Towards a Reverse Turing Test Framework for Chatbots

Yuya Ong
DS 310: Machine Learning Final Exam

Introduction

Since the birth of the term "Artificial Intelligence" has emerged, researchers have been working on various systems which attempts to build models which attempts to emulate the functionalities of human capabilities. In particular, chatbots, or a conversational agent which communicates between a given user or entity through the means of speech or text-based platforms, have often been used as a method to benchmark intelligence and their capabilities to test whether they possess human-like features and attributes. Over the past decade, there have been many attempts in building such systems - beginning from the mid-60s with the ELIZA bot and bots based on AIML, which later emerged in the late 80s with bots such as A.L.I.C.E., which utilizes pattern matching methods to craft conversational patterns. Currently, modern systems such as Siri, Cortana, and Google's Allo have made way for the development of sophisticated interactive chat systems which utilizes speech based input to interact with the user - however, much of the interactive based functionality are often scripted or primarily a query based approach, making it less-so human than how it appears to be. However, efforts towards building and modeling a realistic conversational flow have been progressing with the recent development in generative language modeling through the use of recurrent neural network based models - notably demonstrated with Microsoft's chatbot systems such as Xiao Ice, Rinna, Tay, Zo, and Ruuh. These platforms utilize crowdsourced based interaction data to model the most appropriate response and utilize a generative modeling techniques from existing chat logs to respond to user's queries.

One important and most notable test that has been used to gauge such "humanness" is the Turing Test, which was proposed by Alan Turing. The test is defined by having three different entities - A, B, C who are respectively a computer, human, and a mediator judge - in this case defined as a human being. The judge, C, will converse with A and B - not knowing who they are chatting to, who then will determine whether the entity they spoke to is a human or a bot. Turing predicted that if the computer is able to fool the judge, on average, by at least 70% of the time that it's a human, then the computer passes the Turing Test.

In this paper, we define a problem framework to address an inverse problem, where instead of C having to be a human based judge, we utilize a computer instead to determine whether the entity the computer is conversing with is a computer or a human. To conceptualize this problem, we have further defined two variants of the problem based on the given model's perspective of the interaction between the entities - through the first and third person. The shift in perspective thus provides a framework for both a retrospective based learning on past data as well as an online based learning mechanism, which allows for the model itself to influence the direction of the conversation to affirm its objectives. The following sections will address the problem formulation, the core features and data inputs, baseline models for modeling such a problem, and the definition of an evaluation framework to determine how successful the model is able to perform the Reverse Turing test.

Problem Formulation

In this section we discuss the key problem formulation of the Reverse Turing Test problem. First, we define an entity E, which can be defined as a function that takes in input and outputs a response given the input value provided by some external source. This entity E is the subject of interest that the judge J, will attempt to discern correctly whether it is a computer or a human.

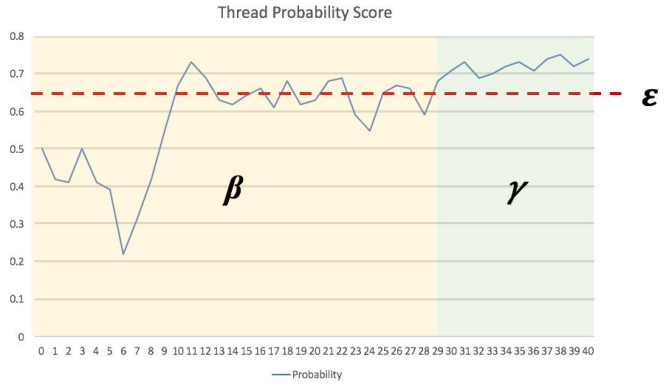
However, the empirical assessment of discriminating E being either a computer or a human can be classified between two specific problem types based on *first and third person perspectives*. While both models attempt to understand and assess whether the entity of interest is either a computer or a human, the key difference in these modeling techniques relies on whether the judge is able to influence the conversation between the entity to provide signals and better contexts to better make informed decisions during the classification process. In other words, the third person perspective only allows the model to view a conversation log between two entities E_1 and E_2, which has already occured in the past - thus, the model J having no influence on the conversation itself. On the other hand, the first person modeling process where the conversation is between only E and J, allows for judge J to provide some influence in the empirical analysis of whether entity E is a human or computer.

Thus, given a conversation stream between two entities, E_1 and E_2, we are given a conversation, $X = \{ x_1^1, x_2^1, \ldots, x_i^j \}$, which is defined as a set of responses defined as x_i^j, where i is the entity identifier and j is the sequence index of the conversational replies with respect to i. For every i, we can attribute the given entity with a binary label, either by 0 or 1, to indicate whether the entity

is a human or a computer, respectively. From this, we define a vector Y, which correspondingly provides the binary label for each of the conversation's corresponding replies, x_i, within the conversational log of set X.

Based on the given input, corresponding label, we define J, as a model that describes a probability function $J = Pr(Y_i \mid x_1^1, x_2^1, ..., x_i^j)$, which models the probability that the corresponding entity i belongs to either class 0 or 1, human or computer, given a subset or entire set of the conversation from the conversation log X. This probability function models the confidence in how likely the given entity is determined to be a computer or a bot given the text based features and interactions held between the two entities conversing with each other. At $Pr(Y_i \mid X)$, where X is a null set, or when the model is not provided with any reply from either entities 1 or 2 initially, we define the probability as 0.5, as we can assume by chance that the entity can either be a human or computer by chance of guessing.

Given this probabilistic framework for modeling the likelihood that the given entity is a human or computer, we now define our objective and cost function for the corresponding problem formulations. In this setting, we define our framework as an interactive process such that we can evaluate the probabilistic convergence rate of the model. We first define $\boldsymbol{\varepsilon}$, a threshold that we define as the *average probabilistic convergence value,* which we establish as the convergence criteria that indicates the pass/fail of the turing test on the average probabilistic confidence the model establishes the given entity is either a human/computer. Furthermore, since the model is based on an interactive and iterative process for understanding the likelihood of the entity being either a human or computer, we want to also evaluate both the rate of convergence to the conclusion that it reached, as well as establishing a consistent prediction of the entity's class - or the precision of the model's ability to successfully discriminate the entity for a given duration of iterations. More formally, we define the parameter $\boldsymbol{\gamma}$, as being the duration or the number of consistent replies such that the average confidence/probability of the model exceeds the defined threshold $\boldsymbol{\varepsilon}$. For parameter $\boldsymbol{\gamma}$, this may also be another parameter which may be a user defined to establish a stopping criteria which determines when to terminate the given Turing Test. As another artifact or criteria for this test is to establish a rate of convergence, or a measurement of how fast the model is able to conclude the belief of the entity being either a human or computer by the parameter β. We define this parameter as being the number of replies it takes for the model to be able to reach and satisfy the stopping criteria of $\boldsymbol{\gamma}$. In short, given we have a large n (or as n tends to infinity), we can define a very successful model as a model that is able to reach a large $\boldsymbol{\varepsilon}$ with respect to a large $\boldsymbol{\gamma}$ consistency with a very small β. The figure, below demonstrates a sample plot that we could generate from a sample conversational model between an entity and the judge with the corresponding probability of the judge based on the given response of the entity.



Model

In this section we describe the various architectures we can utilize to model the two different types of problems. The core modeling architecture we can utilize is a recurrent neural network based model - as we are analyzing sequential information with temporal dependencies on each of the given inputs of the data. Given the corresponding perspective of the modeling method, whether we utilize a first person or third person model, the architecture of the model will differ slightly due to the added generator model in the third person model. However, at the most fundamental level of the model, we have a base discriminator where the input of the model is fed the corresponding text representation as a word embedding vector as well as a single digit value representing the entity identifier and the output is based on a logistic regression based classifier which outputs a probability between 0 and 1.

For a third person based classifier model, we can correspondingly train use an existing word embedding corpus (whether pretrained from an existing corpus set - such as the Google News, Wikipedia, or it's own corpus set), and apply the vectorization process to convert the sequence of words within the sentence to formulate a set of vectors we feed into the model until we reach an <EOS> (end of sentence) signaling vector that indicates termination of a phrase. The vectorization operation is done through the inference process of inferring the given word embedding representation of the raw text data. We assume at this point the input is tokenized and normalized to ensure consistent input with regards to the given input source. Additionally, we feed in the corresponding entity identity value as another input channel into the model. The corresponding temporal weight parameter for the model will utilize a Recurrent Neural Network structure - such as a vanilla RNN, LSTM, Bi-LSTM or GRU to model the temporal dependencies of the word vector. Then, prior to the temporal layer, we can further append a dropout layer as a mode of regularization factor to prevent the model from overfitting. In this scenario, we can utilize a dropout of approximately 20% of the overall weights at random. As the model then gets fed the word vectors and reaches the <EOS> token, we then can analyze the output logistic state of the model to determine the probability that the given sequence of inputs correspond to a computer or a human being's reply. We repeat this cycle interactively as the model continues to observe the corresponding conversation between the two entities as we alternate the inputs between the first and second entity to capture the semantic and sequential dependencies of the conversation.

Conversely, the first person perspective will utilize a very similar model, except, the data we used for returning the response to the entity we are evaluating will be based on a generative model - notably a sequence-to-sequence based model. The objective of this particular sequence to sequence based model is a chatbot model which will attempt to send specific responses given a input response from the entity in which they are currently conversing with to maximize the hypothesis of whether the entity they are conversing with is likely to be a human or a computer. When generating a response, which is essentially sequence of word embedding vectors, the system will correspondingly take two inputs: first the text response from the entity, which will be vectorized through a word embedding based model and the previous prediction generated by the discriminator of the model as a prior belief which the model will use as a background information to prove or disprove whether the model is in fact a computer or human. The intuition behind this process allows us to take the prior belief, say the discriminator believes that the entity is a computer with a probability of 0.6, which will then produce a response which will likely increase the confidence of the model when it receives a response from the entity that will try to maximize the probability of the discriminator confidence to indicate that the entity is in fact a computer. Through introducing a generative processing unit within the model, we can introduce an external influence factor which will help to dynamically update the model's prediction based on the information it has already obtained through conversing with the agent. In particular, other alternative approaches to this modeling problem may also use a variant of a reinforcement learning technique which aims to use a goal oriented approach to prove or disprove whether the entity is a computer or human using methods such as Q-Learning.

Dataset Curation

To curate a dataset for this given task, the dataset we utilize will differ with respect to the modeling perspective we take and how we train the model for the given task at hand. For curating a data set for the third person modeling, we can correspondingly conduct a live study between a pool of users which will join a web based chat room interface, where they will be chatting with either another human or a computer for a given timeframe based on a randomly selected process that pairs the two up. To better analyze the human psychological patterns in how they perceived the other user, the human subjects will also be used to determine whether the other entity was a human or a computer - this data, although will not be used a ground truth data, but as an artifact metadata to analyze particular patterns and strategies that humans also use to be able to determine whether the person is a chatbot or not and how they derive to that conclusion. Simultaneously for the dataset which requires a bot-to-bot interaction, we can relay two models to talk to each other to build a dataset which balances out the number of threads generated based on the sample size of the data.

For the first person dataset collection, we can utilize the same data collection strategy used in the third person modeling to train a chatbot that aims to help the discriminator determine whether the given entity is a chatbot or a human. Prior to training the generative chatbot model, we then can hold another study where we pair up the chatbot and discriminator model against a human and a bot and jointly train the discriminator model and the chatbot and collect the data as we perform the training experimentation through a reinforcement learning procedure.

Conclusion

In short, we have briefly describe a functional framework for how we can potentially conduct a reverse Turing Test through the proposal of a concrete problem formulation, evaluation framework and metric as well as potential modeling and training procedures that can be utilized for this particular problem. We believe that this framework will allow for a standard evaluation benchmark to build a robust model that can dynamically learn to evaluate the identity of a human and computer apart within a rigorous and empirically driven method to identify the entity.