Ana: Automated Nursing Agent

by

Kevin Quinn

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science
University of Alberta

Abstract

Loneliness among the elderly is a growing problem in today's society. Many of them live in isolation either due to their ailments or losing loved ones. Measures need to be taken to improve the living condition of these people.

We propose a system that engages in meaningful conversation with the elderly. This system can extract useful information from the dialogue as well as use past information to generate a response.

In order to show our system can converse in a meaningful way we evaluate our system with two evaluation studies. The first examines the quality of the information extraction. The second puts the users in a scenario and actively engages in conversation with two agents.

Preface

Section 3.7 of this thesis contains published content on information extraction. This work was published as Xu, Ying and Kim, Mi-Young and Quinn, Kevin and Goebel, Randy and Barbosa, Denilson, "Open Information Extraction with Tree Kernels," Proceedings of NAACL-HLT, 868-877. My contribution to this work includes labelling data and providing feedback to the primary author, Ying Xu.

Acknowledgements

I would like to begin by thanking my supervisor Prof. Osmar Zaiane. I would not have made it this far without his belief in me and his guidance. I would also like to thank Prof. Greg Kondrak, Prof. Randy Goebel, and Ying Xu. Finally a big thanks to my family and friends.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The goal of most artificial intelligence work is to create a system that can mimic a human. This is usually intended to replace or replicate the function of a human to make our lives a little easier. One area that has recently become popular is using artificial intelligence to aid in people's lives. In this work we extend this idea to using artificial intelligence to aid the elderly. We believe that systems like this will improve the quality of life for many individuals and promote the well-being of senior citizens.

1.1 Elderly

Around 41 million Americans are 65 and older, and they make up about 13% of the total U.S. population. Due to rising life expectancies world-wide, the population of seniors living alone at home is increasing. According to the U.S. Census Bureau projections, by 2050, one-in-five Americans will be 65 or older, and at least 400,000 will be 100 years or older [6]. Not all will be living assisted in collective dwellings. A large proportion will be living in their own homes, and very often alone. In 2011, the Census of Population in Canada counted nearly 5 million seniors aged 65 and over of which 92.1% lived in private households [39]. The benefits of companionship for the elderly to maintain a good state of their physical and mental health has been demonstrated [34] and many seniors' companionship services exist. However, these services are not always available and certainly do not scale with the expected significant rise of seniors' population. One solution is a software agent that could converse intelligently and that could be embedded in tablets or other domestic appliances [11].

1.2 Automated Nursing Agent

Our system ANA, Automated Nursing Agent, is an intelligent conversational agent that aims to aid the elderly in their daily lives. ANA is designed to engage the elderly in meaningful dialogue while also reminding them and making sure their medical needs are met. The elderly can be quite lonely and ANA would provide an avenue for them to speak with a companion which they may not have access to. This conversation is important for their cognitive well-being.

A number of different mental disorders affect the elderly. Among them are Alzheimer's, anxiety, depression, and dementia. These disorders drastically reduce their quality of life. We believe that ANA has the potential to alleviate their symptoms by providing conversation. It is not always possible for the elderly to engage in conversation with others, but if a system were developed it could be distributed to any who were interested.

An area of life that most take for granted is the ability to remember. We know what we had for dinner yesterday, but a senior may not. ANA has the potential to improve their forgetfulness and reduce anxiety. The user is able to query ANA about various topics such as information about themselves or their family. We believe that this will both improve their memory and reduce their anxiety.

The elderly can have multiple medical issues that need to be monitored. With ANA we would like the ability to identify symptoms or precursors such that the nurse or doctor could be alerted. ANA would monitor the incoming dialogue for indicators. This data would be invaluable for nurses and doctors. It could potentially save or extend a life.

ANA's process at a high level is illustrated in Figure 1.1. The process begins with a pre-processing step which extracts information from the input. After this step, we determine the speech act of the input. By this we mean whether the input is a statement, question, or request. Next we insert relevant information into the knowledge base and ultimately respond to the user. Note that this is a pipelined approach which means that the improvement of individual components may not necessarily improve the performance of the entire system.

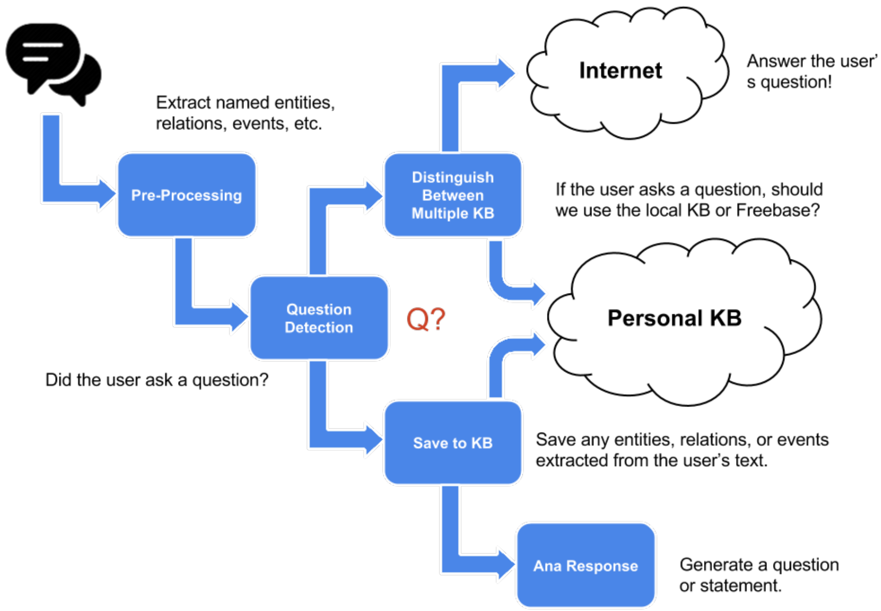


Figure 1.1: High level structure of ANA.

The current prototype of ANA has focused on extracting information from text, building a personalized knowledge base, and generating a response. Our information extraction process is able to extract named entities, relations, attributes and social events from text. This information is placed inside a knowledge base. Our response generation makes simple responses using pre-defined questions combined with a similarity measure.

The main portion of the work in this thesis focuses on extracting relations, events, and attributes as well as identifying the speech act of the input, and ranking questions. Our preliminary prototype used in our user study is able to converse with a user by asking about the mentioned entities and their attributes. It can also ask questions about various events that the user mentions.

The primary objective of this initial prototype of ANA is to build a system that is able to collect useful information and generation responses that are based on a personalized knowledge base. However our prototype is still in the early stages of development and will need further refinement before using it in the context of the elderly.

1.3 Contributions

In this section we will discuss the contributions that ANA makes. Our system is built upon a basic information extraction framework using Stanford's CoreNLP tools. ANA extends this system to incorporate relations, attributes, and events. We also provide a process for generating meaningful response to human generated input. This process uses pre-defined questions and a similarity measure to select an appropriate output.

1.3.1 Knowledge Bases

A knowledge base is a collection of information describing various entities and their attributes. An example of a knowledge base would be Freebase or Wolfram Alpha. These knowledge bases aim to create a database of structured information for influential entities. This information includes relationships such as the spouse of Barack Obama or geological information such as the length of the Nile river. In contrast, a personalized knowledge base contains entities and relations specific to an individual. This information includes family members, favourite foods, or their education. Our assumption is that a conversational agent that possesses a personalized knowledge base will be able to maintain a higher level of dialogue than an agent without a knowledge base. Ideally we would like to leverage both types of knowledges bases.

1.3.2 Information Extraction

Information extraction is the task of taking a collection of unstructured documents, extracting important information, and converting it into a structured format. Extracting useful information from conversation is essential to fluent conversation. Agents that do not remember facts previously stated from past dialogue will sound less human. ANA is able extract named entities, relationships, events, and medical information from text. In addition it performs basic pronoun resolution in order to accurately assign information to entities. Information extraction on web documents is hard, but attempting to extract information from conversational text is even harder. This is because conversation is normally informal and can lack punctuation, grammar, and capitalization. Our hypothesis is that a personalized knowledge base will lead to better conversation.

An aspect of conversation that engages and interests the elderly are events. Specifically, events that we engage in our everyday life. These include menial tasks such as going shopping or special events like birthdays or celebrations. Discussing these events with each other makes us feel connected to one another. We feel it is important to extract these events when taking part in a conversation.

Our definition of an event is when a person or group of people gather for an activity. The correct event could be a single word or a phrase. Phrases such as "going out for dinner" or words such as "concert" can both be considered events.

1.3.3 Response Generation

When participating in a conversation your response depends on a variety of factors. Amongst these factors are past dialogue, entities discussed, relations mentioned, implied information, and so forth. ANA attempts to leverage information extraction to both generate and store useful information within a dialogue. Our hypothesis is that the response generated by ANA will provide meaningful conversation for the elderly and assist them in their day to day lives. The current iterations of conversational agents, such as chatterbots, usually do not consider past dialogue when determining how to respond. We believe that this is flawed and a system that truly wants to respond intelligently needs a way to save past information and use it to respond. Google Now, a conversational search tool, examines past search queries to clarify pronouns in the current search query. If the pronoun "he" is present in the current query Google Now examines the previous query for male persons. If found, the query will be modified with the correct entity inserted. We hypothesize that a response can be generated that is superior to a chatterbot.

1.4 Outline

This thesis is organized as follows. The second chapter discusses the previous work and how it applies to the current research. The third chapter presents how information extraction is used and expanded upon to extract information from conversational data. The fourth chapter determines when to respond and what the response should be. The last chapter elaborates on final thoughts, problems, and future work.

Chapter 2

Related Work

We will now discuss the related work for this thesis. Most of the work revolves around different technologies for aiding the elderly, such as the robot Wakamaru, and components such as machine learning or natural language processing. ANA does not reinvent the wheel. It uses existing technology in addition to various new extensions. The technologies used will be discussed below.

First we discuss various technologies that aid the elderly and attempt to describe how they achieve their goal. Next we focus on the components that ANA uses such as machine learning and information extraction. Lastly we describe previous speech identification systems and question generation.

2.1 Elderly Assistance Technology

We are not the only group interested in aiding the elderly. There are software systems, hardware systems, and event robotics programs that are being designed to improve the lives of the elderly. Each have a specific focus whether it is medical, supportive, or emotional needs.

Japan is home to the longest estimated lifespan of a human being, which is approximately 83 years old. The growing number of elderly combined with limited access to family or support systems is a problem. To address this issue a small robot named Wakamaru has been developed [36]. Its goal is to assist a user or family of users by providing conversation and noting irregularities such as minimal movement or sound. Their focus is providing a physical being that the user can see and interact with.

Toyota is taking a more functional approach with developing robots that aid in specific aspects of elderly life. These robots attempt to take an area, such as movement, and build a robot to assist the individual. These robots focus more on the physical needs of the seniors rather than emotional. An example of these robots was a device that allowed the elderly to use the restroom with ease. The device would gently lift and place the user on the seat.

There is a project named Companions that has focused on human-computer interactions with a majority of the interaction being spoken. They developed a system that can listen and respond appropriately to a user [38]. The example they use is a "how was your day" scenario where the user discusses their day to the companion and it responds accordingly. They have a spinoff project called SeniorCompanions [15] [31] which is very similar to what we are attempting to accomplish. They provide a dialogue system which displays personal images to a senior. They listen and speak to them while they inspect a set of images. While going through these images the system attempts to learn about the user and their family. It also provides shallow humour and news from the internet. We were unable to determine details of how they capture information and use it. However their parent project briefly describes a few of the components they use. Their Natural Language Understanding module uses a Hidden Markov Model for part of speech tagging and a named entity recognition system using a combination of rules and a gazetteer [38].

Each of these systems tries to construct a system or device with the goal of helping our senior citizens. It is inevitable that as time progresses our medical techniques will improve and thus our health will be prolonged. Given that humans are living longer it is appropriate to investigate better ways of monitoring the elderly population.

2.2 Machine Learning

The idea of teaching a machine to replicate a human has been around for a very long time. The field of machine learning finds generalizations and patterns in data.

Machine learning makes predictions about tasks that is usually reserved for a human to make. These predictions usually use large amounts of data to make focused predictions in tasks such as classifying letters, digits, or event entire images. Machine learning has also been used for natural language processing. In language we would like to understand how words are represented or how books are modelled. For these type of tasks we can use machine learning.

Machine learning can be categorized into supervised and unsupervised models. A supervised model means that the model is trained on data which has been collected and labelled. An example of this would be recognizing all written variations of the number '3' as the number '3' label as shown in Figure 2.1. Each of these instances would have the label '3'. An unsupervised model does not have access to the labels of the collected data. This would be similar to having written variations of different numbers, but not knowing which number each instance is.

Figure 2.1: Digit recognition.

There are many types of machine learning models that aim to detect patterns, clusters, or labels. The two models that are used in this thesis are the support vector machine (SVM) [12] and the conditional random field (CRF) [23]. A CRF is a model that uses statistical probabilities to label sequential data. Tasks such as named entity recognition and part of speech tagging rely not only on current information, but also previous information. An SVM finds specific points within the data that are called "support vectors". These are data points that are identified as important or influential. Next, the SVM will try to label new points based on the proximity to these vectors. The SVM method can also incorporate a kernel into its method which allows the input of different objects such as trees or strings.

A kernel is a function that computes the similarity between two objects. The famous "kernel trick" is a method to avoid unnecessary computation and ensure the function is computationally feasible. It works by making the observation that mapping from the input space into the feature space is not needed and can be instead replaced with a kernel function. The dot product between kernel functions will be equivalent in the original space. The type of kernel used depends on the task. Examples of different kernels are linear, polynomial, gaussian, and exponential. The kernel function allows the dot product of any structure such as trees or strings. In ANA's case we use a tree kernel [8] for our relation extraction and a linear kernel for our other classifiers.

2.3 Stanford CoreNLP

The tools that Stanford's NLP group provides are used in this thesis. Almost every component within ANA uses one of the following tools. We will briefly describe what we use.

2.3.1 Part of Speech Tagging

In language, words are categorized according to their function or behaviour. This is called a words' part of speech (POS) tag. Identifying the POS tag for each word in a sentence can improve the quality of the entities and relations that can be extracted. In ANA we use Stanford's POS tagger which is based on the paper from Toutanova et al. [41]. Their paper introduces the idea of using a dependency network to apply part of speech tagging. This means that each label in a sentence can rely on other labels. In addition to this dependency network they use various lexical features combined with a conditional log linear model.

2.3.2 Named Entity Recognition

Named entity recognition (NER) is the act of labelling a sequence of words with their named entity representation. This could be a person, location, organization or nothing at all. In this thesis we use Stanford's NER system to identify named entities. Their system uses the model in Finkel's 2005 paper to extract entities from conversation [16]. Their paper relies on a machine learning construct called a conditional random field [23]. The CRF is a model that uses statistical probabilities to label sequential data. Tasks such as named entity recognition and part of speech tagging rely not only current information, but also past information. Finkel's CRF model uses Gibbs sampling instead of the standard Viterbi decoder for inference. A Viterbi decoder attempts to take hidden states combined with given observations to calculate the most likely path. Gibbs sampling also does this, but uses a multivariate distribution with sampling that can speed up the computation.

2.3.3 Pronoun Resolution

Pronoun Resolution systems aim to find pronouns that are connected to other pronouns or to named entities. This step is necessary in many tasks such as question answering or machine reading. Co-reference systems usually use a function with different features to determine whether two mentions refer to each other. The pronoun resolution tool we use in this work uses the model based on Raghunathan's work [33]. Their paper uses an unsupervised multi-pass sieve approach to ensure that high precision models are used first to eliminate potential ambiguity problems. A sieve is a pattern or filter that removes unwanted instances. For instance they will use a sieve that is very precise first in order to ensure the lesser precise sieves do not remove a correct example.

2.4 Information Extraction

To build a knowledge base it is necessary to use tools that allow the extraction of unstructured text. Information extraction is the task of taking unstructured text and identifying important information within it. Next it transforms that useful information into a structured format. We will now discuss the related work in the following sub-fields of information extraction.

2.4.1 Attribute Extraction

The traditional attribute extraction task usually revolves around taking an entity and attempting to extract their attributes from a collection of web documents. The list of possible attributes is usually pre-defined. This is called closed attribute extraction where the opposite would be open attribute extraction where no list is defined ahead of time.

Lan et al. discuss disambiguating people from search results by comparing their attributes [25]. They use a combination of rules, regular expressions, and gazetteer-based matching. They use a different approach depending on which group of attributes they are identifying. For example to identify a person's relative or birth place they use traditional Named Entity Recognition (NER). However for degree, nationality, or occupation they use a gazetteer, which is a dictionary of entities. Overall they were able to achieve an f-measure of 10% which is low, but they had issues linking attributes to people which is less of a problem for ANA.

Both the Text Analysis Conference (TAC) [20] and the Web People Search (WePS) provide tasks that aim to leverage attribute extraction. TAC's slot filling task takes web documents and asks participants to extract pre-defined attributes. This is very similar to WePS attribute extraction task. TAC uses newswire, wikipedia, and other sources for its dataset. WePS uses a dataset constructed by querying search engines for particular entities. For evaluation WePS used Mechanical Turk to label the evaluation data. The annotators choose between correct, incorrect, partial short, and partial long. The TAC evaluation does not tag the entire corpus, but instead a series of questions are asked such as "Who is the queen's brother" which each system tries to answer. This process determines the precision. The recall is based on the comparison between systems because the exact number of correct documents is not labelled.

2.4.2 Relation Extraction

In traditional relation extraction, SVM tree kernel models are the basis for the current state of the art.

Fader et al. developed REVERB, which solves the problem of incoherent extractions and uninformative extractions of two previous systems citeFader2011. Instead of extracting entities first, they extract verbal relation sequences based on a set of POS patterns. Then entities are identified around the relation sequence, so their system only extracts relation tokens between two entity tokens, e.g. relations such as <he, live in, city> in “Living in this city, he loves the city.” are ignored. Finally, relation triple candidate noise is filtered by a supervised model which is based on lexical and POS features.

Mausam et al. present an improved system called OLLIE, which relaxes the previous systems’ constraints that relation words are mediated by verbs, or relation words that appear between two entities [27]. OLLIE creates a training set which includes millions of relations extracted by REVERB with high confidence. Then OLLIE learns relation patterns composed of dependency path and lexicon information. Relations matching the patterns can then be extracted.

Xu et al. use a support vector machine with a tree kernel to extract relations from text [46]. Instead of extracting explicit features, the tree kernel compares the similarity between dependency trees. The weight learned is the weight for every tree instance. The tree kernel they use is from Moschitti’s 2006 paper [28]. This is an unlexicalized approach. This means that it does not consider the word just the behaviour of the word.

2.4.3 Event Detection

The task of event detection has a different meaning for different areas. Frequently it refers to relations amongst entities. It can also refer to historical events such as great wars, coups, and revolutions in news text. In the paper from Sakaki et al. the authors predict the occurrence of natural events, such as typhoons or earthquakes, by monitoring twitter [35]. They consider each user a sensor in their models which are used to pinpoint where the event has occurred. To detect these events they use a support vector machine with keyword, context, and statistical features.

The paper by Agarwal and Rambow discusses the construction of a “social network” of characters from literature works [3]. To aid in this construction they attempt to extract social events. Their definition of a social event is very broad and includes examples such as “John thinks Mary is good” which is a “talking” event. They use a support vector machine with various tree kernels on the ACE dataset. They use both the partial tree kernel and the subset tree kernel. They evaluate different tree structures such as the dependency word tree or grammatical relation tree.

There has not been a paper describing the task we are interested in, however there has been similar work done with extracting specific phrases. Their definitions differ from ours, but the techniques are still applicable.

2.5 Knowledge Bases

A knowledge base is a collection of structured information about specific entities. This information can differ but generally it describes attributes and relations of multiple entities. A notable example of a knowledge base would be Google's Freebase which stores thousands of entities along with their attributes and relations [5]. This is an example of a general knowledge base which covers as much information as possible. They state that in 2008 they had more than 125,000,000 tuples of data. These tuples contain various attributes such as spouses, date of birth, or event places lived. Other such knowledge bases exist such as UMLS, a medical term knowledge base, and Wolfram Alpha, a mathematical and fact knowledge base. This differs from ANA's approach which is to store the information of an individual, entities or events related to that individual, and useful medical information. ANA prioritizes entities related to the individual when finding information and generating responses. If the user has a friend named Barack Obama and they refer to Barack, ANA will assume they are referring to the friend not the president.

In the Text Analysis Conference they propose a knowledge base population task which aims to extract structured information from text. Specifically they target attributes about people, places, and organizations. The basic foundation of ANA's knowledge base is built upon their guidelines. Many groups and teams participate in this event which must mean it is a worthwhile research area.

An aspect of knowledge base construction that is very important to ANA is entity disambiguation. Entities can have exactly the same name, but refer to different objects. The user could have multiple grandchildren that have the same name and it is important to make sure ANA knows which one is being referred to. The task of entity linking is taking a textual meaning and identifying which entity in the knowledge base is being referred to. The paper by Drez et al. disambiguates entities by using a combination of intricate candidate selection, a ranking machine learning model, and a NIL predictor [13]. The NIL predictor is a method to guess whether the mention is actually within the knowledge base. They consider abbreviations, nicknames, and other short hand styles.

2.6 Speech Act Identification

For the task of speech act identification a sentence is classified into different speech acts. These acts vary from task to task. Various work has been done in the field of question detection and request identification.

A few papers have attempted to label emails with speech acts. The main idea here is to identify “intent” or “purpose”. In Cohen et al. they discuss finding requests for meetings or sentences that provide information [7]. They have many different classes such as requests, commitments, proposals, and reminders. Cohen et al. use an SVM with features such as word bigrams, part of speech tags, and temporal information to determine the speech act. Lampert et al. classify emails as urgent or not [24]. These emails are labelled to indicate that the user must respond with haste. They use an SVM classifier with email specific features such as the inclusion of “Re:” or “Fwd:”. Other features include wh-words, punctuation, and n-grams.

Jeong et al. use a semi-supervised approach to identify speech acts that utilizes subtree pattern mining [19]. The subtree pattern mining they employ is similar to the 2004 paper by Kudo and Matsumoto [22]. Their paper enumerates the entire subtree space and prunes various paths based on an estimated upper bound function. They introduce classes such as statements, rejections, yes-no-questions, wh-word-questions, or uncertain responses. Their work is evaluated on forum and email data.

The work by Qadir and Riloff attempts to identify speech acts within message board posts [32]. The language used on message boards is similar to the conversational data we use in ANA. They extract four different speech acts from the first post: commissives, directives, expressives, and representatives. The key difference is that they assume a message can contain multiple speech acts where as we categorize a sentence into one speech act. They implement an SVM classifier with various hand crafted features to solve this problem.

Li et al. identify tweets, messages posted on the social network Twitter, that contain questions and tweets that seek information or assistance [26]. The first task is to find tweets that contain questions or are interrogative. Next they use both a unsupervised and supervised approach with various features to detect specific question types within tweets. They use a set of rules, such as if the sentence contains a wh-word, for their unsupervised approach. They use an SVM with features constructed by the Prefix Span algorithm for their supervised method. These features are wordsets that are frequent sequential patterns. The Prefix Span algorithm finds frequent sub sequences or sets of words to use as features [30] for their SVM. The data that this paper uses for evaluation is collected from Twitter. In most instances this data lacks grammar and punctuation which similar to our task.

Shrestha and McKeown detect questions and answers from emails [37]. They also use a machine learning approach with discriminative POS bi-grams combined with beginning and ending n-grams. Cong et al. detect question/answer pairs, but they focus on forum posts [10]. They leverage labelled sequential patterns which combines POS tags and lexical information. For instance, the sentence “I want to buy an office software and wonder which software company is best” a sequential pattern would be “wonder which ... is”.

All of the previous works have different definitions for speech acts. They choose the classes that suit their task and our work is no different. Most of the methods use both patterns and supervised learning. The data used is a combination of informal and formal language.

2.7 Response Generation

We will now discuss various methods of generating a response.

2.7.1 Educational Question Generation

Heilman et al. implement a learning system in which questions are generated based on reading material [18]. Students are assigned reading assignments in certain courses and in order to test whether they have read the material the teachers will give their students a quiz. However it is quite tedious for the teacher to create such quizzes. This system attempts to automatically generate questions based on the reading assignment. They use a combination of linguistic rules and statistical ranking. An example of a linguistic rule is the subject-auxiliary inversion. An example of this would be "Sam has read the paper" turning into "Has Sam read the paper". The statistical ranking portion focuses on using a linear regression model over a binomial distribution. Although ANA does not currently use this method in the future it would be interesting to develop a question generation module based on this concept.

2.7.2 Chatterbots

The market has been flooded with intelligent companions attempting to assist people in their day to day lives. Many of these agents focus on a specific task such as organizing a person's email, alerting them of calendar events or simply entertaining users. This is similar to the task ANA is solving.

A more general type of intelligent companion is called a chatterbot. These systems use simple word cues to generate a response. They also use a markup language called AIML which allows them to match patterns to responses. This was introduced by Richard Wallace for the ALICE project [44]. These patterns can be literal or have wildcards, "WHAT IS YOUR *?".

The earliest form of an intelligent agent came from ELIZA [45] in 1966. ELIZA is based on finding keywords and following a tree of rules to generate a response. This approach has a few problems. First if ELIZA does not find a keyword it cannot produce a meaningful response. Another problem with their approach is that they do not save anything from the input. Once they have generated a response the input is simply thrown away.

In 1950, Alan Turing developed the Turing test [43]. The idea of this test was to determine if a computer could think as if it were human. The test begins with a human judge who is behind a wall communicating via a terminal to two entities one of which is a computer and the other is a real human. If the judge cannot deduce which entity is human then the computer is deemed intelligent.

The Loebner Prize encourages the development of chatterbots. However, until now there are not many chatterbots that are close to leading a "human" conversation. The current state of the art chatterbots presently are Jabberwacky, A.L.I.C.E. (Artificial Linguistic Internet Computer Entity) and D.U.D.E.. They are constantly developing, learning new responses and trying to take advantage of new technologies.

The current iterations of chatterbots compete for the Loebner Prize. The most well-known chatterbots are Cleverbot, ALICE, Jabberwacky, and DUDE. The Loebner Prize is awarded to the system that is the closest to completing the Turing test. The prize was first established in 1990 with a grand prize of $100,000.

2.8 Conclusion

All of these works are in some way related to what we are trying to accomplish with ANA. The key differences is that we are focused on the elderly, conversational data, one-on-one discussions, and will eventually use a speech to text system.

Chapter 3

Information Extraction from Conversation

This thesis introduces a new intelligent agent system named ANA which provides conversation for the elderly. Currently ANA assumes that the speech-to-text component has been implemented and only considers text as the input. ANA attempts to produce meaningful and useful conversation by extracting information from the textual dialogue and generating questions to further build its knowledge. ANA slowly constructs a knowledge base that is personalized for a specific person. Our hypothesis is that conversation with an intelligent agent can be greatly improved with the inclusion of a personalized knowledge base. This knowledge base includes entities, relations, medical information, attributes, and past discussion. The presence of the knowledge base will allow an agent to sound more personable because it has access to the user's information. The knowledge base is built by extracting information from the conversation between the user and ANA. At first the knowledge base will be quite empty, but as time progresses it will attempt to expand its knowledge about the user and people associated with the user.

3.1 Introduction

Information extraction is an umbrella term that applies to various research areas. The goal of information extraction is to parse unstructured text and extract structured data in a specific manner. This could be taking a large collection of news text to identify named entities or identifying temporal information from emails. The most important step of building a knowledge base is the information extraction. If the information stored in the knowledge base is flawed then the conversation will sound strange. Extracting information from conversation has its own problems and challenges that sets itself apart from the traditional task. These problems include slang, incomplete sentences, and improper punctuation. In this task, utterances are considered the documents and are usually much shorter in length than usual.

In this chapter we will first discuss ANA's pipline. Then we will describe the medical and named entity recognition that ANA utilizes. Next we will review entity disambiguation and relation extraction. Finally we explain our methods for event and attribute extraction.

3.2 System Pipeline

To extract useful information from the input an information extraction system must be created and a pipeline must be designed. It is difficult to construct a system where each module is independent. With a pipeline approach each component depends on the previous one. This makes it difficult because if the first component fails or is misclassified then the following modules may also contain errors. ANA's pipeline is shown in Figure 3.1. The pipeline starts with extracting part of speech tags and named entities. Next we extract dependency paths and relations. Finally we disambiguate entities and relations. The components of this pipeline are described with more detail later in this chapter.

3.3 Identifying Medical Concerns

The elderly are very susceptible to becoming ill or forgetting important information. Part of our system involves asking the user medical information such as if they have taken their medication, how they are feeling, and if they are not feeling well then what symptoms they have. In order to detect these situations we built a simple pattern based system. This system is based on a list of frequent elderly medical concerns, from Homemeds, and a list of the most common drugs used for them [1]. Drugs such as Tylenol, Vicodin, and Lexapro are frequently prescribed to the

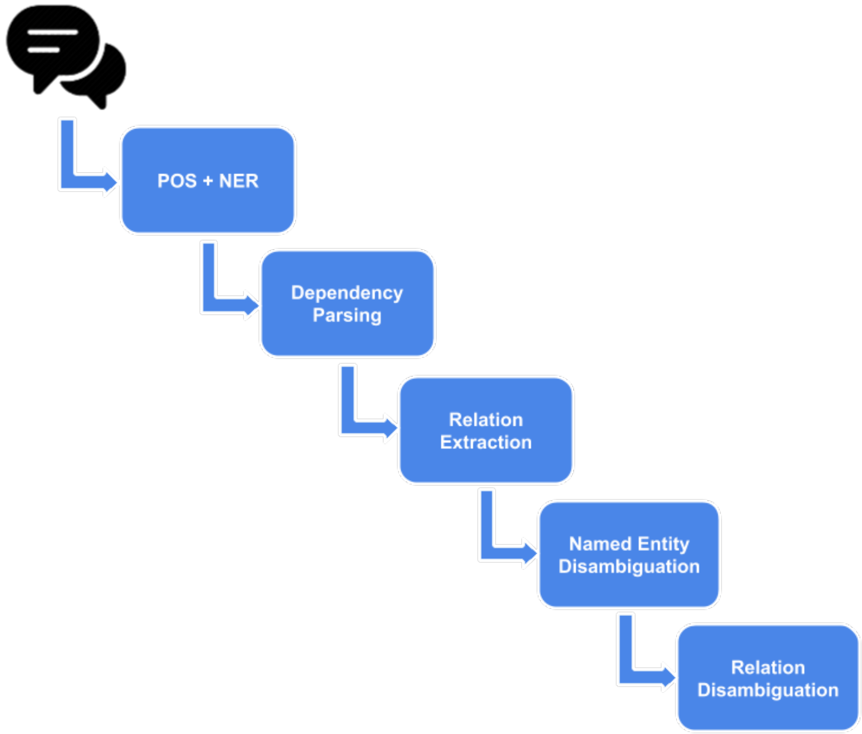


Figure 3.1: High level structure of the information extraction module.

elderly and thus it is important to know when they are referenced.

In a conversation ANA has the capacity to monitor the input for words or phrases that may indicate the user is uncomfortable or may be ill. If ANA deduces that the user could be ill it will ask the user to elaborate on their symptoms by asking "What are your symptoms". We use the presence of the following keywords and phrases to indicate a potential illness.

	Examples
Keywords	"uncomfortable", "headache", "cramped", "sweating", "irritated", "bloated", "pain", "agonizing", "sore", "dry", "sick"
Phrases	"not feeling well", "feeling poorly", "feeling hot", "feeling ill", "feel ill", "pass out", "is wrong", "having difficulty"

Table 3.1: Illness keywords.

Terms such as "sore", "pain", and "irritated" can indicate the user is experiencing a symptom. These terms coupled with phrases such as "not feeling well" or "quite awful" can indicate a problem. If the combination of a term and a phrase is detected then ANA will store the term and ask the user to elaborate on their symptoms. We would also like to store their response. They could indicate that they are in pain because of a sports injury or the cause is unknown. Either case would be useful for their physician. Listed in Table 3.2.

	Examples
Keywords	“runny nose”, “fever”, “sore throat”, “tired”, “weak”, “trouble breathing”, “insomnia”, “dizzy”, “light headed”, “sweaty”, “blurred vision”, “nausious”, “nauseated”, “vomiting”, “chest pain”, “cold”, “dry mouth”, “short of breath”, “ear pain”, “nose bleed”, “cough”, “eyes hurt”, “upset stomach”, “headache”, “blurry vision”

Table 3.2: Symptom keywords.

A key component of taking care of the elderly is making sure they are consistent with taking their medication. Unfortunately a percentage of seniors need assistance in remembering to take their medication. In this scenario ANA could ask periodically throughout the day to determine if they have taken it. If they are sure they took it then ANA can note the time otherwise ANA will ask them to speak with the nurse. The user can also inquire whether they have taken it or not. In order to detect whether the elderly have taken their medication we monitor the input and look for keywords and patterns. We look for words such as “forgotten”, “misplaced”, “remember” and phrases such as “have i taken”, “did i take” and “not sure if”. Refer to Table 3.3 for all keywords used.

	Examples
Keywords	“forgotten”, “misplaced”, “remember”, “meds”, “pills”, “medication”, “drugs”, “lozenge”, “medicine”, “dosage”, “antibiotics”, “antivirals”
Phrases	“have i taken”, “did i take”, “not sure if”, “not sure”, “should i take”, “swallow”

Table 3.3: Forget keywords.

Our goal is to have ANA listen and record useful information such as medication taken, how the user is feeling, and potential symptoms. It is possible that in the future doctors and nurses could examine ANA remotely if the user is experiencing symptoms. Our hypothesis is that by extracting medical concerns ANA will reduce the stress on the user, the nurses, and their doctors.

3.4 Named Entity Recognition

When having a discussion with the elderly it is important to identify and remember important entities mentioned such as people, places, or dates. For this task we use Stanford's CoreNLP package that allows the extraction of various named entities. They use a conditional random field model to calculate the most probable sequence of entity labels for the entire sentence. The named entities that are extracted are: time, location, organization, person, money, percent, and date.

The input is a raw sentence and the output is a labelled array. The output is the same length of the input with each token either "O" or the named entity label.

```
Kevin is a student at the University of Alberta.
['PER', 'O', 'O', 'O', 'O', 'O', 'ORG', 'ORG', 'ORG']
```

From this we can check if Kevin exists in our knowledge base and if we already know he attends that specific university.

3.5 Entity Disambiguation

Once the named entities and relations are identified it is necessary to check whether that are already inside the knowledge base. There can be multiple people with the same name or people with multiple relations. It is crucial to make sure that you have identified the correct entity in these situations or the conversation will not make sense.

Each time an input is read the system extracts all entities and relations within that sentence. These entities and relations are then compared with what is contained within the knowledge base. If there is a name that is contained within the knowledge base then nothing happens, however if the name does not exist then it is created. We always assume that if the name exists then the person is speaking about that entity. An example of this would be if the knowledge base contained an entity named Kevin Quinn and the current input mentioned Kevin. We assume that these are the same unless otherwise stated. Of course this is not always the case, but this is the restriction within our system.

Family titles, like nephew, need to be disambiguated before any information can be extracted. An example of a title needing disambiguation would be “I went to the store with my nephew”. In this situation we do not know which entity is being referred to. First we look through the knowledge base and check whether the speaker has the relation “nephew”. If they do, then we need to check if they have more than one. In the event of multiple nephews the system will need to ask for more information such as “Which nephew”. If there is only one nephew then our system assumes the speaker is discussing that entity.

3.6 Attribute Extraction

Attributes are characteristics of people or events that are interesting for conversational purposes. These include ages, dates, or places. Attributes are very important in terms of ANA’s question generation. ANA attempts to learn as much as possible about a given entity and this happens because it asks questions based on missing attributes. The attributes that we used for people is a subset of the attributes used for the TAC 2011 slot filling task. This is shown in Table 3.4.

Object	**Attribute**
PERSON	Name
PERSON	Age
PERSON	Gender
PERSON	Residence
PERSON	Educational Institute
PERSON	Educational Degree
PERSON	Profession Position
PERSON	Profession Institute
PERSON	Likes
PERSON	Dislikes
EVENT	Start
EVENT	End
EVENT	Who
EVENT	Where

Table 3.4: Entity attributes.

The attributes for people used in this project are: gender, age, name, education, profession, residence, likes, and dislikes. For event attributes we chose to use: start, end, where, and who.

There are many different ways to describe an attribute in conversation. Take the "age" attribute from a person. You could say "I am 25", "Kevin is 25 years old", or even just "25" if they were asked a question previously. There are a vast number of ways to tell the speaker what your age is. This presents a problem when trying to extract these values. We cannot assume that there will always be a keyword present or the expected type will used.

Extracting these attributes in our system is accomplished by using a combination of part of speech tags, presence of keywords, and dependency paths from parsing. Each attribute has its own patterns, keywords, and paths that allow us to extract the attribute value. For example the profession attribute has keywords such as work, profession, employed, or occupation. The input is passed through each of these attribute pattern sets, as shown in Tables 3.5 and 3.6, and if they find a candidate the corresponding property is updated.

Consider the example, "Kevin is a student at the University of Alberta", there is a dependency link between Kevin and student, as well as, student and University of Alberta. From this we can link student with the organization in Figure 3.2.

Figure 3.2: Example of a dependency pattern.

In this example the root must be student and the child nodes must be either a noun or noun phrase.

3.6.1 Experiments

Conversational data is text between two or more people that is informal in nature. This is the type of data that we are interested in. Extracting information from this type is much hard than news articles. People use slang, improper grammar, and misuse punctuation. The data used was not entirely annotated and thus needed

	Examples
Age	“years”, “old”, “age”, “turning”, “born”, “PRP$ NN”, “VBZ CD .”
Residence	“lives”, “resides”, “resident”, “live”, “reside”, “occupant”, “citizen”, “LOC”, “VB* IN NN”, “NNP NNP”
Likes	“likes”, “favorite”, “loves”, “enjoys”, “like”, “love”, “enjoy”, “adore”, “relish”, “VB-dobj-NN”, “VB-xcomp-VB”, “VB-xcomp-VB-dobj-NN”
Dislikes	“hates”, “dislikes”, “loathes”, “hate”, “dislike”, “loath”, “detest”, “VB-dobj-NN”, “VB-xcomp-VB”, “VB-xcomp-VB-dobj-NN”
Education	“attends”, “goes to”, “enrolled”, “studies NN in”, “will go”, “applied to”, “was accepted to”, “NN IN DT NNP NNP”, “NNP IN NNP”
Degree	“computing science”, “physics”, “math”, “chemistry”, “engineering”, “medicine”, “pharmacy”, “science”, “biology”, “education”, “physical education”, “IN-prep-NN”, “NN major”, “NN degree”, “VBD IN NN”, “studies”, “study”, “taking”, “degree”, “specialize”
Profession	“works for”, “work at”, “profession”, “job with”, “occupation”, “employed IN”, “NN VBZ IN NN”, “NN-prep-NNP”, “ORG”, “VB* IN NNP”
Position	“doctor”, “engineer”, “analyst”, “student”, “scientist”, “teacher”, “dentist”, “professor”, “is a”, “i’m a”, “was a”, “i am a”, “is an”, “’s an”, “’s a”, “working as a”, “currently a”, “NN NN”, “JJ NN”, “VBZ DT NN”, “NN-nsubj-NN”, “NN-nsubj-PRP”

Table 3.5: Person attributes.

labelling. The labelling was entirely done by ourselves. We needed to label relationships, events, and attributes within these sentences. Our data selection process begins by selecting a random sentence from our dataset. If that sentence contains a relevant component, such as an event attribute or relation, it is included within our training or testing set. We will now discuss the three datasets that we chose to use in order to evaluation our system.

The objective of the Automatic Content Extraction (ACE) 2008 task [29] is to extract entities and relations from various text sources. One of those sources is telephone conversations which is useful for our purposes. We parse each file and extract the text from the telephone conversations. From these the XML needed to be stripped away as well as various transcription marks such as “%um” or “[laugh]”. These conversations discussed various topics such as politics, vacations, and ordinary conversation. This data had relationships and entities annotated.

	Examples
Who	“with”, “together”, “take”, “PER”
Where	“is at”, “hosted at”, “held at”, “IN NN”, “NNP NNP”, “IN-pobj-NN”, “LOC”, “ORG”
Start	“arive”, “begins”, “starts”, “will begin”, “CD”, “DATE”, “TIME”, “MD VB* IN CD”
End	“depart”, “ends”, “finish”, “will leave”, “CD”, “DATE”, “TIME”, “MD VB* IN CD”

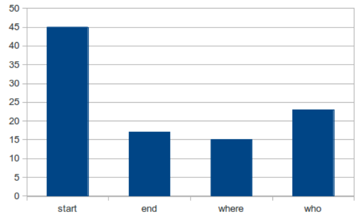
Table 3.6: Event attributes.

We chose to use CALLHOME American English Speech dataset. It contains 120 telephone conversation transcriptions that are unscripted. Each call lasted at most 30 minutes and took place in North America [4]. These conversations were unlabelled and thus we needed to label them by hand.

Another dataset that was used in this system was the Switchboard-1 Telephone Speech Corpus. This dataset is a set of 2,400 telephone conversation transcriptions between two people. It consists of 543 speakers from the United States. These people were instructed to discuss various topics such as their family life, interests, or news [17]. Again the switchboard data was not labelled and we needed to annotate it.

To evaluate the extraction of person attributes we chose 133 pseudo-random sentences that each contained one attribute that needed to be extracted. These sentences were chosen by using “seed” queries to search through each dataset. An example of a seed query could be “works” or “education”. The assumption is that sentences that matched these queries might contain attributes. A secondary method was to randomly select a sentence from a dataset and use a human to decide if an attribute existed. After all of the sentences are gathered we annotated each with the appropriate label. This is biased, but necessary due to lack of labelled conversation data.

Once the test set is constructed we assume that there will be at most one attribute per line, but that line can be repeated. For example if a line contained an “age” and “degree” attribute we would add two lines to the test set, one for each attribute. For a single test input there are three possibilities, our system can correct, partially



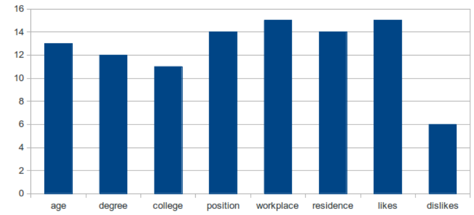


Figure 3.3: Event attribute distribution. Figure 3.4: Person attribute distribution.

correct, or be incorrect entirely.

Evaluating the event attributes follows a similar formula to the person attribute evaluation. First we choose 100 pseudo-random sentences that contain an event attribute. These are chosen in the exact same manor as the person attribute test set. Each line contains one attribute. Our system can either be correct, partially correct, or totally incorrect. Below are the results.

To evaluate both the event and person attributes we use correct, partial, and incorrect. It is difficult to identify attributes that have more than one word, therefore we decided to keep track of partial matches. A partial match is when our system's guess is a substring of the correct answer. A correct match means our system's guess was fully correct and incorrect means it was totally incorrect. An example of a partial match would be extracting "picnic" when the correct answer would have been "going on a picnic".

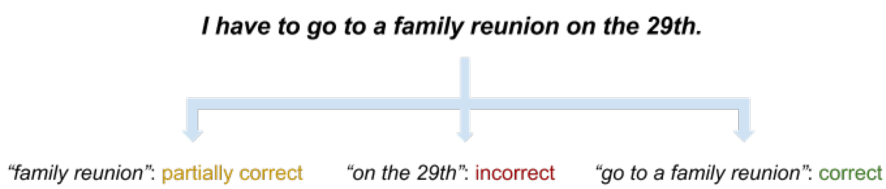


Figure 3.5: Metrics example.

Below are the results of our person and event attribute extraction. For this task we were unable to find implementations to compare with.

Attribute	**Correct**	**Partial**	**Incorrect**
Person	0.18	0.34	0.48
Event	0.20	0.67	0.13

Table 3.7: Results with correct, partial, and incorrect

3.6.2 Discussion

The patterns and heuristics used for attribute extraction were simple and generic. We are able to extract the correct attribute 41% of the time. However there are areas in which our method encountered difficulties. Each attribute category required its own set of keywords and dependency paths which consumed a majority of the time. A more general method could be introduced in the future. We hypothesize that hand-crafted patterns for each attribute in addition to the lack of training data reduced the accuracy.

3.7 Relation Extraction

We are interested in extracting family relations from the conversation with the elderly. We assume that most of the conversation between ANA and the elderly will involve family members. In order to extract these people and their affiliation with the user we needed to build a relation extraction system. This system assumes that within a sentence there will be two entities and one relation word, such as father or mother. The entities can be either names or references, such as pronouns.

We used a support vector machine that uses a partial tree kernel (PTK) to extraction relations from text. Instead of extracting explicit features, the tree kernel compares similarity between dependency trees. The weight learned is the weight for every tree instance. The tree kernel we are using is from Moschitti's 2006 paper [28].

This convolution kernel attempts to capture the latent information stored between different data representations. In our situation we care about the similarity between two sentences. The sentences need to be converted into a tree format that can be used by a kernel function. The main issue when attempting to compare these trees is the computation complexity. The key idea from the paper is that trees can be broken down into their components and if a suitable kernel function can be built for the parts then a kernel function can be constructed for the entire tree. The kernel function is constructed by first taking the inner product between trees:

$$K(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2) \tag{3.1}$$

N_{T_1} and N_{T_2} are the set of trees' nodes [8]. The Δ function provides the basis for identifying subtrees of nodes, which is the essential distinction between different tree kernel functions. Here we adapt the partial tree kernel proposed by Moschitti et al., which can be used with both constituent and dependency parse trees [28]. They define $\Delta(n_1, n_2)$ to be the frequency of common subtrees between n_1 and n_2.

1. if the node tag n_1 and n_2 are different, $\Delta = 0$.

2. otherwise $\Delta(n_1, n_2)$ is:

$$\mu(\lambda^2 + \sum_{J_1, J_2, l(J_1)=l(J_2)} \lambda^{d(J_1)+d(J_2)} \prod_{i=1}^{l(J_1)} \Delta(c_{n_1}(J_{1i}), c_{n_2}(J_{2i}))) \tag{3.2}$$

J_1 and J_2 are sequences of child nodes. The λ and μ terms represent a decay factor that penalizes large trees and child subsequences that contain gaps. When actually constructing the kernel it is important to note that there will be more tree fragments of a larger size, depth 4 versus depth 3. To remedy this the kernel can give less weight to the larger tree fragments. This also applies to fragments that contain gaps. A gap occurs if any sequence of children are not consecutive.

The following figure is an example of subtrees for the PTK. The tool we are using is from Alessandro Moschitti [28]. The example in Figure 3.6 is a subtree from the input "My brother Nathan likes pizza".

The tree structure we employ is similar to the tree structure for the binary task in Xu's 2013 paper [46]. First we extract the shortest path between two entities' heads. Then we need to re-organize the nodes so that link labels of dependency trees are on the nodes instead of on the edge. The root node is the POS tag of the first node in the path. Others have the link label as the parent, the POS tags on the middle and words as the children of the POS tags. Note that in [46] lexical information is not used, but in this thesis words are used as the leaves of the dependency tree.

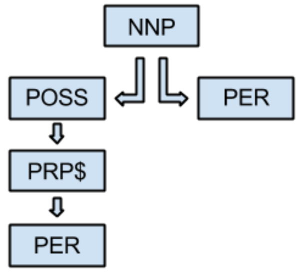


Figure 3.6: Tree structure of "my brother Nathan" excerpt.

3.7.1 Experiments

In order to analyze how the relation extraction system performs, we need to run the following experiments. The relation extraction evaluation uses the process described in the attribute extraction section to gather test data. We extracted tagged instances from the ACE data mentioned previously. We only used the instances that had the "telephone" label because they were only two people conversing. In total, there are 100 sentences which contain family relations. We randomly sampled another 100 sentences with other relations such as employed or located. Next we ran 10-cross validation to make sure the results were stable. Table 3.8 shows the average precision, recall and f-measure.

The evaluation metrics of the relation and event extraction are recall, precision and f-measure. Recall is the ratio of the number of relevant items found to the number of total relevant items. The precision is the number of relevant items retrieved to the number of items retrieved. F-measure is a metric that balances both recall and precision to give a better idea of performance.

$$Precision = \frac{items\, guessed\, positive}{items\, labelled\, positive} \tag{3.3}$$

$$Recall = \frac{items\, guessed\, positive}{all\, positive\, items} \tag{3.4}$$

$$F-measure = 2 \cdot \frac{precision \times recall}{precision + recall} \tag{3.5}$$

Task	Precision	Recall	F-measure
Relation Extraction	100%	100%	100%

Table 3.8: Relation extraction results on family relation.

3.7.2 Discussion

Our relation extraction module achieves 100% for f-measure. This is not because our system is perfect, but rather because the data is very simple. Each of the training samples must contain two entities and a relation word. Sentences such as "My brother went to the store" cannot be used because only one entity is mentioned. The number of ways to express a relation from a first or third person point of view is quite limited. These are the reasons why the relation extraction performs well.

3.8 Event Extraction

In casual conversation people discuss various aspects of their day. Many of these include events such as sports, concerts, dates, and much more. Discussing these events with ANA can help the elderly feel like they are actually speaking with a human. We have implemented an event extraction framework that allows the identification and extraction of events. We have created a machine learning model to identify events and a pattern based approach to extract event phrases.

Our definition of an event is a period of time that an activity occurred with one or multiple participants. This could be a hockey game, music concert, or even just having dinner. This could be simply "going out" or "went for a run", but most of the time events are represented by a single word. Our extraction process is two fold: create a binary classifier to determine whether an event exists within a sentence, if an event exists use a separate heuristic method to extract the exact phrase.

Our first classifier examines an input sentence and determines whether the sentence contains an event. It has access to a list of of popular or frequent events constructed by examining the training data. Obviously this list can never be complete, but a starting collection of words will help the classification process. When a sentence contains an event it usually contains words such as "attending", "went out", or "I went". In order to extract these words or phrases we find all of the most frequent uni-grams and bi-grams surrounding the event word. In addition word n-grams we consider the part of speech n-grams in these sentences. For example the POS pattern "VB NN" is quite frequent within event phrases. It is important to check for named entities within the utterance. A speaker could mention a date, time, or location for an event.

The most important aspect of this classifier is using the PTK with the dependency tree. This is used alongside the explicit features to detect events. The input of the PTK is the entire dependency tree of the sentence. We use a combination of tree kernel and polynomial kernel. The features for the polynomial kernel are:

- Presence of certain word clusters.
- Presence of event keywords.
- Word uni-grams and bi-grams.
- Presence of named entity.

After the event identification we need to extract the event phrase if it exists. Our pattern based approach uses patterns gathered from the training set. The first step is to examine the training data and construct the POS patterns. For example the phrase "went for a run" is the event for a sentence and its POS representation is "VBD IN DT NN". If in the test set we see a sentence that has that pattern then we assume that is the correct event phrase.

Next we need to examine the dependency links within the training data. Specifically we are looking for the VB $\rightarrow$ NN dependency link. We use this link because most event phrases have a link between the main verb and a noun. We examine our training data and look for instances of the VB $\rightarrow$ NN link. We found that the prep and dobj link label appear the most. Then if, in the test set, the input does not contain the above POS pattern and it contains the VB $\rightarrow$ NN pattern with the correct link label we assume that the verb-noun phrase is correct.

There are many different words that refer to events. For these unknown words we wanted a method to measure how similar they were our list of known events. We constructed a list of known events from the training data. These include “game”, “concert” or “barbeque”. We use a word clustering result [42] that assigns a label to a word based on how similar it is to other words with that label. Our assumption is that events in our known list that belong to a cluster and unknown words in these clusters could be potential events.

3.8.1 Experiments

The event extraction evaluation has two parts. The first evaluates if a sentence contains an event and the second evaluates if the correct event word can be found. This assessment uses the 100 data points from the event attribute extraction with 100 more sentences that do not contain events.

First we evaluate how often we can detect the presence of an event. There are 200 sentences of which 100 contain events. We compare four different methods for our evaluation. The baseline simply looks at a sentence and attempts to look for specific keywords such as “attending” or “going” and for event keywords such as “game” or “concert”. The explicit features method was our first attempt and uses an SVM with features such as POS and word n-grams. The tree kernel method only uses the dependency parse tree to distinguish between inputs. Finally the last method uses a combination of explicit features and a tree representation. We use average precision, recall, and f-measure to evaluate the quality of each of the following four methods in Table 3.9.

Method	**Precision**	**Recall**	**F-measure**
Baseline	0.6725	0.8300	0.7425
Explicit Features	**0.9375**	0.4825	0.6325
Tree Kernel	0.8225	**0.8625**	0.8400
Explicit + Tree	0.8550	0.8450	**0.8500**

Table 3.9: Event results with average precision, recall, and f-measure.

To ensure that our second method is indeed better than the baseline we perform a t-test for statistical significance. The input for the test is gathered by conducting 4-fold validation. Each column below represents each of the different test sets. The values in Table 3.10 are the f-measures.

Method	Set1	Set2	Set3	Set4
Baseline	0.79	0.70	0.72	0.76
Explicit Features	0.72	0.49	0.62	0.70
Explicit + Tree	0.85	0.81	0.86	0.88

Table 3.10: 4-Fold f-measure results.

From this input we calculated the t-test and ended up with a t-value of -4.31 and a p-value of 0.006. From this result we can see that our second method is statistically better than the baseline.

Once we have determined that an event does indeed exist within the utterance we need to extract the specific word. There are 150 sentences that contain an event and an event phrase. Of this we use 50 sentences to construct our patterns and keyword lists.

To evaluate the strength of our system we will compare three methods. The baseline uses a dictionary of event words to check whether the input has a word from it. If it does the baseline takes every word to the left of it until it hits a verb. For example in the input “I went to the ball game” it has the keyword “game”, which is in the dictionary, so it would add each word to the left until the current word is a verb. In this case it would guess “went to the ball game” because went is the first verb on the left encountered.

The tree method is a simple method which attempts to extract a VB $\rightarrow$ NN dependency link. It will add all words in-between the verb and the noun. The last method uses the tree method in addition to using previously found POS patterns. In the training set we find all of the event phrases and their POS representation. In the test set if we see that it contains a pattern we extract that phrase. As with the attribute extraction we use correct, partial, and incorrect as the evaluation metrics as shown in Table 3.11.

Method	Correct	Partial	Incorrect
Baseline	0.1882	0.2765	0.5352
Tree	0.3705	0.3327	0.2966
POS patterns + Tree	**0.4131**	0.3628	0.2240

Table 3.11: Extraction results with average correct, partial, and incorrect.

3.8.2 Discussion

The results from the event detection are quite promising. Using just the PTK we can achieve an 84% f-measure. This can partially be attributed to the quality and lack of data, but it is a surprising result. Adding more explicit features could increase the f-measure further given enough time. Unfortunately it cannot handle certain tree structures it has not seen before. In the input, “3 o’clock in the afternoon is the reception”, the sentence is phrased in an unusual manor. Another issue are sentences that have a very simple tree structure. Sentences such as “I’m going to sleep” or “The game ends at 6pm” have a very common tree structure which makes it difficult to classify them.

The word extraction results are also very good. The best method manages to guess the phrase correctly or partially 78% of the time. This is a good result, but there are a few problems. Before we explain the problems we need to discuss the phrases contained within the training data. The most frequent type of event phrase in the training data was single word nouns. Because our system uses the POS pattern from the training data it will attempt to find nouns as potential answers. In the example, “Tomorrow I leave for France”, our system incorrectly extracts “Tomorrow” as the event phrase because it is a single noun. Another issue with the event word extraction is when the sentence contains multiple VB $\rightarrow$ NN dependency links. When this happens and there is not an obvious POS phrase present then the system has to choose the first one.

3.9 Summary

Extracting information from conversation is a difficult task. There are a number of problems that prevent the standard information extraction techniques from being used in conversational data. Issues such as poor grammar, punctuation and spelling errors are the primary problems. Other problems exist such as phrasing a sentence awkwardly or making assumptions on what the user knows. The extraction of attributes, relations, and events are essential to maintain a meaningful conversation with a person.

Our relation extraction module achieved 100% on our small dataset. This is very encouraging and proves that ANA is able to identify family relations with ease. The dataset is very small, but being able to extract all family relations is very good. This component is still being worked on and is not fully complete.

The event extraction, like relation extraction, is difficult to define. Initially we only focussed on single word events such as game, concert, or wedding. These are important, but an event extraction system should not be limited to only those. That is why we chose to extend our system to using phrases. The PTK worked quite well in distinguishing between sentences that contained events. This is not surprising consider how similar this task is with relation extraction and the results of aforementioned relation extraction.

We have implemented three modules which allow the extraction of entity attributes, relations, and events. The attribute extraction is a work in progress, but the relation and event extraction perform well. These modules can assist in the extraction of knowledge from conversation and help build an intelligent conversational agent.

Chapter 4

Generating a Response from an Agent

In this work we introduce our intelligent agent ANA which engages the user in meaningful conversation. A large component of this system is the output generation process. The output must make sense and be on topic. Our hypothesis is that ANA can generate text that is suitable for the current situation and resembles what a human might say.

4.1 Introduction

Once ANA has the ability to extract necessary information from the dialogue we need to determine when and how to respond. These conversation phases are easy to identify and handle as a human, but for an NLP system it is quite difficult. To begin we identify how to respond. To determine this we classify sentences into three categories: *Declarative* (statements), *Interrogative* (questions) and *Imperative* (requests). In each of these scenarios ANA may need to respond differently.

Before we decide how to respond we need to check if ANA was expecting any input. If ANA has asked a question previously then the question and expected answer type are stored in a buffer. When the next input is given ANA checks whether the buffer has any content within it. If it does, ANA uses the buffer to check if the user correctly answered the question. The buffer has a limited window in which it will stay in memory. After this window has elapsed the expected answer is removed from the buffer. This buffer improves both the extraction and response process.

Next we need to determine how to respond. This can either be a question or statement. An example of a question would be asking the user where they went to lunch. A potential statement could be answering a question the user had. A question is generated if ANA wishes to clarify a statement or build knowledge. A statement is returned when ANA either does not understand the input or does not consider it is appropriate to ask a question. For instance, if ANA has asked three questions consecutively then perhaps ANA should not reply with a fourth question.

Once the output has been chosen ANA stores the response into the buffer and checks if the response needs to be modified. This process considers if the speaker or object is female, if the proper tense is used, and even which "person" the response should be.

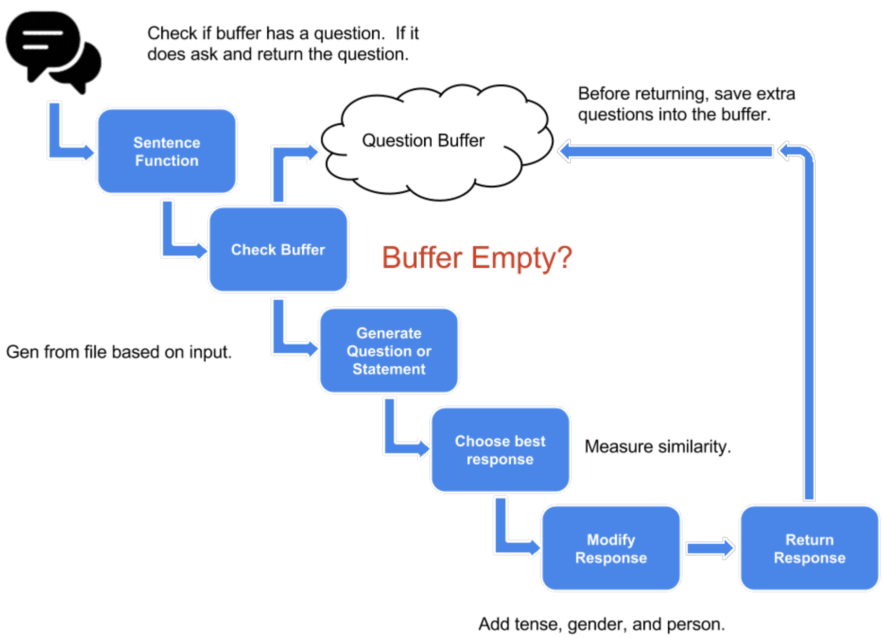


Figure 4.1: High level structure of the response generation.

This chapter will begin by describing how the input is categorized into either a statement, question, or request. Next we describe how the buffer functions. After discussing the buffer we explain how statements and questions are chosen. Finally we go over how the output must be modified by changing the tense, gender or person before the response is returned.

4.2 Speech Act Identification

Much work has been done in the field of speech act identification (SAI). This field attempts to determine the function of a particular utterance. These functions can vary depending on the domain needed. An example of this would be in emails it is necessary to know when a person is requesting the help of another person. In this scenario the function or speech act we would like to identify are requests or commands. Another example would be trying to identify questions or inquiries. There are many different ways to categorize sentences and in our thesis we assume there are three types: *Declarative* (statements), *Interrogative* (questions) and *Imperative* (requests).

You walk to school everyday. (dec)

You walk to school everyday? (int)

You walk to school everyday. (imp)

To make an intelligible response ANA must be able to classify the input into either a question, request, or statement. If a question is detected ANA will attempt to answer that question. Potential questions could be related to the news, weather, or about the user's family. Similarly if a request is detected ANA will attempt to fulfil the request. This could mean alerting the nurse, opening the window, or calling a family member. It is important to identify the type of input, because this will lead to a better response. Applications such as Google Now or Siri most likely distinguish between questions, commands, and statements.

There are many problems in determining the speech act of a sentence. Our primary assumption is that we do not have access to the proper punctuation. Without punctuation this task becomes much more difficult. Usually if the sentence ends with a question mark we can assume it is a question. However the data in which we run our experiments does not have access to this. In this thesis we assume that a sentence can only belong to one class. This is not always true in practice. For example, "Do you have the time?". This can be a request (imperative) or question (interrogative).

Another aspect that is difficult to detect is the intonation of the sentence. The way a sentence is intonated can determine the class even if the content is identical. Take almost any simple sentence, such as “I like fish”, change the intonation, and it will become a different class.

There are times when this is impossible to decipher. However, there are still distinguishing features among the three classes that can aid us in our task.

In order to create our classifier we use the svmlight library [21] in conjunction with the Java layer from Martin Theobald [40]. We create three binary support vector machines, one for each class. We use a linear kernel with the default settings supplied from Theobold’s code.

We will now discuss the features we decided to use. Most of these features are very popular in the current literature such as lexical and syntactic features. However we have not seen other attempts to leverage dependency parsing or word clusters for the speech act identification task. We will now briefly describe the features we use.

Word N-grams: There are phrases and sequences of words that are very influential in certain sentences. We will evaluate different uni-grams, bi-grams, and tri-grams with pointwise mutual information to see if they would be valuable to use as features. The motivation behind this feature set is that there are phrases that are specific to one class. Tag questions usually end with a similar phrase. “You like pizza, right?” or “You like pizza, don’t you?”. Again we used both bi-grams and tri-grams in our task.

Discriminative POS N-grams: The idea here is that many of these sentences contain a certain grammar structure that could potentially identify them. This is why we chose to use POS n-grams. We went through each class of sentence and found all frequent POS bi-grams. Next we needed to eliminate the bi-grams that were not very frequent. This was to make sure that a bi-gram was important to a particular class. We found that bi-grams and tri-grams were the most effective.

The reason we wanted to use POS n-grams was because various phrases within the same class have similar part of speech patterns. Here are two examples: “Open the window”, “Call the Doctor”. They both possess the same POS representation

"VB DET NN".

Dependency Links: We suspect that parsing could potentially be useful in our task. Certain dependency links are present in sentences that determine which speech act it is. We consider pairs of links as features in this work. First we find the root of the sentence and extract all of the outgoing links. Second we find all of the dependent nodes and find all of the outgoing links of those nodes. So for one sentence we will extract multiple pairs of dependency links. Take the link pair "nsubj $\rightarrow$ det". This link pair occurs very frequently within declarative statements such as "The concert begins in two hours.". In Figure 4.2 the extracted pairs would be "nsubj det" and "prep_in num".

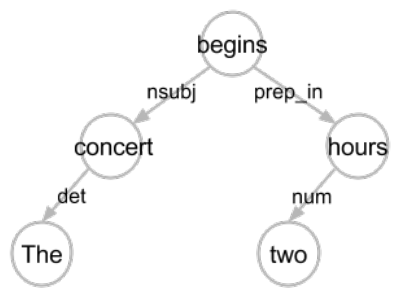


Figure 4.2: Dependency link example.

Word Clusters: In the paper from Turian et al. they evaluate various corpora using word clusters to improve supervised learning [42]. They evaluate the Brown clusters, Collobert and Weston (2008) embeddings, and HLBL (Mnih & Hinton, 2009) embeddings of words on various NLP tasks. We believe that using these clusters could improve the performance of our task. Common imperative verbs such as open and close will be clustered together and could help identify unusual syntactic phrases.

We needed to choose how many word clusters to use. We evaluated the 100, 320, 1000, and 3200 class clusters and found that the 1000 class clusters to be the best. For a given sentence we iterate through every word and determine which cluster it belongs to. A sentence can have multiple word clusters as features.

4.2.1 Experiments

In order to determine whether our classifier is performing well we implemented two baselines to compare against. The first baseline is simply choosing a random class for a given sentence. Each class is weighted equally.

The second baseline implemented is an unsupervised rule based approach. We went through the dataset and attempted to find frequent patterns for each class. These patterns included certain words, part of speech fragments, and phrases. The position of these patterns are also important. For example if the sentence ends with "right" or "eh" or the sentence begins with a verb. Below is a list of the patterns used:

- Sentence begins with a verb.
- Sentence ends with "tag" word (eh, right, really, etc).
- Presence of certain interrogative phrases such as (don't you, do you, isn't it, can't he, etc).
- Sentence begins with "wh-word".
- Presence of emotion or sentiment words.
- Presence of request verbs such as (can, pass, give, will, etc).

Our hypothesis is that our classifier should out perform both of these baselines. We compare these baselines against our five different feature sets. Traditionally in information retrieval the metrics used are a combination of precision, recall, and f-measure. For our task we use: per class precision (P), recall (R), f-measure (F), and total accuracy (A). We calculate the precision, recall, and f-measure for each binary classifier in addition to the overall accuracy of our classifier, and this is evaluated on different combinations of our selected features in Table 4.1.

In order to evaluate which cluster size is optimal we re-ran our experiments on the last feature set (POS+WRD+Clus) with different cluster sizes. This is shown in Table 4.2.

	Declarative			Interrogative			Imperative		
Metrics	P	R	F	P	R	F	P	R	F
Rand	.25	.14	.18	.31	.30	.30	.41	.34	.37
Rule	.39	.90	.55	.83	.13	.23	.91	.58	.71
POS	.69	.83	.76	.81	.66	.72	.83	.88	.85
POS+WRD	.72	.86	**.78**	.81	.63	.71	.86	.88	.87
POS+WRD+Lnks	.70	.82	.75	.76	.63	.69	.86	.87	.87
POS+WRD+Lnks+Clus	.75	.83	**.78**	.85	.70	.77	.84	.92	.87
POS+WRD+Clus	.75	.83	**.78**	.88	.70	**.78**	.86	.92	**.89**

Table 4.1: Speech act results.

Cluster Size	**Accuracy**	**Average F-Measure**
100	79	79%
320	81	80%
1000	82	81%
3200	67	74%

Table 4.2: Cluster size results.

4.2.2 Discussion

The cluster size results shown in Table 4.2 show that the optimal number of clusters should be 1000. As the number of cluster increases the performance seems to increase, but after using 3200 clusters the performance drops considerably.

From our results in Table 4.1 we can see that our SVM approach outperforms both the random and rule-based approaches. The rule-based approach simply does not offer much, because the patterns contain many false-positives. With regard to each class the imperatives achieve the highest f-measure.

Because our task is slightly different than other systems it is difficult to determine how good the results are. That said there are components of our task that are very similar to other work. Much of the related work deals with question detection which is essentially the interrogative class. The paper by Cong et al. achieves an f-measure of 97% on question detection with the use of labelled sequential patterns [10]. This is most likely because their data is much more formal than ours.

We can make a few comparsions to the paper from Qadir and Riloff [32]. They extract speech acts from message board posts. One of their classes is called a directive which is a combination of a question, request, or invitation. They are able to achieve a 86% f-measure.

As stated before a sentence can be classified entirely on how it is intonated. The sentence “I like fish”, can be simultaneously a declarative sentence and interrogative sentence. This is because the emphasis is missing from pure text. Unfortunately there is no way to distinguish between these without the vocal information.

In our dataset there were numerous one or two word sentences. These are tricky because they offer little in the way of context. If given the sentences “yeah”, “right”, “move”, or “yes” out of context could a human classify them into their correct categories? It might be useful to consider the previous sentence or sentences in this case.

The dependency parsing was quite disappointing. It appears that pairs of links were only occasionally useful, and mostly detrimental in classifying sentences. There were a few instances where the parsing did help. The sentence “After the snow storm the air smelled fresh and clean.” is originally labelled as an imperative sentence mostly because of the POS “det nn”, but because the sentence contains the “nsubj $\rightarrow$ det” link it is labelled correctly as a declarative statement.

	Pred Dec	**Pred Int**	**Pred Imp**
Gold Dec	59	5	7
Gold Int	**17**	53	6
Gold Imp	5	2	78

Table 4.3: Confusion matrix.

Table 4.3 shows where the primary problems lies. When attempting to predict the value of an interrogative sentence our system is choosing instead to predict declarative. Instances beginning with “You” such as “You know” or “You are, ... right” are where our system fails consistently. Other issues lie with the intonation problem. For example, the sentence “she enjoyed them” can be both declarative and interrogative depending on factors other than text.

4.3 Knowledge Base Buffer

When having a conversation with another person it is important to remember what has been said. A conversation would be silly if you asked a question again or needed to be corrected multiple times. If a person states they have a sister and then later refers to that sister it is expected that the other person knows that information. ANA employs a buffer process that stores the dialogue and its responses in order to improve the quality of the dialogue. The main issue is that people can reply in such a way that much of the necessary information is not present. If ANA asked "How old are you" and the user responded with "25" it is easy to see that the user is 25. However if you take away the question and just examine the input "25" the meaning becomes less clear.

When ANA receives an input it can store a maximum of two possible questions related to that input. These questions are generated by ANA, not questions within the input. The first question is returned while the second is stored in the buffer. When the first question is answered by the user ANA can check the buffer. In this situation there will be one question in the buffer which ANA can then ask.

In the case when ANA is expecting a particular response, for example ANA is expecting an age, and the expected response is not found then that question still stays in the buffer. It will exist for at most two responses. After this it will disappear. In this way if the user corrects themselves and returns an age in the next response then ANA will connect the previous question with the current response.

4.4 Generating a Response

The majority of the responses from ANA are questions. We believe that questions are a key component in making a conversation feel real. These questions usually are used to gather more information about entities and relations. ANA generates questions based on missing attributes or to disambiguate entities. A list of pre-generated questions is loaded into memory. There are multiple questions for each attribute category. An example of an attribute category and a corresponding question would be "person:age" and "How old is he?".

Questions are meant to fill in missing information from the knowledge base. Each person or event contain a pre-defined list of attributes. ANA attempts to gradually fill in this information by asking questions. A person always has an age, gender, and name. If one of these is missing then ANA may ask a question such as "How old is he?". For each attribute we have defined we have gathered questions that filled in that attribute. We gathered 100 questions total. For the objects and attributes please refer to Table 3.4

```
{'question': 'How old is he?', 'type': 'age'}
{'question': 'What is his age?', 'type': 'age'}
{'question': 'What does he like to eat?', 'type': 'likes'}
```

When choosing a question ANA will attempt to compute the similarity between the input and all possible questions. This process begins by first selecting all possible questions that could be asked. For example let's say we have a person in our knowledge base named Kevin and he is 25 years old. If the input is, "Kevin is going to school", the list of all possible questions includes questions for all unfilled attributes for a person. Once this list is built ANA computes the similarity between the input and this list. To compare S_1 and S_2 all stop words and named entities are removed from both. Then ANA measures the distance between all words in S_1 with those in S_2. The distance is the smallest distance between a single pair. To compute a distance between two words we use a vector representation of words [9] and calculate the euclidean distance between them. Let's say one question from the list was "Where does he go to university?" then the pair "school:university" would have a small distance. Likewise if the question was "Where does he live", the pair "live:school" might have a larger distance. In this way we can choose a more appropriate question.

Disambiguation questions are meant to clarify misunderstandings from the current utterance. If a user references a person that is absent from the knowledge base or discusses a relationship which could have many possible values, an appropriate question needs to be asked. If the user has multiple brothers and references the word "brother" ANA will need to ask which brother the user is referring to. Another scenario could be that the user mentions a person that has not been seen before. In this case ANA needs to ask “Who is $< NAME >$?”.

The are many instances within a conversation where ANA needs to generate a statement. A simple example is if the user greets ANA with a simple hello.

Another type of statement we are interested in are answers. The user has the ability to query ANA for knowledge that she knows. Perhaps the user needs to know what a person likes. They can query ANA with “What does $< NAME >$ like?”. In this scenario ANA needs to check whether she has that information or provide an alternative.

There are times when there is no good indication of what to respond with. In these situations we can respond with a phrase that does not really mean anything, but tries to make the conversation continue. For instance in the sentence, “The sky is blue” there are no entities, relations, or events. In this case we would return a randomly generated meaningless statement such as “yes” or “OK”.

4.4.1 Experiments

Evaluating the response generated by ANA is a difficult task because of its subjective nature. Different people have different ideas on what is more human.

In order to evaluate the quality of ANA’s response generation we created a web application that asked human users to converse with two agents. The user could see both agents and speak with them simultaneously. We decided not to use the classical drug evaluation where half of the participants spoke with ANA and half spoke with an alternative. This is because we did not think we could create questions that could adequately determine the calibre of the agents.

One of these agents was ANA while the other was a chatter bot. This chatterbot was initially Cleverbot which was chosen based on simplicity. After investigation we found that Cleverbot was used more for humour than conversation and we started to use PandoraBot. We chose PandoraBot over alternatives such as Jabberwacky because its dialogue was what we wanted and there was an API available for it.

There were two sets of evaluators. The first set was mostly our peers and the

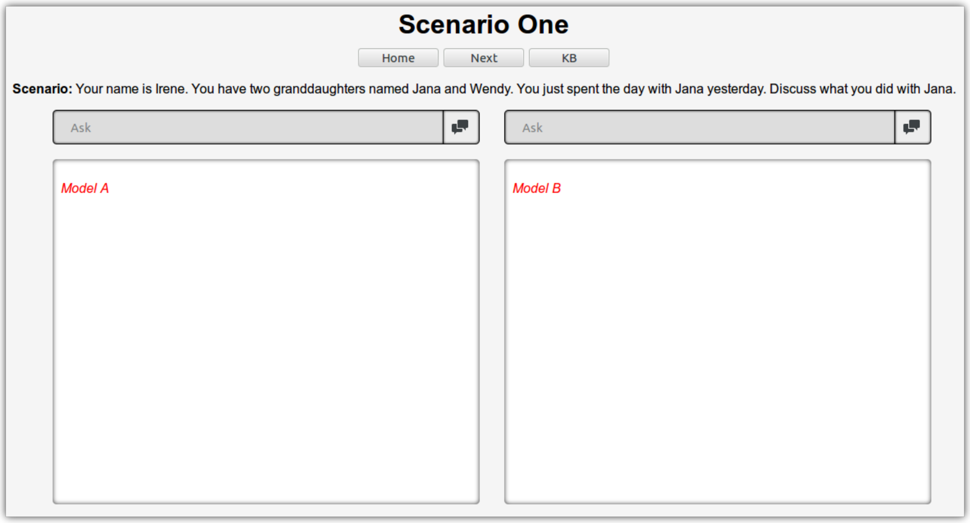


Figure 4.3: The evaluation web application.

second set was mostly students of Dr. Zaiane. It is difficult to say how many people participated because not everyone completed the evaluation completely and there were duplicates that needed to be removed. Our best guess is that in the first round we had around 15 evaluators while the second round had 13.

The evaluator's task was to follow a scenario, constructed by us, and determine which agent sounded more "human-like". They were provided with textual instruction as well as a short 5 minute video. In our context we define more "human-like" as staying as close as possible to a given conversational scenario. These scenarios were designed to restrict the conversation and show what ANA is capable of. An example of a scenario is:

```
Your name is Irene. You have two granddaughters named
Jana and Wendy.  You just spent the day with Jana
yesterday. Discuss what you did with her.
```

The hope is that the user will discuss their day with Jana and perhaps describe her a bit. The scenario was required in order to make sure the users stayed within a certain area of conversation. Our results for both rounds are displayed in Table 4.4

We needed to run this experiment twice because the results in the first round were not statistically significant. There were a few technical problems that may

Round	Chose ANA	Chose Other	Neither	Total	ANA Accuracy
1	38	27	14	79	0.59
2	44	18	11	73	**0.71**

Table 4.4: Results with average correct, partial, and incorrect.

have influenced the users and thus we decided to attempt a second evaluation. One such mistake was using a static variable for part of the knowledge base. For this second evaluation we added two new features, the context window and sentence-question similarity measure. We also fixed the numerous technical problems. Each evaluator was given a choice between choosing ANA, the alternative, or neither. The results are shown in Table 4.5.

After the first round results trickled in we wanted to calculate the statistical significance. To do this we randomly selected a ten subsets of ten test points and calculated the percentage of ANA being chosen. To select the subsets we used the following SQL query:

```
select count(*) from
    (select * from chat order by rand() limit 10)
        as t where (model = ''a'' and correct = ''a'')
            or (model = ''b'' and correct=''b'');
```

The results are shown in Table 4.5. Each column represents a single subset. The float number represents the percentage of people that chose ANA or did not. Each column sums to one. These two arrays are the input to the t-test.

Sample #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
ANA	0.5	0.4	0.5	0.7	0.7	0.4	0.3	0.7	0.3	0.5
Other	0.5	0.6	0.5	0.3	0.3	0.6	0.7	0.3	0.7	0.5

Table 4.5: Round 1 random sample results.

Next we hypothesized that ANA was indeed better by setting the null hypothesis to be the two array's were the same. We ended up with a t-value of 0.0 and a p-value of 1.0. This meant that our results were not statistically significant. After we received the new results from the second round we again calculate the statistical significance as shown in Table 4.6.

Sample #	1	2	3	4	5	6	7	8	9	10
ANA	0.6	0.6	0.4	0.5	0.8	0.8	0.5	0.6	1.0	0.6
Other	0.4	0.4	0.6	0.5	0.2	0.2	0.5	0.4	0.0	0.4

Table 4.6: Round 2 random sample results.

This scored a p-value of 0.0024 and a t-value of -3.52. This proves that our results are statistically significant.

We created another evaluation web application that was designed to evaluate the quality of our information extraction. The idea is that the users would speak with ANA and gradually build up the knowledge base. They would see a graphical representation of the knowledge base and determine if there was any missing information. Unfortunately many of the evaluators chose to skip this part and the results were not clear. In general the users agreed that the relation extraction, named entity recognition, and event detection were good, while the attribute extraction and medical component were not working as intended.

Figure 4.4: The unused evaluation application.

4.4.2 Discussion

In terms of the user study, the results in the first round of user evaluation were not statistically significant. Although approximately 59% of people chose ANA over the alternative, there were only about 15 people who partook in the experiment.

There were a few significant reasons why the results were so poor. First was that priority was incorrect. ANA preferred to ask questions about people rather than events. In the situation where both are present ANA might ask a question of each type rather than focusing on the event. This sounded strange and less human.

Another problem that occurred was when multiple persons or events were mentioned. ANA defaults to using the first person or event that it comes across if there is a conflict. If multiple people are mentioned ANA will only ask about one. This leads into another problem of conjunctions. Once ANA found an interesting aspect of the input it ignores the rest.

An issue that occurred frequently was when there was a problem in the beginning of the information extraction pipeline. Whether the named entity extraction failed or the pronoun resolution was incorrect, this happened quite a bit. If an earlier component fails then the rest will fail. In one example the user told ANA, "Hi, this is my cousin Paul", and failed to correlate "my" with the speaker. This ensures that the relation "cousin" is not found. This is the problem of the pronoun resolution component of the system. Other issues like this include not being able to recognize names, especially foreign names, or locations.

The different methods of phrasing a request caused confusion in ANA. Formally, when requesting an action from another you will include a pronoun or mention of the person involved. Unfortunately in conversation you can request an action by implying an action around another individual. In the situation where a person mentions "I want to know what should we get for our nephew's birthday", this is technically a request, but is phrased in a strange manor. You could also say "I need a suggestion for a birthday gift" which is a request, but lacks a verb indicating the request such as "give".

There were also smaller bugs like not recognizing names such as "You" or "Yi", or guessing the wrong gender for a person. Here is a humourous example of ANA failing. In the example the user is a girl named "You Fu".

```
You: hi.
Ana: Hello! What is your name?
You: i am you.
```

```
Ana: What is your name?
You: you.
Ana: What is your name?
You: Yoyo.
Ana: Nice to meet you!
You: nice to meet you too.
```

The second round evaluation featured much better results. However there are still glaring semantic problems that occur. In the following example ANA cannot realize that a child usually does not have a profession or attend school.

```
You: My nephew's birthday is coming.
Ana: Which university did he graduate from?
You: He is 5.
Ana: What is his profession?
You: He is a kid.
```

These semantic problems occur frequently and should be fixed in the future. The system still struggles when the user changes the subject or it cannot find the answer to its question. ANA has a short window that tries to resolve unanswered questions. In the next example ANA cannot find the correct answer, but still tries to look for it.

```
You: I wonder what Jacob would like as a gift for his
birthday?
Ana: What is his profession?
You: I'm trying to find a gift for Jacob's 5th birthday.
Ana: Where did he go to college?
You: He is in kindergarten
Ana: So.. Jacob has a degree in kindergarten?
```

This is an example of ANA preserving the previous question the buffer. More excerpts are shown in Appendix A. These showcase ANA's strengths and weaknesses.

4.5 Response Modification

There are different aspects of a response that need to be modified. The gender of the speaker or entity can influence the pronouns used in the response. The tense of the sentence can also play a role in which morphemes are used for various verbs. Another aspect that must be addressed is which "person" is being spoken. When speaking between two people the question must be in "second person", while if the conversation is about another individual the dialogue must be in "third person".

In order to make sure the correct gender is used we needed to make a guess as to the gender of the speaker or the subject of the conversation. We downloaded a corpus of names and their frequencies from the US Social Security Administration [2]. We used these frequencies to guess the gender from a given name. If the name was not included in our lists then we defaulted to male. All questions that were pre-generated were in a male specific format. If the name or title is detected to be female then the pronouns are replaced with the female version.

The tense of question is determined by examining the part of speech tags of a given line. Verbs such as "watched", "listened", or "went" indicate the sentence could be past tense and a question should also be in the past tense.

For determining which person to use we simply detect if there are any entities other than the speaker within the text. If only the speaker is present such as "I am ..." then second person is used. Otherwise third person is used.

4.6 Summary

In this chapter we have discussed a process for communicating with a single user that introduces the generation of questions and a process to control the flow of conversation. Given the correct scenario and situation ANA can effectively communicate with an individual. The process we created is a preliminary step for creating a truly intelligent conversation agent. When speaking with another individual it is important to focus on events first before discussing the known entities.

It is also important to make sure that the possible response is appropriate for the given situation. ANA wants to grow its knowledge, but if the response is far out of scope then the agent will sound silly.

In this chapter we have discussed various methods for generating a response within a conversation. We are able to detect the type of input with our sentence class identification which enables ANA to respond in the correct fashion. Our knowledge base buffer is able to remember past dialogue and provide a limited window that allows ANA to sound more intelligent. Finally we described how the response is chosen, ranked, and modified before it is returned to the user.

Chapter 5

Conclusion

Creating an intelligent agent that can converse with an elderly person, and also assist them with their medical needs is a challenging task. This system has many modules and components that need to exist together flawlessly. We made several assumptions that eased the issues with combining multiple modules, but these assumptions are a convenience and any real world application will need to consider a way to minimize these problems.

The potential effect ANA can have amongst the elderly community could be huge. As stated previously the elderly are very likely to live alone and not only need assistance remembering simple tasks but also with having friendly conversation. If an intelligent agent were able to successfully decrease depression and anxiety amongst the elderly then we can consider our goal achieved.

The current state of the system is quite good. The code and methods have been documented well. A new student or team could easily pick up this project and continue towards a more polished product. The use of a pipeline within ANA is inherently flawed. The potential for error propagation is enormous. In addition when a module is improved it does not mean that the entire system is improved. Time spent increasing the quality of a component, such as relation extraction, could have minimal impact on the quality of the entire system.

Moving forward, it will be critical to replace the current pipelined approach with a more suitable architecture. An architecture that allows the potential errors of certain modules. We are thinking about a system similar to what IBM's Watson uses [14]. Watson, an artificial intelligence program, uses multiple approaches to generate a response and chooses the best choice.

Our system's process for dealing with conversation is simple, but effective. In the scenarios provided to the evaluators ANA was able to maintain a decent dialogue and the knowledge extracted was sufficient.

There are assumptions that are made in this thesis in order to make this task feasible. The primary assumption made is that the speech-to-text component will always translate the audio correctly into text. This is a difficult problem. Our system requires proper punctuation, grammar, and capitalization in order to provide a decent conversation. In the future it will be necessary to include a system of adding these dependencies into the speed-to-text module.

The attribute and medical extraction were two components where we did not dive very deep. Each use shallow patterns and limited use of dependency parsing. Neither were evaluated particularity thoroughly. Future research could explore more attributes and a way of generalizing whether an attribute is present, similar to our event extraction. This also could be applied to medical extraction where first ANA detects whether a medical utterance is present, before attempting to establish which medical issue the user is discussing.

A very interesting question that we came across while building this system is what to do when you have no idea what the user is saying. This can frequently occur within a conversation and an agent will need a method of dealing with it. Our system has a list of phrases where there is not a direct meaning, but it is a means to continue the conversation. The chatterbots will often change the topic which can be effective, but that cannot be the solution everytime.

Appendix A

Dialogue Excerpts

The following are dialogues taken from the evaluation. They showcase ANA's strengths and weaknesses.

```
You: Hi, this is my cousin Paul
Model: Hello.
You: Paul is a teacher.
Model: What is his profession?
You: He is an elementary teacher.
Model: Which university did he graduate from?
You: University of Ottawa
Model: Sure.
```

```
You: When is Jacob's birthday?
Model: Jacob was born on November 11, 2008
You: Does he like lego?
Model: Ooook.
You: Do you have any ideas on what to buy for him?
Model: Alright.
You: What do you think I should buy him?
Model: Interesting.
```

You: My name is Phil and I am a retired teacher.

Model: Nice to meet you!

You: I like to watch sports. Do you like sports?

Model: Where did you go to college?

You: I went to the U of A

Model: I see, Phil is a U.

You: My name is Phil and I like todo puzzle games.

Model: I see, Phil is a games.

You: Jana and I went to lunch

Model: When did it start?

You: at 11

Model: Where was it?

You: We went to Tim Horton’s

Model: Jealous.

You: Their soup is awesome. Have you ever had it?

Model: Oh ok.

You: It was her treat because she had a gift card.

Model: What is your profession ?

You: I am a teacher.

Model: Where did you go to college ?

You: University of Alberta

Model: Oh ok.

Bibliography

[1] Drugs Commonly Prescribed to the Elderly. http://www.homemeds.org/images/medialibrary/0620A6A37F028F33FEDC05FC1272963D.pdf. [Online; accessed 01-September-2013].

[2] US Social Security Administration. Popular Baby Names. http://www.ssa.gov/oact/babynames/limits.html, 2012. [Online; accessed 01-September-2013].

[3] Apoorv Agarwal and Owen Rambow. Automatic detection and classification of social events. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1024–1034. Association for Computational Linguistics, 2010.

[4] David Graff Alexandra Canavan and George Zipperlen. CALLHOME American English Speech. http://catalog.ldc.upenn.edu/LDC97S42/, 1997. [Online; accessed 01-September-2013].

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[6] Population Division Census Bureau. Projections of the Population by Selected Age Groups and Sex for the United States. http://www.census.gov/population/projections/data/

national/2012/summarytables.html, 2012. [Online; accessed 30-January-2013].

[7] William W Cohen, Vitor R Carvalho, and Tom M Mitchell. Learning to classify email into“speech acts”. In *EMNLP*, pages 309–316, 2004.

[8] Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press, 2001.

[9] Ronan Collobert and Jason Weston. Word Representations. http://metaoptimize.com/projects/wordreprs/. [Online; accessed 01-September-2013].

[10] Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. Finding question-answer pairs from online forums. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 467–474. ACM, 2008.

[11] B Jack Copeland. The essential turing. the ideas that gave birth to the computer age. *AMC*, 10:12, 2004.

[12] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[13] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING ’10, pages 277–285, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[14] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[15] Debora Field, Roberta Catizone, WeiWei Cheng, Alexiei Dingli, Simon Worgan, Lei Ye, and Yorick Wilks. The senior companion: a semantic web dialogue system. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 1383–1384. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[16] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[17] John J. Godfrey and Edward Holliman. Switchboard-1 Release 2. `http://catalog.ldc.upenn.edu/LDC97S62/`, 1997. [Online; accessed 01-September-2013].

[18] Michael Heilman and Noah A Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617. Association for Computational Linguistics, 2010.

[19] Minwoo Jeong, Chin-Yew Lin, and Gary Geunbae Lee. Semi-supervised speech act recognition in emails and forums. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1250–1259. Association for Computational Linguistics, 2009.

[20] Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. Overview of the tac 2010 knowledge base population track. In *In Third Text Analysis Conference (TAC*, 2010.

[21] Thorsten Joachims. Making large scale svm learning practical. 1999.

[22] Taku Kudo and Yuji Matsumoto. A boosting algorithm for classification of semi-structured text. In *EMNLP*, volume 4, pages 301–308, 2004.

[23] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[24] Andrew Lampert, Robert Dale, and Cecile Paris. Detecting emails containing requests for action. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 984–992. Association for Computational Linguistics, 2010.

[25] Man Lan, Yu Zhe Zhang, Yue Lu, Jian Su, and Chew Lim Tan. Which who are they? people attribute extraction and disambiguation in web search results. In *2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference*, 2009.

[26] Baichuan Li, Xiance Si, Michael R Lyu, Irwin King, and Edward Y Chang. Question identification on twitter. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 2477–2480. ACM, 2011.

[27] Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics, 2012.

[28] Alessandro Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *Machine Learning: ECML 2006*, pages 318–329. Springer, 2006.

[29] E. Nist Ac. Automatic Content Extraction 2008 Evaluation Plan (ACE08), April 2008.

[30] Jian Pei, Helen Pinto, Qiming Chen, Jiawei Han, Behzad Mortazavi-Asl, Umeshwar Dayal, and Mei-Chun Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *2013 IEEE 29th Inter-*

national Conference on Data Engineering (ICDE), pages 0215–0215. IEEE Computer Society, 2001.

[31] Hugo Pinto, Yorick Wilks, Roberta Catizone, and Alexiei Dingli. The senior companion multiagent dialogue system. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, pages 1245–1248. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[32] Ashequl Qadir and Ellen Riloff. Classifying sentences as speech acts in message board posts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 748–758. Association for Computational Linguistics, 2011.

[33] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501. Association for Computational Linguistics, 2010.

[34] Parminder Raina, David Waltner-Toews, Brenda Bonnett, Christel Woodward, and Tom Abernathy. Influence of companion animals on the physical and psychological health of older people: an analysis of a one-year longitudinal study. *Journal of the American Geriatrics Society*, 1999.

[35] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[36] SHIGETOSHI Shiotani, T Tomonaka, K Kemmotsu, S Asano, K Oonishi, and RYOUTA Hiura. World's first full-fledged communication robot" wakamaru" capable of living with family and supporting persons. *Mitsubishi Juko Giho*, 43(1):44–45, 2006.

[37] Lokesh Shrestha and Kathleen McKeown. Detection of question-answer pairs in email conversations. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[38] Cameron Smith, Nigel Crook, Simon Dobnik, Daniel Charlton, Johan Boye, Stephen Pulman, Raul Santos de la Camara, Markku Turunen, David Benyon, Jay Bradley, et al. Interaction strategies for an affective conversational agent. *Presence: Teleoperators and Virtual Environments*, 20(5):395–411, 2011.

[39] Canada Statistics. Living arrangements of Seniors. `http://www12.statcan.ca/census-recensement/2011/as-sa/98-312-x/98-312-x2011003_4-eng.cfm`, 2013. [Online; accessed 30-January-2013].

[40] Martin Theobald. SVMLight Layer. `http://adrem.ua.ac.be/~tmartin/`, 1999. [Online; accessed 01-January-2013].

[41] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[42] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[43] Alan M Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

[44] Richard Wallace. The elements of aiml style. *Alice AI Foundation*, 2003.

[45] Joseph Weizenbaum. Elizaa computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[46] Ying Xu, Mi-Young Kim, Kevin Quinn, Randy Goebel, and Denilson Barbosa. Open information extraction with tree kernels. In *Proceedings of NAACL-HLT*, pages 868–877, 2013.