?else|(a|the|your)\s+
manager|a\s+((live|real)\s+)?(human(\s?being)?|
person)|reservations|number).+
```

For our classifier, we initially use SciKit-Learn's HashingVectorizer and LinearSVC to construct a pipeline [136]. The SVC is trained with the default parameters, and we use a grid search against the dev-test set to find optimal parameter values. These parameters and their function are described in [137] and disscussed at length in [138]. The optimal parameters found were *C=2.1, penalty=l2, loss=squared_hinge* (Table 7.3, Method 1).

7.3.2 Methods for Classifier Improvement

This classifier only produces 83% recall, so we focus on feature improvement (Table 7.3, Methods 2-4). A basic first step is to incorporate raw term frequency for term weighting and N-grams to include context. Changing to SciKit-Lean's TfidfVectorizer, we optimize both the SVC parameters and the lower and upper range boundary of n-words. The additional optimal parameters were *C=1.7, ngram_range=(1,3)* (Table 7.3, Method 2).

To further isolate language specific to escalations, we turn on Inverse Document Frequency (IDF) to scale down the impact of words that are less informative [139, 140]. The additional optimal parameters were $C = 2.5$, $ngram\_range = (1, 4)$ (Table 7.3, Method 3).

Including Part Of Speech (POS) tags can benefit SVC for text categorization [141]. So, we add POS tagging as a preprocessing step. Using the Natural Language ToolKit (NLTK) PerceptronTagger [142], we add the POS tag to each word in the input text and discard all punctuation before feeding the text into the pipeline (Table 7.3, Method 4).

7.3.3 Rejected Methods

We also considered the following methods and determined that they did not significantly increase performance over Method 4.

Automatic Spelling Correction

Using the TextBlob *correct* method [132], which replaces words with the highest probability of known words with edit distance 1-2 from the given word, we observed a minor increase in false negatives and a 0.5% decrease in false positives. The gains were not high enough to justify the additional complexity, but it remains an interesting possibility. There are much better means of automatic spelling correction that take into account context [143, 144] and applying them is subject to future research.

Stemming

Using the NLTK English SnowballStemmer [145], we stem the words after generating the POS tag and replace the original word with its stem. In agreement with the conclusions of [146], stemming introduced ambiguity in the SVC that led to a loss of performance. We observed a negligible increase in false negatives and a 10% increase in false positives.

Stopword Filtering

Using the NLTK English stopword list, we filter all stopwords before the POS tagging. Also in agreement with [146], we found that the removal of stopwords was unnecessary for SVC accuracy. Stopword removal resulted in a minor increase in false negatives and a 12% increase in false positives.

Sentiment Analysis

Using the NLTK NaiveBayesAnalyzer [147], which is trained on the Movie Reviews corpora, we add turn sentiment polarity and subjectivity as two additional feature columns to the feature vector produced by TfidfVectorizer. By adding them to the feature vector, we allow the SVC to discover if any correlation is present. This resulted in no change to false negatives and a negligible increase in false positives. As there exists better methods for sentiment classification ([148], for example) and these scores can be integrated in other ways, we will revisit sentiment analysis in future work. We did observe, however, that despite users feeling frustrated while the agent is misunderstanding them, many of them still politely request an escalation. This makes it difficult to rely on sentiment polarity as a feature of an escalation request.

7.4 Algorithm for Escalation Categorization

Using the escalation classifier in Table 7.3, Method 4, we now focus on precision by designing an algorithm that can separate the positive matches into the three classes (Methods 5-8). We begin with filtering Class I escalations, where the user has no intention of using the agent; they simply request contact information or immediate transfer.

If the initial turn is an escalation, three directions can be taken on the next turn: the conversation ends, the user is not satisfied with the response and rewords his or her question or asks for additional clarification, or the user decides to attempt to resolve their issue with the agent and states their concern. To eliminate the first case, we ignore all conversations with only a single turn. This results in a 43.2% drop in false positives as seen in (Table 7.3, Method 5).

We handle the second case by feeding the remaining turns through the SVC. We ignore every sequential positive match until either a negative match occurs or the conversation ends. We can then remove all conversations containing only escalation language, and if the user decides to use the agent after all, we consider any escalation occurring after the user identifies their issue as a Class III. This results in a further 32.4% drop in false positives (Table 7.3, Method 6).

In reviewing the evaluation data, we noticed conversations where the first turn consisted solely of a greeting (*"hows it going?", "Good morning."*). These add noise to the above mentioned Class I detection and should be ignored. To remove them, a separate binary SVC is trained on a dataset created for this purpose. As all of our virtual agents have a *Hello* intent for responding to greeting language, we gathered 226 unique user inputs from across several US domestic and international deployments that were assigned to this intent. International and US domestic greetings are mixed to make the classifier more robust. We then added 59,960 turns assigned to other intents from the various deployments as negative examples. This classifier achieved 0.986 precision and 0.89 recall on the first turn of the $7,967$ conversations in the evaluation set. Using this classifier to ignore greetings in the fist turn further reduced the false positives by 2% (Table 7.3, Method 7). This proved to be a minor improvement, however, and can be considered an edge case.

Next, we focus on filtering Class II escalations. With each turn, we have the responding intent and the response text. Therefore, if a turn is positive for escalation, we ignore it if the previous intent was in a set of escalation intents. This ignore set was created by reviewing all agent responses and collecting intents responding with instructions to escalate the issue. Leveraging this conversational context removes 73% of the remaining false positives (Table 7.3, Method 8).

Any remaining escalations are considered members of Class III. The final algorithm using both the classifier and set of intents to ignore is given in Figure 7.1.

7.5 Application of Escalation Detection

We make no assumptions on the underlying IVA implementation; therefore, our algorithm can be applied to conversational data from both bounded and open-ended systems regardless of channel. Our results demonstrate that by combining classifiers with language features and narrowing our search to a specific class of escalations, we can successfully detect failure-driven escalation with high accuracy on real world data.

As previously mentioned, the risk indicator **precedes_esc** is assigned to turns that are immediately followed by an escalation request. Using the algorithm in Figure 7.1, the CRS detects explicit requests for escalation after the user first attempted to converse with the IVA. It then labels the preceding turn in the conversation with the **precedes_esc** risk indicator.

```
def LabelConv(conversation):
    Data: A conversation as an ordered list of turns containing (input text,
          responding intent)
    Result: The conversation with labeled turns
    if length(conversation) < 2:
        /* Filter single turns */
        label turn as false;
        return conversation;
    skip = true;
    for turn in conversation:
        tagged_text = POS_tag(turn.user_input);
        feature_vect = vectorizer(tagged_text);
        if hello_clf.predict(feature_vect) == 1:
            /* ignore greetings */
            label turn as false;
            continue;
        if esc_clf.predict(feature_vect) == 1:
            if skip:
                /* ignore 1..n Class I */
                label turn as false;
                continue;
            if prev_turn.intent in ignore_intents:
                /* ignore Class II */
                label turn as false;
            else:
                /* Class III */
                label turn as true;
        else:
            label turn as false;
        skip = false;
    return conversation;
```

Figure 7.1: Detect escalations and categorize

7.6 Escalation Prediction

Recall from Section 4.4 within the conversation level risk indicators there is the **conv_should_esc** indicator. This indicator is applied to all turns in a conversation where the conversation *should have* escalated, regardless if the user explicitly requested to escalate or not. It is important to note that escalations may be user or system-initiated depending on which party first determines that progress in completing a task is not being achieved. For the means to label the **conv_should_esc** indicator and related **should_esc_point** turn-level indicator, we now focus on the topic of system-initiated escalation and how to determine which conversations should escalate. The following work originally appeared in [113].

IVAs can be configured to accept user interaction through channels such as voice input, text input, or fully multi-modal interfaces that allow for any combination of voice, text, clicking-on-controls, and web content. Many IVAs provide the means to escalate a conversation to a human operator if necessary. Our task is to build a means to recommend when an escalation should occur, regardless of the input channel. Dialog systems contain a component for Natural Language Understanding (NLU) which maps a textual representation to a representation of the meaning, or *intent*, expressed by the user [149]. As the textual input to the NLU is the lowest common denominator of the various interfaces supported by multi-modal IVAs, we use this textual form for escalation recommendation. In this way, we ensure that regardless of the channel employed by the user, the recommendation system can still function.

In reviewing current literature on proactive escalation methods, we found no method suitable for our application. All existing methods rely to some extent on acoustic features generated by the Automatic Speech Recognition (ASR) component or other features dependent on language model implementation. Such features may not be present in a multi-modal IVA. Therefore, we set out to create a method with no reliance on features specific to channel; only the input and output of the NLU are consulted. We first evaluated several standard Machine Learning (ML) techniques but were not satisfied with their performance. We developed an algorithm that not only outperforms the ML approaches using textual features but also does not require a large labeled training set for each language domain *and* is considerably more time efficient.

We begin with a review of the literature involving the recommendation of escalations. We discuss our data and the evaluation of several standard ML techniques. However, we obtain even higher precision on an escalation criteria we develop. We also detail how this algorithm is implemented. Finally, we discuss the results, addressing limitations and outlining future work.

7.7 Related Works

Similar research on recommending escalations typically involves Spoken Dialogue Systems (SDS) such as AT&T's *How May I Help You* (HMIHY) application [150]. These systems have access to a wide range of acoustic features that are used in machine learning to identify problematic conversations and transfer a customer to a human customer care representative before the conversation fails. In [151], a machine learning program, RIPPER, achieved an accuracy of 72% in identifying problematic dialogues after the very first exchange and 86.7% accuracy given the whole conversation. However, over 50 features were available from spoken dialogues, of which almost a quarter were acoustic and ASR features. Acoustic features were also used in [152] where 55% accuracy was achieved on over 40,000 phone calls and [128] where 92.9% accuracy was achieved on 4,692 dialogues collected with the HMIHY system. A slightly higher accuracy of 93% was obtained using JRip in [153], but this is for *offline* detection of miscommunication which consumes the completed conversation whereas we are attempting *online* recommendation of escalation.

In [154], call duration was used as an indicator for when to escalate, and probabilistic models were constructed to generate policies identifying the best point in time to transfer callers to human operators. Although a useful feature for auditory assistants, duration of a conversation is not applicable to textual assistants; some customers may read or type more slowly than others, and pauses between turns are common. To demonstrate this, we collected a set of 8k conversations with a text-based IVA that logged explicit user escalation requests in order to measure correlation with duration. A correlation coefficient of .269 (calculated using the Scipy stats module) was obtained. Alternatively, one might consider the number of turns in the conversation before escalation. For this, we obtained a correlation coefficient of .26; therefore, turn or duration based strategies alone do not appear fruitful in predicting escalation.

Our strategy for recommending escalation is inspired by [94] where a SDS provides train timetable information. Although their goal was to detect a single misunderstood turn and not necessarily recommend escalations, we find that the negative cues used by these authors are helpful in the recommendation of escalation. The authors assume the Principle of Minimal Collaborative Effort; both the user and system want the dialogue to be finished as efficiently as possible and with success. Cues are examined and certain combinations of cues have the best predictive potential for discovering the presence or absence of problematic conversations. Cues include turn length, marked or unmarked word order (topicalization or extraposition), confirmation, the presence or absence of an answer, corrections or repetitions, and new information. The highest precision is achieved with a combination of correction and repetition cues on a small set of 120 dialogues; users tend to repeat their requests and correct the system in its interpretation of these requests when there are communication problems. Our criteria for escalation recommendation is heavily influenced by this work.

7.8 Methods

We collected 7,754 conversations (20,808 user turns) across two commercial multimodal IVAs deployed on corporate travel websites and mobile applications. The IVAs help with booking and changing flights and answer various travel-related queries. The conversations were manually tagged for whether or not they should have escalated by 2 reviewers, generating a Cohen's κ of .6. A third reviewer was used to break ties in cases of disagreement, and the majority determined the final tag. In addition to determining if a conversation should escalate, reviewers also notated the turn on which an escalation should occur. The escalation point, if existent, was averaged between the reviewers and rounded down to the nearest integer. Rounding down favors the user as he or she will be escalated faster. Rounding up benefits the company by saving money from delaying the use of human customer assistants. Of the 7,754 conversations, 1,268 were marked for escalation by a majority. A random 80-20 split was used to create training and testing sets. The training set consisted of 6,203 conversations of which 1,027 should escalate. The testing set consisted of 1,551 conversations of which 241 should escalate.

7.8.1 Preliminary Experiments

We initially experimented with several standard machine learning algorithms. As we were not only interested in a model's ability to determine *if* an escalation should occur but also *when*, the models were trained and tested on a cumulative turn basis. For example, if a conversation in the training set is four turns long and is tagged for escalation on the final turn, the model will be trained on $(turn1, 0), (turn1 + turn2, 0), ..., (turn1 + ... + turn4, 1)$.

As high accuracy was achieved using JRip in [153], we tried WEKA [155] JRip. Preprocessing of conversations was done using WEKA String to Word Vector. Fourteen rules were generated from the training set. A precision of .387 and recall of .051 was obtained on the test set for escalations. We also experimented with WEKA's Random Forest (RF) (with 100 trees), resulting in higher precision (.735) but equally terrible recall (.036). WEKA's SVM had better recall (.562) and precision (.317) compared to JRip. Default parameters were used. We then trained a Convolutional

Neural Network (CNN) inspired by [156] using Keras[1]. We used the same parameters and CNN build in [157]. GenSim's[2] Word2Vec was used to preprocess data. Over 100 epochs on CNN-rand, we obtained .776 precision and .718 recall on escalations. We also tried several Recurrent Neural Network (RNN) methods, but it appeared there was not enough data to train a useful model.

7.8.2 Point of Escalation

If a model chooses to escalate a conversation earlier than the tagged turn, the model is *aligned* with the customer as the customer will be escalated faster. If the model escalates a conversation later than the tagged turn, the model is aligned with the company. Alignments were calculated from conversations where the model and reviewer majority agreed that the conversation should escalate (even though the point of escalation may differ). So either reviewer 1 and reviewer 2 both believe the conversation should escalate, or they disagree, but reviewer 3 believes the conversation should escalate. In both cases, there are two reviewers that tag a 1 for the conversation. Suppose conversation Z has a majority vote to escalate, *and* the model chooses to escalate Z also. Suppose reviewer A and reviewer B tag Z for escalation, forming the majority. Let

$$X = \text{model's Predicted Turn} - \text{A's tagged turn}$$

$$Y = \text{model's Predicted Turn} - \text{B's tagged turn}$$

The alignment for a model on Z is the average of X and Y.

Thus, if an alignment is negative, the model prefers escalating early and favors the customer. If the alignment is positive, the company is favored. The number of conversations where the model and reviewer majority agreed that the conversation should escalate will be notated by $agree_{model}$. All frequencies were normalized

[1] `https://keras.io`

[2] `https://radimrehurek.com/gensim`

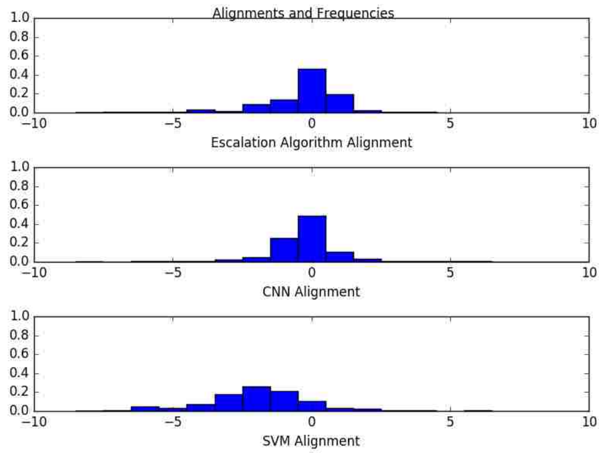


Figure 7.2: Alignments of CNN, SVM, and Algorithm 7.3

by their respective $agree_{model}$. Figure 7.2 includes alignment charts for the CNN ($agree_{CNN} = 173$) and SVM ($agree_{SVM} = 130$) discussed above. JRip and RF were not included due to exceptionally poor recall.

CNN is more fair to both the customer and company whereas the SVM tends to favor the customer and escalate earlier. Indeed the skew value (calculated using SciPy) is .444 for the CNN and 6.96 for the SVM. A skew value greater than 0 indicates left skew whereas a skew value less than 0 represents right skew. A normally distributed dataset should have close to 0 skew. The alignment of our escalation algorithm ($agree_{EA} = 156$) is also shown in Figure 7.2 and will be discussed in the following sections.

7.8.3 Escalation Algorithm Development

While the CNN model produced nearly acceptable results, the number of false positives was still too high for production use. In addition, the burden of collecting and tagging sufficient data from each IVA in order to train a CNN for its language domain was too great. For example, a model trained from insurance claims language

	User Turn	**Intent Hit**
1	im trying to buy upgrade, cant find it	Paid Upgrades
2	its not allowing me to buy an upgrade. Im already checked in	Paid Upgrades
3	this virtual assistant is NO assistance	IDK
4	I want to buy an upgrade for my flight today. Your links are not allowing me to do it	Paid Upgrades

Table 7.2: A customer *clarifying* his or her request with repetition, ultimately ending with frustration.

cannot be reused in product support, so a new model must be constructed. We require an approach that can be reapplied to any IVA with a minimal amount of language domain specific tuning.

Inspired by [94], which achieved good results on single turn misunderstanding detection, albeit on a small test set, we construct a similar detection strategy for escalation. Using only the set of 6,203 training conversations, we performed a manual analysis to determine if there were common structures in the conversations or user turns that would indicate persistent communication issues. If problems are detected, the system should perform an escalation *before* the user explicitly asks for one. As false positives would be very confusing to the users and expensive for the company, we must be conservative in our approach by favoring precision over recall. The result of this manual analysis is presented below in Figure 7.3 and is hereafter referred to as the Escalation Algorithm (EA).

At the heart of the EA is the presence or absence of a *clarify section*. A clarify section consists of an explanation of the problem, an optional reaction to the automated assistant's response, and a restatement of the first explanation (lines 1-10 in Figure 7.3). Notice that a clarification is not considered an exact repeat of the previous input (lines 3,8). Clarify sections occur frequently since many customers initially do not know they are speaking to an automated assistant. Upon this realization, customers tend to repeat and clarify their requests (Table 7.2). If the assistant is not providing an appropriate response to the user's question, a customer typically responds with *correctional language* (e.g. "No, that is not what I meant."). If a clarify section is found, we determine if it is followed by the presence of correctional language, an explicit escalation request, remarks on the uselessness of the assistant, or anger. Otherwise, we check if there are 3 or more user turns following the clarify section that contain similar requests. If so, we choose to escalate the conversation as, by this point, the automated assistant has returned an unsatisfactory response at least 4 times.

For the same reason, if no clarify section is present, recommend escalation if there are 4 or more similar requests, or if correctional language, explicit escalation requests, remarks on the uselessness of the assistant, or anger are present in the conversation after the first turn. We make these checks *after* the first turn to give the automated assistant a chance to rectify the situation if the customer begins the conversation frustrated or immediately requests to speak to a different party. Finally, we check if the agent has returned the same response multiple times (lines 1,2,4 in Table 7.2), or has responded with an I Don't Know (IDK) multiple times. An IDK response occurs when the IVA cannot determine with high confidence what the user means and will reply with something like "I'm sorry I didn't understand you, try rewording your question" (line 3 in Table 7.2).

7.8.4 Implementation Details

To determine the presence of correctional language (lines 13 and 26 in Figure 7.3), a set of 34 regular expressions was created from manual analysis of our training data[3]. For example, to detect the correctional language in line 2 of Table 7.3, a pattern such as "`^you did( not|nt) answer (my|the) question.+`" could be used. Each user turn was case-normalized, and punctuation was stripped prior to correctional

[3]`https://s3-us-west-2.amazonaws.com/anon-share/FLAIRS2017_correctional_res.txt`

```
1 clarify = false;
2 sim = calcSimilarity(userTurns[0],userTurns[1]);
3 if sim ≠ 1 and sim ≥ simThresh:
4     clarify = true;
5     index = 2;
6 else:
7     sim = calcSimilarity(userTurns[0],userTurns[2]);
8     if sim ≠ 1 and sim ≥ simThresh:
9         clarify = true;
10        index = 3;
11 if clarify == true:
12     afterClarify = userTurns[index:];
13     if detectCorrectionLang(afterClarify):
14         return true;
15     if detectEscalationLang(afterClarify):
16         return true;
17     if minSent(afterClarify) ≤ sentThresh:
18         return true;
19     sameReq = countSame(afterClarify,simThresh);
20     if sameReq ≥ 3:
21         return true;
22 else:
23     sameReq = countSame(userTurns,simThresh);
24     if sameReq ≥ 4:
25         return true;
26     if detectCorrectionLang(userTurns[1:]):
27         return true;
28     if detectEscalationLang(userTurns[1:]):
29         return true;
30     if minSent(userTurns[1:]) ≤ sentThresh:
31         return true;
32     if numRepeats(agentTurns) ≥ rptThresh:
33         return true;
34     if numIDKs(agentTurns) ≥ idkThresh:
35         return true;
36 return false;
```

Figure 7.3: Escalation Recommendation Algorithm

language determination.

For the determination of explicit escalation requests (lines 15,28), two methods

	User Turn	**Intent Hit**
1	Hi -agent-, the seat map only shows rows D to F. Where did A to C go?	Seating Chart
2	you did not answer my question. Rows A, B and C are not visible on the website.	-agent-'s Misunderstanding
3	Seat A, B and C are not visible on the online seating chart.	Seating Chart
4	what is the telephone number to contact a human?	Contact Phone Numbers

Table 7.3: A customer clarifies his or her request with repetition, corrects the automated assistant, and ends the conversation with an explicit request for escalation.

can be used. If the IVA understands escalation language, we can simply test if the IVA detected this in any of the user turns so far. If not, a stand-alone classifier for escalation language can be used. For our experiments, we implemented the latter exactly as described in [114] so that we did not rely on any specific IVA implementation.

Our algorithm makes use of thresholds for similarity and sentiment polarity, as well as number of repeated answers and IDKs. These are tuned using grid search and discussed in the following section. Polarity was measured using TextBlob's sentiment classifier, and a threshold is set for what constitutes an escalation (`sentThresh` in Figure 7.3). Sentence similarity is used to determine the number of same requests in a conversation or if a clarify section is present (lines 2,7,19,23). The similarity threshold is tunable to the sensitivity of the method in use (called `simThresh` in Figure 7.3).

Two methods for measuring similarity were compared: cosine similarity and Elasticsearch. A simple implementation of cosine similarity, which ranges from 0 (least similar) to 1 (identical), was used to measure surface similarity without considering semantics. We also experimented with Elasticsearch where every user turn was stored in an index along with the conversation ID and the order it appeared in. To measure the similarity between two turns, A and B, we queried the index using the text of A, the conversation ID of B, and the order ID of B. This results in a single match, the turn B, along with a relevance score. We treated relevance as a measure of similarity. The relevance score was calculated by the *practical scoring function* within Lucene, the underlying engine used by Elasticsearch. A relevance score can be 0 (no similarity) or any positive number. The greater the value, the more similar. Score thresholds are very specific to the query structure, and data and must be optimized appropriately. We did not observe a statistically significant difference between Elasticsearch and cosine similarity performance. As cosine similarity is both faster and less complex, we choose it as the similarity function in our algorithm.

7.8.5 Determination of Optimal Conversation Features

In order to determine the optimal values for the four thresholds in Figure 7.3, we perform a grid search on the training dataset. We set the values of the thresholds to be all combinations of the following: `sentThresh` $\in \{-0.3, -0.4, \ldots, -1\}$, `simThresh` $\in \{0.4, 0.5, \ldots, 1\}$, `rptThresh` $\in \{1, 2, \ldots, 6\}$, and `idkThresh` $\in \{1, 2, \ldots, 6\}$. Running the EA against the $6,203$ conversations in the training set, we measure the precision, recall, and F-1 score of each combination.

To determine the optimal values over the $2,016$ combinations we first rounded the F-1 scores of the results to the nearest hundredth. As many combinations of precision and recall can create similar F-1 scores, we simply selected the values with the highest precision (.883) in the top F-1 score (.75). As previously stated, precision is by far more important than recall as false positives are confusing to users and expensive for the companies. False negatives, on the other hand, are less essential as they present no difference to the user experience with the addition of our system. The optimal values selected were `sentThresh` $= -0.7$, `simThresh` $= 0.4$, `rptThresh` $= 4$, and `idkThresh` $= 4$. These are the threshold values used to measure EA performance on the test set in Table 7.4.

Model	$\mathbf{P}_E$	$\mathbf{R}_E$	$\mathbf{F1}_E$	$\mathbf{P}_{NE}$	$\mathbf{R}_{NE}$	$\mathbf{F1}_{NE}$
EA	**.876**	.647	.744	.938	.983	**.960**
CNN	.776	**.718**	**.746**	**.942**	.957	.950
JRip	.387	.051	.090	.834	.983	.902
SVM	.317	.562	.405	.892	.750	.815
RF	.735	.036	.069	.833	**.997**	.908

Table 7.4: Precision (P), recall (R), and the F1 score (F1) for all models. A subscript of E indicates a metric for the escalation class whereas NE represents no escalation.

7.9 Results

Precision and recall for both classes (E and NE) are displayed for all models in Table 7.4. Although CNN performance may increase if given more training data than we collected, it is important to realize that our EA has two advantages over the CNN. One, the EA needs considerably less training data for threshold tuning. To discover this, we took random samples from the training data and used them to both train a CNN and tune the EA thresholds. We then measured their performance on the test set (Figure 7.4). The EA precision remains stable as the training set decreases until more than 98% of training data is removed, whereas the CNN drops steadily until around 9% where performance becomes unpredictable. This also demonstrates that EA threshold tuning is not sensitive to overfitting. By 4% of training data (248 conversations), EA precision is still identical to that of 100% of training data! Two, the EA requires less time for threshold tuning than CNN training time (Table 7.5). Times were generated from training the CNN and tuning the EA on a server with 48 2.2GHz cores and 64 GB of RAM. A 4.5x average increase may not seem significant until one considers that this step will be required for every IVA deployed, and periodic retraining/retuning may be needed.

We included the alignment graph for the EA earlier in Figure 7.2. The EA appears to be somewhat fair to both the customer and company, although with a skew value of -1.52, there is a slight right skew, showing favor to the company.

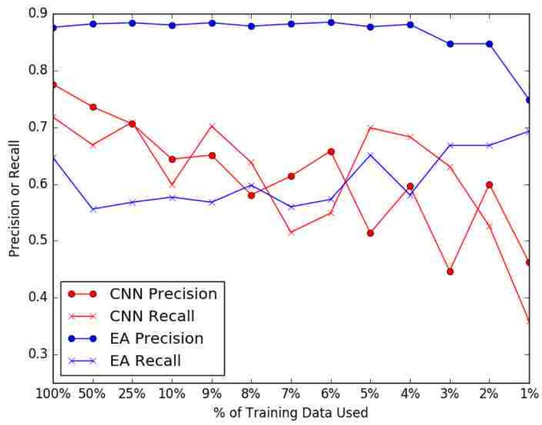


Figure 7.4: Precision and recall for CNN and EA on the test set given a random sample of the training set.

7.10 Discussion and Future Work

Some limitations of our work and different methods to detect features were considered, and a few are worthy of discussion.

For sentiment polarity, a Naive Bayes classifier was trained using Twitter data from [158]. Of the 5,513 manually tagged tweets available from [158], only 3,648 could still be accessed (due to deleted or banned accounts), and 80% were used as training data. Accuracy on the remaining 20% was 72%. TextBlob's sentiment classifier was used in our study instead as it demonstrated better performance in our system. However, TextBlob's sentiment classifier is pre-trained on a movie reviews corpus which may differ significantly from automated assistant chats. Alternative data for training a sentiment classifier could be addressed in future work as well as improved methods for sentiment analysis ([148], for example).

In our algorithm, polarity is determined on a by-turn basis. It would be interesting to split a turn into emotionally homogenous parts and assign a sequence of emotions to a turn like in [97], as we have observed that many longer turns are

% Training Data	EA Time (s)	CNN Time (s)	Increase
1	11	70	6.36x
5	94	339	3.61x
10	139	619	4.45x
25	361	1,466	4.06x
50	672	2,881	4.23x
100	1,375	6,469	4.71x

Table 7.5: Time needed to either tune the thresholds for EA or train the CNN given a percentage of the training set.

multipart with respect to sentiment. Also in [97], we could potentially add to our algorithm by determining common conversational structure sequences for sessions that do not escalate and detect deviations from this order.

There are limitations to using regular expressions (REs) to detect correctional language; variation in word ordering or spelling is difficult to account for. More sophisticated methods for this detection such as tagging data for correctional language and training a classifier for this task could be addressed in future work. However, the check for correctional language can be a considered a minor optimization. Care was taken to only construct REs that did not include any IVA or domain specific language to maximize re-usability. We disabled lines 13 and 26 in Figure 7.3 so that the REs are not even used. We measured no loss of precision and only a 4% drop in recall on the test set from this removal. As this is the only component of the EA that may require manual analysis for a new language domain, it is important to note it has a minor role.

Finally, both IVAs considered in this paper are in the domain of travel. We would like to extend our study to include other language domains and then compare it to our results.

Our algorithm features many strengths. As clearly displayed in Figure 7.4 and Table 7.5, significantly less training data and time is needed to achieve precision and recall comparable to that of standard machine learning techniques. We observed that reviewers tagged 5.1 conversations per minute on average. The EA may require only a few hours of human time in tagging to tune for each IVA. The CNN method may require over 20 times more tagged data and, therefore, carries a much higher human cost. With significantly lower tuning time for the EA, it is conceivable that humans could tag a number of conversations, re-tune the weights, and deploy them within a single day if needed. Also, our system only relies on conversational structure and text features which will be present in both speech and text based agents, and multi-modal agents. Regardless of which channel the human is using, it must be in text form by the time it reaches the NLU where our system would receive its input. Even in simple IR tasks such as a request for a specific document, we observed users restating their request if they were misunderstood and using correctional language.

Our results show that it is possible to achieve high precision (.876) in the recommendation of escalations without access to acoustic features. Whereas the techniques in [150, 128] are applicable to only systems that have access to acoustic features, our methodology can apply to systems with and without such features. For such a subjective task as determining when a conversation should escalate (recall we obtained a Cohen's κ of .6), our escalation algorithm performs better than several standard machine learning techniques *and* requires considerably less data and time for parameter tuning.

Many of the reviewed papers on alternative systems report only accuracy measurements instead of precision and recall. As the accuracy in our system was 93.1%, it is comparable to the accuracies reported in systems relying on acoustic features (55 to 93%). As our method only considers conversational structure and text features, it is less restrictive and can even work in multi-modal environments where turns can alternate between text and speech.

7.11 Application of Escalation Prediction

Within the CRS we apply the Escalation Algorithm (EA) outlined in the previous sections to each conversation during risk analysis. For each turn in a conversation that is marked for escalation by the EA, we assign the risk indicator **conv_should_esc**. In addition, on the turn that the EA indicates escalation should occur, we assign the risk indicator **should_esc_point**.

In addition to the escalation specific risk indicators, components of the EA are used as risk indicators in isolation. Recall that the purpose of the EA is to predict if a conversation should escalate and at which turn. It does not directly determine which turns in a conversation are misunderstood. Therefore, we use some of the features the EA uses to determine if a conversation should escalate to also determine if a particular turn in the conversation was potentially misunderstood by the IVA.

The first of these additional risk indicators is **precedes_corr**. In lines 13 and 26 of the EA, it calls a function that consults a set of regular expressions to determine the presence of correctional language. An example expression "`^(.+)?not what i.+`" would match comments like "That's not what I meant, I wanted ...." This same function is used to apply the **precedes_corr** risk indicator to any turn preceding a match for correctional language. We would expect any instance of correctional language to immediately follow a turn-response pair where the IVA misunderstood the user's intent.

Another feature the EA considers is the similarity between turns in the same conversation. On lines 2 and 7 of the EA, it calls a function to calculate the cosine similarity between two turns and considers turns with a small cosine distance to be restatements of the same intention. For the CRS, we reuse this function except we measure the similarity between a given turn and all following turns in the same conversation. For each turn that is considered a restatement of the following turn we apply the risk indicator **restated** to the current turn. Therefore, if the first turn is restated two times later on in the same conversation, the first turn will have two counts of **restated**, the first restatement will have one count of **restated**, and the second restatement will not have any as we have no further restatements to provide evidence that the third instance of similar language was misunderstood.

The EA in combination with the failure-driven escalation detection algorithm presented in Section 7.4 are used by the CRS to find conversations in which the user is unable to complete their task using the IVA. These conversations are very likely to contain turns that involve intents in need of repair in the IVAs language model. Therefore, escalation detection and prediction are important parts of the risk analysis process.

Chapter 8

Relational Language

In this chapter, we explore another important factor in the identification of missed intents, the inclusion of additional language within a turn that is unnecessary for the determination of the user intention. The contents of this chapter previously appeared in [75].

To better assist humans, IVA designers strive to support human-like interactions. Take, for example, Amazon's Alexa Prize competition where student developers attempt to build IVAs that can carry on meaningful, coherent, and engaging conversations for 20 minutes [159]. As IVAs become more human-like, we theorize that users will increasingly use *relational strategies* (e.g. self-exposure and justification) with IVAs similar to conversing with humans. There is a large body of work on development of trust between humans engaged in virtual dialog [160, 161, 162, 163]. The focus of these works is on how relational strategies contribute to trust between human speakers. From this literature, we predict the types of strategies humans may employ with IVAs as they relate to them in an increasingly human manner.

In customer service and personal assistant domains, trust is necessary between the human agent and customer. The customer's issues must be viewed by the agent as legitimate for proper attention to be given. Likewise, customers must trust that the agent is capable of assisting them and will not mistreat their information. Current research shows that human-like virtual agents are associated with not only greater user trust but also trust resilience when the agent makes mistakes [164]. To build trust with the agent, customers may establish credibility through small talk, self-exposure, and by providing justification of their requests [165].

In interactive question answering, such as dialogs with an IVA, understanding *user intent* is essential for the success of the IVA [166]. The intent can be defined as the interpretation of a user input that allows an agent to formulate the *best* response. However, when relational strategies are applied to IVAs, the additional language introduced is often unnecessary and can even obfuscate user intent. Such language can lead to confusion in the IVA and a degradation of user experience in the form of clarification questions and wrong information.

Example 1

I need a ticket to Boston this Saturday, my son is graduating!

In Example 1, the fact that the customer's son is graduating is unnecessary for determining the user's intent to purchase a ticket. By including unnecessary background information, the IVA may incorrectly deduce that the customer is booking a ticket *for* his or her son instead. Thus, the identification of relational segments is a useful feature for an IVA; unfortunately, no corpus of annotated relational segments exists to develop identification techniques [167].

This lack inspired us to create such a corpus[1]. Within this corpus, we needed to not only identify the location of relational language but also label its type (Gratitude, Greetings, etc.) so that automated methods to determine the relational strategy in use can be explored.

If these strategies are practiced by users of IVAs, it is important to identify

[1]https://s3-us-west-2.amazonaws.com/nextit-public/rsics.html

them; enabling IVAs to separate such language can help better clarify the users' main intention. For IVAs to become more human-like, determining which segments of a request are relational is necessary to allow these IVAs to both understand the user intent correctly and to include empathetic or reciprocal relational strategies.

8.1 Related Work

The identification of relational strategies in a single conversational turn can be structured as a multi-intent detection problem. The user not only wants the task completed (the *primary* intent); they may also attempt to build credibility or some common ground with the IVA (the *secondary* intent). Segments of text such as justification or backstory can be annotated as secondary intent and ignored while determining the primary intent. Once relational language is isolated, a separate classification can determine what relational strategies are in use and how to properly respond.

Multi-intent detection within dialog systems is still an emerging field; in recent work, only one intent is assumed to be present per turn [168]. A few methods exist such as [71] which uses multi-label learning and [18] which employs a two-stage intent detection strategy. However, [71] provided no explanation of how data was annotated nor any mention of annotator agreement. In [18], multi-intent data was fabricated by concatenating all combinations of single-intent sentences.

In the following sections, we discuss in detail how the data was collected, annotated, and merged to create *highlighted* sections. Another round of review was then done on these highlighted sections to determine the class of language present in these sections (e.g. Greeting, Gratitude, etc). We then measure and compare the frequency of relational strategies when users present their requests to IVAs versus humans. Finally, we conduct an experiment with three commercial IVAs, demonstrating that removal of relational strategies lowers confusion and leads to improved responses.

8.2 Data Collection

Next IT - Verint designs and builds IVAs on behalf of other companies and organizations, typically for customer service automation. This unique position allows access to a large number of IVA-human conversations that vary widely in scope and language domain. We selected IVAs for data collection based on the volume of conversations engaged in, the scope of knowledge, and the diversity of the customer base.

For diversity, we considered whether the target user base of the IVA was local, regional, national, or international and mapped the locations of the users engaging in conversations to visually verify. We only considered IVAs that had a national or international target user base and did not appear to have a dominate regional clustering to ensure that conversations were well distributed across users from different regions. This was to control for relational styles that may differ between regions.

IVAs deployed in domains that were highly sensitive, such as human resource management or health care, were not considered. As a result, human-computer data was collected from three live customer service IVAs in the language domains of airline, train travel, and telecommunications. Each agent met our criteria of a broad knowledge base, sufficient conversation volume, and a very diverse user base.

The selected IVAs are implemented as mixed-initiative dialog systems, each understanding more than 1,000 unique user intentions. The IVAs have conversational interfaces exposed through company websites and mobile applications. In addition, the IVAs are multi-modal, accepting both speech and textual inputs, and also have human-like qualities with simulated personalities and interests. A random sample of

2,000 conversations was taken from each domain. The samples originate from conversation logs during November 2015 for telecommunications and train travel and March 2013 for airline travel. There were 127,379 conversations available in the logs for the airline IVA. The telecommunications and train travel logs contained 837,370 and 694,764 conversations, respectively. The first user turn containing the problem statement was extracted. We focus on the initial turn as a user's first impression of an IVA is formed by its ability to respond accurately to his or her problem statement, and these impressions persist once formed [169, 170]. Therefore, it is imperative that any relational language present does not interfere with the IVA's understanding of the problem statement.

Finding a large mixed-initiative human-human customer service dataset for comparison with our human-computer dialogs proved difficult. Despite mentions of suitable data in [171] and [172], the authors did not release their data. Inspecting the human-human chat corpora among those surveyed by [167] revealed only one candidate: the Ubuntu Dialogue Corpus [173]. The corpus originates from an Internet Relay Chat (IRC) channel where many users discuss issues relating to the Ubuntu operating system. After a user posts a query on the channel, all following threads between the querying user and each responding user are isolated to create two-way task-specific dialogs. However, we want to study the initial problem statements to compare their composition with those extracted from our data. In the Ubuntu corpora, these are posed to a large unpaid audience in the hopes that someone will respond. The observed relational language and behavior was, therefore, no different than problem statements inspected in other IRC or forum datasets, and, for our purposes, was no more fitting than any other forum or open IRC dataset.

In addition, we desire to not just measure relational language content but also feed the problem statements into an IVA and measure the effect of any relational language on its understanding of the user intent. To do this, we needed requests that were very similar to those already handled by one of the selected IVAs to have any hope of the user intent already existing in the agent's knowledge base. Unsatisfied with the Ubuntu dataset, we instead focused on publicly visible question and answering data in domains similar to those of the selected IVAs.

Upon searching publicly visible chat rooms and forums in the domains of travel and telecommunications support, we found the TripAdvisor.com airline forum to be the closest in topic coverage. This forum includes discussions of airlines and polices, flight pricing and comparisons, flight booking websites, airports, and general flying tips and suggestions. We observed that the intentions of requests posted by users were very similar to that of requests handled by our airline travel IVA. While a forum setting is a different type of interaction than chatting with a customer service representative (user behavior is expected to differ when the audience is not paid to respond), it was the best fit that we could obtain for our study and subsequent release. A random sample of 2,000 threads from the 62,736 present during August 2016 was taken, and the initial post containing the problem statement was extracted. We use *request* hereafter to refer to the complete text of an initial turn or post extracted as described.

8.2.1 Annotation

From our four datasets of 2,000 requests each, we formed two equally-sized partitions of 4,000 requests with 1,000 pulled from every dataset. Each partition was assigned to four reviewers; thus, all 8,000 requests had exactly four independent annotations. All eight reviewers were employees of Next IT - Verint who volunteered to do the task in their personal time. As payment, each reviewer received a $150 gift card.

The reviewers were instructed to read each request and mark *all* text that appeared to be additional to the user intention. The reviewers were given very detailed instructions, shown in Appendix B, and were required to complete a tutorial demonstrating different types of relational language use before working on the actual

	All Requests	Multi-Intent	Single Intent	Unnecessary	Avg. Length
TripAdvisor	2000	734	1266	94.1%	93.26
Telecom	2000	149	1851	77.3%	19.81
Airline	2000	157	1843	68.6%	21.64
Train	2000	201	1799	55.3%	20.07

Table 8.1: Dataset statistics. The Multi-Intent column represents the count of Requests where one or more reviewers flagged it as containing more than one user intention. The Unnecessary column represents the percentage of Single Intent requests where one or more reviewers selected *any* text as being unnecessary in determining user intent. Avg. Length is the number of words present in All Requests, on average.

dataset. As the data was to be publicly released, we ensured that the task was clear. If more than one user intention was observed, the reviewer was instructed to flag it for removal. This was a design decision to simplify the problem of determining language necessary for identifying the user intention. Furthermore, as mentioned in section 8.1, IVAs with the ability to respond to multiple intentions are not yet commonplace. Although flagged requests were not used for further analysis, they are included in the released data to enable future research. After discarding all multi-intent requests, 6,759 requests remained. Per-dataset statistics are given in Table 8.1.

A request from the TripAdvisor data is given in Example 2 below. A reviewer first read over the request and determined that the user intent was to gather suggestions on things to do during a long layover in Atlanta. The reviewer then selected all of the text that they felt was not required to determine that intent. This unnecessary text in Example 2 is shown in gray. Each of the four reviewers performed this task independently, and we discuss in the next sections how we compare their agreement and merged the annotations.

Example 2

Original Request: Hi My daughter and I will have a 14 hour stopover from 20.20 on Sunday 7th August to 10.50 on Monday 8th August. Never been to Atlanta before. Any suggestions? Seems a very long time to be doing nothing. Thanks

Determine User Intent: *Things to do during layover in Atlanta*

Annotated Request: Hi My daughter and I will have a 14 hour stopover from 20.20 on Sunday 7th August to 10.50 on Monday 8th August. Never been to Atlanta before. Any suggestions? Seems a very long time to be doing nothing. Thanks

Reviewers averaged 1 request per minute over 1,000 requests on TripAdvisor data and 4 per minute over 3,000 requests from the three IVA datasets. We observed that each of the eight reviewers required 29 hours on average to complete their 4,000 assigned requests.

8.3 Annotation Alignment

To compare the raw agreement of annotations between two reviewers, we use a modification of *alignment* scores, a concept in speech recognition from hypothesis alignment to a reference transcript [174]. We modify this procedure as insertions and deletions do not occur. Reviewers mark sequences of text as being unnecessary in determining user intention. When comparing annotations between two reviewers, an error (e_i) is considered to be any character position i in the text where this binary determination does not match between them. The alignment score can be calculated as:

$$align = \frac{n - \sum_{i=1}^{n} e_i}{n}$$

where n is the total number of characters. Thus, $align \in [0, 1]$ where 1 is perfect alignment. Reviewers may or may not include whitespace and punctuation on the boundaries of their selections which can lead to variations in e_i. Therefore, when two selections overlap, we ignore such characters on the boundaries while determining e_i. Figure 8.1 shows a fabricated example of alignment between two annotations. In the first selection, the trailing whitespace and punctuation are ignored as they occur within overlapping selections. Notice, however, that whitespace and punctuation count in the last selections as there is no overlapping selection with the other reviewer; therefore, there is no possibility of disagreement on the boundaries.

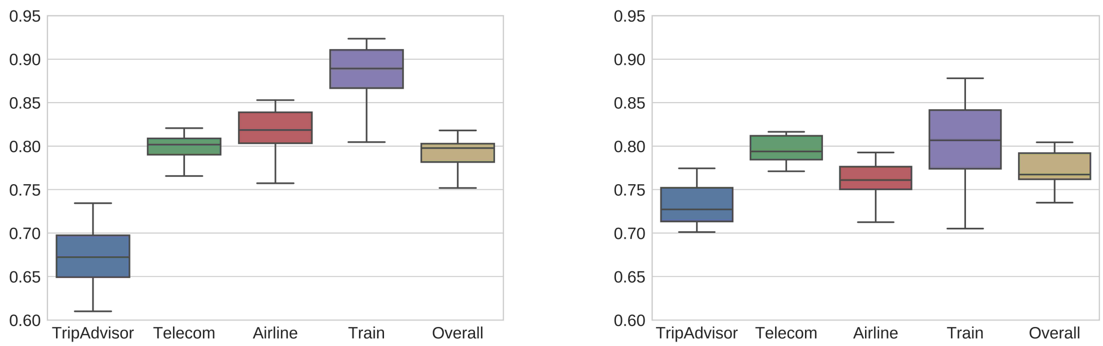
A: [Hi,]I need a new credit card[<u>, my old doesn't work any more.</u>] Can you help?
B: [Hi], I need a new credit card, my old doesn't work any more.[<u> Can you help?</u>]

$n = 73$ $\quad \sum_{i=1}^{73} e_i = 45$ $\quad align_{AB} = \frac{73-45}{73} = 0.384$

Figure 8.1: Example alignment scoring between two fabricated annotations A and B. Text between "[" and "]" was marked as unnecessary for intent determination. Positions with an alignment error are underlined.

The alignment score was calculated for every request between all four annotations and then averaged. For example, an alignment score was calculated for each request between reviewer A and B, A and C, A and D. The same process was repeated between reviewer B and C, B and D, then C and D. Finally, alignment scores between all unique pairs of reviewers over all requests were averaged per dataset. The distribution of average scores per dataset is shown in Figure 8.2 **(a)**. It may appear, at first, that two annotators could inflate the dataset alignment score by simply making annotations infrequently. However, as each request had four annotators, the average alignment score would actually be lower as those reviewers would have large error compared to the other two. The per dataset alignment averages can, in fact, be higher if a dataset has a large number of requests where *no* reviewer selected any text.

Therefore, it is interesting to remove the effect of these cases and compare the ability of reviewers to agree on the selection boundaries given they both agree that selection is necessary. To measure this, we compute average alignment scores where both reviewers agree that additional language is present, shown in Figure 8.2 **(b)**. Observe that although the Train dataset has the highest overall alignment in both cases, it is lower when the reviewers both select text, indicating it has many cases where no reviewers selected anything (which is in agreement with Table 8.1). In the case of TripAdvisor, it appears that there are a significant number of requests where one or more reviewers do not select text, but the others do, lowering the overall alignment score in Figure 8.2 **(a)**.



(a) Overall alignment scores

(b) Alignment scores when reviewers agree that additional language is present

Figure 8.2: The distribution of average alignment scores between all four annotations per dataset is shown in **(a)**. We compute average alignment scores where all reviewers agree that additional language is present in **(b)**.

Alignment based on word-level instead of character-level agreement was also considered. For each word, if the reviewer selected at least 50% of the word it was considered to be marked. This resolves situations where a reviewer accidentally

	TripAdvisor	Train	Airline	Telecom
κ	0.270	0.450	0.405	0.383
1	1192	995	1264	1431
2	1092	709	948	1154
3	863	458	644	795
4	534	205	292	410

Table 8.2: Reviewer agreement on if any text should be selected. For example, row 3 is the number of requests with selections by at least three reviewers.

missed the first or last few characters of a word in their selection. However, this may introduce errors where two letter words have only one character selected. In this case it is impossible to automatically decide if the reviewer meant to select the word or not as always selecting such words will be susceptible to the same error. Besides this ambiguous case, we felt it was safe to assume that words of longer length with less than half of the word selected were not intended to be marked.

Selected words were then used in place of selected characters in calculating the alignment scores between the reviewers in the same manner as Figure 8.1. We discovered that the alignment scores were only 0.2% different on average across the datasets than the character level alignment scores shown in Figure 8.2. This indicates that reviewers are rarely selecting partial words, and any disagreement is over *which* words to include in the selections. Therefore, in the released corpus and in this article, we consider selections using absolute character position which retains the reviewers' original selection boundaries.

8.3.1 Agreement Between Reviewers

As it is difficult to determine how often all reviewers agree additional language is present from alignment scores alone, we measured reviewer agreement on the presence of additional language and multiple user intentions. For additional language presence, we calculated Fleiss' κ over the annotations where the classes compared were if a reviewer did or did not select text. As demonstrated in Table 8.2, regardless of domain, this is a subjective task. While there is moderate agreement in the Train and Airline sets, the TripAdvisor set, in particular, is lower in agreement which reinforces our previous observations in Figures 8.2 **(a)** and **(b)**. Due to the sensitivity of κ measurements [175, 176], these values must be interpreted in light of the task. Despite the lower values, we are only measuring presence or absence of unnecessary language, and these two categories did not necessarily occur in equal frequencies. Under these conditions, according to [177], a κ between 0.2 and 0.45 may suggest reviewer reliabilities between 80% to 90%, respectively. Therefore, despite the lower values for κ, the individual reviewer annotations appear reliable and can be further improved when merged based on agreement as discussed in the following section.

Example 3

$R1$: Our tv reception is horrible. is there an outage in my area?

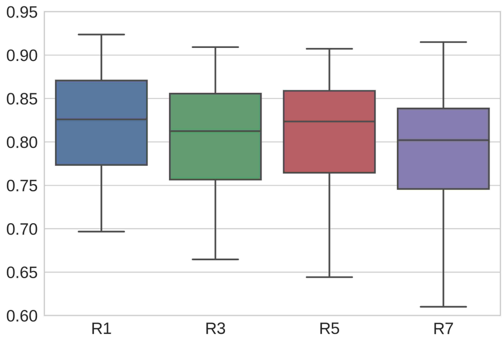
$R7$: Our tv reception is horrible. is there an outage in my area?

We did observe situations where two reviewers disagree on the real intent of the user, therefore, causing conflict in the selection of unnecessary text. While these were rare, Example 3 demonstrates how even humans sometimes struggle with determining the intention of written requests. Reviewer 1 appears to believe that the primary intent of the user is to notify the agent about poor television reception, and the query about the outage in the area is out of curiosity. However, reviewer 7 appears to believe the primary intent is to discover if a cable outage is present in the area, and the complaint about reception justifies the query. The effects of these disagreements on intent can be mitigated by merging the annotations based on the number of reviewers who agreed on a selected character.

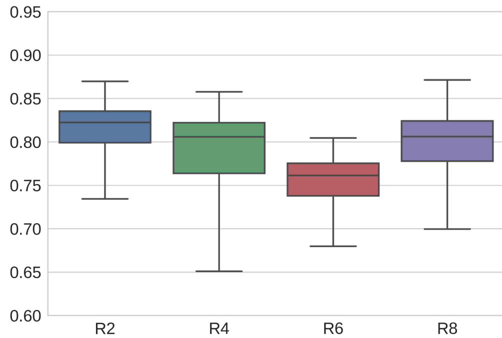
Next, we considered the reviewers' determination of multiple intentions. A κ

	TripAdvisor	Train	Airline	Telecom
κ	0.415	0.374	0.434	0.386
1	734	201	157	149
2	480	85	69	56
3	275	50	38	32
4	71	8	15	11

Table 8.3: Reviewer agreement on multi-intent detection. For example, row 3 is the number of requests flagged as containing multiple intentions by at least three reviewers.



(a) Alignment between group 1 reviewers.



(b) Alignment between group 2 reviewers.

Figure 8.3: Alignment scores between each reviewer and the other three members of their group, averaged across the four datasets.

was calculated over how reviewers flagged requests containing more than one user intention. As shown in Table 8.3, we see somewhat similar performance in this task as in the previous selection task. This table demonstrates the difficulty of multi-intent detection, even for humans. The domain does not seem to be a factor as κ is similar across datasets. It is apparent, however, that in the forum setting, users are much more likely to insert multiple intentions in a single request than in a chat setting.

How reviewers compare to the rest in their selections is another aspect to be considered. Figure 8.3 **(a)** compares how each reviewer agreed with the other 3 in the first group. We can see that, overall, the mean is very close. However, reviewer 7, in particular, had more variation in his or her selections. Similarly, Figure 8.3 **(b)** compares how each reviewer agreed with the other 3 in the second group. In the second group, we see slightly more disagreement, particularly with reviewer 6. This could be because he did not interpret the user intention the same as others or because the reviewer was more generous or conservative in selections compared to the others in the group.

8.3.2 Merging Selections By Agreement

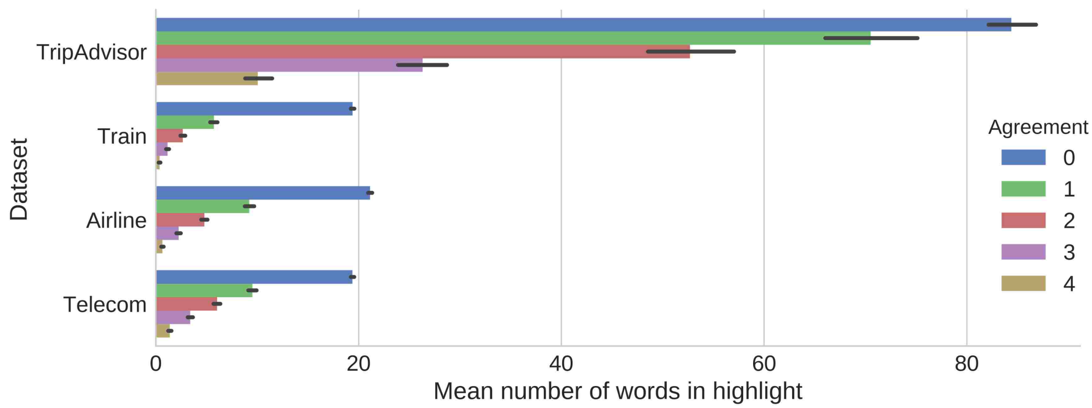


Figure 8.4: Mean number of words highlighted per request by dataset. Agreement is the number of reviewers who marked the same word for removal, where 0 is the original request length.

The four annotations per request were **merged** using the following strategy: for every character position in the request, if at least a threshold of two annotations contained that position, **highlight** it. To quantify the average reduction of request size, we count the number of words highlighted for each level of reviewer agreement. In Figure 8.4, we can see that as the agreement required increases, the size of the highlight decreases significantly.

8.4 Annotating Relational Content

To determine the use of relational strategies, a second round of manual analysis was performed. An increase in agreement corresponds to a significant removal of remaining annotations as can be seen in Figure 8.4. Therefore, in order to have sufficient data for analysis given the sample size, an agreement of two is used. A comparison of relational annotation using all agreement levels is left for future works.

Once merged, highlighted sections were analyzed by the authors to determine the classes of language present. Each such section was evaluated and given one or more of the following tags: *Greeting, Backstory, Justification, Gratitude, Rant, Express Emotion, Other*. See Figure 8.7 for an overview of the entire process.

Greetings are a common relational strategy humans use to build rapport with other humans and machines [178].

Backstory is a method of self-exposure that may be employed by the customer. In Example 1 given in section 8.1, the customer included the fact that he or she is attending a graduation as a means of self-exposure. This may be an attempt to build common ground with the agent or it may indicate the importance of the trip and motivate the agent to help the customer succeed.

Justification is used by the customer to argue why the agent should take some action on the part of the customer. For instance, when trying to replace a defective product, a customer may explain how the product failed to establish credibility that the product was at fault.

Gratitude, like greetings, are used by humans to also build rapport with humans and machines [178].

Ranting is a means of expressing dissatisfaction when a customer feels frustrated, ignored, or misunderstood. In computer-mediated conversations, the non-verbal emotional cues present in face-to-face conversations are missing; thus, humans resort to such negative strategies to convey their emotions [179]. For tagging purposes, we define a *Rant* to encompass any excessive complaining or negative narrative.

Expressing emotions can be a means of showing displeasure when a customer feels a conversation is not making adequate progress or in reaction to an unexpected or disagreeable agent response. This can also indicate joking or other positive emotional expression. The tag *Express Emotion* is used as a catch-all for any emotional statement that is not covered by *Rant*. Examples would be: *"i love that!"*, *"UGH!"*, *"WHY???"*.

The **Other** tag indicates that some or all of the section does not contain any relational language. This is commonly a restatement of the primary intent or facts that reviewers marked as unnecessary.

8.4.1 Analysis of Relational Tags

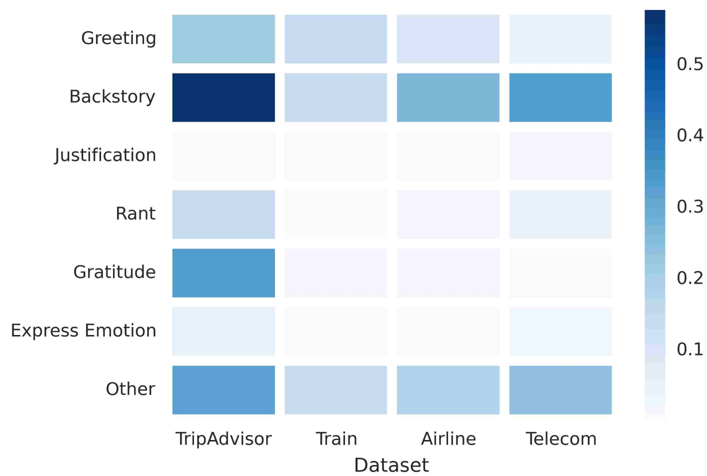


Figure 8.5: Incidence of relational language per dataset. An incidence of 0.5 means the tag is present in 50% of all requests.

As shown in Figure 8.5, we see that backstory is more common in human-to-human forum posts. However, both Airline and Telecom IVAs also have a significant amount of backstory. Although minimal, ranting and justification were present in Telecom. The Train dataset appeared to contain the least amount of relational language. It is difficult to speculate why without deeper analysis of the user demographic, the presentation of the IVA on the website, and the IVA knowledge base.

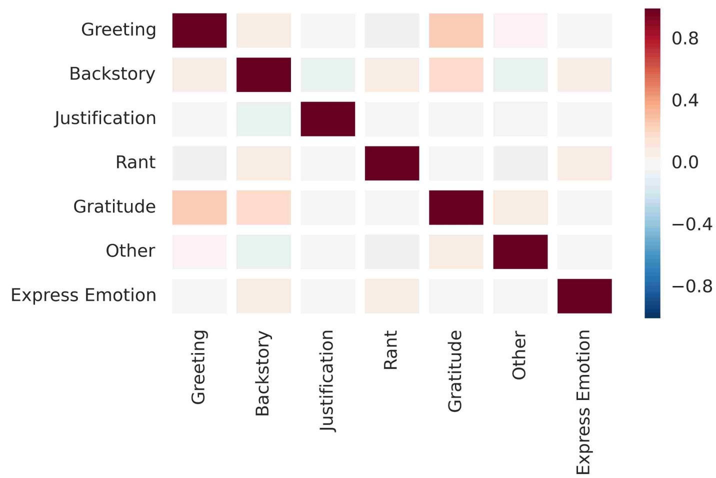


Figure 8.6: Pearson coefficients of tag correlation across datasets.

The correlation between tags is shown in Figure 8.6. When greetings are present, it appears that there is a likelihood there will also be gratitude expressed which agrees with the findings in [178] and [180]. Also interesting is the apparent correlation between backstory and gratitude. Those that give background on themselves and their situations appear more likely to thank the listener. Ranting appears to be slightly negatively correlated with greetings, which is understandable assuming frustrated individuals are not as interested in building rapport as they are venting their frustrations.

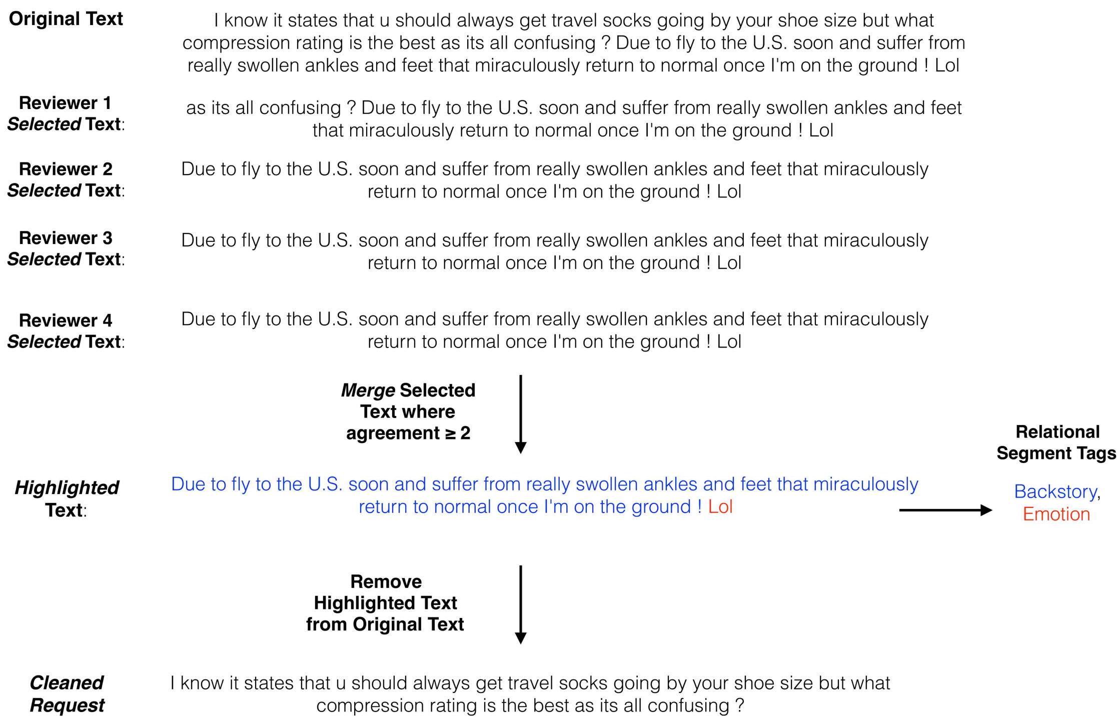


Figure 8.7: An overview of the review and merging process. In this example from the TripAdvisor corpus, reviewers 2, 3, and 4 all agree on which text is unnecessary. *Selections* are *merged* to form *highlighted* text that is then removed from the original text to create a *cleaned request*. A second round of annotation was done on highlighted texts to determine the classes of language present. The colors of the text correspond to the class present.

8.5 Experiments and Results

To measure the effect on IVA performance and determine what level of reviewer agreement is acceptable, we first constructed highlights for the 6,759 requests using all four levels of reviewer agreement. Next, four *cleaned* requests were generated from each original request by removing the highlighted portion for each level of agreement resulting in 27,036 requests with various amounts of relational language removed.

Every unaltered request was fed through its originating IVA, and the intent confidence score and response was recorded. We then fed each of the four cleaned requests

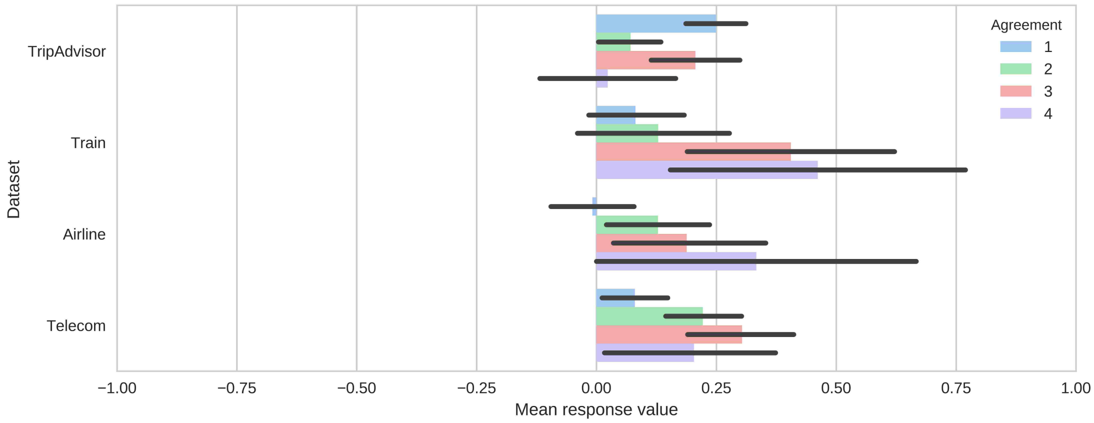


Figure 8.8: Results of the A-B test on IVA response to original request versus cleaned request. Black bars indicate 95% confidence intervals.

to the IVA and recorded the confidence and response. The TripAdvisor data was fed through the Airline IVA as it provided the most similar domain. This was also a test to see if lengthy human-to-human forum posts could be condensed and fed into an existing IVA to generate acceptable responses. We observed an increase in confidence in all domains with an average of 4.1%. The Telecom set, which had the highest incidence of backstory outside of TripAdvisor, gained 5.8%.

In addition to intent confidence, we measured the effect of relational language removal on overall IVA understanding. An A-B test was conducted where four reviewers were shown the user's original request along with the IVA response from the original request and the IVA response from a cleaned request. They were asked to determine which, if any, response they believed better addressed the request. If the original IVA response was preferred, it was assigned the value -1. If the response to the cleaned request was preferred, it was assigned the value 1. Finally, if neither response even remotely addressed the user's request or if both responses were comparable, it was given the value 0.

This A-B test was done only on responses that changed as a result of the cleaned request (3,588 IVA responses changed out of the 27,036 total responses). The result of this analysis is shown in Figure 8.8. Note that the lower bound is -1, indicating the original IVA response is preferred. If language is removed, the IVA response to the cleaned request is more likely preferred as made evident by the significantly positive skew. 95% confidence intervals are included, and although they may seem large, this is expected; recall that a 0 was assigned if both IVA responses address the user request comparably or neither did. In 10 of the 16 cases, the skew is towards the cleaned response within the 95% confidence interval.

This is evidence that the current usage of unnecessary language has a measurable negative effect on live commercial IVAs. TripAdvisor is an interesting exception, especially when the threshold is 4. However, this can be somewhat expected as it is a human-to-human forum where user inputs are significantly longer, and primary intent can be difficult to identify even for a human.

Although, in general, the removal of language is preferred, how *much* removal? This is another question addressed in Figure 8.8. The higher the threshold, the more reviewers need to agree on the removal of the same segment of text. Thus, although language may still be removed, less language is removed with a high threshold than if the threshold was lower due to low kappa (see 8.3.1). In effect, the higher thresholds may remove less unneeded language but the language that *is* removed is more likely to be actually unnecessary which appears to improve the IVA understanding. However, using a threshold of 4 seems to have limited improvement over 3 due to the reviewer disagreement.

Through the collection of this corpus and the annotation of relational segments, we have shown that users of commercial IVAs are already applying relational strategies to these IVAs. It is our prediction that these strategies will only increase as IVAs become more ubiquitous and human-like. We have also shown that the removal of unnecessary language during intent determination not only increases intent classifier confidence but also improves response by reviewer standards.

8.6 Application of Relational Language Detection to Risk Analysis

As demonstrated in Section 8.5 above, the presence of relational language can add confusion to single intent classifiers and therefore can be used as an indication of possible missed intent. Detection of relational language can be structured as a multi-intent detection problem, as described in Section 8.1. For our purposes, we are not concerned with the exact location of the relational language within the text or the classification of its type, we only want to detect its presence. Therefore, we follow a simple partitioning strategy for multi-intent detection proposed by [18]. Using the set of punctuation combined with a dictionary of common English conjunctions[2] such as "and", "but", "because", "so that", et cetera; we split each turn on every occurrence of punctuation or conjunction to form the set of all possible hypothesis pairs H, demonstrated in Example 4 below.

Example 4

Original turn h_{orig}*:* My mother and I just returned from Florida and they lost our bags. Who do we contact

Hypothesis pair 1: <My mother>, <I just returned from Florida and they lost our bags. Who do we contact>

Hypothesis pair 2: <My mother and I just returned from Florida>, <they lost our bags. Who do we contact>

Hypothesis pair 3: <My mother and I just returned from Florida and they lost our bags>, <Who do we contact>

In [18], the left and right segments, h_l and h_r, from every pair $h \in H$ are then fed into the intent classifier independently and the confidence score of classification on

[2]Obtained from http://grammar.ccc.commnet.edu/Grammar/conjunctions.htm

each is recorded. For the purpose of multi-intent detection, the authors determine if two separate intents are present by comparing the scores for h_l and h_r to the score for h_{orig} as shown in Equation 8.1.

$$\frac{min\{score(h_l), score(h_r)\}}{score(h_{orig})} > threshold_{multi-intent} \tag{8.1}$$

If Equation 8.1 holds, the authors conclude that there are two different intents present in the original turn. For our application, if this equation holds using the author's value of $threshold_{multi-intent} = 1$, we apply the risk indicator **multi_intent** to the original turn. As discussed in Section 4.4.2, we assume the IVA expects single intents per turn, therefore the presence of more than one intent can cause confusion in intent classification.

Furthermore, we can extend this idea of partitioning the original turn into segments and using the intent classifier's confidence on each segment to detecting the presence of unnecessary language. If we observe that either of the following equations hold we conclude that h_l (in Eq. 8.2) or h_r (Eq. 8.3) contains language that is unknown to the intent classifier and is therefore out of the expected scope for intent recognition.

$$[score(h_l) < score(h_{orig}) \times s] \wedge [score(h_r) > score(h_{orig})] \tag{8.2}$$

$$[score(h_l) > score(h_{orig})] \wedge [score(h_r) < score(h_{orig}) \times s] \tag{8.3}$$

If either equation is satisfied, using the arbitrary scaling factor $s = 0.75$, we apply the risk indicator **backstory** to the original turn. These equations may indicate the presence of unknown and potentially unnecessary language such as relational language.

Example 5

$score(h_{orig}) = 0.65$

Hypothesis pair 1: $score(h_l) = 0.01$, $score(h_r) = 0.7$

Hypothesis pair 2: $score(h_l) = 0.1$, $score(h_r) = 0.9$

Hypothesis pair 3: $score(h_l) = 0.4$, $score(h_r) = 0.5$

Continuing from the previous example, in Example 5 we see that either Hypothesis pair 1 or 2 would trigger the **backstory** risk indicator to be applied to the original turn, but none would trigger the **multi_intent** risk indicator. Either segment "I just returned from Florida and they lost our bags. Who do we contact" or "they lost our bags. Who do we contact" contains less unnecessary information to determine the user intent of <baggage_claims_contact_info> than the original turn. Therefore, the precedence of unnecessary language in the original turn can be used as an indicator of potential missed intent.

Chapter 9

Response Classes

Having covered indicators of missed intent that focus on the contents of user turns, we now look at the structure of the IVA responses that may cause confusion on the part of the user. When evaluating the performance of the Natural Language Understanding (NLU) component using features derived from the human inputs, we have to somehow control for the fact that the IVA did indeed understand, but the human user did not comprehend the response. If we do not attempt to control for this, we may see indication that user queries were not understood, such as repeating the query or including correctional language, when in fact the failure to understand was on the part of the user.

Human-IVA communication is a two party activity, and therefore both parties must understand each other in order to achieve success [181]. We cannot assume that human users will always achieve 100 percent perfect understanding of every possible IVA response. In this chapter, we explore ways to determine if the IVA did in fact understand, but did not present the response to the user in a way that was easy for them to comprehend. If this is the case, our risk features based on human user behavior can indicate the NLU is in error, when in fact it was the Natural Language Generation (NLG) component that was the cause of miscommunication.

The contents of this chapter to appear in [182] and are an extension of the work appearing in [183].

9.1 Introduction

With the continuing rise of Intelligent Virtual Assistants (IVAs) [3] and analysts predicting that human customer service agents will be altogether replaced by IVAs in the near future [8], discovering means to optimize human-computer interactions is necessary. As a company that builds IVAs primarily for customer service, we are interested in cases where there is no apparent misunderstanding on the part of the IVA, but the user continues to restate their query.

Restatements in conversation are problematic because they can break the *principle of least collaborative effort*: both the user and system want the dialogue to be finished as efficiently as possible and with success [95]. Restatements are a type of error correction mechanism employed by users when they sense the conversation is not progressing as it should [94, 104]. Even if the IVA understood the user's query, restatements may follow because the answer was not specific enough, the user did not fully read or understand the response, or the response was presented in a format that did not appeal to the user. In the latter case, as these IVAs are increasingly multimodal [81, 184], we theorize that not only is the formulation of the response important, but so is the media it is presented on.

For example, a user may prefer the IVA to answer their query directly in the response text instead of displaying a web page with the answer contained in it. If such a user were to ask an airline IVA the maximum carry-on dimensions, and in response, the IVA displays a web page containing all of the airline's baggage policies instead of directly answering the query in text form, the user may restate the query hoping for a more direct answer. Even though the IVA understood the user's request and displayed the correct page containing the answer, the user restated because they would prefer a direct answer from the IVA. This additional back and forth to resolve a query can lead to user dissatisfaction in the IVA and increase the time required to resolve customer support issues. Therefore, we are motivated to discover the causes of such restatements so that we can design IVA responses in a way that minimizes them.

As users have different preferences in communication, we concede that there is no one "right" way to formulate a response. Following the above example, a different user may actually prefer to see the web page with the entire carry-on policy as it would provide more detailed information. In light of this, we resort to statistical analysis to determine which features of a response are correlated to user restatements so they can be taken under consideration when designing responses. Knowing which features do and do not have an effect on user restatement gives dialog designers the tools to make more informed design choices.

In this article, we explore interactions with an IVA that communicates with customers over embedded live chat on a large company website as well as the company's mobile application. In both cases, the IVA is exposed on multimodal interfaces that use audio, text, images, User Interface (UI) controls, and web content as media. After tagging numerous features in these interactions, we perform statistical analysis to determine why the IVA response can appear acceptable to a reviewer (where "acceptable" means a reviewer has deemed the IVA response to have answered the user's question) but still fail to satisfy the user. Our contribution is to provide designers of multimodal IVAs guidance for intelligently selecting the media to present information to the user and the linguistic features of response text to minimize in order to reduce confusion.

9.2 Related Works

In [185], the authors provide a comprehensive overview on the evaluation and usability of spoken language dialogue systems (SLDS). The authors state that the inner workings and evaluation of commercial SLDSs are typically kept secret, but note that it is a well-known fact that a high rate of transaction success does not guarantee happy users. Test subjects may judge differently and more positively than real users. Thus, in our work, we not only determine if an IVA's response is acceptable to reviewers, but also consider *why* a user may repeat his or her request multiple times despite a reviewer's positive judgement call.

[186, 187, 188] all conclude empirically that multiple input and output modalities go well together for the user, but in our paper, we determine that it is not so simple; combinations of media and textual linguistic complexity need to be simultaneously considered for user satisfaction. This is also a very difficult problem given that users may score the same system very differently; one user may prefer related topic links over web content, but another may just prefer text.

Similar research involving the analysis of user behavior for the evaluation of IVAs exists such as in [81] where sequences of user behavior patterns (commanding, selecting, or confirming actions) are used to determine user satisfaction. However, user repetition is not analyzed. In addition, although this work involves interactions with intelligent assistants, the authors note that their approach works best on device function tasks (making phone calls, checking calendar events, etc.) and the worst on chat tasks.

Error spotting in conversation between IVAs and humans is well covered in [94, 104, 91, 105]. However, these works focus on the detection and recovery of errors resulting from misunderstanding or non-understanding on the part of the IVA, not the user. In addition, they do not take into account possible effects of modality as all IVAs involved were SLDSs, and therefore, communicating over a single media. [104], in particular, does highlight user-initiated error correction mechanisms and breaks user restatement into four specific actions: user repeats command, user repeats info, user rephrases info, and user rewords query. There is no discussion of the circumstances that these actions occur in as the data set consisted of only 40 dialogs, and there were few occurrences of each type of restatement.

While restatements can be an indication of conversational error correction, they may be motivated by ignorance on the part of the user to design a proper query [96]. A restatement as a result of a poor query would be the user trying to make his or her question more specific. For example, the authors in [189] discuss the problem of geolocation with Speak4it, a consumer-oriented application that uses multimodal input and output to help users search for local business information. Typically, local search systems assume the device's location for queries when the location is not explicitly stated by the user. The authors discover that users repeat queries and add locations to overcome errors arising from this basic assumption. Some information search systems also offer query suggestions, and this can be another source of repetition. Query suggestions can help users execute searches when it is difficult to formulate a query, especially if the user does not not know what kind of vocabulary to use [190]. The authors in [191] discovered that reformulation and system query assistance accounted for almost 45 percent of query reformulation actions. Users may make a series of small queries instead of one large one in the hopes of obtaining the best search results [192]. Thus, one cannot assume that all user restatements are automatically detrimental to the conversation.

To the best of our knowledge, ours is the only work involving the direct analysis of combinations of response media and linguistic complexity on user restatement. As mentioned previously, we consider *why* a user may repeat his or her request multiple times despite a reviewer determining that the IVA has correctly answered the user's questions.

9.3 Methods

Next IT - Verint designs and builds IVAs on behalf of other companies and organizations, typically for customer service automation. This unique position allows access to a large number of IVA-human conversations that vary widely in scope and language domain. During routine review of these conversations to improve IVA understanding, we frequently noticed that conversations would be flagged for review due to user restatements within them, but upon further inspection, they were not due to errors in IVA understanding. If the IVA correctly understands the user but the query is immediately restated, we reasoned that the response presentation must be somehow unacceptable to the user. To better understand why this phenomenon was occurring, we conducted the following experiment to determine what features of the IVA response can lead to these restatements.

9.3.1 IVA Selection

We reviewed our multimodal IVAs and selected a large international airline IVA for our analysis. The IVA interacts with users on the airline's website and mobile application, providing general travel advice such as flight status information, baggage and security rules, and even helps with the booking process. This particular assistant was selected as user interactions are a good middle ground between an Information Retrieval agent, as it must fetch flight status and travel documents, and a dialogue system, as it contains several tasks such as collecting everything needed to book a flight or transfer award miles between accounts. In addition, it is a very active IVA with a diverse user base. On average, it responds to 4.6 user inputs per second and engages in 115.5 unique conversations per minute with users located around the world. It supports mixed-initiative conversational dialog and can recognize 1,230 unique *user intentions*, which, in the context of Natural Language Processing, are interpretations of a user input or action that allows one to formulate the *best* response.

The input media supported by this agent are voice, text, UI elements, and web page events. Voice service is provided by the speech application programming interfaces available on the mobile device or browser; therefore, we have no access to Automatic Speech Recognition (ASR) features or original audio. We are simply given the resulting text translation. Example UI elements may be additional links provided by the agent as suggestions of related topics or drop down selection boxes used for tasks like indicating a country code. Web page events may be clicking on a help icon next to text on a webpage which will launch the agent with a query asking for more information on that topic or clicking navigation links to pages the agent is designed to help with.

The output media include the agent response in both text and audio format using Text To Speech (TTS), related topic links that, when clicked, will submit the topic to the agent for additional information, and pushing web pages related to the current topic to the user's browser window or mobile application.

The determination of which combination of output media to present in response to a user intention is up to the designers of the specific IVA. For the selected airline IVA, dialog designers determine what other topics in the IVA knowledge base are related to each intention and will manually associate links. They also determine if the airline has web content available on the company website to be associated with a given response. Other IVAs may do this automatically through Information Retrieval methods. The combination of media for a given response can therefore be predetermined by the dialog designers or automatically generated at run time. Even in instances where content is automatically generated, dialog designers have a choice of when to allow this additional content to be displayed and can therefore benefit from this study.

From conversation logs associated with the IVA, we selected 14,000 user input and IVA response pairs contained in 2,998 conversations where the user had at least two interactions with the IVA. The threshold of two was chosen as it is impossible to restate a query in a conversation with only a single interaction. To protect the identity of the human users by allowing time to pass before using data for research these conversations were taken from a random 24 hour period in 2013.

The conversation logs contain the user input to the IVA and its media, the IVA response text, related topic links displayed through the UI if any, and the URL to a web page that would have been displayed to the user if one was associated with the response. Although the IVA can respond with audio using TTS, the user is able to mute the audio in both the web interface and mobile application; we do not know if they have done so. Therefore, we exclude this medium of response from our analysis.

9.3.2 Data Annotation

Three rounds of tagging (see Tables 9.1, 9.2, and 9.5) were conducted on the data to discover user restatements, classify IVA response media, and determine if restatements were due to a clarification of task or not. Due to the size of the dataset and the expense of manual review, the data was divided equally among three reviewers for the first two rounds of tagging. To avoid bias, the chosen reviewers had never used the specific IVA and were not familiar with its knowledge base. The reviewers were given detailed instructions with examples and a general definition of the feature they were tagging for that round.

Although any URLs to web content displayed to the user were known to us, we selected conversations recorded three years prior to the analysis out of concern for user privacy. There is no way to verify that the web page content available now is identical to what users saw. Therefore, only the presence or absence of a URL is included in our analysis.

User Turn Features

In the first round of annotations (see Table 9.1), three reviewers were asked to identify if the turn appeared to be a restatement of some previous turn in the same conversation. In [104], this was broken into four categories: repeating command, repeating info, rephrasing info, and reword query. As the reviewers in that particular paper only tagged 40 conversations in this manner, and we have nearly $3,000$, we collapsed these four categories into one. As noted in [104], this may occur multiple times within the same conversation, and every query may result in multiple restatements until the user is satisfied or gives up. For our purposes, restatements include verbatim repetitions as well as rephrasing of the same info or query previously given. Rephrasing can contain additional information (clarification) or less information (generalization) as a previous turn. For example, turn 3 in Table 9.1 is a clarification of turn 2, and therefore tagged as a restatement of turn 2.

Turn	**User Input**	**Restatement?**
1	How do I change my ticket?	no
2	i want to upgrade my flight	no
3	i want to upgrade my flight today and purchase	yes
4	print baggage id	no

Table 9.1: Round One of Annotation: Is the user turn a restatement of some previous user turn in the same conversation?

Agent Response Features

For the next round, three reviewers tagged two features of each response by the IVA. For every input-response pair, reviewers determined if the response was acceptable given the user input. A reviewer tags a response as "acceptable" if they deem the IVA response to have answered the user's question. If not, the response was tagged as a misunderstanding on the part of the IVA. For example, Table 9.2 shows a conversation where a reviewer has determined that all IVA responses except for the last one are acceptable given the user's inputs.

User Input	**IVA Response**	**Acceptable?**	**Class?**
How do I change my ticket?	If your flight has not departed and you would like to change your reservation, enter your confirmation number in the space provided at the bottom of this Web page. See the links below for more information.	yes	C2c
i want to upgrade my flight	See the upgrade priority section of this Web page for information about the order in which upgrades are granted, or choose from the links below for information on specific types of upgrades.	yes	C2c
i want to upgrade my flight today and purchase	To upgrade to a premium cabin, you can pay a fee on this Web page. For more information about upgrading on the day of departure, select the link below.	yes	C3
print baggage id	In general, each traveler is allowed two pieces of checked baggage, one carry-on bag and one personal item for both domestic and international flights. For further information, please select a link below.	no	C2b

Table 9.2: Round Two of Annotation: Did the IVA's response address the user's input? Based on the combination of output media (text, web content, links), what is the IVA response media class?

Our tagging scheme corresponds to roughly 8 of the 13 *evidence of misunderstanding* features defined by [104] and the five *response-level errors* defined in [193]. However, as the focus of our analysis is only on cases where the IVA appeared to understand, we used a single tag for misunderstanding on the part of the IVA for any reason. This also greatly reduced tagging time.

The second response feature tagged for was the *response media class*. Recall in Section 9.3.1 that there are three response media in use by the IVA: the primary media of text and optional speech, related topic links, and web content URLs. These can occur in any combination; therefore, we define seven classes of response based on which combination of output media are referred to within it (see Table 9.3). If there is just one medium, it belongs in the class of **C1**. Likewise, if there are two media present, the class is **C2** and **C3** if three are present. We further divide these three classes based on what specific media are used. Note that links differ from web content; links, when clicked, will direct the user to additional information whereas web content refers to pages that are automatically pushed into the user's browser window or mobile application.

The first class, **C1a**, corresponds to responses seen in a typical text or speech only IVA. No links or web content are present. Next, we have class **C1b** where the agent displays web content in the browser or mobile application. Although the agent may also include a textual response, this response does not attempt to directly address the user query. Instead, the response may be something along the lines of, "See the following webpage for more information". Whether or not an attempt is made to "directly address" a user's query is determined by reviewers. No links are present. The final class involving just one medium is **C1c** which provides links and is similar to **C1b** where a textual response may be included, but it does not attempt to directly address the user query as determined by reviewers. The links provided give the user an indication of similar or more specific knowledge the IVA has on the current topic and allows the user to simply click on them instead of formulating clarifications or new queries. This response class may be used by the IVA as a form of clarification of the user's task based on the closest matching intentions in its knowledge base. Once clicked, the IVA will respond to the query associated with the link. No web content is present.

For classes involving two media, **C2a** consists of response text that actually

	Text	Web Content	Links
C1a	✓		
C1b		✓	
C1c			✓
C2a	✓	✓	
C2b	✓		✓
C2c		✓	✓
C3	✓	✓	✓

Table 9.3: There are seven classes of response based on which combination of output media are referred to within it: addressing the user query in text and optional speech, web content URLs, and related topic links.

attempts to directly address the user query along with web content containing additional information on the topic. No links are present. Next, **C2b** consists of response text that attempts to directly address the user query along with related or alternative topic links. No web content is present. The last IVA response in Table 9.2 is an example of this. The final class consisting of two media, **C2c**, consists solely of links and web content; any response text present does not attempt to answer the user query. The first two IVA responses in Table 9.2 are examples.

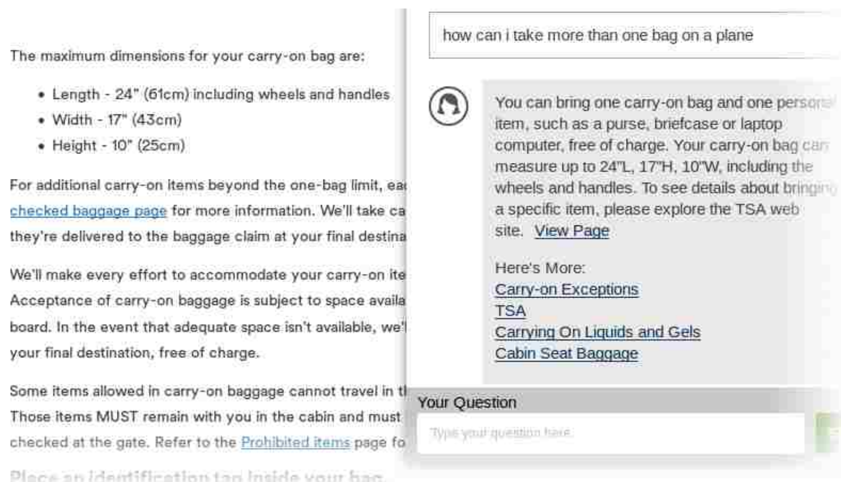


Figure 9.1: Example of a **C3** response from an IVA which involves all three media

The last class (**C3**) contains all three media of information. The IVA provides a textual response attempting to address the user's query, directs the user to web content containing specifics, and suggests related or alternative queries all at once. An example screenshot of a **C3** response is shown in Figure 9.1. Example screenshots of all response classes from various live IVAs on corporate web sites are given in Section 9.6. The distribution of these response classes over all of the responses in our sample is given in Table 9.4.

	Total	**C1a**	**C1b**	**C1c**	**C2a**	**C2b**	**C2c**	**C3**
Total	14,000	1,669	110	535	177	2,354	2,837	6,318

Table 9.4: Distribution of response classes over the 14,000 input-response pairs selected in Section 9.3.1

Task Clarification

For the final round of tagging (see Table 9.5), two reviewers were given conversations containing any restatements tagged in the first round. They were asked to determine if each restatement was identical to the task of its original statement or a more specific task. When the user inputs a request, the IVA responds. At this point three things can happen: the user accepts the response and continues the conversation, the user abandons the conversation, or the user restates their query or information. Not all restatements indicate a failure in communication however. As covered in [96], sometimes speakers are ignorant of the proper way to express their task or what tasks are even possible. The IVA response can therefore clue them in on the difference between their requested task and their real task, and lead them to properly formulate their request. In this case, although the restatement is still indicative of a user error correction strategy, it is indicating an error in the user knowledge, not in understanding the IVAs response.

Suppose the user asks "How do I change my seat?", and the IVA responds with instructions on how to modify a seat, but, in addition, includes a link called *Same*

Turn	User Input	Restatement	Clarifying?
1	How do I change my ticket?	no	-
2	i want to upgrade my flight	no	-
3	i want to upgrade my flight today and purchase	yes	yes
4	print baggage id	no	-

Table 9.5: Round Three of Annotation: Are the restatements that are present in a user turn identical to the task of its original restatement or a more specific task? The annotator must fill out the third column of the table for any turns that were marked as restatements in round one. In turn 3, the user specifies that the upgrade is for a flight that day, which is a more specific task than upgrading a flight in general.

Day Seat Changes. This link indicates to the the user that changing a seat on the same day of departure is a *different* task than what he or she originally asked for. The user might click on the link to get information on what was their real task all along; the user was not aware of such a distinction when stating the initial query. In this scenario, the restatement is not actually a negative user experience since the IVA has informed the user on the difference in task. This clarifying tag helps us separate restatements due to responses that may actually be *helpful* to the user from restatements where the user appears dissatisfied in the response. Note that with this example, the clarifying restatement came from link suggestions, but users may also infer they did not properly formulate their intended task by the IVA response text.

Figure 9.2 shows another example differentiating clarifying from non-clarifying restatements. In the first case, the user restates the same task with less descriptive information (generalization). In the second case, the user restates their task directly with little or no variation (repetition). In the final case, the user provides additional information for their task, differentiating it from the original general task (clarification). While the task still involves gathering information about baggage, the additional pertinent information allows the IVA to disregard other related tasks such as baggage claim numbers or where to drop off baggage. Without this additional information, the IVA's only recourse is to ask the user to clarify their task.

Non-clarifying - Generalization:

User: How do I transfer miles from my husband who passed away in 2007?
User: how to transfer miles from my late husband

Non-clarifying - Repetition:

User: Need flight info
User: Need flight information.

Clarifying - Task Specialization:

User: Checked baggage
User: Checked baggage allowances and policies

Figure 9.2: Some restatements are identical to the task of its original statement (non-clarifying) while other restatements narrow or clarify the task.

Reviewers only tagged restatements where the previous user turn was not misunderstood, as determined by the previous round of annotation in Section 9.3.2. Therefore, as the volume of data to be tagged was much lower than the total set of turns, we were able to have both reviewers tag all of the data and analyze agreement. For any restatement where the two reviewers disagreed, a third reviewer was used to break the tie. Therefore, in all cases, two reviewers agreed on the tag. To determine the reliability of the first two reviewers, we calculate the total reviewer disagreement D.

For measures of disagreement between two different expressions of a set of categories, allocation and quantity disagreement have been proposed as a replacement to a κ measure [194], which, as a ratio, is highly sensitive to the number of categories and the variability of their probabilities [177]. With only one binary category and an expected variability in the probabilities over conversations due to human communication differences, even a κ as low as 0.4 can still indicate observer accuracy anywhere from 80-95% [177]. For our task only involving two reviewers and one category, a simple proportion correct measurement C is sufficient for understanding reviewer agreement [194]. But, for the sake of completeness, we also discuss allocation and quantity disagreement and their implications.

Allocation disagreement, A, is how far off two expressions are from each other spatially. We would expect allocation disagreement to be high if the bottom-left and top-right cells in the 4×4 confusion matrix below were near equal. That would mean one reviewer annotated a similar quantity of "Clarifying" as the other reviewer but assigned the tags to different samples. Quantity disagreement, Q, is the difference in the two expressions due to the proportions in each category. We would expect the quantity disagreement to be higher if the bottom-left and upper-right cells in the 4×4 confusion matrix below were unbalanced. With this inverse relationship between Q and A, it makes sense that the total disagreement, D, between the reviewers is simply the sum of the quantity and allocation disagreement as we see in Equation 9.1.

$$D = 1 - C = Q + A \tag{9.1}$$

Given the following confusion matrix from the two initial reviewers, we can calculate C, Q, A, and D.

		Reviewer 1	
		Clarifying	Non-Clarifying
Reviewer 2	Clarifying	484	656
	Non-Clarifying	45	2328

The reviewer agreement, C, is the sum of the major diagonal of the confusion matrix divided by the sum of the elements of the matrix. Therefore, $C = \frac{484+2328}{484+656+45+2328} = \frac{2812}{3513} = 0.8$. Using Equation 9.1, we can calculate the total disagreement $D = 1 - 0.8 = 0.2$. Q is simply the proportion of restatements that Reviewer 1 marked as "Clarifying" compared to the proportion of restatements that

Reviewer 2 marked as "Clarifying". See [194] for full equations and example calculations for the multiple category case. Therefore, $Q = |\frac{484+45}{3513} - \frac{484+656}{3513}| = 0.1739$. Using Equation 9.1, we find $A = D - Q = 0.2 - 0.1739 = 0.0261$.

In summary, we can gather from the confusion matrix that there are nearly five times as many Non-Clarifying restatements as there are Clarifying. Remember that only the restatements where the previous user turn was *not* misunderstood were tagged. This indicates that something in the IVA responses is not satisfying users. As this was a subjective task, the reviewer agreement was good at 80%, and most of the disagreement came from an imbalance in the *quantity* of categories not the *location* of them. In other words, it is not the case that the two reviewers perceived a similar number of Clarifying restatements but allocated them differently. Instead, Reviewer 2 was more often tagging a restatement as Clarifying than Reviewer 1. Perhaps this tendency was because Reviewer 2 was more likely to perceive the addition of information in the restatement as a different task than Reviewer 1. Or perhaps one of the reviewers had a better understanding of the airline domain and what tasks are possible. At any rate, the 701 disagreements were tagged by a third reviewer which broke the ties. With good agreement between the first two reviewers and the enforcement of a majority decision, we are confident in the categorization of restatements for our analysis.

9.3.3 Linguistic Complexity

In order to measure the effect of response text complexity on restatements, we pass the text through the L2 Syntactic Complexity Analyzer (L2SCA). This generates 14 different measures covering length of production units, amounts of coordination, amounts of subordination, degree of phrasal sophistication and overall sentence complexity [1]. Although the L1 Lexical Complexity Analyzer generates more complexity indicators [195], it is also limited to text containing at least 50 words. As we are

- **MLS**: mean length of sentence
- **MLT**: mean length of T-unit
- **MLC**: mean length of clause
- **C/S**: clauses per sentence
- **VP/T**: verb phrases per T-unit
- **C/T**: clauses per T-unit
- **DC/C**: dependent clauses per clause
- **DC/T**: dependent clauses per T-unit
- **T/S**: T-units per sentence
- **CT/T**: complex T-unit ratio
- **CP/T**: coordinate phrases per T-unit
- **CP/C**: coordinate phrases per clause
- **CN/T**: complex nominals per T-unit
- **CN/C**: complex nominals per clause

Figure 9.3: The linguistic features of complexity as defined in L2SCA [1].

dealing with microtexts that are typically shorter than 50 words and the L2SCA does not have this limitation, we use it for measuring complexity.

Figure 9.3 lists the 14 measures of complexity returned by L2SCA. A T-unit is the smallest word group that can be considered a grammatical sentence regardless of punctuation. A clause is the smallest grammatical unit that can express a complete proposition, or subject and predicate [196]. Clauses can be split into two categories, main clauses which can stand alone in a sentence, and dependent clauses which require the presence of a main clause. A nominal is a word which is not grammatically a noun but functions as one [197]. Therefore, a complex nominal is a group of words that take the place of a noun such as “the man in the corner with a red hat” in place of “him” or “Bill”. A verb phrase is a type of predicate composed of a verb and its dependents and completes an idea about the subject in the sentence. A coordinate phrase is a complex structure that links together two or more elements with a coordinator such as “and”, “or”, “but”, or “then”. Further detail into grammatical form and structures and how they are used in English can be found in [196].

9.4 Results and Analysis

We first consider conditional probabilities and the co-occurrence of misunderstanding and restatement. Next, we proceed with a Mann-Whitney U statistical test for significance on the difference between syntactical complexities of agent responses depending on whether the next user turn is a restatement. We look at the response media classes and their relationship to user restatements in a logistic regression model while controlling for syntactical complexity. This experiment is repeated, but we consider the relationship between response media classes and clarifying restatements. The possibility of particular web pages causing more non-clarifying restatements is also addressed. Sections 9.4.1 to 9.4.3 covers all restatements whereas Section 9.4.4 is narrowed to just clarifying restatements.

9.4.1 Co-occurrence Matrices and Conditional Probabilities

We begin with an analysis of the co-occurrence of user restatement (**RE**) with IVA misunderstanding (**M**). A co-occurrence matrix is created on a conversational (Table 9.6(a)) and turn (Table 9.6(b)) basis.

In Table 9.6(a), there are 639 conversations that contain at least one user turn that was misunderstood by the agent and at least one user turn that was a restatement of some previous user turn. The misunderstood turn is not necessarily the restated turn; they just have to occur in the same *conversation*. Note that while 60% of the $2,998$ conversations contain at least one restatement, only 34% of those conversations also contained misunderstanding on the part of the IVA.

It is also worth investigating the co-occurrence of these features on a turn-by-turn basis. Since it is reasonable to believe that a user is likely to restate immediately after an agent misunderstanding has occurred, we consider whether a turn was misunderstood directly *before* a restatement.

	M (IVA)	**RE (User)**
M (IVA)	1019	639
RE (User)	639	1812

	PM (IVA)	**RE (User)**
PM (IVA)	1408	566
RE (User)	566	4104

Table 9.6: Co-occurrence matrices of **(a)** IVA misunderstanding (**M**) and user restatement (**RE**) at the conversation level and **(b)** previous turn misunderstood by the IVA (**PM**) and user restatement (**RE**) at the turn level.

In Table 9.6(b), there are 566 user turns where the previous turn was misunderstood (**PM**) by the agent and the current turn was a restatement of some previous user turn. We then calculated how likely a turn directly before a restatement was misunderstood ($p(\mathbf{RE}|\mathbf{PM})$), but this only yielded a probability of .4. More surprising was how many turns were apparently *not* misunderstood by the agent, but the user repeated his or her request anyway! Understanding the cause of this scenario ($\mathbf{RE} \wedge \neg\mathbf{PM}$) is the focus of this work. Although some of these restatements are a clarification of the task as described in Section 9.3.2, most restatements are not, and, therefore, may be related to the complexity of the agent response and the media in which it is presented. While the former type of restatement is, in fact, good as the IVA aids the user in uncovering their true task, the latter type is bad as it breaks the principle of least collaborative effort in conversations [181].

Taking into account the differences in response class and restatement type, Table 9.7 shows the distribution and occurrence of response media classes ($\mathbf{C_i}$) in relation to the next turn being a restatement (**NR**). We see the number of unique responses observed per class overall (row 1) and the number of unique responses observed when the next input is a restatement (row 2). Rows 3 and 4 give the distri-

		C1a	**C1b**	**C1c**	**C2a**	**C2b**	**C2c**	**C3**
1	Unique responses overall	49	38	28	43	55	126	311
2	Unique **NR** responses	23	15	20	19	40	88	212
3	Occurrence overall	1669	110	535	177	2354	2837	6318
4	Occurrence with **NR**	484	26	140	51	704	909	1790
5	$p(\mathbf{NR} \wedge GOOD \mid \mathbf{C_i})$	0.026	0.027	0.090	0.074	0.069	0.075	0.040
6	$p(\mathbf{NR} \wedge BAD \mid \mathbf{C_i})$	0.129	0.173	0.135	0.203	0.195	0.225	0.212

Table 9.7: Distribution of IVA responses by media in rows 1 through 4. Rows 5 and 6 show the probability of the next turn being a restatement that clarifies the task ($\mathbf{NR} \wedge GOOD$) given the class $\mathbf{C_i}$ and the next turn being a restatement of the identical task ($\mathbf{NR} \wedge BAD$) given the class $\mathbf{C_i}$.

bution of all $14,000$ responses and the $4,104$ responses preceding a restatement over the response classes. The final two rows give the probabilities of the next turn being a restatement that clarifies the task (row 5) and the next turn being a restatement of the identical task (row 6) for each response class.

Inspecting row 6, the classes using only a single medium of response (**C1a,b,c**) have the lowest probabilities of being followed by a bad type of restatement, with **C1b** (web content only) having a slightly higher probability than the other two. Adding web content (**C2a**) to textual responses (**C1a**) and adding web content (**C2c**) to topic links (**C1c**) appeared to increase the probability of bad restatements more than adding topic links (**C2b**) to textual responses (**C1a**) and web content (**C1b**). In addition, adding web content (**C3**) to the class already using text and links (**C2b**) increased the probability slightly more than adding topic links (**C3**) to the class already using text and web content (**C2a**). However, adding text (**C3**) to the class already using topic links and web content (**C2c**) slightly decreased the probability.

It is apparent from this table that the relationship between response class and restatement type is a complicated one and warrants deeper study. In the remainder of this section, we attempt to uncover the possible causes of bad restatements and explain their associations to text complexity and media. Through this article, it is our hope that designers of IVA responses can use the information we glean to prevent non-clarifying restatements attributable to presentation.

9.4.2 Complexity and the Mann-Whitney U Test

We first only considered turns where the previous turn was not a misunderstanding. We then partitioned this dataset to user turns where their next turn is a restatement (**NR**) and user turns where their next turn is not a restatement ($\neg$ **NR**). A Mann-Whitney U test was conducted to determine if **NR** tends to have stochastically greater values than $\neg$ **NR** for each of the 14 features of complexity. As mentioned in 9.3.3, the fourteen measures of syntactic complexity are discussed in detail in [1] and are defined in Figure 9.3.

We also repeat this test on different media of agent response using the seven classes described in 9.3.2. Complexity was calculated solely on agent response text and not on text in web content and links as there is no way to verify that such content available now is identical to what users saw as mentioned previously in Section 9.3.2. The final row, the average number of words for an agent response (**AW**), is not part of the 14 features but is included to give some intuition on the text content differences between classes.

For example, consider an agent response belonging to class **C2b** (text and links). The complexity value for **VP/T** (number of verb phrases per T-unit) is likely to be higher if the next turn is a restatement than if the next turn was not a restatement (sixth column, last row of Table 9.8). But if we instead consider all classes (first column, last row), **VP/T** has the reverse effect. Note that a tie occurs for **C1b** (web content only) under the complexity feature **T/S** (number of T-units per sentence). See Section 9.5 for a discussion of the implications of these results.

	All	C1a	C1b	C1c	C2a	C2b	C2c	C3
MLC	F	T^{***}	F	T^{***}	T^{*}	T^{***}	F^{***}	F^{***}
MLS	T^{***}	T^{***}	F^{**}	F	F	F^{**}	T	T^{*}
MLT	F	T^{***}	F^{**}	F	F	F^{***}	F	T^{**}
C/S	F^{***}	T	F	F^{***}	F^{***}	F	T^{***}	T^{***}
C/T	F	F^{**}	F	F^{***}	F^{***}	F^{***}	F^{***}	T^{***}
CT/T	F^{***}	F^{*}	F	F^{***}	T	F^{***}	F^{***}	T^{***}
DC/C	F^{***}	F^{**}	F	F^{***}	F^{*}	F^{***}	F^{**}	T^{***}
DC/T	T^{***}	F	F	F^{***}	T	F^{***}	F^{***}	F^{***}
CP/C	F^{***}	F^{**}	F^{*}	F^{***}	F	F^{***}	F^{***}	F^{***}
CP/T	F^{***}	F^{**}	F	F^{***}	F	F^{***}	F^{***}	F^{***}
T/S	F^{***}	F^{***}	TIE	F^{*}	F	F^{***}	T^{**}	F^{**}
CN/C	F^{***}	T^{***}	F^{*}	T^{***}	F	F^{***}	F^{**}	T
CN/T	T^{***}	T^{***}	F^{***}	F^{***}	F	T	F^{***}	T
VP/T	F^{***}	T^{***}	T^{**}	F^{***}	F^{*}	T^{***}	F	F^{***}
AW	38.66	13.97	16.56	17.92	29.42	46.43	25.57	48.93

Table 9.8: Mann-Whitney U results comparing **NR** and ¬ **NR** using the 14 complexity features in [1] for all data (**All**) and separate classes in Section 9.3.2. A value of T indicates that the values in **NR** tend to be greater than the values in ¬ **NR** for the complexity feature. TIE indicates there is an equal chance of restatement or no restatement. Otherwise, the value is F. $^{*}: p \leq 0.1$, $^{**}: p \leq 0.05$, and $^{***}: p \leq 0.01$. The final row represents the average number of words for an agent response in that class (**AW**).

9.4.3 Complexity and Logistic Regression

The previous analysis indicates that there are significant differences in agent response complexity between turns which lead to a restatement and turns that don't on an individual class basis. Now we wish to consider the effects of all response media classes on the next turn being a restatement while *controlling* for syntactical complexity. We fit a logistic regression model where the independent variables are the value of a complexity feature and x_i, where $x_i = 1$ if the agent response belongs to media class i (see Table 9.9). We only choose to use complexity features that are not entirely composed of a row of F's (regardless of significance) in Table 9.8. Thus, **CP/C** and

	C1b	C1c	C2a	C2b	C2c	C3
MLC	−0.0090	0.0706	0.3634	0.2898	0.4211	0.2161
MLS	−0.0710	0.0628	0.2803	0.2162	0.2820	0.1206
MLT	0.0241	0.0794	0.4110	0.3288	0.4542	0.2623
C/S	0.0478	0.1085	0.3888	0.3386	0.3866	0.2412
C/T	−0.0209	0.0630	0.3627	0.2814	0.4178	0.2135
CT/T	0.0933	0.1187	0.3277	0.3770	0.3606	0.2159
DC/C	0.1392	0.1647	0.3182	0.4047	0.4449	0.1902
DC/T	0.1192	0.1587	0.3701	0.4025	0.4704	0.2405
T/S	0.0356	0.1103	0.4245	0.3227	0.1779	0.2436
CN/C	−0.0900	0.0148	0.3037	0.2149	0.3749	0.1429
CN/T	−0.0632	0.0536	0.3030	0.2497	0.3712	0.1574
VP/T	0.0966	0.1213	0.3857	0.3152	0.4834	0.2203

Table 9.9: Coefficients for a logistic regression model where the dependent variable is whether or not the next turn is a restatement. The independent variables are x_i which represents class membership for class i, and complexity is a control variable. **C1a** is the base class; thus, it is not included in the table. Statistical signifiance (95% CI) was present for all features.

CP/T are not included (coordinate phrases per clause and T-unit). The dependent variable is whether or not the next turn is a restatement. So, for example, a restatement is more likely for class **C2c** (web content and links) controlling for number of verb phrases per T-unit (coefficient of .4834) than for class **C1b** (web content only) controlling for that same complexity feature (coefficient of .0966). To ensure that our predictors maintain independence, one class (**C1a**, text only) is not included in the regression, and it serves as the base class [198]. We use a likelihood-ratio test [199] to test for statistical significance. Significance was present for all features using a 95% confidence interval; response class has a significant effect on restatement.

9.4.4 Clarifying Restatement Analysis

We previously considered the effects of all response media classes on the next turn being a restatement while controlling for syntactical complexity. Recall that in Section 9.3.2, reviewers were asked to determine if the restatement was, in reality, a more specific task than the original statement. Considering only input-response pairs that are not misunderstood and have restatements in the next turn, we fit a logistic regression model where the independent variables are the value of a complexity feature and x_i, where $x_i = 1$ if the agent response belongs to media class i (see Table 9.10). Similar to Table 9.9, we did not include **CP/C** and **CP/T**, **C1a** serves as the base class, and a likelihood-ratio test [199] was used to test for statistical significance. Significance was present for all features using a 99% confidence interval. The dependent variable is whether or not the restatement in the next turn is a clarification or narrowing of scope (0 for no and 1 for yes). This table demonstrates to what extent combinations of response media either tend to help the user narrow their task (positive coefficients) or somehow dissatisfies the user (negative coefficients) compared to the IVA's primary medium alone (**C1a**).

Column **C1b** in Table 9.10 is completely dominated by negative coefficients. Recall in Table 9.3 that class **C1b** consists of web content, no links, and does not have an agent response that addresses the user request according to reviewers. These negative coefficients imply that directing users to web content may increase the number of restatements that neither clarify nor narrow scope. In contrast, column **C1c** (links only) has the highest positive values. These positive coefficients imply that providing related topic links may increase the number of clarifying restatements. However, before drawing conclusions about **C1b**, we must determine if there are particular web pages that cause more problems than others; this may unfairly skew the results. We cannot conclude that web content is a problematic response media in general as it could be the case that the content of these websites is the problem and not the medium itself. Thus, we check if the content of particular web pages

	C1b	C1c	C2a	C2b	C2c	C3
MLC	−0.6725	0.9323	0.1318	−0.0533	0.2934	−0.5283
MLS	−0.3091	1.2328	0.3972	0.3652	0.1736	−0.2756
MLT	−0.1614	1.2218	0.7555	0.6918	0.6265	0.0854
C/S	−0.2177	1.2229	0.6051	0.5824	0.5094	−0.0552
C/T	−0.3898	1.0559	0.5539	0.4274	0.5388	−0.0608
CT/T	−0.2973	1.1514	0.6269	0.5177	0.5679	−0.0545
DC/C	−0.4319	1.0423	0.7120	0.4368	0.5005	−0.0024
DC/T	−0.4575	1.0093	0.6492	0.3964	0.4784	−0.0371
T/S	−0.0093	1.4311	0.8534	0.5979	−0.0584	0.1016
CN/C	−0.5906	0.7859	0.2117	0.0446	0.2778	−0.4678
CN/T	−0.2424	1.1315	0.4512	0.4828	0.3903	−0.2057
VP/T	−0.5530	0.9115	0.6157	0.5623	0.3009	−0.0162

Table 9.10: Coefficients for a logistic regression model where the dependent variable is whether or not the restatement in the next turn is a clarification or narrowing of scope. The independent variables are x_i which represents class membership for class i, and complexity is a control variable. **C1a** is the base class; thus, it is not included in the table. Statistical signifiance (99% CI) was present for all features.

were dissatisfying by measuring the effect of their URLs on restatements.

To check for possible biases due to web content displayed, we need a way to rank the URLs that referred to the web content in terms of how problematic they were. We perform this ranking determination using odds ratios (OR) [200]. The *exposure* variable is the presence or absence of a particular URL. The *outcome* can be one of two categories: there is either no restatement or a clarifying restatement in the next turn which would indicate a non-problematic URL (or not obviously problematic), or there is a non-clarifying restatement in the next turn which would indicate a problematic URL. If the OR is greater than 1, then the exposure is associated with higher odds of the first outcome. If the OR is less than 1, the exposure is associated with lower odds of first outcome. Finally, if the OR equals 1, the exposure does not affect the outcome's odds.

We began with the set of all URLs the IVA could return in a response. Only considering input-response pairs that were not misunderstood, we first filtered URLs where either the number of non-clarifying restatements when the URL is present was 0 or the URL was present less than 5 times overall; 75 URLs remained.

Next, to determine the magnitude of effect of URLs with very low ORs (problematic URLs) on Table 9.10, we chose to remove any that were below an arbitrary cutoff of 0.5. If the logistic regression model used to create that table changes significantly due to this removal, this indicates that the problematic URLs have great influence. However, if the model does not change significantly, the problematic URLs do not have great influence on our results.

Although we could have chosen a more "natural" cutoff of 1, URLs close to 1 are not suspect of containing overtly unhelpful content as they have only a slight effect on the outcome. As the OR decreases from 1, the odds that the URL's contents will lead to a positive outcome diminish.

There were 13 URLs with ORs of less than 0.5 in the set of all URLs. To measure the effect of these suspect URLs on non-clarifying restatements, we removed all input-response pairs containing them and redid our logistic regression model in Table 9.10. Once again, if the model exhibits great change in behavior as a result of this removal, this signifies that the problematic URLs have great influence. However, the maximum change in any coefficient in the logistic regression model was around 0.3, and most changes were less than 0.1. There were no significant changes in positive or negative influence of any response class on non-clarifying restatements. Thus, we have considered the possibility of particular web content causing more non-clarifying restatements and determined its effect to be minimal.

9.5 Discussion

In Section 9.4.1, we investigated manually tagged features of agent misunderstanding and user restatement. Restatements were far more likely when the previous turn was *not* misunderstood. This indicates that wording and media selection of IVA response may be just as important as agent understanding. Although correct understanding is a necessary condition for correct response, an apparently correct response does not ensure resolution; users appeared to restate their issue over six times more often when there was *no* apparent misunderstanding on the part of the IVA than when there was (see the user restatement column in Table 9.6(b)). Recall that the determination of misunderstanding is based off of the *IVA response text* which accounts for correct natural language understanding as well as natural language generation. This greatly justifies our work.

In the domain of customer service, neither the system nor users want to spend more effort than necessary on completing their tasks [94]. These restatements on the part of the user are probably not exploratory, where the user probes the knowledge boundaries of the IVA out of curiosity as is expected with chatterbot interactions. Instead, this may indicate that the user's query is not being answered efficiently. This could be due to poor wording of the response which we cannot completely control for. Asking reviewers to tag responses that do not appear to answer the user's question has problems of its own; what is clear to a reviewer may not be clear to the original user. However, the difference between user restatement with and without prior IVA misunderstanding is so great that this cannot be the only reason. To help explain this difference, we turn our attention to response complexity and media.

9.5.1 Response Complexity

Considering Table 9.8, we can safely eliminate the two complexity features with a row consisting only of F (coordinate phrases per clause and per T-unit) as they appear to have no effect on restatements *regardless of the media used*. This makes intuitive sense given that coordinate phrases are a common part of speech, and, therefore, English speakers are used to parsing them. For example, the sentence "I would like to edit the auto check-in *and* have the confirmation emailed *and* print it from Kiosk at airport." has three coordinate phrases linked by "*and*". It may feel "wordy" but it is still easy enough to read and comprehend.

For the remaining features, a statistically significant T indicates that the incidence of a high value for the complexity feature may affect the user's comprehension and cause a restatement. Looking at each class individually, it does appear that some classes are more sensitive to response complexity than others. **C1a** (text only), for instance, has six significant positive features and five significant negative features, using $p \leq 0.05$. This implies that the designers of text-only responses need to pay attention to the mean length of clauses, sentences, and T-units, the number of complex nominals per clause and T-unit, and the number of verb phrases per T-unit (**MLC**, **MLS**, **MLT**, **CN/C**, **CN/T**, and **VP/T**). Compare this to **C2b** (text *and* links) where designers of responses only need to minimize the mean length of a clause and number of verb phrases per T-unit. **C1b** (web content only) and **C1c** (links only) appear to be insensitive to response complexity; only 1 T appears in **C1b**'s column whereas 2 T are in **C1c**'s column. This is not entirely surprising given that the text of both classes just directs the user to a different medium.

The complexity features **MLC**, **MLS**, **CN/C**, and **CN/T** (mean length of clause and sentence, number of complex nominals per clause and T-unit) for class **C3** (all three media) have a negative impact on helpful responses leading to restatement. Thus, designers of IVA responses will need to consider these features when all three forms of media are included. In addition, both Tables 9.10 and 9.8 indicate that the mean length of clause is one that should be minimized in general; the most negative coefficients in Table 9.10 occur in **MLC**'s row, and there are three extremely significant T in Table 9.8 for **MLC**'s row also. A similar trend is observed with mean length of sentence and number of complex nominals per clause (**MLS** and **CN/C**) but to a lesser degree.

Complex nominals in responses may be confusing to users because they can over-complicate what was being asked for. Suppose a user asks what to transport human ashes in, and the IVA responds with *"Human remains must be sealed in a well-marked small clear airtight box."* The complex nominal "well-marked small clear airtight box" requires the reader to parse several attributes and retain them in memory before reaching the noun they are describing, and then refer back to them to understand the requirements of the box. Consider instead the following rewording that removes the complex nominal: *"Human remains must be sealed in a box that is small, clear, airtight, and well-marked."* In this response it is clear to the user immediately that the ashes must go in a box and not a bag or other container, and the following adjectives narrow down what type of a box is required. This is much easier for a reader or listener to mentally parse and retain. Larger values of **CN/C** and **CN/T** mean more of these complex nominals appear in one clause or T-unit, so users must do the mental exercise of gathering attributes before knowing the subject multiple times within the same clause leading to possible confusion.

Interestingly, the total length of response does not appear to be a factor. In row **AW** (average number of words) of Table 9.8, we see that two classes with a text response (**C2a, b** meaning text with web content and text with links, respectively) have a large difference in average response lengths, yet they have nearly identical impact on restatement in general and acceptable restatements, in specific. However, when we compare **C2b** and **C3** (all three media) which have nearly identical average response lengths, we see that **C3** is slightly more likely to lead to a restatement, and, much more likely, any following restatement will be of a negative type. Practically, sensitivity to **MLC** and **MLS** with no sensitivity to **AW** indicates that users appear to favor text responses with shorter sentences and clauses regardless of the overall length of the response.

At this point, we can work through an example demonstrating how a dialog designer can reduce the complexity features to craft better responses. One of the more complex IVA responses we found dealt with carry-on baggage dimensions. If a user asked about measuring baggage they were presented with: *"We measure the size of bags using units of linear inches or linear centimeters. The linear dimensions of a bag are found by adding the thickest or widest part of the length, width and height together."* The distance between "adding" and "together" requires the reader to retain everything in between before they discover what they are to add the intermediate parts to. By rephrasing the same information into a simpler form such as *"We measure the size of bags by adding together the thickest or widest part of the length, width and height. Linear inches or centimeters may be used."* we reduce **MLS** from 18 to 13.5, **MLC** from 36 to 27, **CN/C** from 6 to 3, and **CN/T** from 3 to 1.5 all while conveying the same information. It takes more mental effort to store and join attributes of nouns across long clauses and complex nominals, therefore by minimizing those complexity features, the response text is easier to comprehend. If dialog designers follow the same exercise when crafting response texts, they may reduce understanding errors on the part of the users. The authors of LS2CA even provide an online tool[1] for comparing the complexity of two texts side by side.

9.5.2 Response Media

To further investigate the effects of response media classes on the next turn being a restatement, we compare the classes directly, controlling for complexity features.

[1] `http://aihaiyang.com/software/l2sca`

In Table 9.9, we see that **C1b** (web content only) and **C1c** (links only) have small coefficients implying that the chance of restatement in the next turn is lower for **C1b** and **C1c**. Interestingly, when two media (**C2a, b, c**) are considered instead of one, the coefficient markedly increases, but when *all* three media are included (**C3**), the coefficient drops. In addition, text with links (**C2b**) has slightly lower coefficients than text with web content (**C2a**) or links with web content (**C2c**).

Although our dependent variable is the presence of restatements in Table 9.9, not all user restatements are necessarily negative. As mentioned in Section 9.3.2, if the restatement is a clarifiation or a narrowing of scope of the user's task, the previous response may actually be helpful to a user. We want to be more precise in our analysis and determine the effect of response media class on unwanted restatements while also controlling for complexity (Table 9.10). For example, although **MLC** (mean length of clause) for class **C3** (all three media) may only increase chances for restatement somewhat (coefficient of .2161 in Table 9.9), when restatement *does* occur, it tends to be the unwanted kind (coefficient of $-.5283$ in Table 9.10). Compare that to **C2c** (web content and links) which appears to have the highest chance of restatement due to the presence of relatively strong positive coefficients in its column in Table 9.9. However, when **C2c**'s restatements occur, they tend to be the clarifying kind (see **C2c**'s column in Table 9.10).

The first column in Table 9.10, **C1b** (web content only), is completely dominated with negative coefficients; directing users to a web page without including any helpful links or text addressing the user's intent has a considerably higher chance of unwanted restatements in the next turn compared to all other response classes. Compare that to the case where only links are provided: **C1c**'s column consists of relatively strong positive coefficients. However, it is important to note that in these situations where a link is provided and nothing else, the user has the option to either click on a link or type text. Thus, some bias may be present due to path of least resistance, but as the coefficients are relatively high, providing links are more likely to help a user determine his or her true intention. For two media, **C2a** (text with web content) appears more helpful than **C2b** (text with links) which, in turn, appears more helpful than **C2c** (web content with links) based on logistic regression coefficients. This analysis, for the most part, agrees with the probabilities seen in Table 9.7.

To summarize, users are generally initiating contact with an IVA to help them navigate and digest a large website or to perform a task. To respond with only web content (**C1b**) is to reduce the IVA to a search engine displaying the top result. This may cause frustration in users expecting a tailored response and lead to restatements. Unless users specifically ask for a web page, they generally would expect the IVA to perform more than just search functionality (particularly since large websites typically already have a search functionality). Dialog designers should craft a response with only web content in cases where users have explicitly requested a web page. In all other cases, designers should take care to address the users request or at the very least provide alternative links (**C2c**).

9.6 Examples of IVA Response Classes

The following are examples of the seven response media combinations taken from various publicly visible IVAs on commercial websites.

Single Response Medium

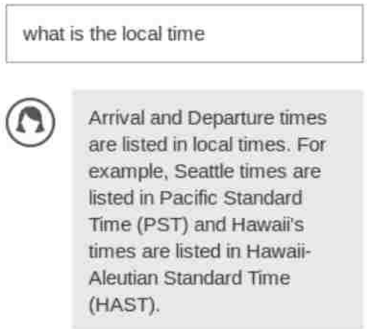


Figure 9.4: Example of a **C1a** response using only the IVA's primary medium

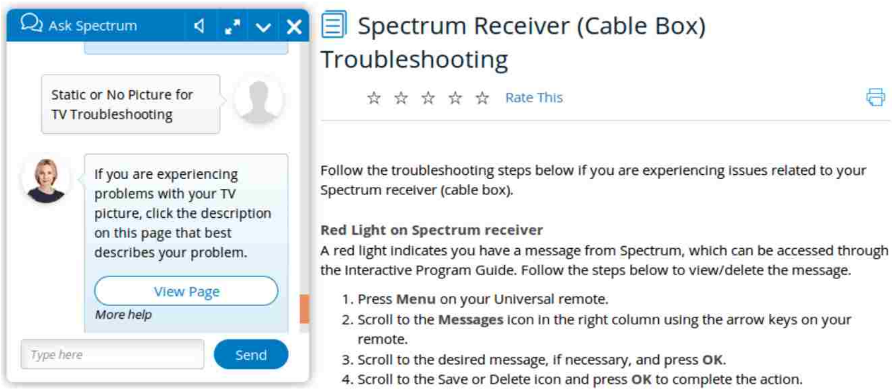


Figure 9.5: Example of a **C1b** response from an IVA. Notice that although the IVA provides a response, it does not address the user query; instead, it directs the user to the web content that the IVA pushed to their screen.

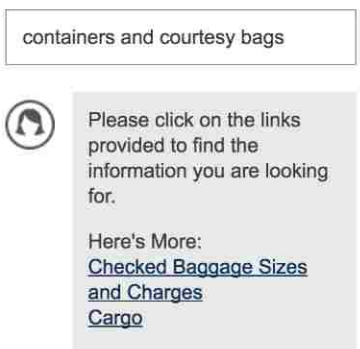


Figure 9.6: Example of a **C1c** response directing the user to choose from related or clarifying topics

Two Response Media

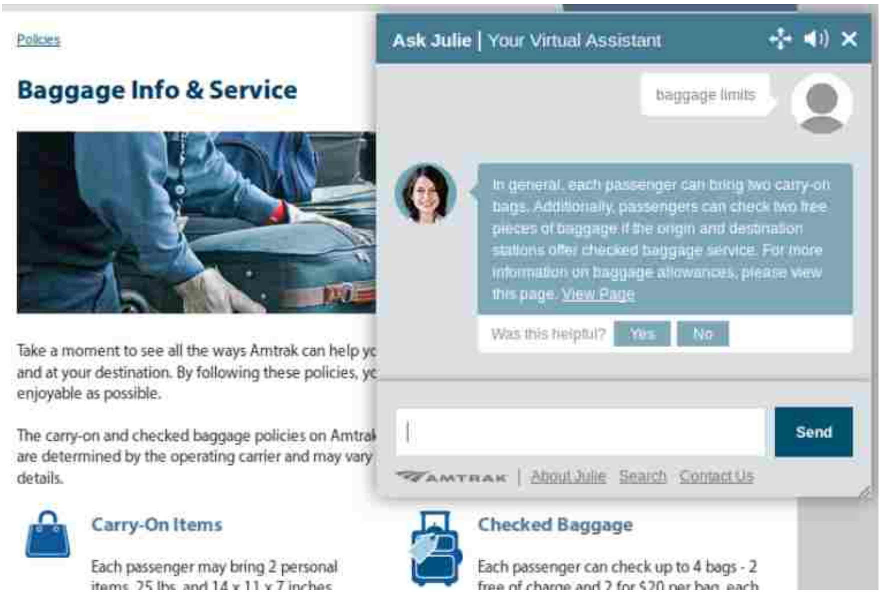


Figure 9.7: Example of a **C2a** response from an IVA. The response contains both an answer from the IVA and web content for further information on the topic.

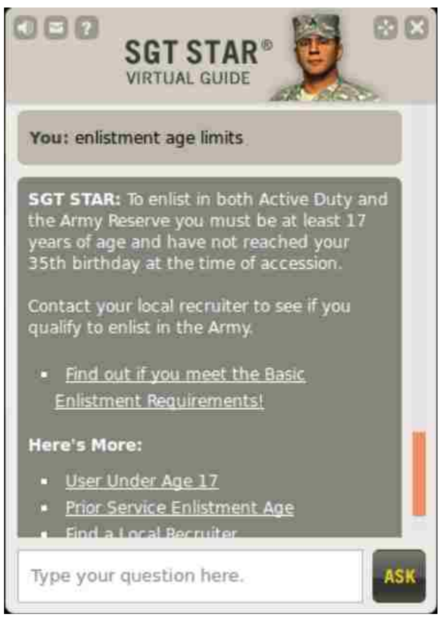


Figure 9.8: Example of a **C2b** response from an IVA containing both an answer and related or clarifying topic links

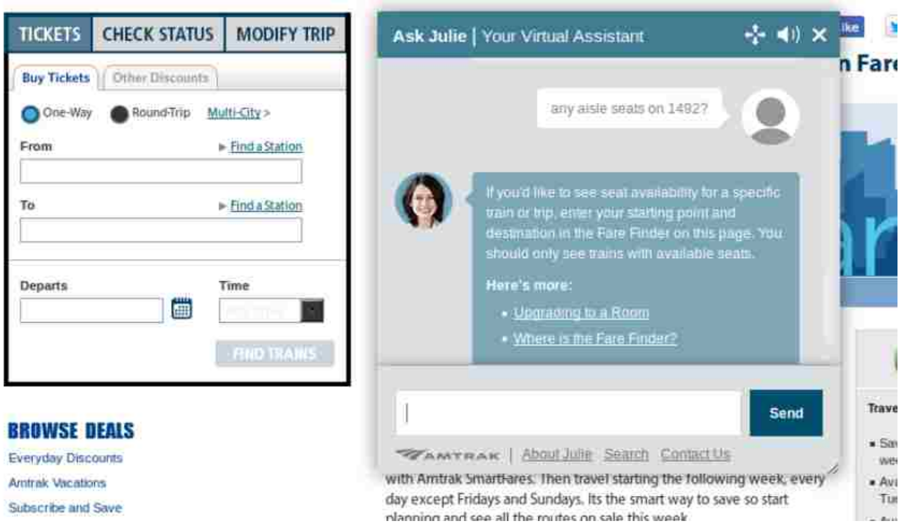


Figure 9.9: Example of a **C2c** response from an IVA. Notice that the IVA does not directly answer the query but instead directs the user to web content it has pushed to their screen as well as providing related topic links.

Three Response Media

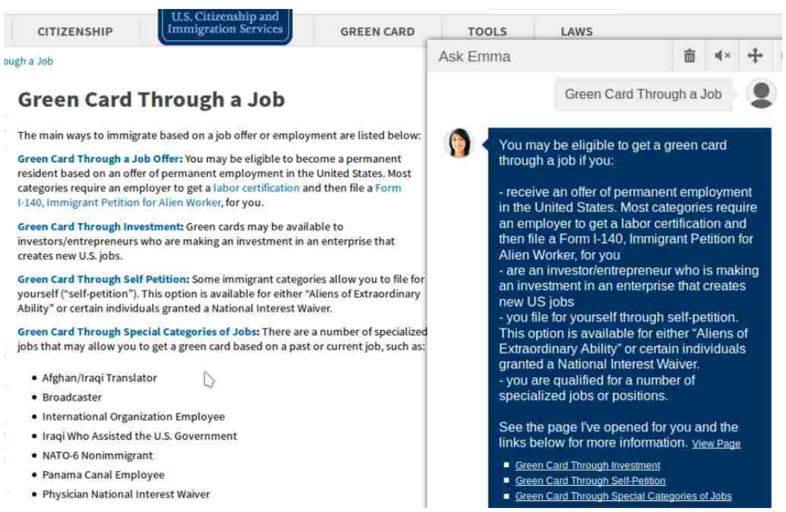


Figure 9.10: Example of a **C3** response from an IVA which involves all three media

9.7 Conclusion

As we have shown in our discussion, when designing multimodal IVAs for customer service, it is not as simple as avoiding long responses and displaying supplementary web content to ensure user comprehension. Although providing web content may be helpful, web content alone without related links or text addressing the user's intention may increase the number of non-clarifying user restatements. Web content may be helpful for additional or detailed information on a topic, but users appear to prefer the IVA to directly address their query.

While a designer might naively believe that including all three forms of media may help a user resolve his or her issue, if the complexity features for **MLC**, **MLS**, **CN/C**, or **CN/T** (mean length of clause and sentence, number of complex nominals per clause and T-unit) are high, this may not be the case. Therefore, once the media of response are chosen, it is necessary to perform complexity analysis on the response text and minimize the use of features that correlate to non-clarifying restatements. This process is demonstrated at the end of Section 9.5.1.

Designers should consider that users appear to favor text responses with shorter sentences and clauses regardless of the overall length of the response and the other media involved. To resolve ties between equally likely intentions, displaying topic suggestions in the form of links is a good strategy. This type of response appeared more helpful to users than showing them a web page related to their query and instructing them to consult it.

9.8 Application of Response Complexity to Risk Analysis

This chapter has presented various response features with statistical correlations to user restatements when there was no apparent misunderstanding on the part of the IVA. As these features may cause confusion or dissatisfaction on the part of the user, we want to take them into consideration when identifying risk of missed intent on the part of the IVA.

9.8.1 Automated Response Classification

In order to identify class of a given IVA response, we train a classifier on the human annotated data described in Section 9.3.2. This classifier is needed to determine whether or not the IVA attempts to address the user query in its response text. This determination is used to separate response classes **C3** from **C2c**, **C2b** from **C1c**, and **C2a** from **C1b**. In each case the only difference between each pair of classes is if the IVA attempts to answer the user in addition to other media or if the IVA just refers them to the additional media. Refer to Table 9.3 for the specific distinction between these classes.

Using the output of the L2 Syntactic Complexity Analyzer (L2SCA) [1] along with several words we found empirically that appeared more frequently when the IVA does not attempt to answer the query, we select the 10 features most positively and negatively correlated to the response classes of **C2a**, **C2b**, and **C3** (those classes where the IVA answered the users query in the response text). The positively correlated features were: number of words, number of sentences, number of chunks, number of T-units, number of verb phrases, number of coordinate phrases, and number of clauses. The negatively correlated features were: presence of word "page", presence of the word "information", and presence of word "form".

Using these 10 features and the human labels for **C2a**, **C2b**, and **C3** as positive samples and **C1b**, **C1c**, and **C2c** as negative samples we constructed training data for a Random Forest classifier. To train and select the best configuration of model parameters, we performed a grid search over the model parameter values using 10-fold cross validation (CV) to optimize on the CV average accuracy. The highest CV accuracy of 84.2% was reached using 22 trees in the forest, information gain to measure split quality, and 9 for the maximum depth of a tree. This classifier is then used within an algorithm which takes in the IVA response text, any additional topic links, and the URL of any web page content shown to users.

```
def LabelResponseClass(iva_response):
    Data: An IVA response tuple (text, topicLinks, URL)
    Result: The class of IVA response
    if topicLinks and URL:
        /* It is a C2c or C3 */
        if response_clf.predict(text) == 1:
            return C3;
        else:
            return C2c;
    elif topicLinks:
        /* It is a C1c or C2b */
        if response_clf.predict(text) == 1:
            return C2b;
        else:
            return C1c;
    elif URL:
        /* It is a C1b or C2a */
        if response_clf.predict(text) == 1:
            return C2a;
        else:
            return C1b;
    else:
        return C1a;
```

Figure 9.11: Categorize IVA response class

In Figure 9.11 the algorithm that determines the IVA response class is shown. Using the presence of related topic links and a URL to web content along with the output of the binary response classifier previously described, this algorithm determines which of the seven classes a particular IVA response belongs to.

9.8.2 Response Class Risk Indicators

Recall that the purpose of IVA response analysis is to find any features that may indicate it is the user *not* the IVA that is misunderstanding at a given point in a conversation. Using the correlations in Table 9.10 we can see that IVA responses that fall under class **C1b** (web content only) are more likely to lead to a non-clarifying restatement than any other class. Therefore, we want to decrease the risk of missed intent following an IVA response in class **C1b**, as it may actually be the *presentation* and not the *understanding* at fault in the IVA. The purpose of the CRS is to evaluate the Natural Language Understanding (NLU) component, in isolation. Thus, we must separate errors in presentation from those in understanding. The CRS will apply the indicator **response_class** to any input following a response that has been classified as **C1b** by the algorithm in Figure 9.11. Later, when combining indicators into a singular risk score (see Section 4.4.4), this indicator carries negative weight.

The second indicator of user dissatisfaction in IVA responses is the complexity of the IVA response text. From Table 9.10 above, we can see there are several cases where higher values of a response complexity feature combined with a particular response class lead to non-clarifying restatements. For example, higher **CN/C** (complex nominals per clause) with a **C3** (all three media) response is negatively associated with a clarifying restatement, but when the same complexity feature is higher with a **C1c** response, it is positively associated. Therefore, these different combinations of complexity and response class are considered.

Using Table 9.10, any cell where the coefficient is less than -0.2 is used as an indication that the combination of text complexity and response class can lead to a non-clarifying restatement by the user. For each such combination, like (high

CN/C, **C3**), the CRS applies the indicator **response_complexity**. Similar to **response_class**, when combining indicators into a singular risk score this indicator carries negative weight. In order to determine when a value for a particular complexity feature is “high”, we calculate the quartiles for the values of each complexity feature in the human labeled dataset described in Section 9.3.2. If the value for a particular complexity feature is above the upper quartile boundary, we consider its value “high”. Recall that the upper quartile boundary is the middle value between the median and the highest observed value in the data set.

We have now covered the implementation details of the majority of risk indicators outlined in Section 4.4. The next chapter describes how the remaining indicators are detected in the risk analysis process. Following this is an evaluation of how the CRS performs both in prioritizing conversations for review by risk, and voting in place of human reviewers.

Chapter 10

Remaining Indicators of Risk

In the last few chapters have discussed several of the more complex risk indicators used by the CRS to determine if a user turn was likely misunderstood by the IVA. In this chapter we cover the remaining indicators. Many of the following indicators were used in a prototype of the CRS, published in [110].

One indicator commonly used in call centers to evaluate customer service agent performance is their call abandonment rate [201]. This metric is measured by how often a customer hangs up or closes an online chat before completing their current task. The rationale is that a customer would not go through the trouble of contacting a support center, begin a task, and then quit partway through unless they were not satisfied with their treatment by the customer service agent. Given this rationale, we also use this metric to evaluate the IVA performance by looking for conversations where the user abandoned the task as an indicator of missed intent.

There are two ways to determine that the user abandoned a task with an IVA. We can either rely upon the IVA to tell us, or we can attempt to infer it from the IVAs final response. The CRS supports either method but will prefer the IVA indicate an abandonment occurred in the conversation log. Recall from Chapter 2 that IVAs are implemented as dialog systems and therefore keep track of the current conversation state, and part of that state is the current task they are attempting to perform on the part of the user. If an IVA is attempting to gather information from the user to perform a task, and the conversation ends before enough information has been acquired to perform the task, the IVA can flag the conversation as an abandonment for reporting purposes.

If, however, a particular IVA under review does not perform this flagging, the CRS attempts to detect abandonment using a simple heuristic. Using the text from the final IVA response in the conversation, it first tests if the IVA posed a question to the user, ignoring generic ending sentences such as "Is there anything else I can help you with?" If so, it considers the conversation to have been abandoned as the IVA was still trying to gather information from the user when it ended. In this case the CRS applies the risk indicator **abandonment** to the last user turn in the conversation.

If the conversation did not end on a question, it next checks if the grammatical mood of the final IVA response was imperative. Imperative mood is typically used for ordering or requesting the listener to do something [196]. Thus, the IVA using imperative mood in its response indicates it is asking the user to do something. If IVA response is not a question, the mood is imperative, and the response text contains references to typing, input, giving, or telling, as determined by regular expressions; the CRS applies the risk indicator **abandonment** to the last user turn in the conversation. Responses that meet these conditions typically look something like "Type the last four digits of your SSN below." or "Please tell me the date of your claim." If a user abandons a conversation on a similar response, it may indicate that the IVA was trying to perform the *wrong* task and the user left in frustration.

10.1 Turn Content

Several remaining risk indicators are based on detecting particular content of the user turn. For example, if a user requests something using vocabulary unknown to the IVA, it may indicate the user is talking about a subject the IVA has not been trained for. The use of such unknown words may cause a misunderstanding on the part of the IVA as it drops the unknown words while performing intent classification.

For example, if an IVA exists on a telecommunications website and a user says to it "Cheers, mate!", thanking the IVA using common British slang, if "mate" is not in its vocabulary it will ignore it and perform intent recognition solely on the word "Cheers". In the television domain this word is the name of a very popular television show from the 1980s, and reruns of the show are still played at the time of this writing. The IVA may misunderstand that the user is asking about the show and respond with a page or reference to the current schedule or channel the show is playing on.

As the CRS has access to the training data used to train the intent classifier, or the grammars or patterns within the model itself if it is a symbolic model, it knows the entire vocabulary of the IVA. When a user turn contains words that do not appear in the IVA's vocabulary, the CRS applies the risk indicator **unknown_words** to that turn.

Other content of interest is the use of profanity by the user. Profanity use by the customer is treated as an indication of poor customer service in quality assurance systems [202, 203]. While this may arise in response to business rules, such as not allowing the customer to transfer a plane ticket to another persons name, it may also be used as an expression of frustration in response to a perceived inability for the service agent to understand the user's request [203]. Therefore, the CRS detects swearing and vulgarities through 3,900+ regular expressions periodically generated from several term lists on Wiktionary[1]. Words with common innocuous meanings are removed. If a user turn matches such an expression, the risk indicator **precedes_profanity** is applied to the preceding user turn in the conversation.

A related indicator is when a user comments on the unhelpfulness or futility of the IVA. To detect this indicator the CRS leans on the originating IVA and the internal agreement classifiers. If any of these intent classifiers maps the user turn to an intent representing unhelpfulness, the CRS applies the risk indicator **precedes_unhelpful** to the preceding user turn in the conversation.

10.1.1 Sentiment

An important feature in analyzing customer satisfaction in reviews and feedback is sentiment polarity [204, 205, 206]. The CRS considers sentiment in two ways. The first is any user turn preceding a turn containing negative sentiment polarity as determined by a sentiment classifier is assigned the risk indicator **precedes_neg**. Negative comments by the user may be in response to IVA misunderstanding and is therefore considered as an indicator of risk.

The second way sentiment is used for risk analysis is by considering how sentiment changes over the course of a conversation. If the user sentiment is positive or neutral in the beginning, but by the end of the conversation has become negative, this is a conversation level indication that misunderstanding may be present. To determine the change in sentiment, the CRS performs a least squares one degree polynomial fit over the sentiment polarity values from the user turns in a conversation. It then measures the slope of this line to determine if sentiment is decreasing over the course of the conversation. If the slope of the least squares line is negative, the CRS assigns the risk indicator **sent_change** to all of the turns in the conversation. Essentially,

[1] Specifically the derogatory, offensive, and vulgarities lists on `https://en.wiktionary.org/wiki/Category:English_terms_by_usage`

the fact that sentiment is degrading over the course of the conversation can mean that multiple communication problems have been encountered and therefore the IVAs handling of the entire conversation is suspect.

To determine sentiment polarity, the CRS uses a sentiment classifier trained from a combination of the IMDB movie review dataset from [207] and Twitter data from the SemEval-2015 Task 11 dataset[2]. The user turns are converted to word embeddings using the Word2Vec [208] embeddings from the 3 million word Google News dataset[3]. The resulting variable length list of word vectors from the user turn are padded or truncated to length 350, then fed into a Recurrent Neural Network consisting of an input layer followed by a single hidden 200-neuron Long Short-term Memory (LSTM) layer [209]. This is followed by a dropout layer for regularization and finally a dense output layer using a sigmoid function for polarity classification. The performance of this network on a hold out test set of 10,000 mixed samples from the two sources was 90% accuracy.

10.2 I Don't Know (IDK) Response Risk Indicators

A subcategory of risk indicators applies to what we call "I Don't Know" (IDK) responses. These occur when the language model does not find an intent that satisfies the user input. An IDK intent may return a response such as "I'm sorry, I don't understand you. Please revise your question.". An example of matching this intent is seen at turn 6 in Table 10.1, which is repeated from Section 4.4 for convenience. Therefore all user turns in that conversation would be tagged with **idk_in_conv**, turn 6 would be additionally tagged with **triggers_idk**, and turn 5 with **precedes_idk**.

[2] `http://alt.qcri.org/semeval2015/task11`
[3] `https://code.google.com/archive/p/word2vec`

Chat Text	**Intent Hit**	**Conversation ID**	**Turn #**
My TV is not working I need to have it fixed.	TV Support	26789	1
My TV will not start I want to talk with support staff	Contact Information Deflection	26789	2
I cannot get my tv to work	TV Support	26789	3
How can I get a person to help me get the TV started?	TV Services	26789	4
How can I speak with a support staff to get my TV to work?	TV Support	26789	5
There are no links showing my problem what now?	I Don't Know	26789	6

Table 10.1: Conversation with Risky Inputs

A language model may also contain intents that are used as "intelligent" IDKs. An example response to one of these may be: "I see that you are asking about liability insurance, but I do not have detailed knowledge in that area." These indicate to the user that the IVA understood the topic of their request, but does not possess the specific knowledge necessary to answer it fully.

Another type of IDK occurs when the same intent is hit more than two successive times within a conversation. This is an *impasse*, since it is clear we cannot give the user a satisfactory response. It may indicate that the language model mapping associated to the intent involved is too broad and therefore the intent is being selected when it should not be, or the IVA is missing domain knowledge. At the user turn in a conversation where an impasse is reached, the CRS applies the risk indicator **triggers_impasse**.

10.3 Repetition in Conversation as Risk Indicators

A second subcategory applies to multiple hits, where the same intent was returned multiple times within a conversation. This is an indication of risk within a customer service conversation as it is unlikely the user would want to see a specific response more than once. An example of this are turns 1, 3, and 5 in Table 10.1. All user turns in this conversation would therefore be tagged with **multi_in_conv**.

If two hits are successive, we label the successive user turns with the indicator **triggers_seq**, indicating that they generated a sequential hit to the same intent. This usually indicates that the response to the first input did not satisfy the user so they are rewording their questions to get a different response. In addition, all turns in the conversation would be labeled with **seq_in_conv**, indicating it is possible the topic of the entire conversation is outside the IVA's current knowledge.

10.4 External Ratings as Risk Indicators

A third category is derived from ratings given by the human users themselves. Typically, customer service interactions present surveys on user satisfaction after a conversation is complete. Popular rating methods include Customer Satisfaction Score (CSAT), Net Promoter Score (NPS), and Customer Effort Score (CES) [210]. All such methods give users a scale to rate their experience on such as one to five stars or 0-10.

If these ratings are present, they can be used as an indicator of risk. If the user reported he/she was unhappy with the conversation, it may indicate the intents present within that conversation need improvement. The ratings are normalized into a range 1-5 where 1 would represent a poor response and 5 would represent an excellent one. Each rating score is given its own risk indicator in order to weigh them individually. For example, if the user from Table 10.1 gave their experience a 1 out of 5 rating, the CRS would assign every turn in that conversation the risk indicator **conv_rating_1**. As discussed in Section 4.4, these ratings do not always indicate misunderstanding occurred. The user may give negative feedback if business rules prevent the user from doing something he or she wanted. The user may also say the IVA was unhelpful when the NLU was indeed working correctly, but the response text was poorly worded.

Some live chat and IVA implementations take this feedback a step further and give the user the option of rating every response. In the chat window there may be a response shown from the agent, followed by a drop down box or scale to click on that allows the user to quickly rate the quality of that response. In the cases where the IVA has this functionality, these can be used as an indicator of per-turn risk. These ratings are also normalized into a range 1-5 where 1 would represent a poor response and 5 would represent an excellent one. Like conversation ratings, each turn rating score is given its own risk indicator in order to weigh them individually. For example, if the user from Table 10.1 gave the first turn a 2 out of 5 rating, the CRS would assign the first turn the risk indicator **user_rating_2**. Similar to conversation ratings, user ratings may not always reflect the understanding ability of the IVA so it must be weighted along with other indicators present.

10.5 Tying Intents

The final category of risk indicators used are those that relate to two or more intents tying during intent determination. Ties can occur in an intent classifier if two intent share similar language, such as initial seat selection on a plane and changing an existing seat selection. Both intents will share similar classification scores given a user turn such as “I need help with my seat selection.” Such an input could mean the user has a seat and needs help modifying it, or it could mean the user does not yet have a seat and needs help selecting one. When multiple intents share similar classification scores, the CRS will assign the risk indicator **triggers_tie** to the user turn that generated the tie. In addition, the CRS will add the risk indicator **tie_in_conv** to all turns in that conversation as the presence of ties may indicate confusion in the language model around the topic of the conversation.

Chapter 11

Evaluation

We have now covered how all of the indicators of the risk of missed intent are detected and applied to both user turns and conversations under review. In this chapter, we perform several evaluations on the performance of the CRS as a whole, and how it compares to human reviewers.

11.1 Data

In order to evaluate the performance of the CRS on various tasks, we first constructed a gold standard corpus to use in experiments. As the CRS must work well regardless of domain, we constructed three datasets, each from a different language domain. Due to annotation budget, we limited our average user turns per dataset for evaluation samples to 8,000.

All turns in a conversation need to be reviewed, however conversations have varying numbers of turns and, with multi-modal IVAs, not all user turns consist of natural language. For example, some user turns in a conversation may be events such as user interface clicks or web page navigations which the IVA responds to.

Dataset	# Conversations	Total User Turns	Natural Language User Turns	Majority Agreement
Train	2,030	13,930	7,270	6,331
Telecommunications	1,342	20,485	7,313	5,252
Airline	1,611	9,103	9,103	6,978
Average	1,661	14,506	7,895	6,187

Table 11.1: Dataset statistics for the evaluation data.

Therefore, collecting a truly random sample of conversations to meet the total turn count but still balanced by dataset was somewhat challenging. Using the average natural language turns per conversation we estimated the sample size per domain. We then selected a random sample of full conversations, using the estimated sample size per domain, from the conversation logs of a live virtual agent in each of three domains.

All natural language turns were selected for voting and released to a group of 14 voters. Three votes per turn was required to control for subjectivity. Voters were all Next IT - Verint employees who were trained on the CRS user interface and voting process prior to actually voting. Employees were used for voting as the data contained personally identifiable information, and therefore external annotation services that provided Non-Disclosure Agreement coverage were not affordable. After voting, the average number of turns per dataset with a clear majority (agree or disagree with the intent chosen by the live IVA) was $6,187$. If there was no clear majority, the turn was not used as an evaluation sample.

As the human reviewers voted on the user turns, the system logged the time required to review each turn and make a determination. The average overall time to receive a user turn and place a vote was 11.12 seconds.

Evaluation dataset statistics are given in Table 11.1. Total User Turns involve all forms of user input including clicking on controls and web page navigation events.

Natural Language User Turns are only those that were processed by the Natural Language Understanding (NLU) component for intent classification. As the CRS is only interested in the discovery of error in the NLU, it is these user turns that are evaluated by humans. Majority Agreement are the number of Natural Language User Turns where a majority of the three voters agreed on the vote.

From these counts we can see that the Telecommunications IVA is very interactive, less than half of user turns are actually in the form of natural language. This IVA responds to many user activities besides typed or spoken input. In contrast, the Airline IVA does not accept anything but typed or spoken input. The Train IVA appears a good balance of the two interaction styles. The Train IVA had the highest level of overall voter agreement, at 87%. The Airline had less at 76.7% followed by the Telecommunications IVA with 71.8% agreement. Inspecting the conversations and IVA knowledge bases, it appears this is due to the complexity of the IVA and the number of intents understood. The Train IVA has 930 distinct intents in its knowledge base, compared to 1,223 for the Airline IVA and 2,173 for the Telecommunications IVA. Not surprisingly, the increase in possible intents to select from appears to decrease voter agreement on the correctness of an intent chosen by the IVA.

11.2 Potential New Intent Suggestions

The first task evaluated was the suggestion of Potential New Intents (PNIs) by the agreement classifiers. Recall from Section 5.1.2 human reviewers or domain experts need to suggest alternative intents when disagreeing with the intent selected by the IVA. Essentially, once an error in the language model is identified, it cannot be fully corrected until the proper intent of the erroneous user turn is known. Then the domain experts can adjust the language model to map the turn involved from the former intent to the proper intent.

The CRS attempts this behavior by using the same agreement classifiers used to generate the **external_clf** and pni_origin risk indicators to predict the proper intent when they confidently disagree with the IVA. Multiple agreement classifiers are supported and any classification method can be used, assuming it is different than that of the method used by the NLU. Review Sections 4.4.3 and 5.1.2 for further background.

In the particular implementation of the CRS under evaluation, we used a single agreement classifier as we had limited human voter resources for validating the suggestions. As our problem involves text classification, we selected Scikit Learn's[1] Support Vector Classifier (SVC) with a linear kernel which has been shown to perform well at this task [133, 134, 135]. It implements a "one-vs-rest" multi-class strategy. To generate a confidence metric, we used SciKit Learn's *decision_function* method which returns the signed distance of a sample to the hyperplane. Using this distance d, we calculate the probability P of class membership. We used the following estimation method since datasets may have input counts in the millions and calculations must be performed on each input:

$$P = \frac{d}{2} + .5 \tag{11.1}$$

As we need an estimation method that will perform at a consistent and efficient speed with a large number inputs, and we do not require a high degree of precision, we used the estimation technique in (11.1) over other techniques such as Platt scaling [211]. Platt scaling has been shown to be an expensive operation on large datasets [212]. Note that if d does not satisfy

$$\epsilon \leq \frac{d}{2} + .5 \leq 1 - \epsilon \tag{11.2}$$

[1] `http://scikit-learn.org`

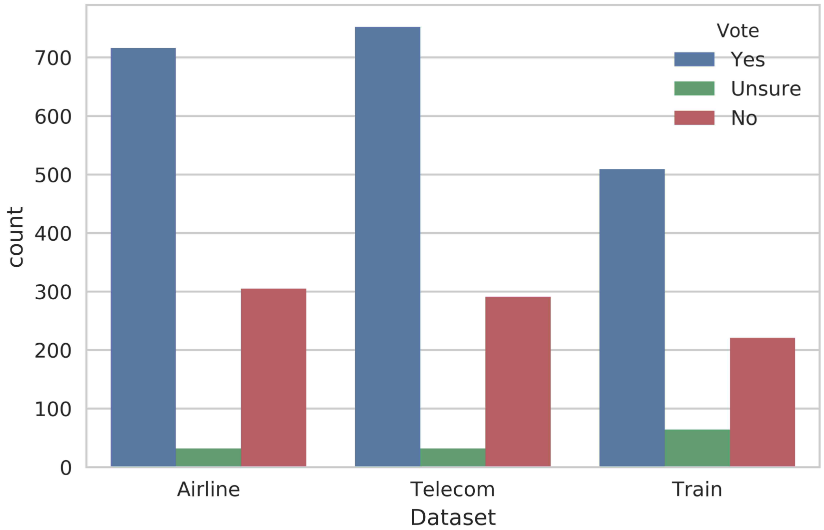


Figure 11.1: Voter agreement with CRS PNI suggestions

where $\epsilon = .0001$ in our case, then P will take on the value ϵ or $1 - \epsilon$, whichever is closer. We used P_i of the predicted intent i for a user turn as the confidence of the SVC in i.

For each user turn where the agreement classifier disagreed with the IVA selected intent, if $P_i > 0.6$ the CRS will construct a PNI with the original user turn and the top intent i from the SVC. As the SVC is selecting between one thousand or more intents, the threshold 0.6 was chosen arbitrarily as a probability high enough to warrant consideration.

In Figure 11.1 the voter agreement with the PNI suggestions is shown. Human agreement over all three datasets is 67.7%, with 4.4% unsure. The Telecom dataset had the highest agreement at 70%, followed by Airline at 68%, and finally Train at 64.1%. While this performance is not high enough to fully replace humans in determining alternative intents, it does appear that the PNIs generated in each of the domains are acceptable to use as suggestions to domain experts. Even if an average of 32.3% suggestions are ultimately discarded by the domain experts, PNI generation prevents any human effort to find alternative intents over 2/3 of the time. Performance of different choices for probability thresholds and altogether different classification methods is left for future works.

11.3 Tuning Risk Indicator Weights

Recall from Section 4.4.4 that each risk indicator can be assigned a weight. Weights are initialized to 0.5 and can be tuned over time as voting data is added. The risk score for turn t, known as z_t, is defined as the sum of these weights:

$$z_t = \sum_{n=1}^{|N_t|} w_n \tag{11.3}$$

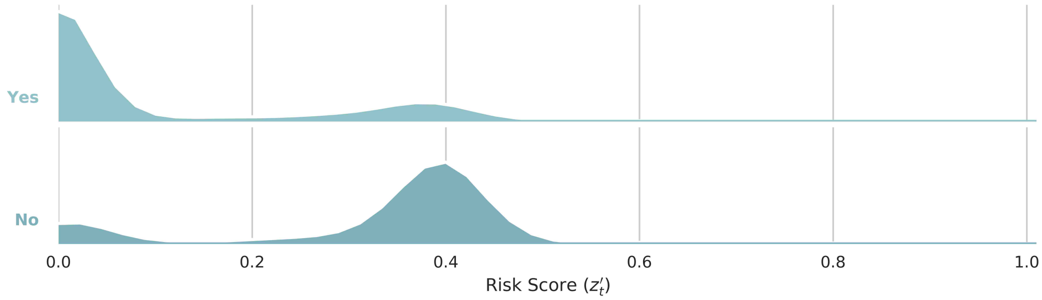
Since there is no upper bound to z_t, it is further normalized by the highest value of z_t observed in the dataset, $MaxScore$:

$$z'_t = \frac{z_t}{MaxScore} \tag{11.4}$$

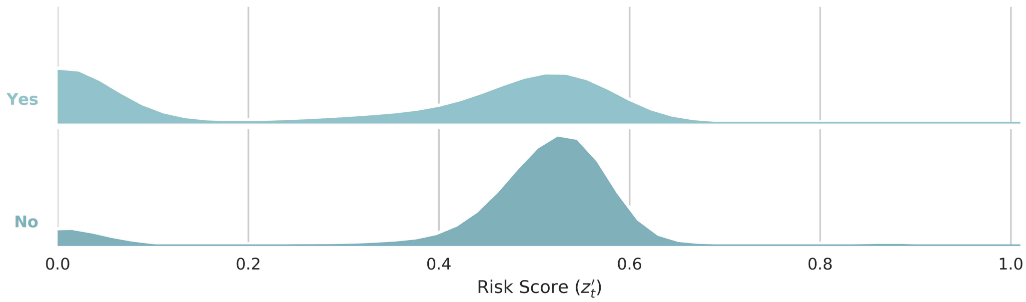
In the following section, we evaluated how well the risk score z'_t using equal weights correlates to vote outcomes. Then we explored how the weights can be optimized and measured the effect of such optimizations.

11.3.1 Equal Weight Scores

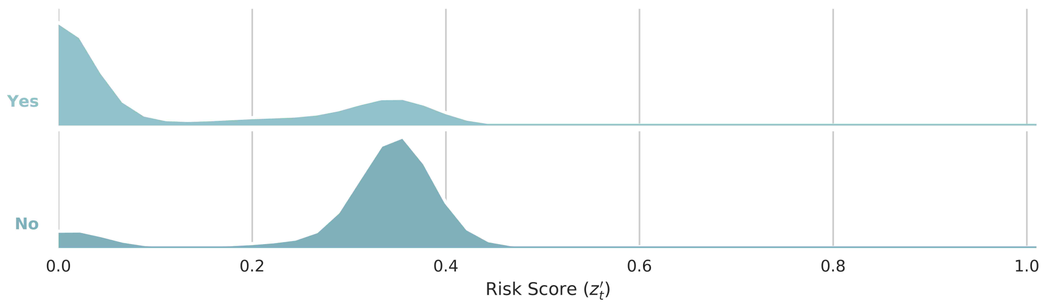
In Figure 11.2, the distribution of risk scores using equal risk indicator weights based on the majority vote is shown. Comparing the distributions between the datasets, we see that risk score is correlated with the majority vote, but it tends to have a bimodal shape. Either user turns have medium risk values or they have low risk values, there is very little in between. There is a long tail out to the riskiest user turn in every case. The Airline set in particular appears to be the best fit of risk score to voter outcome, whereas the Telecom set appears to be the worst fit. If one were to predict the majority vote based only on the risk score, we would expect the Airline and Train datasets to perform well and the Telecom dataset to perform poorly due to the large overlap of scores between the two voting outcomes.



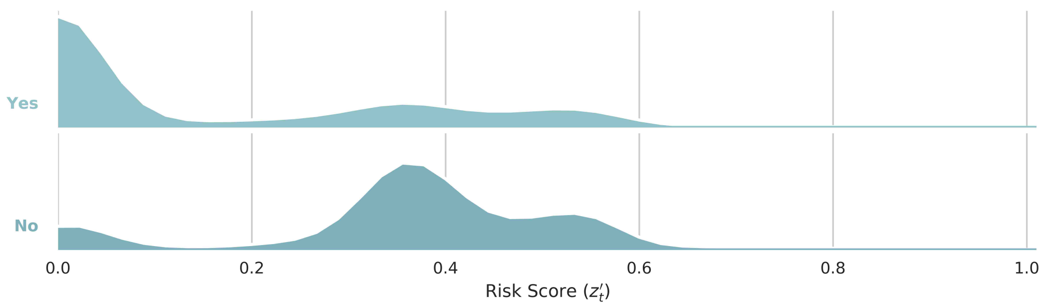
(a) Airline



(b) Telecom



(c) Train



(d) Overall

Figure 11.2: Distribution of risk score by majority vote for each dataset using equal risk indicator weights.

11.3.2 Generating Weights by Odds Ratios

We tune the weights of the risk indicators by dataset, as some datasets may have higher incidence of certain factors such as escalations or out of vocabulary words. To tune the weights, we used the presence or absence of risk indicators on the user turns with majority voting outcomes. First we ignored any risk indicators with less than 25 occurrences and set their weight to 0, as there are not enough samples to build statistics from. We then used Odds Ratios on the remaining indicators to determine which ones are correlated with a majority disagreement on the IVA-chosen intent.

Odds Ratios (ORs) are commonly used in many fields of social science and medical research to compare the impact of risk factors on a selected outcome [213, 214]. The odds ratio represents the odds that an outcome will occur given a particular exposure, compared to the odds of the outcome occurring in the absence of that exposure [215]. As odds are determined from probabilities, another way to think of them is the ratio of the probability of success and the probability of failure, given some probability p of an outcome [216].

$$odds(success) = \frac{p}{(1-p)} \tag{11.5}$$

$$odds(failure) = \frac{(1-p)}{p} \tag{11.6}$$

$$OR = \frac{odds(success)}{odds(failure)} = \frac{p^2}{1-2p+p^2} \tag{11.7}$$

From our voter evaluated datasets, odds ratios for every risk indicator were calculated. Statistically insignificant risk indicators were eliminated using a 95% confidence interval. Remaining indicators were then ranked by their ORs where a higher

OR indicates a larger magnitude of effect. For example, from (11.7), using success to mean the majority disagreed with the IVA-chosen intent, an OR of 3 would indicate that given the observance of a particular risk indicator the probability of success (discovered a misunderstanding) is three times that of failure (no misunderstanding). Finally, ORs were normalized between 0 and 1 to obtain weights for their respective indicators.

To calculate ORs, dichotomized exposures must be delimited. An input chat was *risky* if the majority of voters disagreed with the intent it was mapped to, otherwise it was labeled as *safe*.

$$\begin{array}{rc} & \begin{array}{cc} \text{Risky} & \text{Safe} \end{array} \\ \begin{array}{r} \text{Risk Indicator} \\ \text{No Risk Indicator} \end{array} & \begin{pmatrix} a & b \\ c & d \end{pmatrix} \end{array}$$

$$OR_{Risky} = \frac{a/c}{b/d} \tag{11.8}$$

The value of the OR indicates the effect the risk indicator has on the riskiness of the chat: an $OR \gneqq 1$ signifies that the indicator positively affects riskiness. If the 95% confidence interval of the OR includes 1, we deemed it a statistically insignificant result. Intuitively, using (11.7) an OR of 1 means that the odds of success is equal to the odds of failure, or $p = 0.5$. Therefore, a risk indicator with an OR of 1 is not useful for prediction of missed intent. If using (11.8), the lower and upper bounds for the 95% confidence interval can be calculated in the following manner [215] :

$$e^{ln(OR) \pm 1.96 * \sqrt{(\frac{1}{a} + \frac{1}{b} + \frac{1}{c} + \frac{1}{d})}} \tag{11.9}$$

If risk indicators are not independent, an adjusted OR can be calculated with logistic regression [217, 216, 218]. As we expected some indicators may have slight dependence, for example **tie_in_conv** and **triggers_tie**, we used the adjusted ORs from the logistic regression coefficients, as in [216]. To eliminate the effect of any risk indicators with high multicollinearity, we first calculated the Variance Inflation Factor (VIF) for each risk indicator using the Python statsmodels package[2]. The VIF is a means to determine if a variable has a significant impact on variance of the model, and therefore should be removed. It is a ratio of the variance in a model with multiple terms over the variance in a model with one term [219]. A VIF threshold greater than between 5 and 10 is commonly used to indicate high multicollinearity [220, 221]. Thus, we ignored any indicators with a VIF greater than 5 by setting their weight to 0. Finally, we calculated an adjusted OR using logistic regression on the remaining indicators with an intercept and taking the exponent of their coefficients [218]. Any indicators with 1 in the 95% confidence interval were set to 0. The normalized OR is used as the weight for the rest.

Risk Indicator	**Airline Count**	**Telecom Count**	**Train Count**	**Airline OR**	**Telecom OR**	**Train OR**
backstory	298	186	200	1.9±0.53	1.46±0.5	1.1±0.34
conv_rating_1	0	0	0	-	-	-
conv_rating_2	0	0	0	-	-	-
conv_rating_3	0	0	0	-	-	-
conv_rating_4	0	0	0	-	-	-
conv_rating_5	0	0	0	-	-	-
conv_should_esc	167	186	182	1.9±0.95	0.6±0.29	0.88±0.5
end_rating_1	0	0	0	-	-	-
end_rating_2	0	0	0	-	-	-
end_rating_3	0	0	0	-	-	-
end_rating_4	0	0	0	-	-	-
end_rating_5	0	0	0	-	-	-

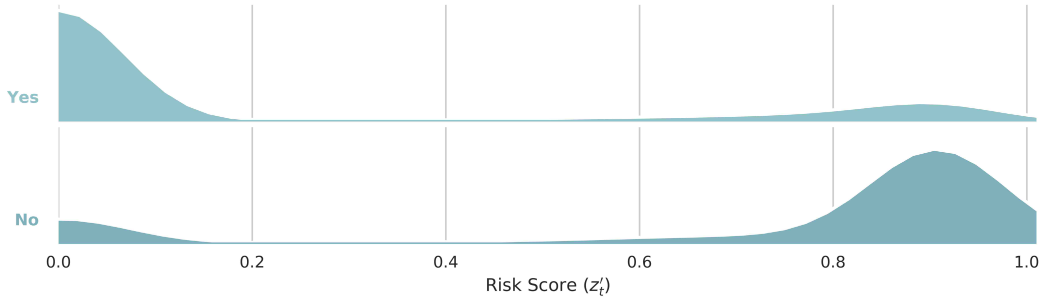
[2]`http://www.statsmodels.org/dev/generated/statsmodels.stats.outliers_influence.variance_inflation_factor.html`

ends_on_imperative	1	2	0	-	-	-
ends_on_q	3	7	5	-	-	-
external_clf	2,055	3,185	2,481	12.9±2.6	4.6±1.1	15.4±3.0
idk_in_conv	141	133	202	1.8±0.97	1.53±0.8	2.47±1.5
multi_in_conv	365	220	324	VIF: 5.1	0.46±0.3	VIF: 6.2
multi_intent	448	193	409	1.34±0.3	1.5±0.45	1.25±0.3
pni_origin	783	1,714	900	1.75±0.3	1.5±0.28	1.0±0.17
precedes_corr	4	5	2	-	-	-
precedes_esc	35	19	16	0.6±0.48	-	-
precedes_idk	21	23	64	-	-	1.38±0.8
precedes_neg	196	206	187	1.39±0.7	1.4±0.59	1.2±0.52
precedes_profanity	0	0	0	-	-	-
precedes_unhelpful	0	0	0	-	-	-
response_class	0	0	0	-	-	-
response_complexity	28	18	9	2.1±1.7	-	-
restated	79	43	49	1.4±0.95	0.54±0.4	0.99±0.6
sent_change	241	226	287	0.95±0.5	0.4±0.19	1.83±0.8
seq_in_conv	232	79	255	2.08±1.0	5.26±3.1	0.79±0.4
should_esc_point	21	24	11	-	-	-
tie_in_conv	318	324	323	0.3±0.15	1.6±0.72	0.27±0.1
triggers_idk	0	0	70	-	-	1.49±0.8
triggers_impasse	0	0	7	-	-	-
triggers_seq	55	15	62	0.67±0.5	-	0.38±0.2
triggers_tie	1,258	1,031	1,153	0.7±0.14	1.2±0.24	1.2±0.2
unknown_words	31	16	30	3.47±2.1	-	1.35±0.8

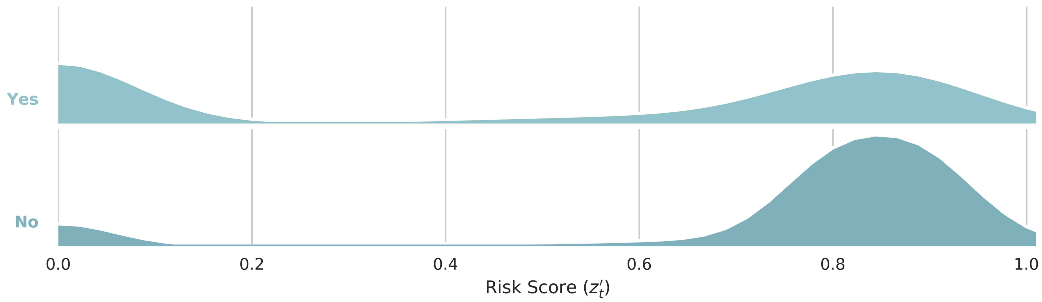
Table 11.2: Risk indicator incidence per dataset. The adjusted OR and 95% confidence interval is given for each with count > 25. Those with a confidence interval spanning 1 are eliminated, as are those with a VIF > 5.

The counts of each risk indicator by dataset, and the adjusted ORs with 95% confidence are given in Table 11.2. Note that many risk indicators were not present or did not meet the 25 count minimum to be considered. When constructing these datasets, we drew a random sample that did not consider any prior knowledge of risk indicators. Therefore, we did not have enough data to draw any conclusions about the effectiveness of these under-represented indicators. In the future, as voters continue to use the system, we can re-evaluate the predictive power of these indicators. Deeper analysis of these OR scores is covered in Chapter 12.

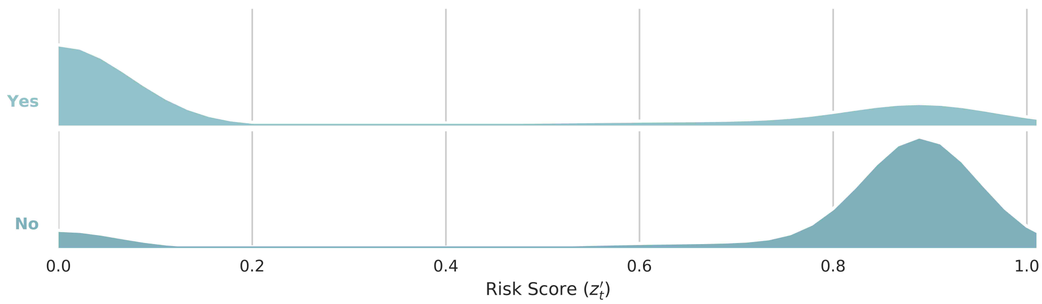
In Figure 11.3, we see the new distribution of risk scores created using the weighted indicators. Notice that when compared to those from the equal weights in Figure 11.2, the two distributions of user turns are pushed further apart on the risk scale. Unfortunately, comparing the overlap between Yes and No it does not appear that the error is necessarily diminished any. We discuss this in detail and the possible reasons for it later in Chapter 12.



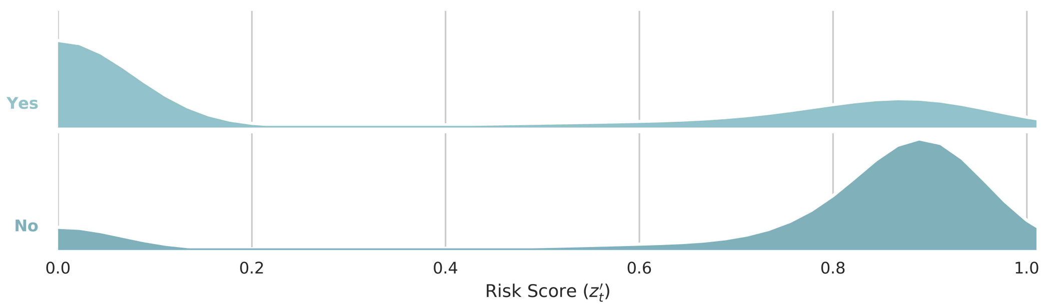
(a) Airline



(b) Telecom



(c) Train



(d) Overall

Figure 11.3: Distribution of risk score by majority vote for each dataset using OR-tuned risk indicator weights. Compare these to Figure 11.2

11.4 Performance in Augmenting the Existing Refinement Cycle

The first configuration of the CRS, introduced in Section 5.1, is to optimize the reviewer's time by prioritizing the user turns for review. The priority ordering is the most likely to be misunderstood by the IVA. In addition, the CRS provides suggestions for alternative intents to the domain experts, which was earlier evaluated in Section 5.1. To evaluate the quality of the prioritization, we compared several strategies using a common practice of selecting a random sample as the baseline.

We are interested in maximizing the number of turns the majority of voters will disagree with the IVA-selected intent in a fixed sample size. This minimizes the wasted time for human review on a sample of that size. Recall that every turn reviewed that was *not* misunderstood is a waste of reviewer time as only misunderstood turns will be given to domain experts for analysis. Therefore, by maximizing the number of misunderstood turns reviewers will see in a fixed sample size, we reduce wasted human effort and, therefore, wasted money. Reduction of human labor costs is a major motivation of this work. Recall that to generate a majority, three votes per turn is desired. Thus, a company pays the cost per turn $\times 3$ for every reviewed turn that is *not* misunderstood, with no improvements to be made in the IVA in return.

For a baseline, we choose a random sample from each evaluation dataset in sizes 100, 200, 300, and 400. We chose a limit of 400 as this is roughly 5% of the average dataset size. Recall from Section 6.2.1 the Telecom IVA processes 50,000 user turns per day, and the human reviewers averaged 11.12 seconds per turn (Section 11.1). Reviewing a 5% sample from that IVA would require 2,500 turns to be reviewed per day at 11.12 seconds per vote. This would require a full 8-hour working day to complete, and thus a 5% sample size is the limit an average human reviewer could process in a day.

For each random sample size, we calculated the percentage of majority voter disagreements with the IVA it contains. To avoid unintentional bias in the random sample, we repeated this process 500 times per sample size per dataset and used the average percentage of disagreement as the baseline.

The first test was a comparison of selection by overall risk score, z'_t, using the equal risk weights. We sorted the evaluation dataset by risk score, descending, and selected the top N turns for each sample size. We then calculated the percentage of disagreements per sample size.

Next, we compared the selection by overall risk score, z'_t, using the OR tuned risk weights. We sorted the evaluation dataset by OR tuned risk score, descending, and selected the top N turns for each sample size. We then calculated the percentage of disagreements per sample size.

Last, we compared the selection using the voting classifier, described in the next section. We sorted the evaluation dataset by the voting classifier decision, and then by the risk score. We used both the equal weight risk score and the OR tuned risk score to compare their effect in combination with the voting classifier. Finally, the top N turns were selected for each sample size. We then calculated the percentage of disagreements per sample size for the equal weight and OR tuned weight in combination with the voting classifier label. The comparison of all 5 selection methods by dataset is shown in Figure 11.4.

Inspecting the charts, we first focus on the random samples. Across all N, the random sample does not change in error discovery performance by more than $\pm 0.2\%$ per dataset. Therefore, it appears this number faithfully indicates the IVA understanding error per dataset, as determined by human reviewers. The Airline IVA has a misunderstanding rate of 11.6% within the evaluation dataset. The Telecom IVA is slightly worse at 14.35%, and the Train IVA is the worst performer at 17.4%

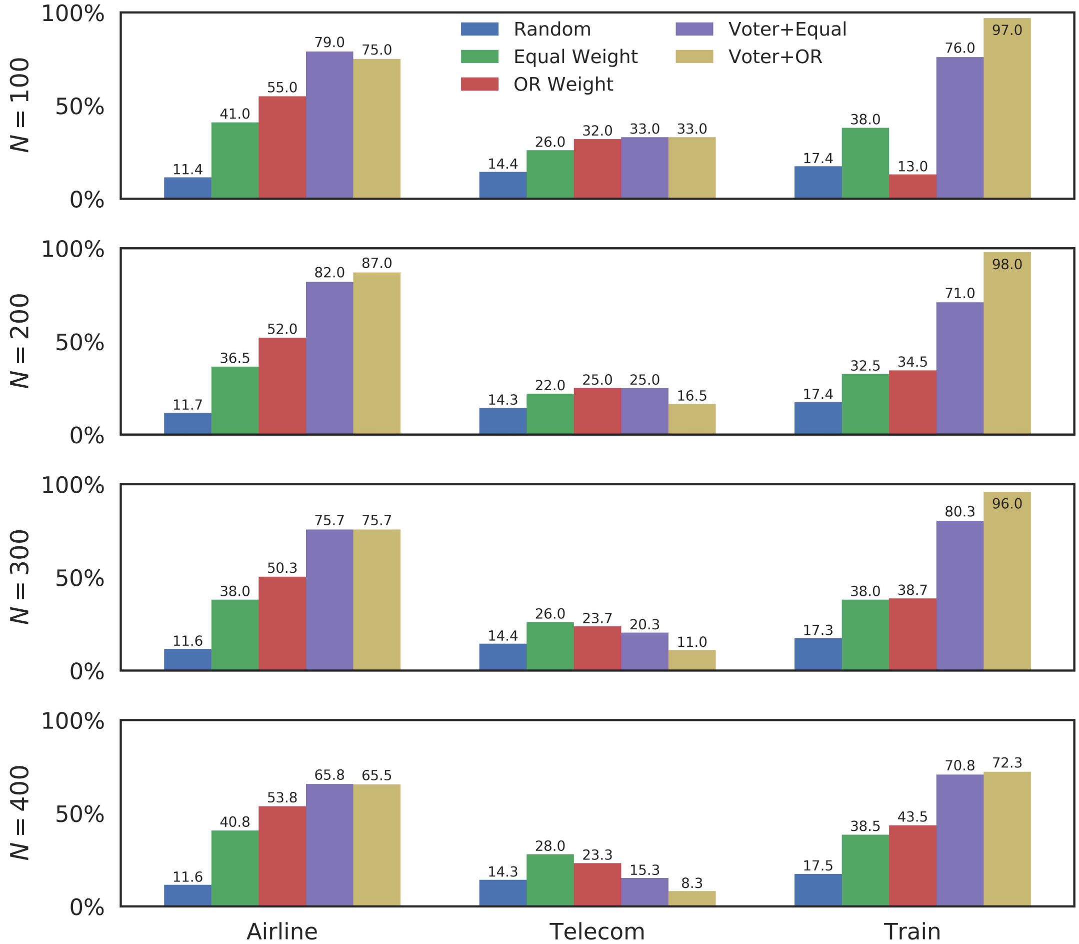


Figure 11.4: Performance of various selection strategies at finding disagreements in a fixed sample size N. The Y-axis is the % of user turns where the majority of voters disagreed with the IVA-selected intent within the sample.

misunderstanding rate. If a human reviewer were to take a random sample of the Airline IVA conversation logs for review, they would only find a misunderstanding in about one out of every ten user turns reviewed. In addition, as the IVA performance improves over time due to the review and language model correction cycle, we would expect this discovery rate to decline.

Next, we focus on the selection using equal indicator weights. At $N = 400$, or a 5% selection size, the increase in reviewer productivity is 3.5x for Airline, 2x for Telecom and 2.2x for Train. The combined risk indicators appear to be effective in finding misunderstandings.

The OR tuned risk weights are more complex. For most cases, they further improve the error discovery rate. However, for the $N = 100$ case they do poorly for the Train dataset, and they appear to degrade as N grows for the Telecom dataset. There is also another issue to consider. Recall from Section 11.3.2 that the OR weights are derived from coefficients of a logistic regression over the three datasets. Therefore, there will be a bias introduced in that the OR weights are used to rank samples that were part of the OR tuning process. We must therefore consider these scores to be a theoretical maximum performance on the part of OR-derived weights.

The poor performance on a small sample size in the Train dataset, and overall in the Telecom dataset can be explained by looking at the risk score distribution graphs in Figure 11.3. Notice that the Telecom dataset has a bimodal distribution for Yes, and there is large overlap between the high risk score values of Yes and No. Thus, the selection by top risk score will contain many errors in the Telecom dataset. For the Train dataset, the mean and standard deviation is smaller for No than the Airline dataset. Small values of N will therefore not retrieve as many misunderstandings as they will for Airline. We do see the performance of the OR risk weights increase as N increases, which would be expected from the distributions.

Introducing the voting classifier to help rank the user turns will only be as successful as the voting classifier is accurate in a language domain. This is evidenced by the performance of the voting classifier across the three datasets. As discussed in the next section, the voting classifier does not perform as well on the Telecom dataset as the others. For the voting classifier combined with the equal weight ordering, the Airline and Train datasets enjoy a 5.7x to 7x and a 4x to 4.6x increase in reviewer productivity over a random sample, while the Telecom dataset only sees a 1.1x to 2.3x increase.

For the last combination of voting classifier and OR tuned weights, we once again see a more complex story. The Telecom dataset appears to pick up the combined error in the voting classifier and the risk distributions as N increases. By $N = 400$, the random sample out performs this selection method. With the Train dataset, we see the opposite effect. For some values of N this approach nears 100% reviewer productivity. We must be careful to point out that the scores for this method will also be subject to the bias introduced by the OR tuning, and therefore should be considered theoretical maximums. Further discussion of these results is continued in Chapter 12.

11.5 Performance in Automating the Existing Refinement Cycle

The second configuration of the CRS, introduced in Section 5.2, is to vote in place of the human reviewers. To select a voting classification method, we evaluated 9 different classification models using several metrics. We performed training and evaluation using stratified 10-fold cross validation on subsets of each dataset, starting from 10% of the total dataset and increasing by tenths until reaching the full set. The increasing partitions were to simulate the growth of the training data as humans continue to vote. We could then see how each model performed with increasing training samples.

The voting classifiers were trained using the risk indicators as features and the majority decision as the outcome. Repeating the example from Section 5.2 for clarity, if voters agreed that turn t belongs to the intent assigned by the IVA, the outcome is 1. If they disagree, the outcome is 0. Then for each turn with a voter consensus we add a row to a feature matrix M, with a column for each risk indicator and a final column for the outcome.

$$
\begin{array}{cc}
 & \begin{array}{ccccc} backstory & multi\_intent & restated & & outcome \end{array} \\
M = \begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_n \end{array} & \begin{pmatrix} 0 & 0 & 1 & \cdots & 1 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}
\end{array}
$$

This feature matrix M is then passed to a classifier training function to produce a binary classification model. When a new turn is under review, the risk indicators present are represented as a feature vector and fed to the voting classifier to predict the majority vote of **Yes** or **No**.

The classifiers were trained and evaluated on each dataset in isolation. It is clear from Table 11.2 that risk indicators are dependent on the language domain, both in frequency and in predictive power. Therefore the CRS maintains and continuously trains an independent classifier for each IVA under review.

11.5.1 Comparison Metrics

Due to the multiple layers of random sampling used to create the datasets and gather the votes, fairly comparing humans to each other and the voting classifier can be difficult. As the 14 human voters in Section 11.1 did not see all of the user turns in a dataset, but were merely given a subset of turns ensuring each turn had three votes each, we cannot calculate a recall, and therefore a F1 score, for the humans. Furthermore, no two humans saw the exact same subset of the turns so that a pair of voters could not always vote the same (always vote "Yes", for example) and generate an inaccurate majority on an entire subset. Equations for precision and recall for the class A are as follows [222]:

$$precision_A = \frac{truepos_A}{truepos_A + falsepos_A} \tag{11.10}$$

$$recall_A = \frac{truepos_A}{truepos_A + falseneg_A} \tag{11.11}$$

Recall, also known as sensitivity, is a measurement that quantifies the fraction of positive samples in the total population that are successfully retrieved [223]. It is a measure of completeness, or sensitivity, of a classifier. The trouble with recall arises when each classifier (human reviewer) has a different total population to select from (subset of turns reviewed). To calculate recall using all possible positive samples (misunderstood turns) in the dataset is unfair as no human had the opportunity to select them all. Because of this, recall scores would be both low and potentially biased towards reviewers with higher numbers of positive samples in their subset as they have more opportunity to reduce $falseneg_A$.

Alternatively, to calculate recall using the count of positive samples in the subset for a particular reviewer is also unfair as each subset may have very different counts of positive samples, so a comparison of recall between two reviewers will have biases. In this case, $falseneg_A$ will have less possibility of an effect on those with low numbers of positive samples in their subset than those with high numbers in their sample. Therefore, to compare the human reviewers to each other and to the voting classifier we considered only the class unweighted (micro) and class weighted (macro) precision. The equations for both in the binary case are given [224]:

$$precision_{micro} = \frac{truepos_A + truepos_B}{truepos_A + truepos_B + falsepos_A + falsepos_B} \tag{11.12}$$

$$precision_{macro} = \frac{1}{2}\left(\frac{truepos_A}{truepos_A + falsepos_A} + \frac{truepos_B}{truepos_B + falsepos_B}\right) \tag{11.13}$$

The micro-averaged precision gives a sense of how many "correct" votes a reviewer made over the sample size they reviewed. Equal weight is given to each turn classification decision without regard to class imbalance [222, 224]. However, as the two classes are very imbalanced (see Section 11.4) this can be misleading if viewed alone. Consider the case of a function that always returns a "Yes" vote. In the Airline dataset, which has only 11.6% misunderstood turns, such a function would get a micro-averaged precision of 0.884. As each sample is weighted individually, the micro-averaged precision becomes a measure of effectiveness on the largest class as the class imbalance grows [224]. In a binary classification, micro-averaged precision reduces to the accuracy score.

In contrast, a macro-averaged precision for this function using (11.13) would be $\frac{1}{2}(0.884 + 0.0) = 0.442$. It should be noted that some toolkits consider no samples from a class as 100% correct on that class as division by 0 is undefined, and they would instead report that function's macro-precision as 0.942. As each class is now weighted equally, the macro-averaged precision gives a sense of effectiveness on small classes [222]. Taking these two measurements together we can get a sense of a classifiers (human or machine) performance overall and performance equally favoring the under-represented class of misunderstandings. In addition, as precision does not consider false negatives, which is difficult to fairly define in differing subsets, we can more fairly compare the metrics between reviewers and the voting classifier.

11.5.2 Voting Classifier Selection Process

The classification methods chosen for comparison either performed well on various datasets[3], or are current standard machine learning methods applied to sparse-feature binary classification tasks (Support Vector Machines [225], Multi-Layer Perceptrons [226, 227], Random Forests [228], et cetera). The specific classification

[3] Such as `http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html`

methods chosen and their configuration parameters[4] are detailed in Table 11.3. A Naive Bayes classifier was also considered, but did so poorly on the preliminary tests it was discarded.

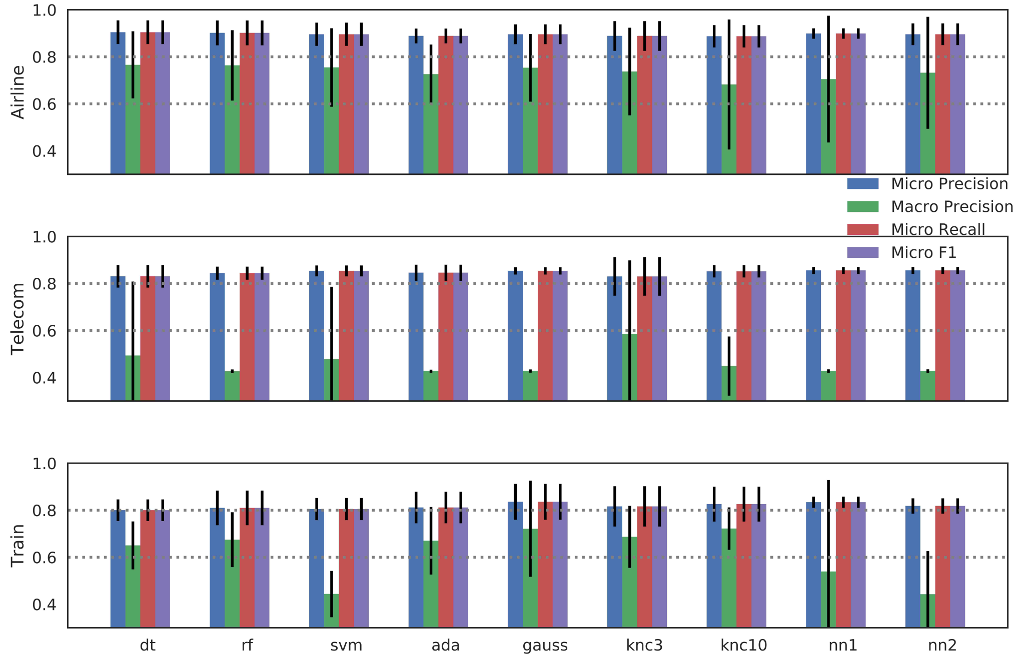
Name	Method	Parameters	References
dt	Decision Tree	{criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1}	[229]
rf	Random Forest	{estimators=15, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1}	[228]
svm	Liner Support Vector	{penalty='l2', loss='l2', C=0.5}	[225, 113]
ada	AdaBoost	{estimator=DecisionTree, n_estimators=50, learning_rate=1.0, algorithm='SAMME.R'}	[230, 228]
gauss	Gaussian Process	{kernel='1.0 * RBF(1.0)', optimizer='fmin_l_bfgs_b'}	[231, 232]
knc3	K-Nearest Neighbor	{n_neighbors=3, weights='uniform'}	[233]
knc10	K-Nearest Neighbor	{n_neighbors=10, weights='uniform'}	[233]
nn1	Multi-Layer Perceptron	{input_neurons=12, input_activation='relu', hidden_neurons=8 hidden_activation='relu', output_activation='sigmoid', epochs=20, batch_size=32}	[226, 227]
nn2	Multi-Layer Perceptron	{input_neurons=64, input_activation='relu', dropout=0.2, hidden_neurons=64 hidden_activation='relu', dropout=0.2, output_activation='sigmoid', epochs=20, batch_size=32}	[226, 227]

Table 11.3: Details on classifiers under comparison for the voting classifier.

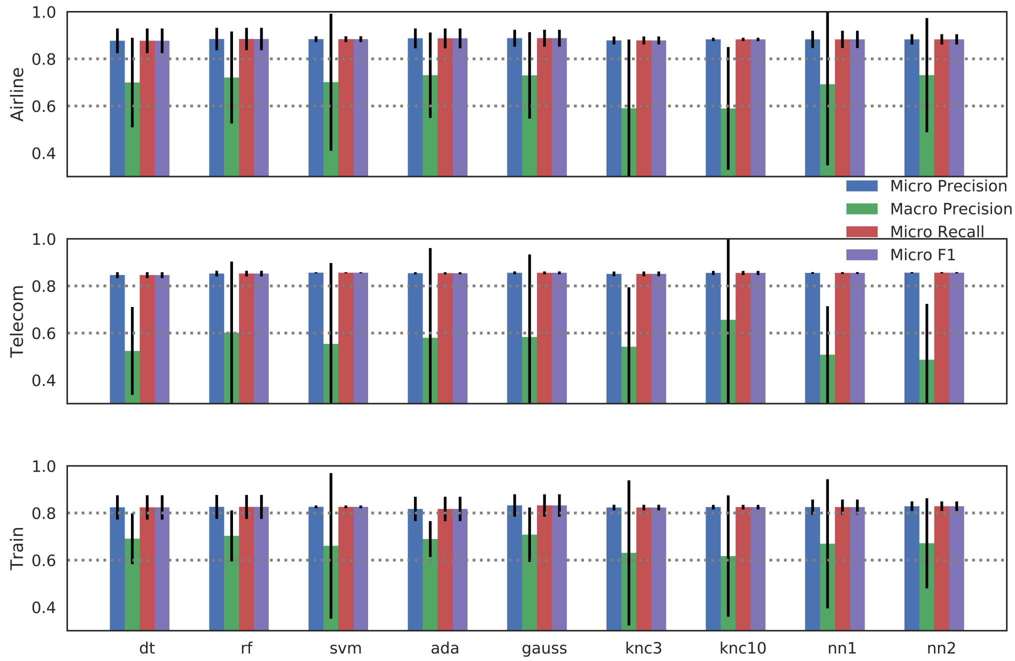
In Figure 11.5, the results of the evaluation using 10% of the data (a) and the final step using 100% of the data (b) is shown. The intermediate steps were not shown as there was surprising little change, outside of the macro precision, as the

[4]Documentation on parameter meanings and possible values is available at `http://scikit-learn.org/stable/modules/classes.html`.

training data increased. With additional data, the **svm**, **knc3**, **knc10**, **nn1** and **nn2** models all saw a slight increase in precision, and most methods saw a decrease in variance across the 10 folds. The tree-based methods (**dt**, **rf**, **ada**) along with **gauss** performed nearly as well with 10% of the data as they did with 100% on the Airline and Train datasets.



(a) 10% dataset

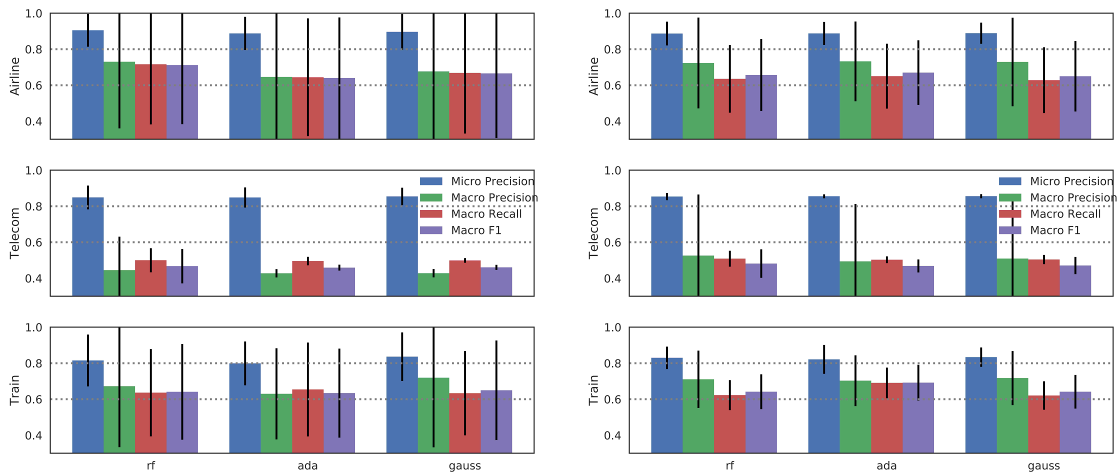


(b) 100% dataset

Figure 11.5: Comparison of classification models on majority voter prediction task, using 10-fold cross validation. The initial evaluation at 10% of the datasets and final size of 100% are shown. 95% confidence interval is represented by black bars.

The top three classification methods that appeared least sensitive to data volume and had the highest combined precision across all three datasets were then selected for a second round of comparison. These classifiers (**rf**, **ada**, **gauss**) were then retrained on the 10% to 100% set sizes using a more rigorous 30-fold cross validation and the training times were recorded. As the CRS is continually retraining these models as human voting data is added, training time and scaling are important considerations, classification performance being equal. The **gauss** classifier was by far the slowest to train, averaging 12 minutes for 30-fold cross validation across the three datasets for the 100% dataset size. Compare this to 8 seconds for **ada** and 2 seconds for **rf** on the same data size.

The performance of the remaining classifiers in this round is shown in Figure 11.6. The volume of data seems to have a minor effect on the precision, but a more noticeable effect on the confidence interval.



(a) 10% dataset (b) 100% dataset

Figure 11.6: Comparison of the top three classification models on majority voter prediction task, using 30-fold cross validation. The initial evaluation at 10% of the datasets and final size of 100% are shown. 95% confidence interval is represented by black bars.

Given the nearly identical performance of the three methods across all datasets, regardless of training data size, we chose the method with the fastest training time (**rf**) as the final classification type to use for the voting classifier. As the CRS maintains separate classifiers per dataset due to the difference in risk indicators, we next optimized the parameters for the **rf** classifier per dataset. Using a stratified 30-fold cross validation, we performed a grid search over the following parameter values:

Parameter	Possible Values
n_estimators	15 through 30 in steps of size 2
criterion	'gini', 'entropy'
max_depth	5 through 12, None

The **n_estimators** parameter sets the number of trees in the forest. The **criterion** parameter is the function to measure the quality of a split, Gini impurity or the information gain. The **max_depth** is the maximum allowable depth of a tree. For each combination of parameters, the macro precision was calculated and later was used to select the best combination. The best parameters found per dataset were:

Dataset	Best Parameters	Precision (Micro)	Precision (Macro)
Airline	{n_estimators=30, criterion='gini', max_depth=10}	0.890	0.740
Telecom	{n_estimators=30, criterion='entropy', max_depth=11}	0.858	0.587
Train	{n_estimators=17, criterion='gini', max_depth=9}	0.831	0.742

11.5.3 Voting Classifier Evaluation

Having selected a voting classifier, we compared its performance using the optimized parameters on a stratified 30-fold cross validation of each dataset to the human voters. For each voter, we calculated the micro and macro precision of their votes to the majority vote.

Airline Dataset

The airline dataset was composed of $9,103$ user turns of which $6,978$ (76.65%) had a majority voter agreement, and the originating IVA can recognize $1,223$ unique intents. The human voter performance, along with the voting classifier average performance over a 30-fold cross validation, is shown below.

Rank	Voter	Precision Macro	Rank	Voter	Precision Micro
1	voter2	1.000000	1	voter2	1.000000
2	voter15	1.000000	2	voter15	1.000000
3	voter17	1.000000	3	voter17	1.000000
4	voter6	0.926611	4	voter9	0.958349
5	voter1	0.904731	5	voter1	0.952381
6	voter9	0.891965	6	voter6	0.947956
7	voter3	0.861885	7	voter3	0.942841
8	voter4	0.844408	8	voter4	0.941643
9	voter5	0.832290	9	voter7	0.933943
10	voter8	0.790549	10	voter8	0.917561
11	voter7	0.772389	11	CRS	0.889902
12	CRS	0.739638	12	voter12	0.878893
13	voter12	0.695378	13	voter11	0.866667
14	voter11	0.666667	14	voter5	0.836735
15	voter10	0.641716	15	voter10	0.790123
	mean±95%	0.844899±0.239		mean±95%	0.926221±0.123

Telecom Dataset

The Telecom dataset was composed of $7,313$ user turns of which $5,252$ (71.82%) had a majority voter agreement. It originated from the most complex IVA of the three. The IVA recognizes $2,173$ unique intents and carries out complex user interactions that take into account meta-data such as the users mobile device, if the user is logged in or not, the location and account type the user has, et cetera. Without understanding this meta-data and how it is used in intent classification, the CRS is at somewhat of a disadvantage in selecting the proper intent as an outside observer. Due to this IVA complexity, the scores are lower on this dataset than the other two.

Rank	Voter	Precision Macro	Rank	Voter	Precision Micro
1	voter1	1.000000	1	voter1	1.000000
2	voter16	0.935140	2	voter10	0.980040
3	voter10	0.914609	3	voter6	0.952590
4	voter6	0.894821	4	voter2	0.943147
5	voter15	0.858721	5	voter13	0.936842
6	voter4	0.833291	6	voter16	0.923913
7	voter9	0.826032	7	voter3	0.911934
8	voter2	0.825104	8	voter4	0.898210
9	voter7	0.815603	9	voter15	0.895105
10	voter5	0.813492	10	voter5	0.887090
11	voter3	0.761177	11	voter7	0.871921
12	voter13	0.738506	12	CRS	0.857446
13	voter11	0.680040	13	voter9	0.803738
14	CRS	0.587383	14	voter12	0.788462
15	voter12	0.564286	15	voter11	0.696275
	mean±95%	0.818630±0.243		mean±95%	0.892090±0.158

Train Dataset

The Train dataset was composed of 7,270 user turns of which 6,331 (87.1%) had a majority voter agreement. The originating IVA can recognize 930 distinct intents, making it the smallest language model of the three. Despite its smaller size, it appeared to have more disagreement between the individual voters and the majority vote, evidenced by the lower means and larger 95% confidence intervals.

Rank	Voter	Precision Macro	Rank	Voter	Precision Micro
1	voter15	0.972222	1	voter6	0.973888
2	voter10	0.918452	2	voter10	0.951338
3	voter6	0.910626	3	voter15	0.950000
4	voter2	0.874898	4	voter4	0.933293
5	voter5	0.853439	5	voter5	0.931871
6	voter4	0.848875	6	voter7	0.931398
7	voter3	0.848343	7	voter8	0.920548
8	voter8	0.838889	8	voter3	0.909347
9	voter7	0.825955	9	voter14	0.909091
10	voter1	0.809436	10	voter11	0.856651
11	CRS	0.742357	11	voter2	0.854251
12	voter11	0.730333	12	voter1	0.836417
13	voter12	0.594160	13	CRS	0.831601
14	voter16	0.562500	14	voter12	0.630252
15	voter14	0.500000	15	voter16	0.222222
	mean±95%	0.792009±0.276		mean±95%	0.843612±0.382

Overall Performance

Averaging the voter performance over the three datasets we can see how each of the voters faired overall. Although all three datasets had 14 human voters, not all of the voters were the same people. There were 17 unique voters overall, and their labels are consistent across datasets (voter3 in the Train dataset is the same person as voter3 in the Airline dataset).

Rank	Voter	Precision Macro	Rank	Voter	Precision Micro
1	voter17	1.000000	1	voter17	1.000000
2	voter15	0.943648	2	voter6	0.958145
3	voter6	0.910686	3	voter15	0.948368
4	voter1	0.904722	4	voter13	0.936842
5	voter2	0.900001	5	voter2	0.932466
6	voter9	0.858999	6	voter1	0.929599
7	voter4	0.842192	7	voter4	0.924382
8	voter5	0.833074	8	voter3	0.921374
9	voter10	0.824926	9	voter8	0.919054
10	voter3	0.823801	10	voter7	0.912421
11	voter8	0.814719	11	voter14	0.909091
12	voter7	0.804649	12	voter10	0.907167
13	voter16	0.748820	13	voter5	0.885232
14	voter13	0.738506	14	voter9	0.881044
15	voter11	0.692346	15	CRS	0.859649
16	CRS	0.689793	16	voter11	0.806531
17	voter12	0.617941	17	voter12	0.765869
18	voter14	0.500000	18	voter16	0.573068
	mean±95%	0.809931±0.251		mean±95%	0.885464±0.252

Chapter 12

Discussion

We begin our discussion with a review of the work presented thus far. The task at hand is to review logs of conversation between Intelligent Virtual Assistants (IVAs) and human users in order to find communication errors in the form of misunderstanding on the part of the IVA. This task is currently reserved for human reviewers, who are unable to scale to the volume and velocity of the logs for commercial IVA deployments. Therefore, sampling strategies are employed to create manageable subsets of the conversations that can be handled by humans.

The work presented in this dissertation is the design and construction of a system that can handle the scale of the logs (Section 6.2), provide an intelligent selection strategy superior to current practices (Section 11.4), suggest relevant alternative intents autonomously (Section 11.2), and even potentially altogether replace human reviewers (Section 11.5). The system proposed to perform all of these tasks is named the Chat Review System (CRS).

The motivations for the construction of the CRS are the cost of human labor, the subjectivity of human language decisions, and the inability for a set of human reviewers to find all of the different error scenarios present in the conversations logs (such as misunderstanding an infrequent intent) as time and money do not permit reviewing all of the logs. Given the overarching task, motivations and goals of the CRS, we now will consider each objective and how the CRS performs on it.

12.1 Load Tests and Ability To Scale

One of the primary needs is the ability to review *all* of the conversations from a live IVA. It is impossible to predict all of the ways human users will attempt to communicate with a natural language interface. Every person brings their own style of communication, assumptions, biases, and problems. Given this, failures in understanding have the potential to be present in any conversation with an IVA. If the CRS is to be viable solution for a company that designs and builds IVAs, it must support not just the log sizes of a single IVA, but of dozens.

In Section 6.2, various scaling tests are performed on a deployment of the CRS. From the scale up tests, we see there is a linear relationship between the number of turn-response pairs analyzed and the total processing time that closely follows a $O(\sqrt{n})$ growth curve. If this were to continue, ignoring storage and memory limitations of the test servers, that configuration could theoretically support a single IVA which generates 8.6 million user-response pairs per 24 hours. Put in perspective, this number is 8 times the number of calls the Los Angeles Police Department handled in all of 2017[1].

Taking into account daily model training overhead per IVA, the CRS as tested could support 31 concurrent IVAs. If models were only retrained once a week, we would use 0.015 as the average number of seconds to process one turn (see Figure 6.3). As the model training time has a large impact on total processing time, training once a week would increase the capacity of the CRS to handle 115 concurrent IVAs averaging 50,000 turn-response pairs per day. This number of concurrent IVAs by

[1]Source: `https://catalog.data.gov/dataset/lapd-calls-for-service-2017`

far surpasses the requirements of a single business, and, if for some reason it was still unsatisfactory, more nodes can be added to the compute and database clusters to increase performance (see Figure 6.5), or a second instance of the CRS could be deployed to double the capacity.

These load tests demonstrate that the CRS, in the hardware configuration tested, has the ability to meet the demands of commercial deployment. One instance could easily handle all of the conversation logs from a single high-volume IVA, even using only a single HPC cluster node.

12.2 Intelligent Selection Strategy

The next area of need is to provide an intelligent selection strategy superior to current practices. If human reviewers are to be productive in their error discovery task, we wish to prioritize the user turns they see to maximize the error discovery. As stated in the introduction, every user turn reviewed by a human that does not contain any misunderstanding on the part of the IVA is not useful to the domain experts to improve the IVA. Therefore, the time spent on review was wasted effort. To evaluate possible selection strategies, several methods were used to select a fixed sample size N of user turns for review in Section 11.4. The number of turns within each sample that resulted in an IVA misunderstanding, as determined by a majority of voters, was counted.

The baseline method was a random sample. To prevent any unintended bias, the random sample was taken 500 times for each value of N and the percentage of misunderstandings found were averaged. We would expect this averaged value to converge to the error rate of the IVA within the dataset. In Figure 11.4 we see the result of each selection strategy by dataset. There are a few issues that are immediately apparent. The first is that the error rate of the IVAs are all low

($< 18\%$) and therefore random selection will result in a lot of wasted reviewer time. The second is that the selection strategies differ in effectiveness by language domain. In the Train dataset we do not see nearly the effectiveness in the OR weighted selection that is seen in the Airline dataset. In the Airline and Train datasets, the voter-based selections are the highest performing.

In the Telecom dataset, the voter-based selection strategies are not effective, due to the lower performance of the voting classifier in this domain. Recall that this was the most complex of the three IVAs considered. The Telecom IVA understands $2,173$ unique intents and allows the most complex user interactions along with extensive business logic that is used to determine correct intent based on many factors of the current user. As the voting classifier performed the worst on this dataset, the selection strategies using the classifier performed worse than the weight based strategies alone. However, the weight based strategies were able to outperform a random sample, even doubling its performance in the case of even weight selection.

12.2.1 Annotation Cost Recovery by Deploying the CRS

Looking at each sample selection method we can get a sense of the human productivity increase we could expect by employing the method over a random sample. To quantify this productivity increase, we need estimates for cost of review per turn. There are two ways a company can conduct reviews. The first is to use employees to review the logs, the second would be to use an external annotation service.

$$HourlyCost = \frac{YearlySalary}{52wks * 40hrs} + Burden = \frac{\$39,231}{2,080} + \$11.38 = \$30.24 \quad (12.1)$$

$$TurnsPerHour = \frac{SecondsPerHour}{TurnsPerSecond} = \frac{60m * 60s}{11.12} = 324 \quad (12.2)$$

$$CostPerTurn_{employee} = \frac{HourlyCost}{TurnsPerHour} = \frac{\$30.24}{324} = \$0.0933 \tag{12.3}$$

To get an estimated cost per turn for an employee, we take the median salary for a Customer Service Representative plus average labor burden and convert this into hourly pay (12.1). We can then find how many turns per hour an average human reviewer can vote on, using the statistic 11.12 seconds per turn measured in Section 11.1. Finally, using (12.3) we can determine the average cost for a company to employ someone with customer service training full time to review IVA conversation logs. For the median salary, we pick a location with known high concentrations of call centers [234] and find the median salary there[2]. For the labor burden, which is the overhead an employer pays per employee such as payroll taxes and insurance, we use the December 2017 average US benefits costs reported by the US department of labor[3]. We arrive at a cost of $0.0933 per turn voted on.

To get a cost for an external annotation service, we use the costs reported by the popular crowd-sourcing platform Amazon Mechanical Turk[4]. The pay per item of work can be anything larger than $0.01; at the time of this writing, between $0.01 and $0.10 per item appears common in listed jobs for annotation tasks such as labeling images and classifying language. There is then the 20% fee per item Amazon charges plus and additional 20% fee per item for batches of more than 10 items. Finally, as this work requires meeting minimum qualifications for proper intent classification, there is an additional $0.40 fee per item to use qualified workers. Using (12.4), the total cost per turn to use the crowd-source platform Mechanical Turk is therefore

[2]Source: `https://www.payscale.com/research/US/Job=Customer_Service_Representative_(CSR)/Hourly_Rate/822cfecf/Dallas-TX`

[3]Source: `https://www.bls.gov/news.release/ecec.nr0.htm`

[4]Source: `https://requester.mturk.com/pricing`

between \$0.414 and \$0.54 depending on how much we pay the workers.

$$CostPerTurn_{MTurk} = (PayPerTurn * 1.4) + \$0.40 = [\$0.414, \$0.54] \tag{12.4}$$

For a fair estimation, we must subtract and costs associated with running the CRS analysis over the data. Using the Amazon Web Services (AWS) hosting cost calculator[5], we can estimate the hourly costs associated with the deployment configuration given in Table 6.1, but only using one HPC node as that is more than sufficient to support a single IVA. Given this configuration, the hourly hosting costs for the CRS on AWS would be \$2.67. Using the time per turn measured to process one days worth of data, 0.055 seconds (see Figure 6.3), we can calculate the CRS processing cost per turn:

$$CRSTurnsPerHour = \frac{SecondsPerHour}{SecondsPerTurn} = \frac{(60m * 60s)}{0.055s} = 65,454.\overline{54} \tag{12.5}$$

$$CostPerTurn_{CRS} = \frac{HourlyCost}{CRSTurnsPerHour} = \frac{\$2.673}{65,454.\overline{54}} = \$0.000040837 \tag{12.6}$$

Now that we have our costs for the employee reviewers, external reviewers, and data processing, we can calculate the cost savings of deploying the CRS at this task. As majority agreement is required to control for subjectivity, the human cost is multiplied by 3. Referring to Figure 11.4 we can calculate the savings in lost human productivity using method x in place of a random sample by the following equation:

$$CostDelta = (HumanCost * 3) - CostPerTurn_{CRS} \tag{12.7}$$

[5]Source: `http://calculator.s3.amazonaws.com/index.html`

$$Recovered = [((Err_x - Err_{random}) * N) * CostDelta] \tag{12.8}$$

Using equation (12.8), the various values of N, $CostPerTurn_{employee}$ and boundaries of $CostPerTurn_{MTurk}$ as $HumanCost$, we populate the following table.

	Dataset	N	Equal	OR	Voter+Equal	Voter+OR	HumanCost
0	Airline	100	\$8.28	\$12.19	\$18.91	\$17.79	\$0.09
1	Airline	100	\$36.72	\$54.11	\$83.92	\$78.95	\$0.41
2	Airline	100	\$47.90	\$70.58	\$109.46	\$102.98	\$0.54
3	Telecom	100	\$3.24	\$4.92	\$5.20	\$5.20	\$0.09
4	Telecom	100	\$14.40	\$21.85	\$23.09	\$23.09	\$0.41
5	Telecom	100	\$18.78	\$28.50	\$30.12	\$30.12	\$0.54
6	Train	100	\$5.77	\$-1.23	\$16.40	\$22.28	\$0.09
7	Train	100	\$25.60	\$-5.44	\$72.80	\$98.88	\$0.41
8	Train	100	\$33.40	\$-7.10	\$94.96	\$128.97	\$0.54
9	Airline	200	\$13.89	\$22.57	\$39.36	\$42.16	\$0.09
10	Airline	200	\$61.65	\$100.15	\$174.67	\$187.09	\$0.41
11	Airline	200	\$80.41	\$130.63	\$227.83	\$244.03	\$0.54
12	Telecom	200	\$4.29	\$5.97	\$5.97	\$1.21	\$0.09
13	Telecom	200	\$19.05	\$26.50	\$26.50	\$5.39	\$0.41
14	Telecom	200	\$24.85	\$34.57	\$34.57	\$7.03	\$0.54
15	Train	200	\$8.46	\$9.58	\$30.01	\$45.12	\$0.09
16	Train	200	\$37.54	\$42.51	\$133.18	\$200.24	\$0.41
17	Train	200	\$48.97	\$55.45	\$173.71	\$261.19	\$0.54
18	Airline	300	\$22.17	\$32.53	\$53.80	\$53.80	\$0.09
19	Airline	300	\$98.41	\$144.36	\$238.75	\$238.75	\$0.41
20	Airline	300	\$128.36	\$188.29	\$311.41	\$311.41	\$0.54

21	Telecom	300	$9.78	$7.82	$5.02	$-2.82	$0.09
22	Telecom	300	$43.39	$34.70	$22.28	$-12.50	$0.41
23	Telecom	300	$56.60	$45.26	$29.06	$-16.30	$0.54
24	Train	300	$17.37	$17.93	$52.92	$66.07	$0.09
25	Train	300	$77.11	$79.59	$234.83	$293.21	$0.41
26	Train	300	$100.57	$103.81	$306.31	$382.45	$0.54
27	Airline	400	$32.66	$47.21	$60.64	$60.36	$0.09
28	Airline	400	$144.93	$209.51	$269.13	$267.89	$0.41
29	Airline	400	$189.04	$273.28	$351.04	$349.42	$0.54
30	Telecom	400	$15.35	$10.04	$1.08	$-6.76	$0.09
31	Telecom	400	$68.14	$44.54	$4.80	$-29.98	$0.41
32	Telecom	400	$88.87	$58.09	$6.26	$-39.10	$0.54
33	Train	400	$23.51	$29.11	$59.62	$61.29	$0.09
34	Train	400	$104.35	$129.19	$264.56	$272.01	$0.41
35	Train	400	$136.11	$168.50	$345.08	$354.80	$0.54

Table 12.1: Annotation cost savings using various selection methods in the CRS over a random sample.

From this table it is immediately apparent that data annotation can become expensive if volume is large enough, and if a company was to deploy an IVA and put in place a continuous review and improvement cycle, it would be much cheaper to hire full time employees with customer service experience at the median salary than use an external data annotation service with quality control guarantees. If quality control was not a concern, and the data could be publicly viewed such as on Twitter, external data annotation could be cheaper if total cost was lower than $0.09 per item. There were some instances where introducing a particular strategy would actually cause a loss, particularly with the Telecom IVA and the voting classifier. Therefore, it would be necessary to evaluate the selection strategies per IVA under review as it appears there is none that performs equally well across all datasets.

Consider that the largest N evaluated was 400, which was 5% of the average dataset size. If a company were to deploy a customer service IVA handling similar volume as the Telecom IVA, 1.6 million user turns per month, and they wished to review 5% for quality control, they would be paying for 80,000 turns to be reviewed every month.

First assume it was a very difficult domain in which there are over 2,000 unique intents and complex intent classification logic, as is the case with the Telecom IVA. If the company was currently using Mechanical Turk and paying workers \$0.10 per item for the annotations, which costs the company \$0.54 per item, they would be paying \$129,600 per month in annotation costs to get three votes per user turn. Using the even weight selection strategy, who's performance in Figure 11.4 is unbiased as no prior training is performed, deploying the CRS would save such a company \$17,774 per month (Table 12.1, row 32).

If instead, the IVA were similar to the Airline language domain and using the same even weight strategy, deploying the CRS would save such a company \$37,808 per month (Table 12.1, row 29). Using the Voter+Even Weight strategy would save such a company \$70,208 *per month*. That savings is over 54 % of their total monthly annotation costs, and greatly justifies the introduction of the CRS.

It is difficult to draw conclusions about the OR turned weights given they were tuned on the data that was used for evaluation, making the performance closer to the theoretical maximum. There does appear to be a theoretical improvement in performance over the even weights, particularly with the Train IVA. Having more samples where the under represented risk indicators are present may further increase this improvement as well.

12.3 Relevant Alternative Intents

The CRS performance on suggesting alternative intents was evaluated in Section 11.2. If an IVA intent classification is deemed inaccurate by the human reviewers, a determination of where it should go must be made in order to fully correct the language model error. The CRS attempts to automate this task by leveraging the output of its agreement classifiers. If an agreement classifier disagrees with the IVA-selected intent, and it has high confidence, the winning intent from the agreement classifier is used as a suggested alternative intent.

To determine how useful this alternative suggestion is, humans were shown the original user turn and agreement classifier selected intent to vote on as if it were an IVA-selected intent. These alternatives are referred to as Potential New Intents (PNIs), and to prevent bias human reviewers did not know the origin of the intent they were voting on. The PNIs were mixed in to the IVA-selected intents under review and presented in the same interface. To control for subjectivity a majority was required to agree on the outcome.

Inspecting Figure 11.1, we see the results of the PNI voting per dataset. Overall, the PNI suggestions were beneficial, with two out of every three having a majority of reviewers agree with the suggested intent. Although the Telecom dataset was the most difficult for the voting classifier, the agreement classifier performed the best there with 70% voter agreement. Consider that this agreement classifier is having to pick an acceptable intent from a pool of $2,172$ alternatives, and does so 70% of the time. It is possible that the introduction of additional alternative classifiers and some agreement strategy between them could increase performance.

It is very difficult to quantify the costs associated with choosing an alternative intent as they will vary based on the complexity of the user turn and the knowledge of the possible intents by the human. In some cases an alternative may be immediately known to the human, in other cases they may have to search the knowledge base and consider several closely related intents. Regardless, having the system suggest an alternative will simply provide no cost saving in a worst case, as the human would immediately disregard the suggestion every time and follow the normal process. But for situations where an alternative is not immediately known to the human, the suggestion will provide a time savings for them, which will translate into a cost savings for the company.

Using the Telecom IVA as an example, in a random sample of 1,000 user turns, 144 will need alternative intents determined due to the 14.4% understanding error observed in that IVA. If we were to arbitrarily guess that searching for an alternative takes 2 minutes for a human to perform and make a decision, the process would require 4.8 hours of human time, or $145.15 using the hourly rate of an employee from the previous section. If the CRS could reduce the time required by 70%, only 1.44 hours would be needed, or $43.55. Like the conversation review costs, this alternative intent finding task is a recurring cost, a 70% time savings will therefore reduce the total monthly operating costs for a company deploying an IVA.

12.4 Human Voter Replacement

The final evaluation performed on the CRS was that of its potential to replace human reviewers altogether. In Section 11.5.3, the performance of the CRS in identifying missed intent compared to that of the human reviewers. There is a bias favoring the humans here in that the gold standard was produced by the majority of human reviewers. If a user turn already has two *yes* votes, and the third reviewer were to vote *yes* as well, they have predicted the majority vote. If the third reviewer votes *no* on that same user turn, they have failed to pick the majority vote. In both of these instances the reviewer is performing the same task as the CRS and can be fairly compared.

The bias arises when a reviewer votes on a user turn where there is one agreement and one disagreement vote. In this case the reviewer is forming the majority either way they vote and will therefore never be penalized on these turns. This creates an advantage for certain humans in cases of disagreement. Note that on that same user turn, the human voter that the final voter disagreed with will still be penalized for not predicting the majority vote.

In light of this, the human voter precision scores given in the tables in Section 11.5.3 may be higher than a true outside observer predicting the human majority vote as the CRS does. It is dangerous to try to correct for this by ignoring votes that form majority however, as turns will not be scored for the humans choosing the majority when only two agree, but will be scored against the one that didn't. This introduces a bias that gives more chances for penalty than reward. We also cannot fairly only consider turns where all three reviewers agree, as the human performance will always be 100 % precision and turns with some disagreement are potentially harder cases we want to evaluate performance on.

Therefore, we leave the precision as is, noting that the bias exists and favors the human voters. To measure the magnitude of possible bias, we count how often humans vote to form the majority when there is an existing split vote, by dataset.

Project	Majority Creating Votes
Airline	16.8%
Telecom	22.7%
Train	25.1%

The human performance numbers at predicting the majority can be considered potentially inflated by a maximum of the above percentages. To properly prevent this effect, a second round of voting should be done over the same gold standard where a different set of voters tries to predict the majority vote without influencing it. Due to time and money constraints, this is left for future works.

Recognizing the bias toward the humans, we can still look at the results of the voting in Section 11.5.3 and consider that the CRS is able to outperform several voters in each dataset. As prior experience looking at tables of precision involving classifier comparisons in literature conditions us to expect the proposed system to be in the top ranks, it can be discouraging to view these tables. But it is important to remember that in each table, the competitor to the CRS is in fact an individual human voter attempting to complete the task to the best of his or her abilities. The humans have the benefit of native language understanding along with the ability to read the full conversation for context around the current turn. The very fact that the CRS is able to perform among their ranks is encouraging. Considering the humans also enjoy some measure of precision increase by occasionally influencing the majority, the CRS appears to perform well at this task. On all datasets for both macro and micro precision, the CRS score is within the 95 % confidence interval of the human voter scores.

The balance between the macro precision and micro precision is whether we are interested in total task performance (micro) or performance considering each class equally (macro). Due to much lower incidence of misunderstanding, as these IVAs have misunderstanding rates from 11.6% to 17.4%, the difference in micro precision to macro precision for a given voter indicates how much better they perform on one class than the other. As micro precision favors the larger class, this would be how much better they perform on correct intents than misunderstood intents.

Looking at the Airline rankings, voter5 appears to have nearly equal macro and micro precision. This would indicate they are performing equally well on the two classes. Voters 2, 15, and 17 in that same dataset all have 100% macro and micro precision, indicating they always agree with the majority for both classes. These three voters do not have this same performance on the other datasets indicating that they are likely all experts on this particular IVA, or at least understand its language model very well. The CRS appears to be performing better on the correct class, but still bests three human voters in macro precision and falls well within the 95% C.I. A domain expert for this IVA would have to decide if the CRS performance is acceptable enough to warrant using its votes in place of humans. Remember that the outcome of this whole process is consumed by the domain experts, who use the voting data and suggested alternative intents to discover errors in the language model and fix them. Therefore, some error in the voting process can still be corrected by the domain experts as they have the ultimate say in what will and will not be changed in the language model.

Moving to the Telecom dataset results in Section 11.5.3, we see the worst performing language domain for the CRS. It appears harder for humans as well, since the means for both micro and macro precision are lower than the Airline dataset. Several voters manage to get higher macro precision than micro, indicating that correct intent may be less obvious with this IVA. The CRS has its lowest macro precision in this set, but still beating voter12, and staying within the 95 % C.I.

Finally, inspecting the Train dataset results in Section 11.5.3, we see the most disagreement between the individual reviewers and the majority vote. This is evidenced by the lowest means and highest confidence intervals, and no voter was able to reach 100% precision of either type. This disagreement may be due to many similar intents that caused confusion, or possibly because no voters were familiar with the Train IVA and its knowledge base. The CRS is better in this dataset, with less difference between the micro and macro precision than the other datasets, indicating better performance on misunderstood turns. Its scores were much closer to the mean scores in both metrics than the other datasets as well. A domain expert for this IVA may be willing to accept the CRS votes in place of a human majority, especially in light of the potentially inflated human scores and lower agreement.

The cost benefits of replacing the humans is very compelling, although at these performance levels it may lead to more work on the part of the domain experts to verify the CRS voting. Using the earlier example of a company currently using

Mechanical Turk and paying workers \$0.10 per item for the annotations, which costs the company \$0.54 per item, they would be paying \$129,600 per month in annotation costs to get three votes per user turn for 5% of the monthly volume. To replace the human voters altogether and accept the CRS votes would only cost \$3.27 for all 80,000 turns. In addition, the CRS could process the entire monthly volume of 1.6 million for only \$65.34 in processing costs. Even if the performance does not currently appear to match the best human reviewers, performance was near that of the average reviewer. With a 40,000x reduction in annotation costs and ability to provide annotations on 100% of the data, using the CRS in place of humans is well worth a reduction in precision.

12.5 Future Work

As covered in the last section, a much larger dataset which contains sufficient volume of all of the risk indicators would be very informative. In addition to providing missing predictive power of these indicators, large enough datasets would allow testing and training partitioning where incidence of all of the indicators are balanced between them. This would allow a true evaluation of the Odds Ratio-based selection methods for increasing voter productivity.

To fairly compare the CRS to humans for predicting the majority, a second pass of annotation would be required on the testing partition. This would require paying a second group of reviewers that were not part of the first group to vote on all of the data where a majority was already established. The CRS could then be fairly compared to the performance of the second group, as their tasks would be identical.

Evaluating different agreement classifiers and ways to form majority decisions such as the creation of Potential New Intents (PNIs) could provide improvement in the area of selecting alternative intents. As the average performance of this task was

67.7%, there is room for improvement.

The automation of even more tasks currently performed by humans (in the refinement cycle) would be highly beneficial. When a PNI is generated, the system could perform a detailed analysis on the language model to determine *why* the turn was incorrectly mapped. Because the analysis would differ based on the implementation of the language model, the format of the language model is vital. For example, less detailed analysis is possible with a statistical language model. The CRS, however, could generate additional training samples from the misunderstood turns using synonyms and rephrasing techniques. These samples can be analyzed by the domain experts to help them correct the model. The number of samples needed could be determined by the margin of error present between the observed and suggested intent. For example, with a Support Vector based model, the distances from the hyperplane for the suggested intent may be used to indicate the number of additional training samples needed.

For regular expression or grammar based models, a more detailed analysis is possible. The CRS can easily trace the path of matching patterns that resulted in the responding intent. The CRS can then determine what patterns were missing from the appropriate intent that would have created a correct mapping.

For example, if the responding intent *Baggage Claim Information* had a greedy pattern like the one below:

```
.+baggage\sclaim.+
```

this would cause a user utterance such as *"The ticket counter agent that took my baggage claimed it would arrive with me, but its not here!"* to be mapped to the *Baggage Claim Information* intent. Suppose the CRS suggests a more appropriate intent: *Missing Baggage*. From there, the CRS can do one of two things. Given that the CRS has access to all of the training and regression data used to construct the language model (see Section 4.4.3), it could iteratively try to constrain the greedy pattern and rerun all of the regression data through this modified model to verify it both fixed the misunderstood turn and did not break any existing behavior in the *Baggage Claim Information* intent. Alternatively, the CRS could attempt to generate a new pattern [235] or modify an existing one for the *Missing Baggage* intent and do the same verification using the regression data. Once such a solution is found, the CRS will present it to the domain experts, saving them further manual analysis and verification time.

In order to add this correctional functionality for both statistical and pattern based models, additional research and testing would be required.

An additional future task would be to integrate the CRS more closely with the IVA, providing on-line evaluation of turn-response pairs. Using the live conversational context the CRS could evaluate the mapped intent *before* the IVA responds. This way the dialog manager can take into account the risk score from the CRS in determining the proper response. Communication between the IVA and the users could be improved by this application by preventing misunderstandings before they occur.

Chapter 13

Summary

This work has covered the introduction of a Chat Review System (CRS) that can optimize various stages of an Intelligent Virtual Assistant (IVA) review and refinement cycle. Currently, costly and incomplete human annotation is required due to the volume of conversations generated by commercial IVAs and the complexity of evaluating natural language communication. The system architecture and its various internal indicators of missed user intention are presented and evaluated in isolation and combined within the CRS.

The CRS is evaluated at three primary tasks in the IVA refinement cycle: the augmentation of the existing review process by maximizing the misunderstood user turns within a fixed sample size for review; the automation of suggesting an alternative intent when disagreement arises; the automation of the existing review process by replacing human reviewers. In addition, the system is shown capable of handling the volume and velocity of conversations logs originating from large-scale commercial IVAs.

The CRS demonstrates significant human productivity improvement when used to prioritize samples for human review. It can generate human labor cost savings by also automating the selection of alternative intents. The CRS is able to perform these tasks on large volumes of conversation logs using only a small number of commodity machines, making it feasible for companies of any size to deploy.

While the performance of the CRS compared to human voters is not so impressive that replacing humans altogether is an easy decision, the comparative performance is hindered by possible bias in the human scores, as well as many of the risk indicators the voting classifier uses as features had no samples in the evaluation data. Despite these drawbacks, performance on some of the evaluation datasets was near that of the average human reviewer, and the substantial cost savings and possible insights gained from processing the entire set of conversations in place of a small sample makes it a viable alternative to human reviewers.

Acronyms

AI Artificial Intelligence. 7, 8, 15

AIML Artificial Intelligence Markup Language. 13

ASR Automatic Speech Recognition. 7, 16, 17, 19, 23, 27, 29, 30, 92, 93, 138

CRS Chat Review System. vii, xi, xiii, xiv, xvi, xxii, 40, 41, 43, 51–76, 78, 79, 90, 107, 108, 171–178, 180–186, 196, 200, 201, 207, 208, 210–217, 219, 221–232

HCI Human Computer Interface. 26

IVA Intelligent Virtual Assistant. v, vi, ix, x, xiii, xv, xvi, xix, xx, 2–5, 7–12, 14–19, 21–30, 32, 34–38, 41–51, 54, 56–64, 70, 72, 74–76, 78, 80, 90, 92, 94, 97–99, 101, 103, 105–116, 125–128, 130, 132–145, 148, 150–153, 156, 158–162, 164–171, 173–180, 182–186, 190, 191, 196–198, 200, 201, 209–211, 214–219, 221–224, 226, 227, 230, 231

NLG Natural Language Generation. 17, 30, 41, 42, 58, 132

NLP Natural Language Processing. 18, 24

NLU Natural Language Understanding. 7, 11, 12, 17–21, 23, 25, 27, 30, 37, 41, 42, 45–47, 49–51, 57, 58, 92, 106, 132, 171, 180, 184, 185

PNI Potential New Intent. xiii, xvi, 59–61, 63, 186, 223, 229

SDS Spoken Dialog Systems. 27–29, 93, 94

TTS Text To Speech. 17, 138, 139

VIF Variance Inflation Factor. xxi, 192–194

XML Extensible Markup Language. 13

References

[1] X. Lu, "Automatic analysis of syntactic complexity in second language writing," *International Journal of Corpus Linguistics*, vol. 15, no. 4, pp. 474–496, 2010.

[2] Anonymous, "Intelligent virtual assistant." Technopedia, Retrieved 8-26-2016. Available online at `https://www.techopedia.com/definition/31383/intelligent-virtual-assistant`.

[3] E. Marois, "Using intelligent virtual agents to improve the customer experience: Brains before beauty." ICMI Blog, 2013. Available online at `http://www.icmi.com/Resources/Self-Service/2013/11/Using-Intelligent-Virtual-Agents-to-Improve-the-Customer-Experience`.

[4] J. Cifuentes, "Introducing the bots landscape: 170+ companies, $4 billion in funding, thousands of bots." `https://venturebeat.com/2016/08/11/introducing-the-bots-landscape-170-companies-4-billion-in-funding-thousands-of-bots/`, Retrieved 1-9-2018.

[5] J. Guynn, "Facebook messenger takes another swipe at bots." `https://www.usatoday.com/story/tech/news/2017/04/18/facebook-messenger-takes-another-swipe-chat-bots/100596798`, Retrieved 1-9-2018.

[6] A. Caramenico, "Aetna improves virtual customer experience." Fierce Healthcare, 2013. Available online at `http://www.fiercehealthcare.com/payer/aetna-improves-virtual-customer-experience`.

[7] J. Frearson, "From new york: How intelligent assistants are cutting customer waiting times." Business Reporter, 2015. Available online at `http://business-reporter.co.uk/2015/10/15/from-new-york-how-intelligent-assistants-are-cutting-customer-waiting-times`.

[8] Auxbreak, "The call center industry may come to an end soon according to some experts." Call Center Talk Blog, 2015. Available online at `http://www.auxbreak.com/call-center-industry-may-come-to-an-end-soon/`.

[9] R. Alzona, "Call centers predicted to disappear in 10 years." Business Mirror, 2016. Available online at `http://www.businessmirror.com.ph/2016/05/07/call-centers-predicted-to-disappear-in-10-years/`.

[10] J. Chu-Carroll, K. Czuba, J. Prager, and A. Ittycheriah, "In question answering, two heads are better than one," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pp. 24–31, Association for Computational Linguistics, 2003.

[11] R. B. Loatman, S. D. Post, C.-K. Yang, and J. C. Hermansen, "Natural language understanding system," Apr. 3 1990. US Patent 4,914,590.

[12] R. Kuhn and R. De Mori, "The application of semantic classification trees to natural language understanding," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 5, pp. 449–460, 1995.

[13] P. Lison, *Structured Probabilistic Modelling for Dialogue Management*. PhD thesis, University of Oslo, February 2014.

[14] A. Sarma and D. D. Palmer, "Context-based speech recognition error detection and correction," in *Proceedings of HLT-NAACL 2004: Short Papers*, pp. 85–88, Association for Computational Linguistics, 2004.

[15] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper, "Enriching speech recognition with automatic detection of sentence boundaries and disfluencies," *IEEE Transactions on audio, speech, and language processing*, vol. 14, no. 5, pp. 1526–1540, 2006.

[16] Y.-C. Tam, Y. Lei, J. Zheng, and W. Wang, "Asr error detection using recurrent neural network language model and complementary asr," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 2312–2316, IEEE, 2014.

[17] A. Ogawa and T. Hori, "Asr error detection and recognition rate estimation using deep bidirectional recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 4370–4374, IEEE, 2015.

[18] B. Kim, S. Ryu, and G. G. Lee, "Two-stage multi-intent detection for spoken language understanding," *Multimedia Tools and Applications*, pp. 1–14, 2016.

[19] B. A. Shawar and E. Atwell, “Chatbots: are they really useful?,” in *LDV Forum*, vol. 22, pp. 29–49, 2007.

[20] R. Cooper, J. McElroy, W. Rolandi, D. Sanders, and R. Ulmer, “Personal virtual assistant with semantic tagging,” Oct. 15 2002. US Patent 6,466,654.

[21] R. Cooper, J. McElroy, W. Rolandi, D. Sanders, R. Ulmer, and E. Peebles, “Personal virtual assistant,” June 29 2004. US Patent 6,757,362.

[22] M. L’Abbate, U. Thiel, and T. Kamps, “Can proactive behavior turn chatterbots into conversational agents?,” in *Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, pp. 173–179, IEEE, 2005.

[23] B. S. Hasler, P. Tuchman, and D. Friedman, “Virtual research assistants: Replacing human interviewers by automated avatars in virtual worlds,” *Computers in Human Behavior*, vol. 29, no. 4, pp. 1608–1616, 2013.

[24] D. Evon, “What is alexa? its amazons new virtual assistant.” `https://www.digitaltrends.com/home/what-is-amazons-alexa-and-what-can-it-do`, Retrieved 1-9-2018.

[25] A. John, R. Klemm, and D. Seligmann, “Method and apparatus for providing a virtual assistant to a communication participant,” Jan. 31 2012. US Patent 8,107,401.

[26] A. Cheyer and D. Guzzoni, “Method and apparatus for building an intelligent automated assistant,” Mar. 18 2014. US Patent 8,677,377.

[27] J. Sharkey, “A virtual travel agent with all the answers.” `http://www.nytimes.com/2008/03/04/business/04road.html`, Retrieved 1-9-2018.

[28] D. Jurafsky and J. H. Martin, *Speech and language processing*, vol. 3. Pearson London, 2014.

[29] K. Morik, “User modelling, dialog structure, and dialog strategy in ham-ans,” pp. 268–273, 01 1985.

[30] F. Bock, “Sgt star goes mobile; prospects get answers to questions anywhere, any time.” `https://www.army.mil/article/103582/sgt_star_goes_mobile_prospects_get_answers_to_questions_anywhere_any_time`, Retrieved 1-9-2018.

[31] A. M. Turing, “Computing machinery and intelligence,” *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[32] J. H. Moor, "An analysis of the turing test," *Philosophical Studies*, vol. 30, no. 4, pp. 249–257, 1976.

[33] M. L. Mauldin, "Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition," in *AAAI*, vol. 94, pp. 16–21, 1994.

[34] J. Weizenbaum, "Computer power and human reason: From judgment to calculation.," 1976.

[35] J. R. Carbonell, "Mixed-initiative man-computer instructional dialogues.," tech. rep., BOLT BERANEK AND NEWMAN INC CAMBRIDGE MASS, 1970.

[36] J. R. Carbonell, "Ai in cai: An artificial-intelligence approach to computer-assisted instruction," *IEEE transactions on man-machine systems*, vol. 11, no. 4, pp. 190–202, 1970.

[37] M. Boden, *Mind as Machine: A History of Cognitive Science*. Oxford University Press, 2006.

[38] A. P. Saygin, I. Cicekli, and V. Akman, "Turing test: 50 years later," *Minds and machines*, vol. 10, no. 4, pp. 463–518, 2000.

[39] D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson, and T. Winograd, "Gus, a frame-driven dialog system," *Artificial intelligence*, vol. 8, no. 2, pp. 155–173, 1977.

[40] R. Wilensky, D. N. Chin, M. Luria, J. Martin, J. Mayfield, and D. Wu, "The berkeley unix consultant project," *Computational Linguistics*, vol. 14, no. 4, pp. 35–84, 1988.

[41] P. R. Cohen, A. Cheyer, M. Wang, and S. C. Baeg, "An open agent architecture," in *AAAI Spring Symposium*, vol. 1, 1994.

[42] S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue, "Galaxy-ii: A reference architecture for conversational system development," in *Fifth International Conference on Spoken Language Processing*, 1998.

[43] R. S. Wallace, "The anatomy of alice," *Parsing the Turing Test*, pp. 181–210, 2009.

[44] A. Spink and O. Gunar, "E-commerce web queries: Excite and ask jeeves study," *First Monday*, vol. 6, no. 7, 2001.

[45] S. Withers, "Flirty bot passes for human." `http://www.itwire.com/your-it-news/home-it/15748-flirty-bot-passes-for-human`, 2007.

[46] R. Raine, "Making a clever intelligent agent: the theory behind the implementation," in *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, vol. 3, pp. 398–402, IEEE, 2009.

[47] D. Mass, "Answers and questions about military, law enforcement, and intelligence agency chatbots." Electronic Frontier Foundation, Retrieved 1-9-2018. Available online at `https://www.eff.org/deeplinks/2014/04/answers-questions-about-military-law-enforcement-and-intelligence-agency-chatbots`.

[48] C. Laorden, P. Galán-García, I. Santos, B. Sanz, J. M. G. Hidalgo, and P. G. Bringas, "Negobot: A conversational agent based on game theory for the detection of paedophile behaviour," in *International Joint Conference CISIS12-ICEUTE´ 12-SOCO´ 12 Special Sessions*, pp. 261–270, Springer, 2013.

[49] K. Kuligowska and M. Lasek, "Virtual assistants support customer relations and business processes," in *The 10th International Conference on Information Management, Gdańsk*, 2011.

[50] D. Hancock, "Comcast apologizes for changing customer's billing name to profanity." `http://www.cbsnews.com/news/comcast-apologizes-for-changing-customers-billing-name-to-profanity`, 2015.

[51] P. Bright, "Tay, the neo-nazi millennial chatbot, gets autopsied." Ars Technica, Retrieved 1-12-2018. Available online at `https://arstechnica.com/information-technology/2016/03/tay-the-neo-nazi-millennial-chatbot-gets-autopsied`.

[52] H. Reese, "Why microsoft's 'tay' ai bot went wrong." TechRepublic, Retrieved 1-12-2018. Available online at `https://www.techrepublic.com/article/why-microsofts-tay-ai-bot-went-wrong`.

[53] S. Haustein, T. D. Bowman, K. Holmberg, A. Tsou, C. R. Sugimoto, and V. Larivière, "Tweets as impact indicators: Examining the implications of automated bot accounts on twitter," *Journal of the Association for Information Science and Technology*, vol. 67, no. 1, pp. 232–238, 2016.

[54] Z. Gilani, R. Farahbakhsh, and J. Crowcroft, "Do bots impact twitter activity?," in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 781–782, International World Wide Web Conferences Steering Committee, 2017.

[55] N. Chavoshi, H. Hamooni, and A. Mueen, "Temporal patterns in bot activities," in *Proceedings of the 26th International Conference on World Wide Web*

Companion, pp. 1601–1606, International World Wide Web Conferences Steering Committee, 2017.

[56] M. Lynley, "With more than 6m daily users, slack opens up cross-organization teams." Techcrunch, Retrieved 1-9-2018. Available online at `https://techcrunch.com/2017/09/12/with-more-than-6m-daily-users-slack-opens-up-cross-organization-teams`.

[57] J. Polifroni and S. Seneff, "Galaxy-ii as an architecture for spoken dialogue evaluation.," in *LREC*, 2000.

[58] D. Bohus and A. I. Rudnicky, "The ravenclaw dialog management framework: Architecture and systems," *Computer Speech & Language*, vol. 23, no. 3, pp. 332–361, 2009.

[59] H. Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira, "Evaluation of a hierarchical reinforcement learning spoken dialogue system," *Computer Speech & Language*, vol. 24, no. 2, pp. 395–429, 2010.

[60] J. Dumoulin, "Using multiple classifiers to improve intent recognition in human chats," in *Modern Artificial Intelligence and Cognitive Science Conference 2014* (A. Inoue and P. DePalma, eds.), no. 1144 in CEUR Workshop Proceedings, pp. 10–21, 2014.

[61] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, pp. 173–182, 2016.

[62] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 4945–4949, IEEE, 2016.

[63] T. Zhao and M. Eskenazi, "Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning," *arXiv preprint arXiv:1606.02560*, 2016.

[64] A. Bordes and J. Weston, "Learning end-to-end goal-oriented dialog," *arXiv preprint arXiv:1605.07683*, 2016.

[65] M. Ghazvininejad, C. Brockett, M.-W. Chang, B. Dolan, J. Gao, W.-t. Yih, and M. Galley, "A knowledge-grounded neural conversation model," *arXiv preprint arXiv:1702.01932*, 2017.

[66] C. Xiong, V. Zhong, and R. Socher, "Dynamic coattention networks for question answering," *arXiv preprint arXiv:1611.01604*, 2016.

[67] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," *arXiv preprint arXiv:1611.01603*, 2016.

[68] J. Hankamer, "The semantic interpretation of anaphoric expressions," *Georgetown University Round Table on Languages and Linguistics*, 1976.

[69] S. Lappin and H. J. Leass, "An algorithm for pronominal anaphora resolution," *Computational linguistics*, vol. 20, no. 4, pp. 535–561, 1994.

[70] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Dept. of Informatics, Aristotle University of Thessaloniki, Greece*, 2006.

[71] P. Xu and R. Sarikaya, "Exploiting shared information for multi-intent natural language sentence classification.," in *INTERSPEECH*, pp. 3785–3789, 2013.

[72] Y.-N. Chen, M. Sun, and A. I. Rudnicky, "Matrix factorization with domain knowledge and behavioral patterns for intent modeling," in *NIPS Workshop on Machine Learning for SLU and Interaction*, 2015.

[73] Y.-N. Chen and X. He, "Learning bidirectional intent embeddings by convolutional deep structured semantic models for spoken language understanding," in *Extended Abstract of The 29th Annual Conference on Neural Information Processing Systems–Machine Learning for Spoken Language Understanding and Interactions Workshop (NIPS-SLU)*, 2015.

[74] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *arXiv preprint arXiv:1609.01454*, 2016.

[75] I. Beaver, C. Freeman, and A. Mueen, "An Annotated Corpus of Relational Strategies in Customer Service," *arXiv preprint arXiv:1708.05449*, Aug. 2017.

[76] T.-H. Wen, M. Gasic, N. Mrksic, L. M. Rojas-Barahona, P.-H. Su, D. Vandyke, and S. Young, "Multi-domain neural network language generation for spoken dialogue systems," *arXiv preprint arXiv:1603.01232*, 2016.

[77] K. Kuligowska, "Commercial chatbot: Performance evaluation, usability metrics and quality standards of embodied conversational agents," *Browser Download This Paper*, 2015.

[78] P. R. Cohen, J. L. Morgan, and M. E. Pollack, *Intentions in communication.* MIT press, 1990.

[79] T. Holtgraves, "Automatic intention recognition in conversation processing," *Journal of Memory and Language*, vol. 58, no. 3, pp. 627–645, 2008.

[80] C. A. Montero and K. Araki, "Enhancing computer chat: Toward a smooth user-computer interaction," in *Knowledge-Based Intelligent Information and Engineering Systems*, pp. 918–924, Springer, 2005.

[81] J. Jiang, A. Hassan Awadallah, R. Jones, U. Ozertem, I. Zitouni, R. Gurunath Kulkarni, and O. Z. Khan, "Automatic online evaluation of intelligent assistants," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 506–516, ACM, 2015.

[82] S. S. Pradhan and W. H. Ward, "Estimating semantic confidence for spoken dialogue systems," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 1, pp. I–233, IEEE, 2002.

[83] F. Wessel, R. Schlüter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 9, no. 3, pp. 288–298, 2001.

[84] A. van den Bosch, E. Krahmer, and M. Swerts, "Detecting problematic turns in human-machine interactions: Rule-induction versus memory-based learning approaches," in *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 82–89, Association for Computational Linguistics, 2001.

[85] P. Lendvai, A. van den Bosch, E. Krahmer, and M. Swerts, "Multi-feature error detection in spoken dialogue systems," *Language and Computers*, vol. 45, no. 1, pp. 163–178, 2002.

[86] W. Kim, "Using prosody for automatically monitoring human-computer call dialogues," *Proceedings of Speech Prosody 2008*, pp. 79–82, 2008.

[87] M. Swerts, D. J. Litman, and J. Hirschberg, "Corrections in spoken dialogue systems.," in *INTERSPEECH*, pp. 615–618, Citeseer, 2000.

[88] J. Hirschberg, D. J. Litman, and M. Swerts, "Generalizing prosodic prediction of speech recognition errors.," in *INTERSPEECH*, pp. 254–257, 2000.

[89] P. Lendvai, A. Van Den Bosch, E. Krahmer, and M. Swerts, "Improving machine-learned detection of miscommunications in human-machine dialogues through informed data splitting," in *Proceedings of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics*, pp. 1–15, Citeseer, 2002.

[90] M. A. Walker, I. Langkilde-Geary, H. Wright Hastie, J. Wright, and A. Gorin, "Automatically training a problematic dialogue predictor for a spoken dialogue system," *Journal of Artificial Intelligence Research*, vol. 16, pp. 293–319, 2002.

[91] J. Hirschberg, D. Litman, and M. Swerts, "Identifying user corrections automatically in spoken dialogue systems," in *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pp. 1–8, Association for Computational Linguistics, 2001.

[92] R. Higashinaka, K. Sudoh, and M. Nakano, "Incorporating discourse features into confidence scoring of intention recognition results in spoken dialogue systems," *Speech Communication*, vol. 48, no. 3, pp. 417–436, 2006.

[93] D. Goddeau, H. Meng, J. Polifroni, S. Seneff, and S. Busayapongchai, "A form-based dialogue manager for spoken language applications," in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, vol. 2, pp. 701–704, IEEE, 1996.

[94] E. Krahmer, M. Swerts, M. Theune, and M. Weegels, "Problem spotting in human-machine interaction," 1999.

[95] H. H. Clark and D. Wilkes-Gibbs, "Referring as a collaborative process," *Cognition*, vol. 22, no. 1, pp. 1–39, 1986.

[96] H. H. Clark, S. E. Brennan, *et al.*, "Grounding in communication," *Perspectives on socially shared cognition*, vol. 13, no. 1991, pp. 127–149, 1991.

[97] S. Roy, R. Mariappan, S. Dandapat, S. Srivastava, S. Galhotra, and B. Peddamuthu, "Qa rt: A system for real-time holistic quality assurance for contact center dialogues," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[98] H. P. Grice, "Logic and conversation," *Syntax and semantics*, vol. Vol. 3, Speech Acts, pp. 41–58, 1975.

[99] J. E. F. Tree and J. C. Schrock, "Basic meanings of you know and i mean," *Journal of Pragmatics*, vol. 34, no. 6, pp. 727–747, 2002.

[100] A. Mauranen, "Signaling and preventing misunderstanding in english as lingua franca communication," *International Journal of the Sociology of Language*, pp. 123–150, 2006.

[101] E. A. Schegloff and H. Sacks, "Opening up closings," *Semiotica*, vol. 8, no. 4, pp. 289–327, 1973.

[102] B. L. Davies, *An empirical examination of cooperation, effort and risk in task-oriented dialogue*. PhD thesis, University of Edinburgh, 1997.

[103] H. H. Clark and C. R. Marshall, "Definite reference and mutual knowledge," 1981.

[104] J. Aberdeen and L. Ferro, "Dialogue patterns and misunderstandings," in *ISCA Tutorial and Research Workshop on Error Handling in Spoken Dialogue Systems*, 2003.

[105] D. Bohus and A. I. Rudnicky, "Sorry, i didn't catch that!-an investigation of non-understanding errors and recovery strategies," in *6th SIGdial Workshop on Discourse and Dialogue*, 2005.

[106] R. Fagin, R. Kumar, and D. Sivakumar, "Efficient similarity search and classification via rank aggregation," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pp. 301–312, ACM, 2003.

[107] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web.," 1999.

[108] S. Brin and L. Page, "Reprint of: The anatomy of a large-scale hypertextual web search engine," *Computer networks*, vol. 56, no. 18, pp. 3825–3833, 2012.

[109] T. H. Haveliwala, "Topic-sensitive pagerank," in *Proceedings of the 11th international conference on World Wide Web*, pp. 517–526, ACM, 2002.

[110] I. Beaver and C. Freeman, "Prioritization of risky chats for intent classifier improvement," in *Florida Artificial Intelligence Research Society Conference*, vol. 29, pp. 167–172, The AAAI Press, Palo Alto, California, 2016.

[111] D. Bohus, *Error awareness and recovery in conversational spoken language interfaces*. PhD thesis, SRI International, 2007.

[112] I. Bulyko, K. Kirchhoff, M. Ostendorf, and J. Goldberg, "Error-correction detection and response generation in a spoken dialogue system," *Speech Communication*, vol. 45, no. 3, pp. 271–288, 2005.

[113] C. Freeman and I. Beaver, "Online proactive escalation in multi-modal automated assistants," in *Florida Artificial Intelligence Research Society Conference*, vol. 30, pp. 215–220, The AAAI Press, Palo Alto, California, 2017.

[114] I. Beaver and C. Freeman, "Detection of user escalation in human-computer interactions," in *INTERSPEECH*, 2016.

[115] J. Onken, R. Hastie, and W. Revelle, "Individual differences in the use of simplification strategies in a complex decision-making task," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 11, pp. 14–27, 2 1985.

[116] B. Settles, "Active learning literature survey," Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[117] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 3–12, Springer-Verlag New York, Inc., 1994.

[118] F. Liu, P. Shu, H. Jin, L. Ding, J. Yu, D. Niu, and B. Li, "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless communications*, vol. 20, no. 3, pp. 14–22, 2013.

[119] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos, "Management of an academic hpc cluster: The ul experience," in *High Performance Computing & Simulation (HPCS), 2014 International Conference on*, pp. 959–967, IEEE, 2014.

[120] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[121] Aetna.com, "Aetna members ask 'ann': Where have you been all my life?." `http://web.archive.org/web/20111111064429/http://www.aetna.com/news/newsReleases/2011/0629_Ann.html`, 2011.

[122] NextIT.com, "Case study: Charter communications." Next IT, Retrieved 2-23-2018. Available online at `http://www.nextit.com/case-studies/charter`.

[123] E. Horowitz, "The secret sauce of sharding." MongoSF, 2011. Available online at `http://www.mongodb.com/presentations/secret-sauce-sharding`.

[124] I. Beaver and J. Dumoulin, "Applying mapreduce to learning user preferences in near real-time," in *Case-Based Reasoning Research and Development*, pp. 15–28, Springer, 2013.

[125] I. Beaver and J. Dumoulin, "Scaling user preference learning in near real-time to large datasets," in *Modern Artificial Intelligence and Cognitive Science Conference 2014*, vol. 1144, pp. 68–73, CEUR-WS.org, 2014.

[126] S. S. Skiena, *The algorithm design manual: Text*, vol. 1. Springer Science & Business Media, 1998.

[127] O. Lemon and O. Pietquin, *Data-Driven Methods for Adaptive Spoken Dialogue Systems: Computational Learning for Conversational Interfaces.* Springer New York, 2012.

[128] M. Walker, I. Langkilde, J. Wright, A. Gorin, and D. Litman, "Learning to predict problematic situations in a spoken dialogue system: experiments with how may i help you?," in *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pp. 210–217, Association for Computational Linguistics, 2000.

[129] E. Levin and R. Pieraccini, "Value-based optimal decision for dialog systems," in *Spoken Language Technology Workshop, 2006. IEEE*, pp. 198–201, IEEE, 2006.

[130] W. Kim, "Online call quality monitoring for automating agent-based call centers," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[131] T. Bosse and S. Provoost, "On conversational agents with mental states," in *Intelligent Virtual Agents*, pp. 60–64, Springer, 2015.

[132] S. Loria, "Textblob: simplified text processing," *Secondary TextBlob: Simplified Text Processing*, 2014.

[133] T. Joachims, *Learning to classify text using support vector machines: Methods, theory and algorithms.* Kluwer Academic Publishers, 2002.

[134] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text classification using machine learning techniques.," *WSEAS Transactions on Computers*, vol. 4, no. 8, pp. 966–974, 2005.

[135] T. Joachims, *Text categorization with support vector machines: Learning with many relevant features.* Springer, 1998.

[136] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[137] Various, "sklearn linearsvc documentation," 2016. Available online at `http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html`.

[138] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[139] J. Ramos, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, 2003.

[140] D. Hiemstra, "A probabilistic justification for using tf× idf term weighting in information retrieval," *International Journal on Digital Libraries*, vol. 3, no. 2, pp. 131–139, 2000.

[141] H. Taira and M. Haruno, "Feature selection in svm text categorization," in *AAAI/IAAI*, pp. 480–486, 1999.

[142] M. Honnibal, "A good part-of-speech tagger in about 200 lines of python." spaCy Blog, 2013. Available online at `https://spacy.io/blog/part-of-speech-POS-tagger-in-python`.

[143] M. A. Elmi and M. Evens, "Spelling correction using context," in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pp. 360–364, Association for Computational Linguistics, 1998.

[144] M. Schierle, S. Schulz, and M. Ackermann, "From spelling correction to text cleaning–using context information," in *Data Analysis, Machine Learning and Applications*, pp. 397–404, Springer, 2008.

[145] M. F. Porter, "Snowball: A language for stemming algorithms," 2001.

[146] E. Leopold and J. Kindermann, "Text categorization with support vector machines. how to represent texts in input space?," *Machine Learning*, vol. 46, no. 1-3, pp. 423–444, 2002.

[147] S. Bird, "Nltk: the natural language toolkit," in *Proceedings of the COLING/ACL on Interactive presentation sessions*, pp. 69–72, Association for Computational Linguistics, 2006.

[148] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 231–240, ACM, 2008.

[149] P. Lison, *Structured Probabilistic Modelling for Dialogue Management*. PhD thesis, University of Oslo, 2013.

[150] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may I help you?," *Speech communication*, vol. 23, no. 1, pp. 113–127, 1997.

[151] I. Langkilde, M. Walker, J. Wright, A. Gorin, and D. Litman, "Automatic prediction of problematic human-computer dialogues in How May I Help You," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, ASRUU99*, pp. 1–4, Citeseer, 1999.

[152] A. Schmitt, M. Scholz, W. Minker, J. Liscombe, and D. Sündermann, "Is it possible to predict task completion in automated troubleshooters?.," in *INTERSPEECH*, pp. 94–97, 2010.

[153] R. Meena, J. L. G. Skantze, and J. Gustafson, "Automatic detection of miscommunication in spoken dialogue systems," in *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 354, 2015.

[154] E. Horvitz and T. Paek, "Utility-directed coupling of spoken dialog systems and human operators for call routing," tech. rep., MSR Technical Report 2003, 2003.

[155] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[156] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[157] A. Rakhlin, "Cnn for sentence classification in keras." `https://github.com/alexander-rakhlin/CNN-for-Sentence-Classification-in-Keras`, 2016.

[158] N. Sanders, "Twitter sentiment corpus." `http://www.sananalytics.com/lab/twitter-sentiment/`, 2011. accessed 2016-05-19.

[159] S. Levy, "Alexa, tell me where you're going next." Backchannel.com, 2016. Available online at `https://backchannel.com/alexa-tell-me-where-youre-going-next-739c53ff10b3`.

[160] C. B. Gibson and S. G. Cohen, "Virtual teams that work," *JosseyBass, San Francisco*, 2003.

[161] D. Ballantyne, "Dialogue and its role in the development of relationship specific knowledge," *Journal of Business & Industrial Marketing*, vol. 19, no. 2, pp. 114–123, 2004.

[162] J. A. Holton, "Building trust and collaboration in a virtual team," *Team performance management: an international journal*, vol. 7, no. 3/4, pp. 36–47, 2001.

[163] N. W. Coppola, S. R. Hiltz, and N. G. Rotter, "Building trust in virtual teams," *IEEE transactions on professional communication*, vol. 47, no. 2, pp. 95–104, 2004.

[164] E. J. de Visser, S. S. Monfort, R. McKendrick, M. A. Smith, P. E. McKnight, F. Krueger, and R. Parasuraman, "Almost human: Anthropomorphism increases trust resilience in cognitive agents.," *Journal of Experimental Psychology: Applied*, vol. 22, no. 3, p. 331, 2016.

[165] T. Bickmore and J. Cassell, "Relational agents: a model and implementation of building user trust," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 396–403, ACM, 2001.

[166] J. Y. Chai, C. Zhang, and T. Baldwin, "Towards conversational qa: automatic identification of problematic situations and user intent," in *Proceedings of the COLING/ACL on Main conference poster sessions*, pp. 57–64, Association for Computational Linguistics, 2006.

[167] I. V. Serban, R. Lowe, P. Henderson, L. Charlin, and J. Pineau, "A survey of available corpora for building data-driven dialogue systems," *CoRR*, vol. abs/1512.05742, 2015.

[168] R. Sarikaya, P. Crook, A. Marin, M. Jeong, J.-P. Robichaud, A. Celikyilmaz, Y.-B. Kim, A. Rochette, O. Z. Khan, X. Liu, *et al.*, "An overview of end-to-end language understanding and dialog management for personal digital assistants," in *IEEE Workshop on Spoken Language Technology*, 2016.

[169] Bentley.edu, "Bentley graduate students reveal how users respond to chatbots compared to human agents." Bentley.edu, 2017. Available online at `http://www.bentley.edu/prepared/bentley-graduate-students-reveal-how-users-respond-chatbots-compared-human-agents`.

[170] P. Madhavan, D. A. Wiegmann, and F. C. Lacson, "Automation failures on tasks easily performed by operators undermine trust in automated aids," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 48, no. 2, pp. 241–256, 2006.

[171] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.

[172] S. Roy, R. Mariappan, S. Dandapat, S. Srivastava, S. Galhotra, and B. Peddamuthu, "Qart: A system for real-time holistic quality assurance for contact center dialogues.," in *AAAI*, pp. 3768–3775, 2016.

[173] R. T. Lowe, N. Pow, I. V. Serban, L. Charlin, C.-W. Liu, and J. Pineau, "Training end-to-end dialogue systems with the ubuntu dialogue corpus," *Dialogue & Discourse*, vol. 8, no. 1, pp. 31–65, 2017.

[174] K. Zechner and A. Waibel, "Minimizing word error rate in textual summaries of spoken language," in *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pp. 186–193, Association for Computational Linguistics, 2000.

[175] A. R. Feinstein and D. V. Cicchetti, "High agreement but low kappa: I. the problems of two paradoxes," *Journal of clinical epidemiology*, vol. 43, no. 6, pp. 543–549, 1990.

[176] I. Guggenmoos-Holzmann, "How reliable are change-corrected measures of agreement?," *Statistics in Medicine*, vol. 12, no. 23, pp. 2191–2205, 1993.

[177] R. Bakeman, D. McArthur, V. Quera, and B. F. Robinson, "Detecting sequential patterns and determining their reliability with fallible observers.," *Psychological Methods*, vol. 2, no. 4, p. 357, 1997.

[178] M. K. Lee, S. Kiesler, and J. Forlizzi, "Receptionist or information kiosk: How do people talk with a robot?," in *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pp. 31–40, ACM, 2010.

[179] A. Laflen and B. Fiorenza, ""okay, my rant is over": The language of emotion in computer-mediated communication," *Computers and Composition*, vol. 29, no. 4, pp. 296–308, 2012.

[180] M. Makatchev, M. K. Lee, and R. Simmons, "Relating initial turns of human-robot dialogues to discourse," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pp. 321–322, ACM, 2009.

[181] H. H. Clark, "Using language. 1996," *Cambridge University Press: Cambridge)*, vol. 952, pp. 274–296, 1996.

[182] C. Freeman and I. Beaver, "The effect of response complexity and media on user restatement with multimodal virtual assistants," *International Journal of Human-Computer Studies*, vol. pending, 2018.

[183] I. Beaver and C. Freeman, "Analysis of user behavior with multimodal virtual customer service agents," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 585–591, Dec 2016.

[184] M. Johnston, J. Chen, P. Ehlen, H. Jung, J. Lieske, A. Reddy, E. Selfridge, S. Stoyanchev, B. Vasilieff, and J. Wilpon, "Mva: The multimodal virtual assistant," in *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 257, 2014.

[185] L. Dybkjaer, N. O. Bernsen, and W. Minker, "Evaluation and usability of multimodal spoken language dialogue systems," *Speech Communication*, vol. 43, no. 1, pp. 33–54, 2004.

[186] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow, "Quickset: Multimodal interaction for distributed applications," in *Proceedings of the fifth ACM international conference on Multimedia*, pp. 31–40, ACM, 1997.

[187] S. Oviatt, "Multimodal interactive maps: Designing for human performance," *Human-computer interaction*, vol. 12, no. 1, pp. 93–129, 1997.

[188] S. F. Roth, M. C. Chuah, S. Kerpedjiev, J. A. Kolojejchick, and P. Lucas, "Toward an information visualization workspace: combining multiple means of expression," *Human-computer interaction*, vol. 12, no. 1, pp. 131–185, 1997.

[189] P. Ehlen and M. Johnston, "Location grounding in multimodal local search," in *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*, p. 32, ACM, 2010.

[190] X. Niu and D. Kelly, "The use of query suggestions during information search," *Information Processing & Management*, vol. 50, no. 1, pp. 218–234, 2014.

[191] B. J. Jansen, D. L. Booth, and A. Spink, "Patterns of query reformulation during web searching," *Journal of the Association for Information Science and Technology*, vol. 60, no. 7, pp. 1358–1371, 2009.

[192] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger, "The perfect search engine is not enough: a study of orienteering behavior in directed search," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 415–422, ACM, 2004.

[193] R. Higashinaka, K. Funakoshi, M. Araki, H. Tsukahara, Y. Kobayashi, and M. Mizukami, "Towards taxonomy of errors in chat-oriented dialogue systems," in *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 87, 2015.

[194] R. G. Pontius Jr and M. Millones, "Death to kappa: birth of quantity disagreement and allocation disagreement for accuracy assessment," *International Journal of Remote Sensing*, vol. 32, no. 15, pp. 4407–4429, 2011.

[195] X. Lu, "The relationship of lexical richness to the quality of esl learners oral narratives," *The Modern Language Journal*, vol. 96, no. 2, pp. 190–208, 2012.

[196] P. R. Kroeger, *Analyzing grammar: An introduction.* Cambridge University Press, 2005.

[197] LinguaLinks, "What is a nominal?," 2004. Accessed: 2017-06-05.

[198] C. J. Nachtsheim, J. Neter, M. H. Kutner, and W. Wasserman, "Applied linear regression models," *McGraw-Hill Irwin*, 2004.

[199] G. Casella and R. L. Berger, *Statistical inference*, vol. 2. Duxbury Pacific Grove, CA, 2002.

[200] M. Szumilas, "Explaining odds ratios," *Journal of the Canadian Academy of Child and Adolescent Psychiatry*, vol. 19, no. 3, p. 227, 2010.

[201] V. Pallotta and R. Delmonte, "Interaction mining: the new frontier of customer interaction analytics," in *New Challenges in Distributed Information Filtering and Retrieval*, pp. 91–111, Springer, 2013.

[202] J. M. Silverman, "Systems and methods for monitoring quality of customer service in customer/agent calls over a voip network," 2014. US Patent 8,665,863.

[203] E. Yellin, *Your call is (not that) important to us: Customer service and what it reveals about our world and our lives.* Simon and Schuster, 2009.

[204] M. Gamon, "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," in *Proceedings of the 20th international conference on Computational Linguistics*, p. 841, Association for Computational Linguistics, 2004.

[205] A. Bagheri, M. Saraee, and F. De Jong, "Care more about customers: unsupervised domain-independent aspect detection for sentiment analysis of customer reviews," *Knowledge-Based Systems*, vol. 52, pp. 201–213, 2013.

[206] D. Kang and Y. Park, "based measurement of customer satisfaction in mobile service: Sentiment analysis and vikor approach," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1041–1050, 2014.

[207] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 142–150, Association for Computational Linguistics, 2011.

[208] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[209] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[210] G. Van Dessel, "Measure customer satisfaction: Csat, ces and nps compared." CheckMarket, Retrieved 3-28-2018. Available online at `https://www.checkmarket.com/blog/csat-ces-nps-compared`.

[211] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in large margin classifiers*, Citeseer, 1999.

[212] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[213] J. W. Parrish, M. B. Young, K. A. Perham-Hester, and B. D. Gessner, "Identifying risk factors for child maltreatment in alaska: A population-based approach," *American Journal of preventive medicine*, vol. 40, no. 6, pp. 666–673, 2011.

[214] J. M. Etchegaray, A. J. Ottenbacher, F. Sittig, and A. B. McCoy, "Understanding evidence-based research methods: Survey analysis, t-tests, and odds ratios," *HERD: Health Environments Research & Design Journal*, vol. 6, no. 1, pp. 143–147, 2012.

[215] M. Szumilas, "Explaining odds ratios," *Journal of the Canadian Academy of Child and Adolescent Psychiatry*, vol. 19, no. 3, p. 227, 2010.

[216] S. C. G. UCLA, "How do i interpret odds ratios in logistic regression?," Retrieved 3-28-2018. https://stats.idre.ucla.edu/stata/faq/how-do-i-interpret-odds-ratios-in-logistic-regression.

[217] S. I. Santos, *Cancer epidemiology: principles and methods.* IARC, 1999.

[218] S. C. G. UCLA, "How do i use odds ratio to interpret logistic regression?," Retrieved 3-28-2018. https://stats.idre.ucla.edu/other/mult-pkg/faq/general/faq-how-do-i-interpret-odds-ratios-in-logistic-regression/.

[219] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[220] M. H. Kutner, C. Nachtsheim, and J. Neter, *Applied linear regression models.* McGraw-Hill/Irwin, 2004.

[221] S. Sheather, *A modern approach to regression with R.* Springer Science & Business Media, 2009.

[222] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to information retrieval*, vol. 39. Cambridge University Press, 2008.

[223] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[224] V. Van Asch, "Macro-and micro-averaged evaluation measures," *Tech. Rep.*, 2013.

[225] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.

[226] U. Orhan, M. Hekim, and M. Ozer, "Eeg signals classification using the k-means clustering and a multilayer perceptron neural network model," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13475–13481, 2011.

[227] R. Linggard, D. Myers, and C. Nightingale, *Neural networks for vision, speech and natural language*, vol. 1. Springer Science & Business Media, 2012.

[228] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[229] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[230] Y. Freund, R. E. Schapire, *et al.*, "Experiments with a new boosting algorithm," in *Icml*, vol. 96, pp. 148–156, Bari, Italy, 1996.

[231] J. Bernardo, J. Berger, A. Dawid, A. Smith, *et al.*, "Regression and classification using gaussian process priors," *Bayesian statistics*, vol. 6, p. 475, 1998.

[232] R. Herbrich, N. D. Lawrence, and M. Seeger, "Fast sparse gaussian process methods: The informative vector machine," in *Advances in neural information processing systems*, pp. 625–632, 2003.

[233] P. Cunningham and S. J. Delany, "k-nearest neighbour classifiers," *Multiple Classifier Systems*, vol. 34, pp. 1–17, 2007.

[234] D. D. Buss, "Call centers: Reshoring instead of offshoring," Retrieved 3-28-2018. http://www.areadevelopment.com/BusinessGlobalization/Q3-2015/call-centers-reshoring-instead-of-offshoring-271444.shtml.

[235] N. Kushman and R. Barzilay, "Using semantic unification to generate regular expressions from natural language," North American Chapter of the Association for Computational Linguistics (NAACL), 2013.