**A Personal Conversational Model**

*Lawrence Lin Murata* *lmurata@stanford.edu*
Stanford University

## 1. Introduction

Conversation with machines has been an important topic in both research and fiction for a long time. It makes sense for natural language to become the primary way in which we interact with devices, because that's how humans communicate with each other. Thus, the possibility of having conversations with machines would make our interaction much more smooth and human-like. In this project, we explored different methods that model distinct aspects of language.

## 2. Defining Metrics

One of our challenges was to define success. To do so, we defined our own test that suits our project goals and the task we want to achieve better.

### 2.1. Turing test

The Turing test can penalize mistakes too harshly. The main problem with the test is that, since it's a binary test ("human" or "not human"), it can hide progress. For instance, a conversational model might have bad grammar and bad semantics. However, if we make improvements in the model that result in better grammar but the semantics are still bad, the Turing test might still classify all the responses generated as "not human," hiding the progress that has been made in the grammar aspect of the model.

### 2.2 Granular Conversation Test

#### 2.2.1 Description

We asked ourselves and other people "what makes a conversation human?" The intention of this new test we developed was to break down a human conversation into a few key components so that we could do grade our conversational models in a more granular way. This test breaks down a conversation into 4 different aspects:

1. Semantics: "Does the response make sense by itself?"
2. Structure: "Is it understandable and does it follow a structure a human would use?"
3. Context: "Does the response make sense in the context?"
4. Feeling: "Does the response feel human?"

Each component receives a score of 0 or 1 ("no" or "yes," respectively). The result of the test is the sum of the score of each of the 4 components.

#### 2.2.2 Existing chatbots

We tested existing chatbots and virtual assistants with a basic conversation. The questions were "hello," "how are you," "what's your name," "when were you born," "what year were you born," "where are you from," "are you a man or a woman," "why are we here," "okay bye," "see you later."

For each response given to each question, we graded each of the 4 components as 0 and 1. The score of the response was the sum of the grades of all the 4 components. The final average score for the basic conversation was the average of all the scores for each of the responses.

Note: ALICE is considered one of the best chatbots available online.

| Chatbot/Assistant | Average score for basic conversation |
|---|---|
| Siri | 2.7/4 = 67.5% |
| ALICE | 3/4 = 75% |

## 5. Methods

### 5.1 N-gram Search

We defined our model as a search problem, where we are searching for the words that will be part of our reply. Thus, in our framework, a state is defined as an ordered set of words that could be added to the reply. Each state has one more word than the previous one. We have started tackling the task of generating replies by using a very basic n-grams model. We start by choosing a word that starts the reply and a number of words we want our reply to have. The starting word is our initial state. Right now, we are choosing the starting word ourselves, but in the future, we plan on generating the starting word based on the last message (the one that we are replying to). We are also manually inputting the numbers of words we want in our reply ourselves, but in the future we want to account for that by using grammar and stopping the reply when a punctuation mark is encountered.

#### 5.1.1 Overview

Starting from that word, we are running n-grams using our dataset, constructing the sentence by matching consecutive tuples of n-words. Thus, the way we go from a state to another one is by taking the last n-1 words from the current state and looking for the n-tuple that starts with those n-1 words that happens with highest frequency and choosing that as our next state. So far, we have tried 2-grams and 3-grams. Basically, in generating the reply, we are always matching the last n-1 tuples of the last part that was generating, taking the match that happens with highest probability. We have a recursive function that takes in n-1 words, a desired number of words in the reply and a level as arguments. The n-1 words account for the n-1 words of the last part of the reply that was generated. Thus, we know that the part of the reply that was generated so far ends with n-1 words when we enter the recursive function. The function finds the tuple that starts with those n-1 words that happens with highest probability and adds that part to the reply.

For example, to start with, we find all n-tuples (in our case, 2 or 3) in our dataset that begin with the starting word. Then, we choose the n-tuple that happens with highest probability. Then, we do the same thing, but match the tuples that start with the previous n-1 words and take the one that occurs with highest probability. We keep repeating the process until we generate the number of words that we want. To know when to stop, we are keeping a level that tells us how many words we have added to the reply. We are stopping when the level equals the desired number of words in the sentence.

Trying on 2-grams and 3-grams, we noticed that the 2-gram method did not work that well, generating non-sense sentences. This makes sense and we expected this would happen, because our 2-gram model always chooses the next word that has the highest probability. Basically, at each step, we look at the last word that was generated and we choose the word that happens with highest probability in the dataset after it and add that to the reply. This does not account for the context in which the words are used, gathering together words that don't make sense to be added in the same context. Another issue that we ran into is that if let's say we have word1 and word2, and word1 happens with highest probability out of all the words that happen after word2 and word2 happens with highest probability among all the words that follow after word1, then we end up with a cyclic sequence [word1 word2 word1 word2...], which obviously doesn't make sense in a conversational model. Moreover, the same issue holds for word1 appearing more times in the same sentence, because word2 will be the one generated each time after word1, so we would end up with something like “I go to the table because the table is the table most] (word2 will always repeat after word1). We accounted for that by having a set to keep track of the words we have added so far in the reply, and only choosing the next highest probability word that has not already been chosen (is not in the dataset). Of course, this is not the most optimal solution, and in the future we want to somehow account for the context in which the word is used.

The 3-gram model, which we stopped on, seemed to give visibly better results. The replies make sense and could definitely be part of an English conversation. This makes sense, because now we are not only taking the next high probability word, but we are matching for sequences of words in our datasets, which is why we get sentences that make more sense. Examples are provided in the Replies section.

**5.1.2 Examples**

Examples of replies are:

| Reply | Dataset | starting word #words/ sentence |
|---|---|---|
| no no no no. Nono, this is so great! | Friends | “no” n=8 |
| You know what? Just get out of the day, | Friends | “You”, n=8 |
| Well then get it, get it! | Friends | “Well” n=6 |
| being polite, but, alright. | Friends | “being”, n=4 |
| maybe it was just being nice, y'know, | Friends | “maybe”n=7 |
| Hey Pheebs, you remember how we were, uh, y'know, back in high school, I was just being nice, y'know, and- But nothing happened, nothing- Ask Joey, | Friends | Hey n=25 |
| why would you just enter and open the light? | Conversation1 | “why” n=8 |
| they have so serious relationships this should not happen | Conversation1 | “they” n=8 |
| hmm okay, will put the photos on fb | Conversation1 | “hmm”n=7 |
| Hey are you gonna be in SF or something! I'm leaving the apt now | Conversation2 | “Hey” n=13 |
| I don't know if I can get in? | Conversation2 | “I” n=8 |
| maybe we can figure smth out/we could still meet | Conversation2 | “maybe”n=8 |
| They often `came down' handsomely, and Scrooge walked out with a touch of such weather as that, instead of being beyond the narrow limits of our ancestors is in the hall cried. `Bring down Master Scrooge's box, there!' and in its Christmas dress; but the two apprentices, who were not to | A Christmas Carol | “They” n=50 |
| He was not alone, but sat by the sad event, but that he was told, and held it up, high up, to shed its light would have been a copy of old Marley's head on every one. `Humbug!' said Scrooge; and walked across the hall, and glancing through the air | A Christmas Carol | He n=50 |

| | | |
|---|---|---|
| winter of his bed were drawn aside, I tell you! humbug!' At this the spirit raised a cry, and shook its chain with such energy of action, that the baby sallied out to dance with Mrs Fezziwig. As to measuring her waist in sport, as they went past? Why was he | A Christmas Carol | winter 0n=5 |

#### 5.1.3 Datasets

We have four datasets that we used as lexicons for our model. One of them consists of all the scripts from the first season of *Friends* TV Show (taken from http://www.friendstranscripts.tk - appx. 12000 lines), two of them consist real personal conversations through iMessaging between one of us and one of her friends (one of them has 1260 lines and the the one 2740 lines) and the last one consists on the integral book *A christmas Carol*, (taken from http://www.fulltextarchive.com).

#### 5.1.4 Limitations of the model:

1. Uses a preset number of words to know when to stop adding words to the reply; ideally, should know to end the sentence based on context/punctuation marks for the end of the sentence. Right now, it keeps adding words even after punctuation marks
2. The datasets are still not comprehensive. The two datasets consisting of real conversations are still not definitory for generating a wide variety of replies; the replies are limited in scope to the specific of the conversations/topics/language used between the people in those conversations; we need to gather more datasets and conversations between different people
3. Only works with a starting word, but should also work with a starting sequence of words.
4. We are presetting the starting word, but we should be able to get it from the last reply, to imitate the flow of a real conversation; we need to add the context factor somehow.
5. The replies are repetitive and don't account for the context, and would obviously not fit any previous reply

### 5.2 Logic

Our logic model parses human utterances into first order logic rules. By enforcing logic rules, the model is able to draw inferences from facts given by the human. Then the model generates responses and answers questions according to the inferences it can draw from facts it has previously learned. Disclaimer: the starter code for this model comes from CS221, an Artificial Intelligence class, but it was extended for this project.

#### 5.2.1 Overview

Logic is an important component of human communication, because it provides a compact language for modeling, which gives us more expressivity. It is interesting that historically, logic was one of the first things that AI researchers started with in the 1950s. While logical approaches were in a way quite sophisticated, they did not work well on complex real-world tasks with noise and uncertainty. On the other hand, methods based on probability and machine learning naturally handle noise and uncertainty, which is why they presently dominate the AI landscape. However, they have yet to be applied successfully to tasks that require really sophisticated reasoning.

At an abstract level, one fundamental thing a good personal assistant should be able to do is to take in information from people and be able to answer questions that require drawing inferences from the facts. In some sense, telling the system information is like machine learning, but it feels like a very different form of learning than seeing 10M images and their labels or 10M sentences and their translations. The type of information we get here is both more heterogenous, more abstract, and the expectation is that we process it more deeply (we don't want to have to tell our personal assistant 100 times that we prefer morning meetings).

Our logic method models language according to logic. It is very limited compared to statistical approaches, yet efficient for some and fast. Another advantage of the model is that it doesn't require any data or training since it relies solely on logic to generate responses. Our model parses utterances from the human user into first-order logic formulas. The model then enforces logic rules for the first-order formulas. It is able to parse the following types of utterances:

1. Every X is a Y.
2. X is a Y.
3. Is X a Y?
4. Every X has a Y.
5. No X has a Y.
6. If a X has a Y, then it has a Z.

#### 5.2.2 Examples

Human: Pluto is a dog.
Machine: I learned something.
Human: Every dog is a mammal.
Machine: I learned something.
Human: Is Pluto a mammal?
Machine: Yes.
Human: No dog has a fin.
Machine: I learned something.
Human: Is Pluto a dolphin?
Machine: I don't know.
Human: Every dolphin has a fin.
Machine: I learned something.
Human: Is Pluto a dolphin?
Machine: No.

#### 5.2.3 Limitations

There are some big limitations about the logic model. First off, the number of replies and utterances it can work with is limited by the logic formulas and by what human utterances are being parsed into logic formulas. It can’t have any conversations outside of logic-related conversations.

The other drawback is that this model assumes that humans always follow logic in their utterances. Natural language is tricky, because it is replete with ambiguities and vagueness. Thus, drawing inferences using natural languages can be quite slippery. This is not true because humans are not always logical, and because their sentences do not always have the same meaning. One example that illustrates this problem is:

A penny is better than nothing.
Nothing is better than world peace.
Therefore, a penny is better than world peace.

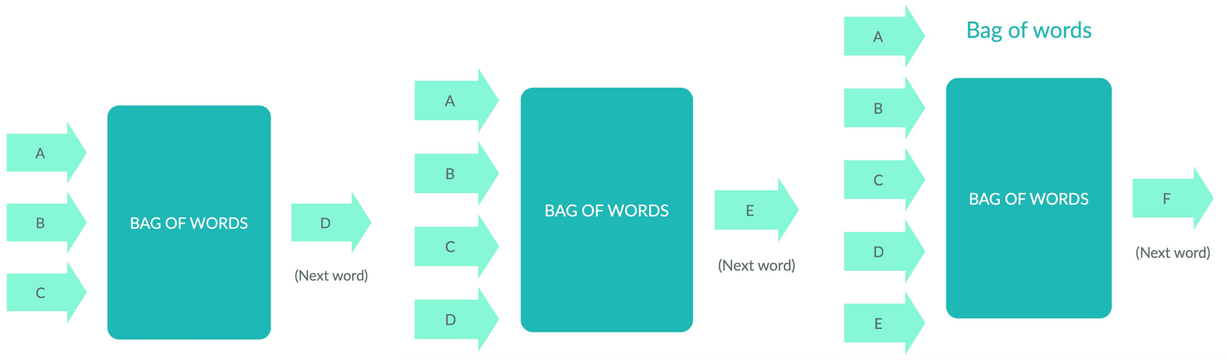
### 5.3 Recurrent Statistical Model

Our statistical conversation model is the one that allowed us to generate the most flexible conversational models, even though responses naturally didn’t always follow a correct grammatical structure. This model takes the human’s utterance as an input for a bag of words. The bag of words is used as an input for our machine learning model (SVM) to predict the next word. Then the next word is added into the bag of words in order to predict the next word. This step is repeated until an <EOS> tag is predicted, indicating the end of the machine’s response.

### 5.3.2 The model

In this example, we have:

Human: “A B C”
Machine: “D E F”




### 5.3.2 Data and feature engineering

We use multiple data sets from movies, TV series and conversations. Since we wanted to build basic conversations, our main data set is made of a collection of conversations that are used to teach English as a second language.

We use TF*IDF, and a mix of uni-grams and bi-grams as features. Our SVM model is a linear SVC that uses the “one-vs-the-rest” multi-class strategy.

### 5.3.3 Examples

In this section, we describe experimental results with three data sets. One of them is a small data set of basic conversations, then a data set with a small portion of the Season 2 of the Friends TV series, and finally a data set with the entire Season 2 of Friends.

#### 5.3.3.1 Basic Conversation

H: hello
M: how are you (4)
H: good
M: what is the purpose of to living (3)
H: what does life mean to you
M: is the purpose of morality (2)
H: why do you keep living
M: to think about he (2)
H: tell me what the meaning of life is
M: the fact that you have child (3)
H: do dogs have a tail
M: yes (4)
H: see you
M: later bye (4)

Average GCT (Granular Conversation Test) score: 3.14/4 = 78.6%

#### 5.3.3.2 Friends Season 2 (small portion)

H: hello
M: hi (4)
H: how are you
M: good (4)
H: how are you today
M: yeah good (3)
H: do you want to go out
M: with me (3)
H: yes
M: oh my god (4)
H: let's head out to grab coffee
M: me yeah (2)
H: do you want to go somewhere
M: c'mon (3)
H: are you having fun today
M: yeah ok (2)
H: i have to pee
M: i'll cancel the sodas (3)

Average GCT (Granular Conversation Test) score: 3.11/4 = 77.8%

#### 5.3.3.3 Friends Season 2 (entire season)

H: hello
M: hey (4)
H: how are you
M: good luck (3)
H: what do you want
M: you me (3)
H: why you me
M: don't know it's just so mean to like be who he's not here yet so and soooooo are you going (1)
H: this will not work
M: is it out' of your league exits (1)
H: i can't do this
M: oh we just delighted so uh you met yes someone huh (2)
H: you are too awkward
M: so much (4)
H: you are so creepy
M: oh much better than the coffee house (3)
H: good bye
M: bye (4)

Average GCT (Granular Conversation Test) score: 2.8/4 = 69.4%

## 6. Related Work

1. Oriol Vinyals and Quoc Le from Google have used a sequence to sequence framework to create a state-of-the-art neural conversation model. http://arxiv.org/pdf/1506.05869v1.pdf
2. Tiedmann has collected a data base of dialogues from movie transcripts. (Tiedemann, J. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In Nicolov, N., Bontcheva, K., Angelova, G., and Mitkov, R. (eds.), Recent Advances in Natural Language Processing, volume V, pp. 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, 2009. ISBN 978 90 272 4825 1)
3. Luong and Sutskever have create a sequence to sequence model. .Luong, T., Sutskever, I., Le, Q. V., Vinyals, O., and Zaremba, W. Addressing the rare word problem in neural machine translation. arXiv preprint arXiv:1410.8206, 2014.
4. Shang et al used neural models for short conversations. Shang, L., Lu, Z., and Li, H. Neural responding machine for short-text conversation. In Proceedings of ACL, 2015.
5. Enarvi, Seppo, and Mikko Kurimo. "Studies on Training Text Selection for Conversational Finnish Language Modeling." *Proceedings of the 10th International Workshop on Spoken Language Translation (IWSLT 2013)*. 2013.