A Framework for Creating Text Parsing Dialogue Systems

A Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment
of the requirements for the degree

Master of Science

by

Martin E Bolger

August 2019

APPROVAL SHEET

This Thesis
is submitted in partial fulfillment of the requirements
for the degree of
Master of Science

Author Signature: _____

This Thesis has been read and approved by the examining committee:

Advisor: Donald Brown

Committee Member: Stephanie Guerlain

Committee Member: Laura Barnes

Committee Member: _____

Committee Member: _____

Committee Member: _____

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, School of Engineering and Applied Science

August 2019

Contents

Abstract

Dialogue-based simulations are simulations designed to teach users methods for interacting with a target population. They have a wide variety of applications including training medical students to interact with patients, teaching military personnel about local cultures before deployment, and giving students learning a foreign language a chance to practice by going through scenarios. Many dialogue-based simulations have used a multiple-choice dialogue system. While there has been recent work on free-input dialogue systems for educational simulations, most frameworks for free-input dialogue systems do not preserve the dialogue tree structure of a multiple-choice dialogue system. This thesis aims to create a framework for transitioning from a multiple-choice dialogue system to a free-input text classification system. The framework includes methods for crowdsourcing for data collection, a binary sub-category data labeling system, and a data generation algorithm. An implementation of the proposed framework in an existing military educational simulation is developed. This implementation replaces the multiple-choice dialogue system with classification models trained on datasets created using the framework. A test of the framework is performed on real user input. The results indicate that this approach may offer a viable method for building free-input dialogue systems for educational simulations.

1 Introduction:

1.1 Problem Statement:

The goal of this research is to develop a framework for converting a multiple-choice dialogue system to a free-input dialogue system. The free-input dialogue system needs to be able to perform detailed classification of input text to provide educational feedback.

1.1.1 Framework:

The main steps in the proposed framework are:

1. Creating a labeled dataset using dataset crowdsourcing methods and a data generation algorithm.
2. Training natural language processing classification models for each section of the simulation using the assembled dataset
3. Testing the models and evaluating the factors that affect their performance

1.2 Implementation:

A series of multiple-choice questions from an existing simulation were converted into free-input text classification models to test the proposed framework. The text classification model for each question is trained on a dataset of free-input responses to the questions from the simulation. The mechanics of the dialogue system are preserved.

1.2.1 Multiple-choice Simulation:

In 2018, Sheridan, et. al. created a VR simulation in which the player acts as a US army officer interacting with a Chinese army officer[1]. The content of the simulation is based on observed interactions between American and Chinese soldiers during a US-China joint training exercise known as the Disaster Management Exchange [1][2]. At specific points in the game, players are asked to make a decision about what to say to the Chinese officer avatar. Each time the player is asked to provide input, they are provided with a text menu containing up to four multiple choice options. Users receive different feedback depending on the appropriateness of the option selected. Table 1 shows the multiple-choice options and feedback for one of the questions from the simulation. In this part of the simulation, the Chinese commander asks the user for a list of the supplies that they are planning to bring on the mission. The user does not have the complete list and is busy preparing for a meeting later, so each response gives an explanation for why the user cannot answer. The multiple-choice options and feedback for all the questions are listed in appendix A

Table 1: Multiple-Choice and Feedback Table Question 4

Multiple-Choice Text	Feedback and Score
I appreciate that you are looking out for your team, but you will get all of it soon when we have a brief this afternoon	Moderately appropriate, displays understanding and empathy towards the officer and his need to know, but is direct in telling him he will not comply. **Score: Best**
Is it alright if this waits until the actual mission brief? I'm busy getting ready for it now.	Has an air of politeness to it that implies if the Chinese officer really needed the information, that you will give it, but it would be inconvenient to do so. **Score: Second Best**
Not all of the details have been finalized so I'm not sure if that it would be of any help to you now.	The response is curt and dismissive. **Score: Worst**

The dialogue trees, multiple-choice options, and feedback for the simulation were created by the West Point Chinese Department [1]. To create the dialogue trees, the dialogue was first translated into Chinese to "[eliminate] any potential variability that could have resulted from the many different ways the English responses could have been translated by the individual raters"[1]. The options in the trees were then rated by three "Chinese cultural experts" (i.e., professors in the Chinese department at West Point) [1]. These ratings give each multiple-choice option a score. This score can be thought of as a representation of the "appropriateness" of the multiple-choice option. After all responses for each prompt were scored, the raters created feedback text to go with each multiple-choice response. The feedback corresponds with the rating they assigned (e.g., a response with a higher score is more "appropriate", so it will have positive feedback text).

Sheridan et. al.'s simulation is only used as an example for the application of the proposed framework to an existing simulation. This thesis does not focus on the validation of the educational effectiveness of the original system, and the non-technical aspects of creating educational simulations for training cross-cultural competence will not be discussed. The background on the educational goals of the simulation is only presented to explain how the original dialogue system functioned. The new dialogue system must be able to give user feedback, but it has to be able to do so by classifying free-input text.

2 Literature Review

2.1 Dialogue-Based Training Simulations:

2.1.1 Multiple-Choice Based Training Simulations:

Educational dialogue-based simulations have been used in variety of fields. Some applications include: teaching medical students effective communication methods for clinical exams [3], exposing military personnel to local culture before deployment [4], and teaching negotiation strategies [5]. Many of education dialogue-based simulations use a multiple-choice dialogue system to simulate conversation. User input can easily be scored based on the educational goal of the simulation because each multiple-choice option has an associated score.

Despite its advantages, multiple-choice training is an oversimplification of real communication. The benefits of free-response questions for testing competency have been studied in past research [6] [7]. Newble et al propose that multiple choice questions may lose their effectiveness due to the "cueing effect of the options"[6]. Therefore, creating a free response classification based dialogue system framework could yield improved educational benefits.

2.1.2 Free-input Based Training Simulations:

While free-input training is more appropriate for creating an educational dialogue system, it is difficult to implement a classification model for a free-input dialogue system. Past research has attempted to address this issue by creating classification models that take advantage of the limited range of input that specific types of free-input dialogue simulations will receive. Others focus on intent-based dialogue classification rather than the classification of subtler text differences like sentiment or politeness.

One area of free-input dialogue system research is computer assisted language learning (CALL) [8]. Students studying a foreign language will naturally have a limited vocabulary, so the classification task is simplified. One of the earliest examples of a CALL application with a free-input dialogue system is Subarashii[9]. Subarashii is a game for helping students practice spoken Japanese[9]. Students go through a variety of scenarios that test their ability to speak Japanese[9]. Speech recognition was performed using a series of Hidden Markov models trained on a corpus of input speech from non-native Japanese speakers. The corpus of speech used to train the Hidden Markov models is created by collecting the text version of potential input from non-native speakers and then recording various speakers vocalizing the input. The dialogue system's functional accuracy was found to be 71.4% and 66.9% in two different rounds of testing[10]. Seneff et al. created a CALL application with a free-input dialogue system for practicing Mandarin Chinese [11]. In this application, users practice by discussing hobbies with a virtual human and arranging a time to "jointly

participate in an activity they are both like"[11]. Speech translation is performed using a " linguistic analysis/synthesis" approach rather than the more prevalent statistical modeling (i.e., machine learning) approach [12]. CSIEC is a CALL application designed to help Chinese students practice English[13]. It is a chatbot. This means that user input is not restricted to specific topics. The author classified the effort as a failure because it lacked the ability to communicate in a realistic way or educate users because it "cant understand the meaning of the sentence syntactically and semantically"[13]. The results of this research emphasize the importance of applying intelligent classification systems to limited domains. If the scope of an educational dialogue-based simulation is too general, it is impossible to give effective feedback.

Other CALL applications use a synergy of tutoring and interactive scenarios to create a free-input dialogue system with a limited range of possible input. Because users are being introduced to specific vocabulary and sentence structures in the tutorial, the classification models are designed to classify the type of sentences that are taught to the user. Ville is a CALL application for learning Swedish that combines a tutoring component and dialogue simulation called DEAL[14]. In DEAL, the user participates in a simulated negotiation with a virtual character. DEAL takes advantage of the limited domain and accompanying tutoring component to simplify the classification problem. The Tactical Language and Culture Training System (TLCTS) is a CALL application that combines language education with cultural competence training[15]. There are tutoring modes in which users learn new dialogue and simulated dialogue sections in which users interact with avatars using foreign languages including Iraqi, Pashto, and French. There is also a Mission Mode that places the user in a scenario that requires them to communicate with avatars from the target culture. The objective of the training is to teach foreign language skills while allowing the player to interact with realistic avatars from the target culture .

There have been attempts to create chatbot style dialogue simulations in which users' input is not restricted by the simulation scenario. One of the most famous is Façade. Façade is a non-educational interactive drama game that allows users to interact through a free-response dialogue system[16]. Façade's text parsing algorithm uses a two step rule-based process. In the first step, the text is sent to a rule-based text parsing algorithm that looks for patterns in the text. The first phase is the Natural Language Understanding phase[17]. The natural language processing system works by first mapping input text to what they call discourse acts. Discourse acts represent the "pragmatic effects of an utterance". Discourse acts are very general versions of responses to user input. Examples include expressing agreement, giving a positive exclamation, expressing confusion, flirting, and giving advice. The second phase involves mapping potential discourse acts to avatar responses . The discourse acts are not associated with specific dialogue until the second phase is reached. The reaction is based on the subsection of the action the story currently occupies and takes into account the user's previous input. The game creators report that their dialogue system

was able to succeed in making a partially or fully sensible response to the user around 70 percent of the time[18]. However, in their game the dialogue system has to only interpret the basic intent of the response. The characters have to react in a believable way when the player speaks to them, but because the game's goal is not to directly critique the player's input, specific text features in the input are not classified.

Other examples of simulations with free-input dialogue systems that utilize a chatbot-like architecture include simulations described in several papers for a research team from the Institute for Creative Technologies at the University of Southern California that focus on building question-answering characters. The design goal for the dialogue system for these question answering characters is "to create an automatic system that uses a set of training question/answer pairs to learn the appropriate question/answer matching algorithm"[19]. Because there are multiple possible answers for each question, they use a two step approach to select an appropriate answer for each input question. First, questions are classified using a multi-class Support Vector Machine model. Next, probabilistic language models are trained to rank potential answers. These language models are based on using the conditional probability of observing specific words in the question. The authors note that questions and answers do not have the same language models (e.g., questions will contain interrogative words like 'who' or 'what'), so they model this task as a "cross-lingual information task" by assuming that the vocabulary for the questions and answers come from different probability distributions[19].

Two systems were created using this question-answer matching technique. The first was SGT Blackwell, a simulation of a soldier who can answer questions about his position in the military[19]. The highest accuracy for question-answer matching for this model was 53.13%. Another project involved the creation of virtual twin scientist characters designed to answer questions from museum visitors about STEM subjects [20]. These virtual human models were installed in the Boston Museum of Science. The authors break questions into two categories: in-domain and out-of-domain. In domain questions are questions the system should be able to answer. Out-of-domain question are questions that the system is not able to answer. Classification is evaluated separately for these two question types. Classification accuracy was also broken into three speech recognition groups: male, female, and child. Testing results indicate that the system was able to correctly identify between 71% and 77% of out-of-domain questions, and in-domain questions were classified correctly between 31% and 72% of the time. The results for transcripts of the input were 77% for out-of-domain questions and 71% for in-domain questions, so a significant portion of the errors were caused by speech recognition errors. The Institute for Creative Technologies has also released a program called the NPCEditor that was used to construct the question-answer matching dialogue systems for these three simulations[21].

2.2 Dataset Collection:

An appropriate dataset for creating a dialogue-based simulations does not typically exist, so dataset creation is necessary. Data crowdsourcing is an online data collection method in which users are paid to answer prompts or label existing data. Crowdsourcing allows the researcher to generate larger datasets that are applicable to a specific problem. Some data crowdsourcing websites include Amazon Mechanical Turk and Figure Eight. Data crowdsourcing methods have been the subject of previous research. Kang et. al. conducted comparisons of various methods for data collection using crowdsourcing by creating "algorithm-independent [metrics] to evaluate the quality of training data and its effectiveness at solving the target [classification] task" [22]. They introduce two metrics: diversity and coverage. Diversity is a measure of the lexical diversity of the data. Coverage measures how well the training dataset covers the space of potential ways to respond to the questions. Coverage measures how well the training set covers the test set. Both metrics were evaluated based on comparison between the classification performance of the collected data and the metric value for the data. It was found that coverage correlates with the training accuracy of the classification model. Past research has also focused on the extraction of text features from large scale online corpuses. These text features can be used to train models to produce a general conversational chatbot system [23]. However, it is difficult to get task specific data from an online corpus, so crowdsourcing is often necessary.

Labeling text is an important consideration when collecting a dataset. Text labels are used to categorize text, and they are supposed to represent the presence or absence of text features. Text labels need to be provided by human raters before data can be used for training classification models. For example, text could be labeled based on the author's use of pathos, logos, and ethos. On a large enough dataset, a text classifier could learn to detect pathos, logos, and ethos in new text data based on features of the text like the presence of specific words. Crowdsourcing of labels for text has been previously investigated by Danescu-Niculescu-Mizi et al. [24]. They were able to use crowdsourced politeness labels for data from the discussion sections of Wikipedia and Stack Exchange to create politeness detection models that were able to achieve near-human-performance on in-domain data (i.e., data from the same website that they were trained using).

2.2.1 Goal of This Research:

The main aim of this thesis is to create a framework for creation a free-response dialogue system from an existing multiple-choice-based dialogue system. Although past research has resulted in free-input dialogue simulations that are able to conduct a conversation and classify input using text features to select appropriate responses, previously proposed dialogue system designs do not take advantage of the structure of a multiple-choice dialogue system. In a multiple-

choice dialogue system, answers are evaluated based on different criteria at each step in the dialogue tree. This means that each step in the dialogue can be treated as a sub-problem with its own classification system based on its own set of labels. The work in this thesis could serve as a starting point for other research into creating dialogue tree-based free-response educational dialogue systems, or the proposed framework could be used to create new simulations with free-input dialogue systems.

3 Text Modeling Background:

This section will give an in-depth overview of the main text modeling techniques that were used for building the text classification models.

3.1 Feature Representations:

Text classification is performed by a machine learning model that is trained to predict the presence or absence of specific features in input text. To train the machine learning models, input text needs to be converted into a numerical representation so that the machine learning model can be trained. These numerical representations of text are called feature representations because they are based on "features" of the text.[1]

3.1.1 Bag-of-Words and Term Frequency Representation:

The most basic feature representation for text data is bag-of-words. Bag-of-words uses the frequency of each word in each document in the corpus to create a vector for each document. For example, given the following three document corpuses:

Document 1: John wants to eat watermelon. Mary wants to eat eggs
Document 2: Mary and I like to eat the same things
Document 3: John and I are going to eat later

The Bag-of-Words representation for each document is shown below:

[1] Note that references to features in text or text features describe the presence of words in the input text whereas feature representations are numerical representations of those text features used to training machine learning models.

D1 = {(John, 0) = 1, (wants,1) = 2, (to,2) = 2 , (eat, 3)=2,
(watermelon,4) =1, (Mary,5) =1, (eggs,6) = 1}

D2 = {(Mary,5) = 1 (and, 6) = 1, (I, 7)= 1, (like, 8) = 1, (to, 2) = 1,
(eat, 3) = 1, (the, 9)=1, (same, 10) = 1, (things, 11) = 1}

D3 = {(John, 0)= 1, (and, 6)=1, (I, 7) = 1, (are = 12) = 1,
(going, 13)=1, (to, 2) = 1, (eat, 3) = 1, (later, 14)=1}

After this bag-of-words representation has been created, it can be converted into a matrix with entries $x_{i,j} = n(t_i \in d_j)$ where n gives the number of times term t_i appears in document d_j.

$$\begin{bmatrix} (John, 0) & 1 & 0 & 1 \\ (likes, 1) & 2 & 0 & 0 \\ (to, 2) & 2 & 1 & 1 \\ (eat, 3) & 2 & 1 & 1 \\ (watermelon, 4) & 1 & 0 & 0 \\ (Mary, 5) & 1 & 1 & 0 \\ (and, 6) & 0 & 1 & 1 \\ (I, 7) & 0 & 1 & 1 \\ (like, 8) & 0 & 1 & 0 \\ (the, 9) & 0 & 1 & 0 \\ (same, 10) & 0 & 1 & 0 \\ (things, 11) & 0 & 1 & 0 \\ (are, 12) & 0 & 0 & 1 \\ (going, 13) & 0 & 0 & 1 \\ (later, 14) & 0 & 0 & 1 \end{bmatrix}$$

Now each column of the matrix gives a numerical representation of each document. There are two main issues with this approach:

1. Words that appear more frequently have a higher weight, but over large corpuses, words that appear frequently have very little predictive power. For example, prepositions and articles will probably appear very frequently, but they will not give any information about the meaning of the text.

2. The proximity of words in the sentence does not affect the output, so the representation is only based on the presence of words, not the way they are used in the sentence.

3.1.2 Term Frequency Document Infrequency

A new feature representation called term frequency inverse document frequency (TFIDF) can be used to deal with the first of these issues. The weights for each term are scaled by the document infrequency of the term. It is based on the intuition that if a term shows up in more of the documents, it will have less

predictive power.

For the i, j^{th} term, we get $weight_i = log(\frac{N}{\sum_{j=1}^{N} u(t_i \in d_j)})$ where N is the total number of documents and:

$$u(x, y) = \begin{cases} 1 & x \in y \\ 0 & otherwise \end{cases}$$

The logarithm is used to reduce the weight difference for larger values. Now the values in the matrix are: $x_{i,j} = n(t_i \in d_j) \cdot weight_i$. The matrix is shown below:

$$\begin{bmatrix} (John, 0) & log(3/2) \approx 0.176 & 0 & log(3/2) \approx 0.176 \\ (likes, 1) & log(3/2) \cdot 2 \approx 0.352 & 0 & 0 \\ (to, 2) & log(1) \cdot 2 = 0 & log(1) = 0 & log(1) = 0 \\ (eat, 3) & log(1) \cdot 2 = 0 & log(1) = 0 & log(1) = 0 \\ (watermelon, 4) & log(3) \approx 0.477 & 0 & 0 \\ (Mary, 5) & log(3/2) \approx 0.176 & log(3/2) \approx 0.176 & 0 \\ (and, 6) & 0 & log(3/2) \approx 0.176 & log(3/2) \approx 0.176 \\ (I, 7) & 0 & log(3/2) \approx 0.176 & log(3/2) \approx 0.176 \\ (like, 8) & 0 & log(3) \approx 0.477 & 0 \\ (the, 9) & 0 & log(3) \approx 0.477 & 0 \\ (same, 10) & 0 & log(3) \approx 0.477 & 0 \\ (things, 11) & 0 & log(3) \approx 0.477 & 0 \\ (are, 12) & 0 & 0 & log(3) \approx 0.477 \\ (going, 13) & 0 & 0 & log(3) \approx 0.477 \\ (later, 14) & 0 & 0 & log(3) \approx 0.477 \end{bmatrix}$$

This approach fixes the issue with frequent words being weighted too heavily, but it does not account for word proximity. Word vectors can be used to account for word proximity.

3.1.3 Word Vectors

The word vectors are initialized by creating a one-hot-vector representation for each word in the corpus:

$$\begin{matrix} \text{W1} \\ \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} \text{W2} \\ \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} \text{W3} \\ \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \cdots, \begin{matrix} \text{Wn-2} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} \text{Wn-1} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} \text{Wn} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{bmatrix} \end{matrix} \in \mathbf{R}_{nx1}$$

Where n is the total number of unique words in the corpus. These vectors create an orthonormal basis to represent the words in the corpus.

Word vectors need to be trained using a neural network. There are two approaches for training the vectors: skip gram and continuous bag-of-words (CBOW). A window size of words to be considered is chosen. For example, for a window size of 3, the following words are considered.

Mary and I like to eat $\boxed{\text{the same things}}$

The center word is 'same'. The context words are 'the' and 'things'.

In CBOW, the neural network is trained to predict the center word given context words.

In skip-gram, the neural network is trained to predict context words from the center word.

In both cases, the purpose of the training is to obtain the hidden layer weight matrix so that after the network is trained, the rows of the weight layer give vector representations of each of the words in the corpus. Not only do the rows represent the words, but they are also positioned in n dimensional space s.t. words that appear in similar context appear close to each other. Since the two approaches are similar, only skip-gram is presented section 3.1.4 below.

3.1.4 Skip-gram:

In the skip-gram training method, each word has two representations: v_i (center word representation) and u_i (the context word representation).

Two weight matrices need to be trained: W (the weights for converting the one-hot-vector representation to a center word representation) and W' (the weights for converting to the context representation (logits)).

After the vector has been converted into the context representation, a softmax function is used to convert it into a vector of probabilities. The softmax function is:

$\frac{e^{x(i)}}{\sum_{i=1}^{n} e^{x(i)}}$ where $x(i)$ is the i^{th} element of the vector.

An example of the architecture for the skip-gram is given below:

$$\underset{\begin{bmatrix}0\\0\\0\\\vdots\\1\\0\\0\end{bmatrix}}{Input\ s} \rightarrow W \begin{bmatrix}0\\0\\0\\\vdots\\1\\0\\0\end{bmatrix} = \underset{\begin{bmatrix}-0.3\\0.5\\1.3\\\vdots\\0.2\\0.1\\0.2\end{bmatrix}}{Word\ Vector : v_c} \xrightarrow{W'^T v_c = u =} \begin{bmatrix}0.08\\0.1\\-0.1\\\vdots\\0.2\\0.3\\3\end{bmatrix} \xrightarrow{Softmax} \begin{bmatrix}0.01\\0.05\\0.05\\\vdots\\0.08\\0.1\\0.6\end{bmatrix}$$

The output vector gives the probability that each word is a context word. The network is trained by comparing the output of the network to the true context words for words in the training corpus. For a sufficiently large corpus, the output can accurately predict the context words.

The accuracy of the predictions is measured by the loss function. A cross-entropy loss function is used in this case. Cross-entropy is often used as a loss function for classification problems. For two vectors $y \in \mathbf{R}_{Vx1}$ and $y' \in \mathbf{R}_{Vx1}$, the cross-entropy function is: $h(y, y') = -\sum_{j=1}^{V} y_j log(y'_j)$

If y is a one-hot vector, this simplifies to: $h(y, y') = log(y'_j)$ because all of the entries are 0 except for the one entry that is equal to 1.

To understand why this is a good loss function, consider two cases:

1. When the prediction is perfect (the output of the softmax is equal to the one-hot-vector representaion). The function is minimized when the prediction is perfect.

$$h(1, y) = -log(y_j = 1) = 0$$

2. The function diverges to negative infinity when the prediction is completely incorrect.

$$\lim_{y_j \to 0} h(1, y) = \lim_{y_j \to 0} -log(y_j) = -\infty$$

Therefore, the cross-entropy loss function behaves the way we would expect the loss function to for this case.

The loss function that needs to be minimized is: $E = -log(p(w_1, w_2, \ldots, w_{2m}|w_c))$. This is the probability of the context words $w_1, w_2, \ldots, w_{2m}$ given center word w_c. This function maximizes the probability of these context words for this center word.

In terms of the output vector, this is:

$$E = -log \prod_{k=1}^{2m} \frac{exp(u_k(j^*)}{\sum_{j=1}^{V} exp(u_k(j')}$$

k is the k^{th} context word output vector and j^* gives the index for the true context word in the output vector.

By expanding using logarithm rules, this becomes:

$$E = -\sum_{k=1}^{2m} u_k(j^*) + \sum_{k=1}^{2m} log(\sum_{j=1}^{V} exp(u_k(j))$$

Let t_k be the target output (the one-hot-vector for the context word at the k^{th} position). The context weight matrix update is:

$$\frac{\delta E}{\delta w'(i,j)} = \sum_{k=1}^{2m} \sum_{j^*}^{V} \frac{\delta E}{\delta U_k(j^*)} \frac{\delta U_k(j^*))}{\delta W'(i,j)}$$

$$\frac{\delta E}{\delta u_k(j^*)} = \frac{exp(u_k(j^*)}{\sum_{j'}^{V} exp(u_k(j'))} - 1 = y_k(j^*) - t_k(j^*)$$

This is simply the difference between the true output and the target output. This gives the error in the prediction.

$$\frac{\delta u_k(j^*)}{\delta W'(i,j)} = v_c(i)$$

This is the i^{th} component of the center word's word vector.

This means the weight update is propositional to the error in the prediction of the i^{th} word vector.

The final value becomes:

$$\frac{\delta E}{\delta w'(i,j)} = \sum_{k=1}^{2m} \sum_{j^*}^{V} (y_k(j^*) - t_k(j^*)) v_c(i)$$

3.2 Machine Learning and Deep Learning Models:

After the text has been reduced to a feature representation, machine learning models can be trained to classify the text. The models considered in this research include K Nearest Neighbors, Multilayer Perceptron, Decision Trees, and Random Forests.

3.2.1 K-Nearest Neighbors:

K-Nearest Neighbors classifies input by looking for the most prevalent class for the k nearest values in the training set. For example, if the input text is represented using TFIDF, a distance measure like Euclidean distance or cosine similarity can be used to measure the distance between the end of two vectors. The Euclidean distance between two vectors is the length of the line segment connecting the end of the vectors. For high dimension spaces, Euclidean distance is less useful than Cosine similarity[25].

Cosine similarity is a distance measure for two vectors. Let $\mathbf{X}$ and $\mathbf{Y}$ be TFIDF vectors for two different sentences in the corpus. Their cosine similarity is:

$$cos(\theta) = \frac{\mathbf{X} \cdot \mathbf{Y}}{||\mathbf{X}||||\mathbf{Y}||}$$

When the vectors are pointing in the same direction, $\theta = 1$. When the vectors are pointing in completely opposite directions, $\theta = -1$. When the vectors are orthogonal, $\theta = 0$. K-nearest neighbors suffers from the curse of dimensionality. This means that when there are a larger number of features, more data is required to complete training.

3.2.2 Multilayer Perceptron Network:

A Multilayer Perceptron Network is a neural network with a single hidden layer. The structure of an MLP is shown in the figure 1.

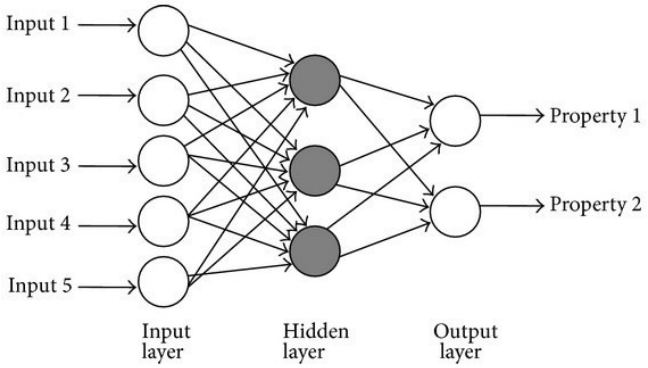

Figure 1: The structure of an MLP Network [26]

The grey circles are the preceptron. Each perceptron contains an activation function. Examples of activation functions include hyperbolic tangent (shown in figure 2), the logistic function (shown in figure 3), and the rectifier function (shown in figure 4). The activation function for the neurons in the output layer is the Softmax function 5, which was introduced above. Graphs of each of these activation functions are shown below:

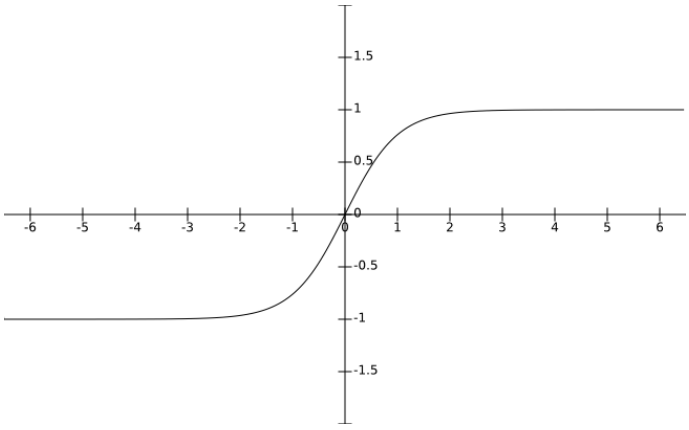

Figure 2: Hyperbolic Tangent $y(x_i) = tanh(x_i)$

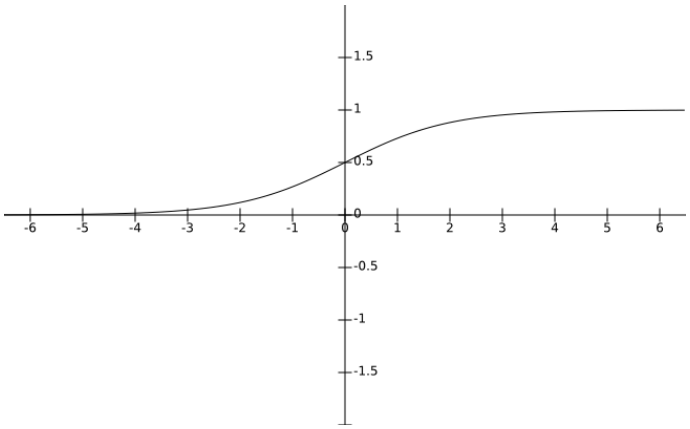

Figure 3: Logistic Function $y(x_i) = (1 + e^{-x_i})^{-1}$

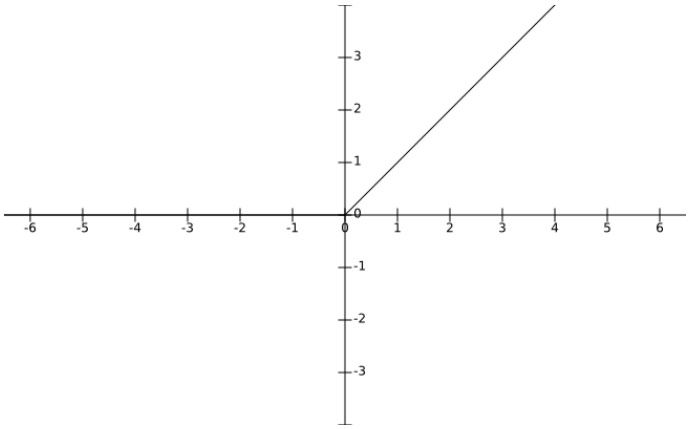

Figure 4: Rectifier Function $y(x_i) = max(0, x_i)$

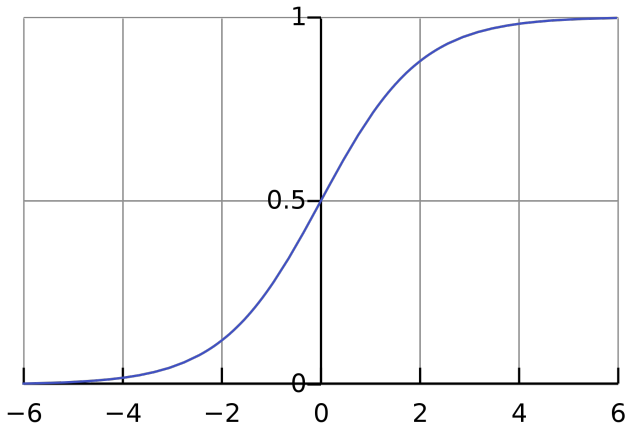

Figure 5: Softmax Function $y(x_i) = \frac{e^{x(i)}}{\sum_{i=1}^{n} e^{x(i)}}$

The input for each of these functions is a weighted sum of inputs from the previous layer of the network. For example, in the figure above, each perceptron is connected to each perceptron in the next layer, so each input is going into each perceptron.

Let n be the number of nodes in the input layer. Let m be the number of nodes in the hidden layer. Let x_i be the i^{th} input from the input layer. Note that i ranges from 1 to n. Let $wh_{i,j}$ be the i^{th} input weight for the j^{th} perceptron in the hidden layer. Note that j ranges from 1 to m. Each node has n weights (one for each input node), so there are there are $n \cdot m$ weights in the hidden layer.

The input for the j^{th} perceptron in the hidden layer can thus be written as: $vh_j = \sum_{i=1}^{n} x_i \cdot wh_{i,j}$. vh_j is then input into the activation function. If the activation function is $\sigma(x)$, its output can be written as: $\sigma(v_j) = h_j$. This output is then used as the input for each of the output layer neurons. Assume there are l output neurons. Let $wo_{j,k}$ be the j^{th} input weight for the k^{th} perceptron in the output layer. Note that k ranges from 1 to l. Each node has m weights (one for each hidden layer node), so there are $m \cdot l$ weights in the output layer.

The weighted sum that is used for the input to the k^{th} output layer neuron is: $vo_k = \sum_{j=1}^{m} h_j wo_{j,k}$. Each of these weighted sums is input into the Softmax function to produce the output for each neuron. For Softmax function $\phi(x)$, the output is $\hat{y}(k) = \phi(vo_k)$

The output layer structure depends on the type labels the dataset has. For a multi-class classification problem, the output layer has a perceptron for each possible class. Each perceptron in the output layer uses a Softmax function and

the activation function. The perceptron with the largest value is the selected class.

Back-propogation

The weights for the MLP network need to be trained for the model to provide useful predictions. The weight training process is called back-propogation. Back-propogation updates the weights based on how close the outputed prediction from the neural network is to the expected value. This is measured by a loss-function. Back-propogation works by applying an optimization algorithm like Gradient Descent to the partial derivatives of the loss-function with respect to each weight. This results in weight updates that decrease the loss-function value for the output. A common loss-function for an MLP network is is cross-entropy. Cross-entropy was introduced in section 3.1.4.

Using the notation from above, we can write: Let $y(k)$ be the label for the k^{th} category. This will be 0 if the data does not belong to the k^{th} category and 1 if the data does belong to the k^{th} category. Let $\hat{y}(k)$ be the predicted value for the k^{th} label. This will be the output of the Softmax function indicating the probability of the input getting label k. The categories correspond to the output nodes, so if there are l output nodes, the output is being classified into l categories.

For a multi-class classification problem, the cross-entropy loss function gives a measure of the similarity between the true value and the predicted value for each class. It is given by:

$$\mathbf{L} = -\frac{1}{l}\sum_{i=1}^{l}[y(k)log(\hat{y}(k)) + (1 - y(k))log(1 - \hat{y}(k))]$$

Now the weight update formula can be created by finding the derivative of the loss function with respect to each of the weights. The weights in the hidden layer have a different update formula then the weights in the output layer.

Output Layer Weight Update:

Suppose the r^{th} weight for the k^{th} perceptron in the output layer needs to be updated. This is $w0_{r,k}$ For the loss function $\mathbf{L}$, the update formula is given below:

$$\frac{\delta \mathbf{L}}{\delta wo_{r,k}} = \frac{\delta \mathbf{L}}{\delta \hat{y}(k)}\frac{\delta \hat{y}(k)}{\delta wo_{r,k}}$$

The first term can be further broken down:

$$\frac{\delta \mathbf{L}}{\delta \hat{y}(k)} = \frac{\delta \mathbf{L}}{\delta \phi(vo_k)}\frac{\delta \phi(vo_k)}{\delta \hat{y}(k)}$$

The derivative of the first term is:

$$\frac{\delta \mathbf{L}}{\delta \phi(vo_k)} = y(k)\frac{1}{\hat{y}(k)} + (1 - y(k))\frac{-1}{1 - \hat{y}(k)}$$

The derivative of the second term is:

$$\frac{\delta \phi(vo_k)}{\delta \hat{y}(k)} = \frac{exp(vo_k)}{\sum_{k=1}^{l} exp(vo_k)} - (\frac{exp(vo_k)}{\sum_{k=1}^{l} exp(vo_k)})^2 = y(\hat{k})(1 - \hat{y}(k))$$

Therefore, the product of the two terms is:

$$\frac{\delta \mathbf{L}}{\delta \hat{y}(k)} = \frac{\delta L}{\delta \phi(vo_k)}\frac{\delta \phi(vo_k)}{\delta \hat{y}(k)} = y(k) - y(\hat{k})$$

The second term in equation (x) is:

$$\frac{\delta \hat{y}(k)}{\delta wo_{r,k}} = \phi(\sum_{j=1}^{m} wo_{j,k}h_j)'h_r = \phi(vo_k)(1 - \phi(vo_k))h_r$$

So the overall update formula is:

$$\frac{\delta \mathbf{L}}{\delta wo_{r,k}} = (y(\hat{k})(1 - \hat{y}(k)))\phi(vo_k)(1 - \phi(vo_k))h_r$$

Hidden Layer Weight Update:

Suppose the r^{th} weight for the j^{th} perceptron in the hidden layer needs to be updated. This is $w0_{r,j}$ The update formula is given below:

$$\frac{\delta \mathbf{L}}{\delta wh_{r,j}} = \sum_{k=1}^{l} \frac{\delta L}{\delta vo_k}\frac{\delta vo_k}{\delta h_r}\frac{\delta h_r}{\delta wh_{r,j}}$$

The first term was found above. The other terms are shown below:

$$\frac{\delta vo_k}{h_j} = wo_{r,k}$$

$$\frac{\delta h_r}{\delta wh_{r,j}} = \sigma(\sum_{i=1}^{n} x(i)wh_{i,j})'x(r)$$

$$\frac{\delta \mathbf{L}}{\delta wh_{r,j}} = \sum_{k=1}^{l}(y(k) - \hat{y}(k)wo_{r,k}wh_{i,j})'x(r)$$

These weight update formulas can now be used in Gradient Descent or another optimization algorithm to find the optimal weight update during each training step.

3.2.3 Decision Trees

Decision Trees is a rule based classifier that classifies data by splitting it into exclusive sets based on features of the data using a tree-like structure. At each node in the tree, the data is split into two sets. At the terminal nodes, the data is classified into one of the possible classes.

The two main methods for measuring the quality of the data split are Gini impurity and entropy. Entropy was introduced in the section on building word vectors using skip-gram, so Gini impurity will be covered here. Gini impurity gives the probability of incorrectly classifying an item in the dataset if the classes are selected based on the distribution of labels in the dataset.

For decision trees, the Gini impurity value can be calculated for each split of the data using the following formula:

$$\sum_{i=1}^{C} p(i)(1 - p(i))$$

where C is the number of classes in the dataset and $p(i)$ is the proportion of the dataset in class i. Note that when the Gini impurity is being calculated for one of the splits, only the data inside that split is considered. The Gini impurity for the full dataset can be calculated using the distribution of each label in the dataset. Let the overall Gini impurity be $Gini_o$. Next, the Gini impurity can be calculated for each branch of the tree. Let the impurity for branch i be $Gini_i$. Now, using a weight based on the proportion of the data points in each split, the Gini gain can be calculated. Let the i^{th} weight be w_i, the Gini gain is then:

$$Gini\ Gain = Geni_o - \sum_{i=1}^{C} w_i \cdot Gini_i$$

3.2.4 Random Forests:

The main issue with decision trees is its tendency to overfit to data. This means that they tend to have low bias, but high variance; they fit the training data very well, but their predictions to not generalize to the test set. Random Forest is a machine learning algorithm based on an expansion of decision trees that uses aggregated bootstrapping to yield better classification results. Aggregated boostrapping means that a series of decision trees are trained on randomly selected subsets of the data. The median of the results for the individual trees is used as the final classification result.

3.2.5 Convolutional Neural Networks:

A Convolutional Neural Network (CNN) is another type of neural network. They were originally applied to image data, but they can be applied to any matrix based input. A matrix representation of a piece of text can be created by replacing each word with its word vector representation and concatenating all the vectors. CNNs uses two main techniques to reduce the dimension of the data: convolution layers and max pooling layers.

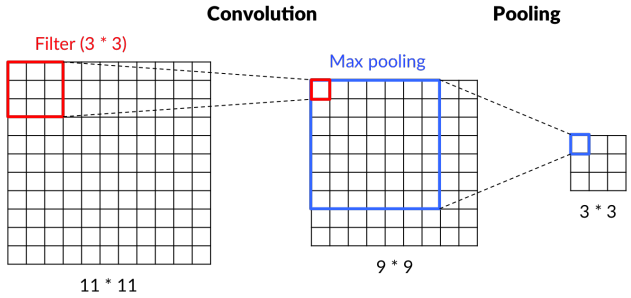

Figure 6: CNN Operations [27]

In a convolution layer, the sum of the element-wise product of each element in the matrix with a weight matrix is calculated. For example, given the following input and weight layer, the convolution is calculated as follows:

$$input = \begin{bmatrix} -3 & 0 & 1 \\ 2 & 5 & 3 \\ 2 & 4 & 1 \end{bmatrix}$$

$$weights = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$convolution = (-3 \cdot 1) + (0 \cdot 0) + (1 \cdot 1) + (2 \cdot 1) + (5 \cdot 0) + (3 \cdot 1) + (2 \cdot 1) + (4 \cdot 0) + (1 \cdot 1) = 6$$

The max pooling layer takes the max value in a subsection of an array. For example, for the following 4X4 input, the 2X2 max pooling layer output is:

$$input = \begin{bmatrix} 4 & 8 & 9 & 6 \\ 5 & 4 & 18 & 4 \\ 7 & 9 & 10 & 8 \\ 7 & 3 & 13 & 5 \end{bmatrix}$$

$$maxpooling(input) = \begin{bmatrix} 8 & 18 \\ 9 & 13 \end{bmatrix}$$

After using a series of convolution layers and pooling layers, the data can be used as input for a multi-layer perceptron newtork to compute the final output. All weights for the multi-layer perceptron network and convolution layers can be trained using back-propagation. Because CNN uses input that has a spacial component, it can potentially pick up features that the previous models are not able to.

3.2.6 Recurrent CNN:

Convolutional Neural Networks and Convoltional Neural Networks are examples of feed forward neural networks because data only moves forward during the calculation. In contrast, Recurrent Neural Networks (RNN) contain perceptrons with the ability to copy and preserve data from previous input. This makes it possible them to detect temporal patterns in data. An RCNN model contains CNN and RNN layers.

3.3 Modeling Metrics:

A modeling metric gives a numerical score for evaluating each model's performance. Performance is evaluated using a test set that contains data with known labels which was not used to train the model. The most straightforward modeling metric is accuracy, i.e. the percent of correct predictions. The main disadvantage of using accuracy is that the model's accuracy does not measure its ability to distinguish between classes of data. This is a special problem when the data is very skewed. As an example, imagine that in a dataset with binary labels, 90% of the data has a true label and 10% of the data has a false label. A model could attain 90% accuracy by always predicting true, but the model could not be said to have the ability to distinguish between the two classes.

The F1 score is a modeling metric that addresses the issue with accuracy. The F1 score is the harmonic mean of the models precision and recall. Precision is the models rate of correctly classifying positive responses. This is:

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

Recall is the model's rate of correctly classifying relevant responses. This is:

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

$$f1 = (\frac{recall^{-1} + precision^{-1}}{2})^{-1}$$

Precision and recall can individual be minimized by minimizing false positives and false negatives respectively. If the arithmetic mean of the two values is used to evaluate the classification, the chosen classifier could be biased toward minimizing precision or recall individually. Classifiers that minimize either false

negatives or false positives by guessing that almost all the elements belong to one of the classes may have high accuracy on skewed datasets, but will not have the ability to distinguish between classes. Taking the harmonic mean of the two metrics ensures that the classifier will not be biased toward false positives or false negatives.

In this classification problem, false positives are assigned when an error in the users response is not detected. This detracts from the educational value of the simulation. False negatives are assigned if a correct response is deemed wrong and corrected. This detracts from the user experience because being corrected after giving a correct answer can cause frustration. There is an interest in minimizing both false negatives and false positives, so the F1 score is an appropriate metric for evaluating the dialogue classifiers.

The F1 score is only defined for binary classification tasks. In multi-label classification problems, the precision and recall can be computed and averaged in different ways to create an overall F1 score. For example, let C be the total number of classes. The overall number of true positives, false positives, and false negatives can be individually summed and used to compute a global value for the precision and recall as follows:

$$Global\ Precision = \frac{\sum_{i=1}^{C} True\ Positive(i)}{\sum_{i=1}^{C} True\ Positive(i) + \sum_{c} False\ Positive(i)}$$

$$Global\ Recall = \frac{\sum_{i=1}^{C} True\ Positive(i)}{\sum_{i=1}^{C} True\ Positive(i) + \sum_{i=1}^{C} False\ Negative(i)}$$

A F1 score termed the micro F1 score can then be computed from these values.

Another method for computing a multi-class analogue to the F1 score is using simple average of the precision and recall for each class:

$$Mean\ Precision = \frac{\sum_{i=1}^{C} \frac{TP(i)}{TP(i)+FP(i)}}{C}$$

$$Mean\ Recall = \frac{\sum_{i=1}^{C} \frac{TP(i)}{TP(i)+FN(i)}}{C}$$

These mean values can then be used to compute the F1 score, termed the macro F1 score.

The macro F1 score can be further modified to account for class imbalance by weighting i^{th} term in the average precision and recall calculations by the number of samples that have the i^{th} label. This means that classes with more

data will be weighted higher in the calculation.

The weighted F1 score is used as the main modeling metric for the data analysis portion of this research because there are large class imbalances in several of the datasets.

4 Methodology

4.1 Dataset Creation:

The goal of the implementation of the framework is to create a series of machine learning classification models that are able to classify input for each of the questions from the simulation based on the same features that distinguish the multiple-choice questions. These classification models need to be trained using a dataset of potential responses to a free-input version of each multiple-choice question. Therefore, the first step in the creation of the free-input dialogue system was the creation of a dataset comprising possible responses for each question from the simulation. The dataset collection process evolved over the course of the research. This section will cover the evolution of the data collection approach. It will also cover an algorithm that was created for generating data.

4.1.1 Initial Approach: Data Collection for Multiple-Choice Classification

The initial data collection approach focused on collecting re-phrasings of the multiple-choice answers for each question from the simulation. This data collection approach assumed that re-phrasing data could be used to create models with the ability to map all possible user input into the original multiple-choice categories. Note that each multiple-choice option was associated with label. This label indicates the "appropriateness" of each response (see section 1.2.1). Each rephrasing of a multiple-choice response is associated with that multiple-choice response's label.

Data collection was performed using the crowdsourcing service Amazon Mechanical Turk. For the initial round of dataset collection, the prompts asked users to provide re-phrasings of the original multiple-choice responses. Two types of prompts were created for collecting data: rewrite prompts and feedback prompts. Rewrite prompts asked users to reword multiple-choice answers from the original dialogue tree. Feedback prompts asked users to give answers that match the feedback of the original multiple choice answers. Examples of these prompt types are shown in figures 7 and 8.

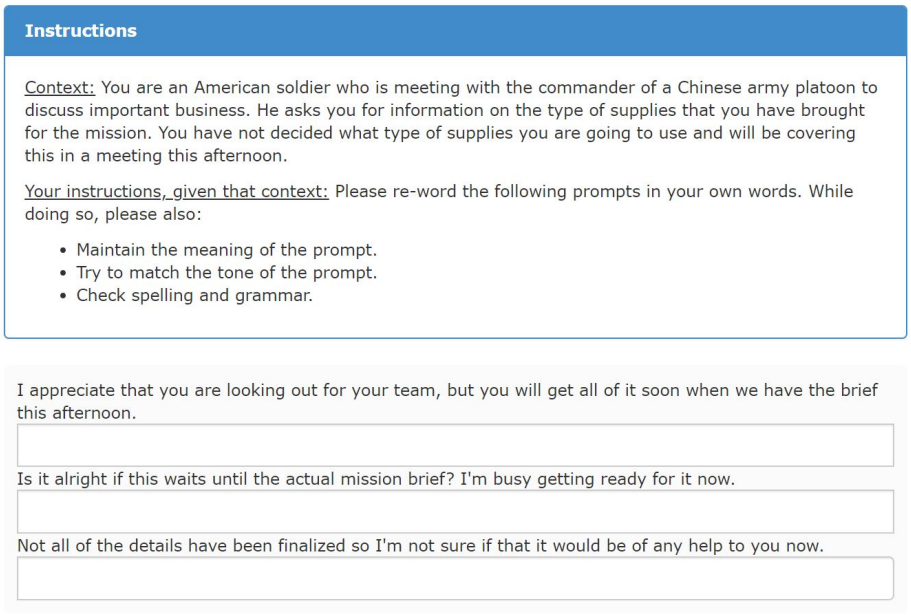

Figure 7: Rewrite prompt example

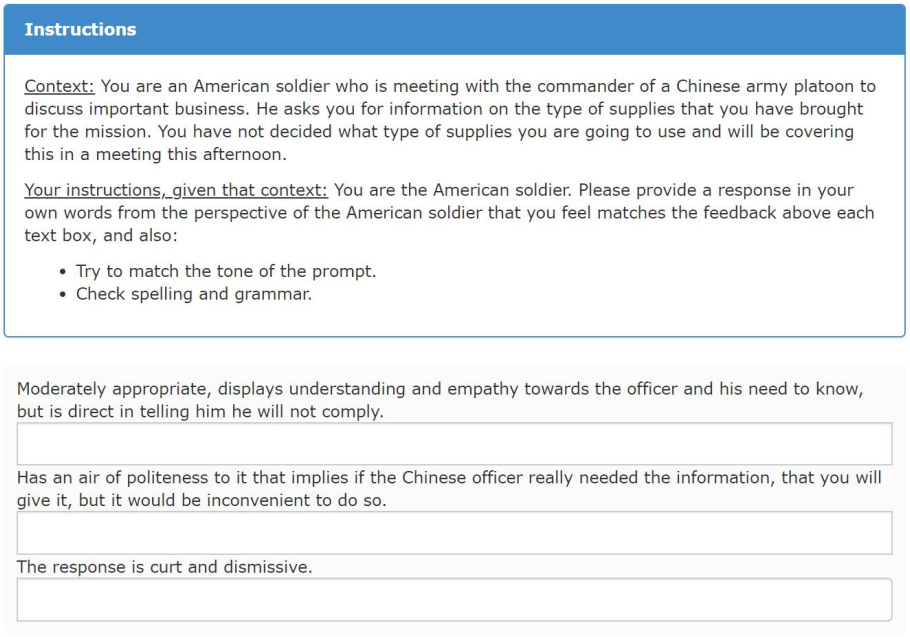

Figure 8: Feedback prompt example

The main issue with this labeling approach is that the labels are not able to capture the range of possible input that could be received by a free-input dialogue system. To illustrate the issue that was encountered when attempting to classify responses into the single label system, consider the responses in table 1 (section 1.2.1). Imagine a response combining parts of the first and third response: "I appreciate that you are looking out for your team, but not all of the details have been finalized, so I'm not sure if that it would be of any help to you now." It is not clear which category this response should go into because it has traits that are considered positive (similar to response 1) and negative (similar to response 3) based on the given feedback.

4.1.2 Final Approach: Context-based Data Crowdsourcing, Binary Category Labels, and Data Generation

Because of the limitations discussed above, the original multiple choice options were ignored in favor of a new approach. The test set for the model should be a set of input created by collecting responses to the free-input versions of the questions from the simulation because this is the dataset that the models are being trained to classify. Therefore, data crowdsourcing was performed using prompts containing only the context and intent information a simulation user would receive while using the game. An example of one of these prompts is

shown in figure 9:

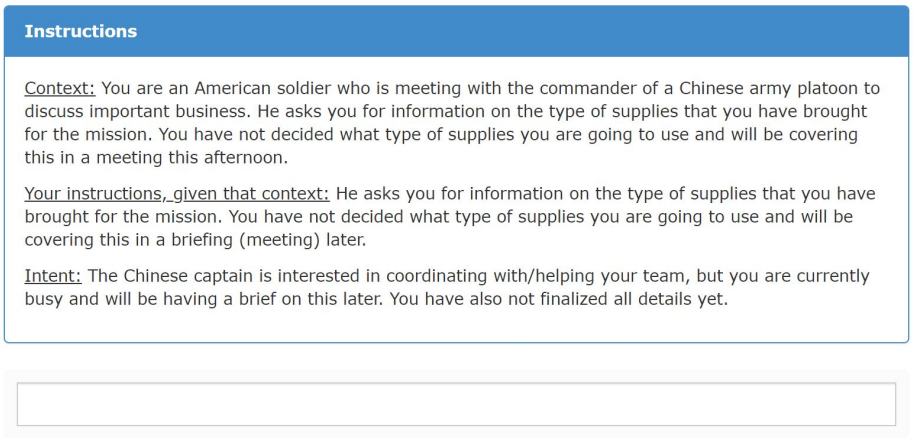

Figure 9: Context prompt example

The responses to these prompts represent real answers to the questions from the simulation, so this data can be used for evaluating the classification models.

A new labeling approach was also used. First, each question from the simulation was considered to be a separate classification problem. This means that each prompt needs its own classification model trained only on data collected specifically for that prompt. Next, each prompt was given its own unique labels. Rather than using labels based on the original multiple-choice categories, the new labels for each prompt were created by breaking the text features mentioned in the original feedback into sub-categories. For example, the feedback for the responses in table indicate that a response should receive a higher "appropriateness" label if it "displays understanding" and "implies that you will give the information". Similarly, a response should receive a lower "appropriateness" label if it "is curt and dismissive". We can define "displays understanding", "is curt and dismissive", etc. as binary sub-categories that can be present or absent in the text. Although these are not truly binary categories (e.g., text could "display understanding" to different degrees), it is reasonable to approximate labels for these categories by using binary labels to represent the presence or absence of text that matches the category description. Table 2 gives an example of sub-categories for question 5. A list of the sub-categories for each question is given in appendix B.

The final label for each question is created by combining the labels for each of the sub-categories. For example, the first piece of text would have a label 101, and the second piece of text would have a label 100. Given that there are n total binary sub-categories for a question, there are 2^n possible labels.

Note that this type of labeling scheme accounts for the two main issues that

Table 2: Sub-categories and Binary Labels for Question 5

Example Responses	Gives time when information will be given	Displays understanding and empathy	Explains that you are not ready	Full Label
Is it alright if this waits until the actual mission brief? I'm busy getting ready for it now.	1	0	1	101
I appreciate that you are looking out for your team, but you will get all of it soon when we have the brief this afternoon.	1	1	0	110

plagued the single label labeling scheme: it easy to generate context specific labels and the labels span a wide array of responses.

Using this type of binary labeling system has several advantages. Each label in the original labeling system had to be explicitly defined. With binary labels, only the individual categories have to be defined. This means that a binary labeling system yields a larger number of possible labels from a smaller number of defined categories. If a new text feature needs to be found, the binary labeling system is also easier to update because the binary labels for the preexisting categories would not change. The binary labeling system is also based on individual text features; each label can be considered to represent the presence or absence of certain text features in the response. This makes the binary labels more definitive and aids in the creation of a data generation algorithm.

4.1.3 Data Generation:

Collecting a sufficient dataset for training classifiers using only crowdsourcing is difficult because crowdsourced responses need to be labeled and are expensive to collect. On the other hand, use of labels for sub-categories based on the presence or absence of text makes it easy to write a list of text snippets that would receive a binary label of 1 or 0 when labeled using the labeling system presented in section 4.1.2 for each sub-category. Full pieces of input data can be created by making combinations of these text snippets. Each piece of output text created by this data generation algorithm is labeled, so it can be used as training data for the classification models.

Tables 3 and 4 give examples of text snippets for the sub-categories for question 4.

Table 3: Example text snippets for set $definite(1)$

Gives time when information will be given	**Displays understanding and empathy**	**Explains that you are not ready**
We will have a brief this afternoon	I know that this is critical to the mission	Not all the details have been finalized
The supplies will be discussed in the meeting	I understand that you need this information	I need time to confirm what we are bringing
Later today we will have all answers		I am accessing the need for supplies and manpower

Table 4: Example text snippets for set $definite(0)$

Gives time when information will be given	**Displays understanding and empathy**	**Explains that you are not ready**
We won't be able to provide that information	I don't understand why you need to know that	(this category is purposefully left blank)
You will not get that information	That is none of your concern	
No, we can't supply this information to you	It is my decision who gets the information and who does not	

The text snippets in each table can be thought of as a set. Let $t_j(i)$ be the i^{th} snippet of example text in sub-category j. Let $text$ be a piece of text created by choosing one or fewer text snippets from each sub-category and combining them. Text snippets in Table 3 are members of the set $definite(1)$. The presence of text snippet from category j in $text$ will result a binary label of 1 for the text snippet's corresponding sub-category. Similarly, the presence of a text snippet on Table 4 will result a binary label of 0 for the text snippet's corresponding sub-category. These sets are called "definite" because the presence of text snippets from them defines the label the text will receive for each sub-category. Figure 10 shows the definition in set notation.

$$t_j(i) \in definite(k) \text{ for } k = 0, 1 \text{ if } t_j(i) \in text$$
always implies $text$ will receive binary label k for category j.

Figure 10: Set notation for sets $definite(1)$ and $definite(0)$

Two additional sets were defined: $isolated(1)$ and $isolated(0)$. These additional sets allow the algorithm to deal with cases when the meaning of the

output text changes depending on how much information is given. Table 5 shows example text snippets for $isolated(0)$.

Table 5: Example text snippets for set $isolated(0)$

Gives time when information will be given	**Displays understanding and empathy**	**Explains that you are not ready**
I will get back to you as soon as possible	(this category is purposefully left blank)	(this category is purposefully left blank)
I will give you an update quickly as I can		

If the only text snippet in *text* from the first sub-category is "I will get back to you as soon as possible", it would receive a label of 0 for the first sub-category. If another text snippet from $definite(1)$ is also in text (e.g., "I will get back to you as soon as possible, but the supplies will be discussed in the meeting"), the label for the first sub-category would become 1 since the text now indicates a specific time that the information will be given. For this question, there are no responses in $isolated(1)$. The set notation definition for $isolated(1)$ and $isolated(0)$ is given in figure 11.

$$t_j(i) \in isolated(k_1) \text{ for } k_1 = 0, 1 \text{ if } t_j(i) + t_j(h) \in definite(k_2),$$
$$\text{where } k_2 = 0, 1 \text{ for any } t_j(h) \in definite(k).$$

Figure 11: Set notation for sets $isolated(1)$ and $isolated(0)$

Note that text snippets from the same sub-category on isolated tables are never combined.

The text snippets on these tables can be combined using a recursive data generation algorithm to create labeled output data that can be used to train a text classification model. Consider tables 7 and 7:

Table 6: Example table $definite(0)$

Sub-category 1	Sub-category 2
a	d
b	e
c	

Table 7: Example table $definite(1)$

Sub-category 1	Sub-category 2
f	h
g	i
	j

The algorithm takes a column number and a blank string as input. It starts by appending each element in the first column to a list and recursively calling

the function on the next row. It does this with a separate loop for each table. This process can be illustrated by the tree like structure shown in figure 12:

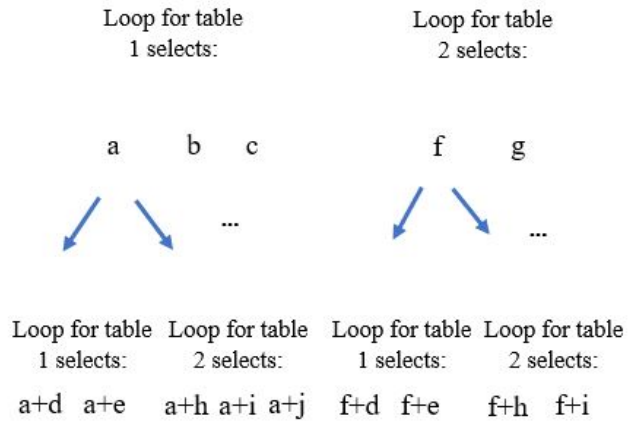

Figure 12: Data generation tree

The recursive function outputs when the number of concatenated text snippets equals the number of categories. Categories can be left blank by inserting blank rows into the table.

The amount of output created by the data generation algorithm is heavily dependent on the number of binary categories for the labels and the number of text snippets per binary category. For example, consider the case when the data generation is being used to combine samples from the sets $definite(0)$ and $definite(1)$. If there are f_i text snippets in category j for set $definite(0)$ and r_j pieces of data in category j for set $definite(1)$, $\prod_{j=1}^{n} f_j + r_j$ pieces of data will be generated by $\sum_{j=1}^{n} f_j + r_j$ pieces of input text. The output of the algorithm (measured by the number of pieces of data generated) is $\mathcal{O}(\sum_{j=1}^{n} f_j + r_j)^2$ for $n = 3$ and $\mathcal{O}(\sum_{j=1}^{n} f_j + r_j)^3$ for $n = 3$

4.2 Training Text Classifiers:

4.2.1 Splitting Data:

The test dataset for each classification model consists of only responses collected using context prompts. These prompts contain the same information that would be given to users in a free-input version of the simulation. Therefore, the data from these prompts is representative of the type of responses that users of the free-input dialogue system would produce.

Resampling is necessary because the rephrasing and feedback prompts are based on the original multiple-choice questions, but the labeling system in-

troduces additional labels for responses that are not equivalent to any of the multiple-choice options. This means that the data from the rephrasing and feedback prompts can be skewed toward certain labels that are not very prominent the in the context dataset.

For example, figures 13, 14, and 15 show the distribution of data collected using context, rewrite, and feedback prompts for dataset 2.

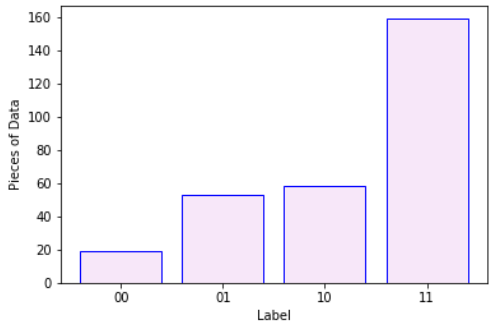

Figure 13: Label distribution for context data for classifier 2

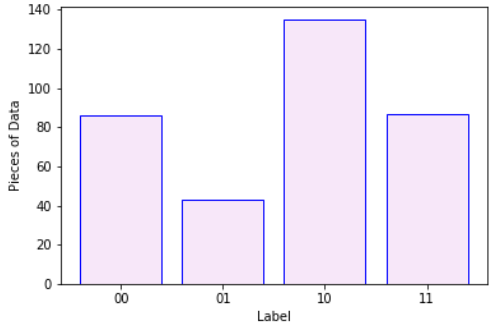

Figure 14: Label distribution for rewrite data for classifier 2

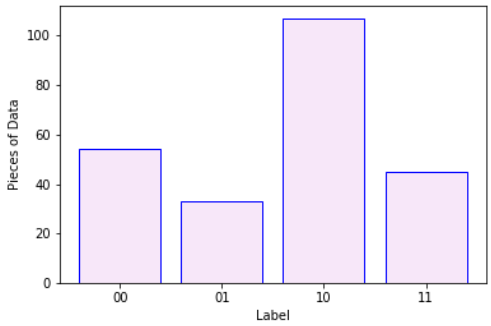

Figure 15: Label distribution for feedback data for classifier 2

The rewrite and feedback datasets have very similar distributions, and the most prominent label in the context dataset is underrepresented in both the rewrite and feedback datasets.

The data collected using context prompts was randomly split to make the training and testing sets. The training set was created by resampling the rewrite, feeedback, and generated data. The goal of the resampling process was to create a training set with a similar label distribution to the label distribution of the context data. The resampling process works as follows:

1. The ratio of each label in the context set is found and saved. For example: $ratio = \{"00" : 0.065, "01" : 0.18, "10" : 0.2, "11" : 0.55\}$.

2. For each label, data with the same label from the feedback, rewrite, and generated data is resampled. The proportion of feedback, rewrite, and generated data in the training set is controlled by a parameter $mult_{type}$ where $type \in [rewrite, feedback, generated]$. The number of samples selected from each type of data is $selected\ samples = ceil(ratio[i] \cdot mult_{type})$ where $ceil$ is the ceiling function. Let $support_{type}(i)$ be the number of pieces of data with label i for type $type$. If there are no samples for a label in the context data, the ratio will be 0. In this case, the number of samples selected is

$$selected\ samples = \sum_{i=1}^{n} support_{context}(i) \cdot \frac{min(ratio)}{3}$$

where $sum(support_{context})$ is the total number of samples in the context data and n is the total number of labels. Even if there are no samples in the context set with a specific label, the classifier should still be trained on that label.

3. If $support_{type}(i) < ceil(ratio[i] \cdot mult_{type})$ then data is resampled with replacement. If $support_{type}(i) >= ceil(ratio[i] \cdot mult_{type})$, data is resampled with-

out replacement. After the final dataset has been created using this resampling process for each type of data, the resampled data is combined together.

After a training dataset was assembled using the techniques covered in the above section, a feature representation and modeling pipeline was created to test various modeling approaches. Feature representations are the numerical inputs used to represent data features (see section 3.1). Models in this context are machine learning or deep learning models trained using the numerical feature representations to classify input text (see section 3.2). Model performance is evaluated by using an evaluation metric (see section 3.3)

5 Implementation Results:

5.1 Crowdsourced Data:

Table 8 gives the total number of samples collected using each of the data crowdsourcing prompt types.

Table 8: Crowdsourced data by prompt type

	Context	Rewrite	Feedback	Total
# of Samples	1,449	1,359	1,314	4,122

Figure 16 shows the number of pieces of text data collected for each dataset using each data crowdsourcing prompt type.

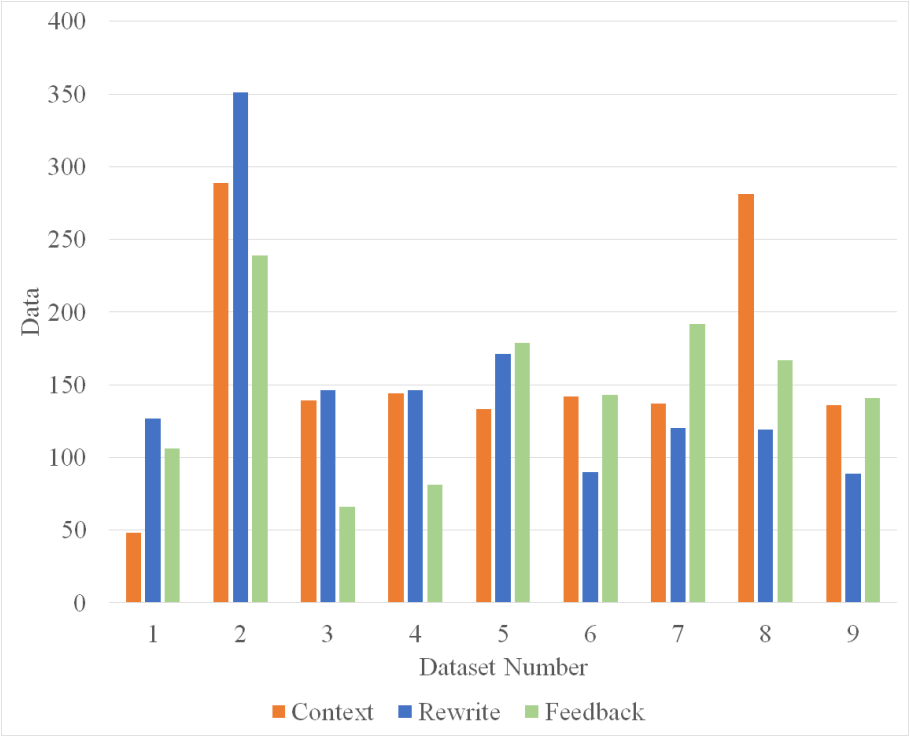

Figure 16: Crowdsourced data by data type

Classifier 2 and classifier 8 are used for two questions from the multiple-choice game, so additional data was collected for both. For a map from multiple-choice question to dataset, see tables 33, 34, and 35 in appendix B.

The lexical diversity and average response length for each type of text data can be used to measure the utility of each data type. However, as figure 16 indicates, the amount of data collected using each prompt type is imbalanced. Types to tokens ratio (TTR) is the standard measure of lexical diversity. TTR measures the ratio to unique words to total words in a corpus. It is biased toward higher values for smaller corpuses, so Measure of Textual Lexical Diversity (MTLD) was used to measure lexical diversity. MTLD differs from TTR because it calculates the mean length of sequential word strings in a text that maintain a given TTR value [28]. It has been found that MTLD is less affected by text length than other measures of lexical diversity [28].

In figure 17, the horizontal and vertical bars give the mean MTLD and average length over all the questions in the data set. Context prompts yield longer answers with higher MTLD values. For some questions (e.g. 3 and 4), there are large differences in average response and MTLD value. Each of these questions

asks the user to provide an explanation, and the context prompts encouraged users to provide more verbose explanations in an attempt to be more polite. The values for other questions, such as questions 1 and 12, are relatively similar regardless of the prompt type. Both of these questions are greetings, so the clustering could be due to the small feature space that these questions have.

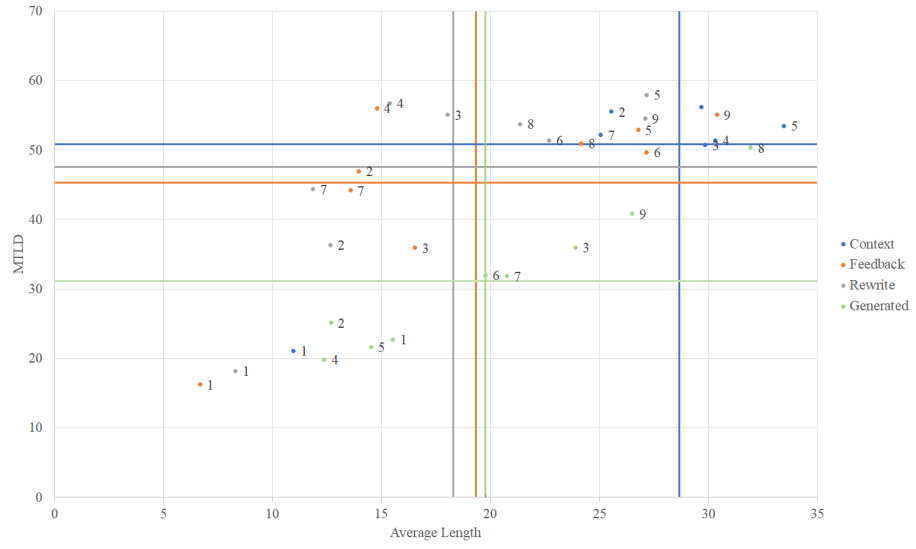

Figure 17: Average Length and lexical diversity (MTLD) by crowdsourcing prompt type

Figure 17 indicates that data collected using context prompts has the highest mean average response length and MTLD values. The average MTLD value for rewrite and feedback data are almost exactly the same, but data collected using feedback prompts has a slightly higher average response length than data collected using rewrite prompts. Data with higher MTLD and higher average response length values can capture more of the variance in potential input to the classification models, so crowdsourcing prompt types that yield data with higher values for these metrics are preferred.

5.2 Modeling Results:

During the modeling process, K-Nearest Neighbors (KNN), Random Forests (RF), Multi-Layer Perceptron (MLP), Convoluational Neural Network, and RCNN models were trained using TFIDF and Glove Word2Vec average, and Glove word embedding feature representations as input. The parameters on these models were also varied. For KNN, the number of neighbors varied between 1 and 10. For RF, the number of estimators (i.e., the number of trees used to generated the prediction) was varied between 2 and 46. The max depth of

the individual trees was also varied between 2 and 46. For MLP, the max number of training iterations was varied between 500 and 1500, and various hidden layer dimensions were tested. For CNN, the epochs were varied between 3 and 7.

Two training sets were used: the first contained generated data and crowdsourced data and the second contained only crowdsourced data. In both datasets, the data was split using the method introduced in section 4.2.1. These two training datasets were created to test the usefulness of the generated data for training models

The best results for each classification model using the training dataset with generated data are shown in tables 9 through 13.

Table 9: KNN Results

Classifier	Dimensions	Neighbors	F1	Feature
1	200	4	0.7079	tfidf
2	100	1	0.6096	tfidf_lem
4	200	10	0.5498	w2v
5	200	1	0.7529	w2v
6	100	7	0.7531	tfidf
9	200	4	0.6310	w2v
10	300	5	0.7826	tfidf_lem
13	300	1	0.6530	tfidf_lem
14	300	2	0.5086	tfidf_lem

Table 10: Random Forests Results

Classifier	Dimensions	Number of Estimators	Max Depth	F1	Feature
1	100	26	6	0.8018	tfidf
2	300	42	38	0.7536	tfidf
3	300	14	38	0.5765	w2v
4	300	30	18	0.8333	tfidf
5	200	6	42	0.7848	tfidf
6	200	22	42	0.7450	w2v
7	100	34	22	0.8383	tfidf
8	300	14	46	0.7154	tfidf
9	100	10	22	0.5722	tfidf

Table 11: Multi-Layer Perceptron Results

Classifier	Dimensions	Hidden Layer Sizes	Max Iterations	F1	Feature
1	300	(50, 50)	1500	0.7986	tfidf
2	300	(100, 100)	1500	0.7616	tfidf
3	300	(50, 50)	1500	0.6680	w2v
4	200	(50, 50)	1000	0.7903	w2v
5	100	(100, 100)	500	0.7621	tfidf
6	300	(100, 100)	1500	0.7493	w2v
7	200	100	500	0.7955	w2v
8	300	50	1500	0.7235	tfidf
9	100	50	1000	0.6196	tfidf

Table 12: CNN Results

Classifier	Dimensions	epochs	input_d	F1
1	300.0	5	300	0.7837
2	200.0	5	300	0.8183
3	200.0	7	300	0.6486
4	300.0	5	300	0.8082
5	200.0	3	300	0.8004
6	200.0	5	300	0.7452
7	300.0	7	300	0.8479
8	100.0	7	300	0.7997
9	200.0	7	300	0.6256

Table 13: RCNN Results

Question	Dimensions	epochs	input_d	F1
1	100.0	3	300	0.7437
2	100.0	3	300	0.8007
3	100.0	3	300	0.6009
4	200.0	3	300	0.7385
5	100.0	3	300	0.7783
6	200.0	3	300	0.7737
7	200.0	3	300	0.7634
8	200.0	3	300	0.8215
9	200.0	3	300	0.5589

The best results for models trained using generated data are summarized in the table below:

Classifier	F1	model_type
1	0.8018	rf
2	0.8182	cnn
3	0.6680	mlp
4	0.8333	rf
5	0.8004	cnn
6	0.7737	rcnn
7	0.8479	cnn
8	0.8215	rcnn
9	0.6256	cnn

Table 14: Best Results With Generated Data

The best results for each classifier using the training dataset without generated data are shown in tables 15 through 19.

Table 15: KNN Results

Question	Dimensions	Neighbors	F1	feature
1	200	4	0.707878788	tfidf
2	100	5	0.59127255	tfidf
3	200	10	0.549770633	w2v
4	200	1	0.752906977	w2v
5	100	7	0.753077652	tfidf
6	200	4	0.630990759	w2v
7	300	6	0.774068323	tfidf
8	100	4	0.621802657	tfidf
9	100	8	0.508535245	tfidf

Table 16: RandomForests Results

Question	Dimensions	n_est	max_dep	F1	feature
1	200	34	18	0.8534	w2v
2	200	22	34	0.7143	tfidf
3	100	38	42	0.6241	tfidf_lem
4	200	34	38	0.9387	tfidf_lem
5	300	22	42	0.7224	w2v
6	300	6	42	0.6077	w2v
7	200	10	42	0.8188	tfidf
8	200	22	34	0.6336	tfidf_lem
9	300	2	14	0.5122	tfidf

Table 17: MLP Results

Question	Dimensions	hls	max_it	F1	feature
1	200	(50, 50)	1000	0.7708	w2v
2	100	(50, 50)	1500	0.7454	tfidf
3	100	50	1000	0.5785	w2v
4	200	(50, 50)	1500	0.8546	w2v
5	100	(50, 50)	500	0.7870	tfidf
6	200	(50, 50)	1500	0.6994	w2v
7	300	(50, 50)	1000	0.8074	tfidf
8	300	(50, 50)	1500	0.7260	tfidf
9	100	50	500	0.6505	tfidf

Table 18: CNN Results

Question	Dimensions	epochs	input_d	F1
1	300.0	7	300	0.8153
2	200.0	7	300	0.7596
3	300.0	3	300	0.6226
4	100.0	7	300	0.9794
5	300.0	5	300	0.7695
6	200.0	3	300	0.6358
7	200.0	5	300	0.7171
8	200.0	7	300	0.7265
9	100.0	3	300	0.5414

Table 19: RCNN Results

Question	Dimensions	epochs	input_d	F1
1	100	3	300	0.7805
2	100	3	300	0.7776
3	100	3	300	0.5893
4	200	3	300	0.9387
5	200	3	300	0.7394
6	100	3	300	0.6390
7	100	3	300	0.6790
8	200	3	300	0.7526
9	100	3	300	0.6506

The best results for models trained without using generated data are summarized in the table below:

Table 20: Best Results Without Generated Data

Question	F1	Model	Better?
1	0.8534	rf	Yes
2	0.7776	rcnn	No
3	0.6241	rf	No
4	0.9793	cnn	Yes
5	0.7870	mpl	No
6	0.6993	mlp	No
7	0.8188	rf	No
8	0.7526	rcnn	No
9	0.6506	rcnn	Yes

The best overall results are:

Table 21: Best overall results from all methods

Question	model_type	F1
1	rf	0.8534
2	cnn	0.8183
3	mlp	0.6680
4	cnn	0.9794
5	cnn	0.8004
6	rcnn	0.7737
7	cnn	0.8479
8	rcnn	0.8215
9	cnn	0.6256

5.3 Data Visualizations:

5.3.1 Principal Components and t-SNE:

One way to visualize a feature representation's effectiveness at distinguishing between classes is through Principle Components and t-SNE plots. Principal Component analysis breaks a data set down by finding the minimum number of variables that explain the maximum amount of variance. It works by finding the eigenvalues and eigenvectors of the covariance matrix of the normalized data. The normalized eigenvectors form an orthogonal basis of the covariance matrix, and the value of the eigenvalues is proportional to the amount of variance explained by each normalized eigenvector.

t-SNE is a non-linear dimensionality reduction technique that works by optimizing a similarity measure between points in the dataset for a high and low dimensional space. In the high dimensional space, a Gaussian distribution is fitted over each point. The probability density of the Gaussian distribution at any other point in the dataset is proportional to the similarity between the two points. Next, a Cauchy distribution is fit over the point in the low dimensional space, and similarity is again calculated using the probability density at all other points. t-SNE attempts to make the similarity measures in the low dimensional space mirror the similarity measures in the high dimensional space.

The dataset that got the biggest boost from being classified without generated data was dataset 4. The principle component and t-SNE plots for the data for this classifier show why the performance boost was so dramatic.

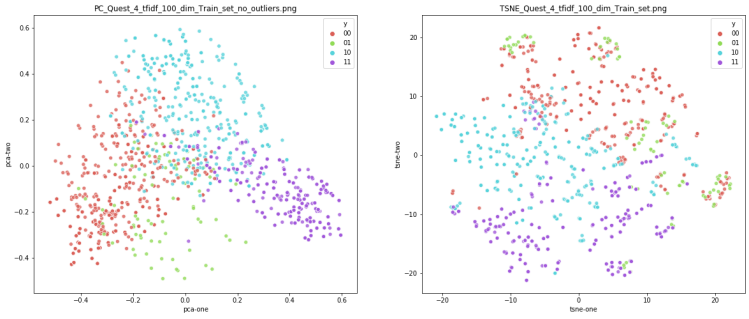

Figure 18: Left: Principal Component Plot for 100 dimension TFIDF for dataset 4 with generated data Right: t-SNE breakdown of 100 dimension TFIDF with generated data

Figure 18 shows that data points with three of the labels are relatively separable (00, 10, 11), but the data points for class 01 are interspersed with

data points with label 00. However, as seen in Figure 19, for the version of this dataset without generated data, the label 01 does not appear.

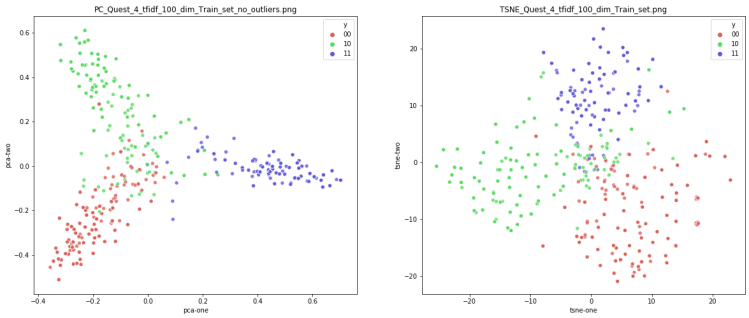

Figure 19: Left: Principal Component Plot for 100 dimension TFIDF for dataset 4 without generated data Right: t-SNE breakdown of 100 dimension TFIDF without generated data

The label 01 does not appear because it is not present in the rewrite, feedback, or context data. For this question, the binary sub-categories are not independent. The first sub-category is based on whether or not the user answered a question. The second sub-category is based on whether or not the answer gave specific details, so a label of 01 would be an answer that gave specific details without answering the question, which is impossible. The data generation algorithm is based on the assumption that the labels are independent, so it generates unnecessary data that weakens the classification accuracy in this case. Without this unnecessary class, the label separation is very clear, and the classifier can easily distinguish between the labels.

The other dataset that got a boost from being trained without generated data was dataset 1. A Principal Components and t-SNE plot using TFIDF for the data for this dataset are shown in Figure 20.

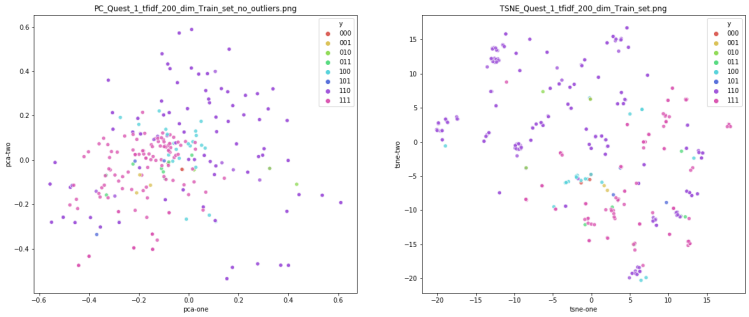

Figure 20: Left: Principal Component Plot for 200 dimension tfidf for dataset 1 with generated data Right: t-SNE breakdown of 200 dimension tfidf with generated data

There is not very clear separation between the labels. In both plots, the data points with the label 110 are spread out across most of the graph, but there are some clusters of points that may be helpful for classification purposes in the t-SNE plot. Figure 21 shows the same plots without generated data.

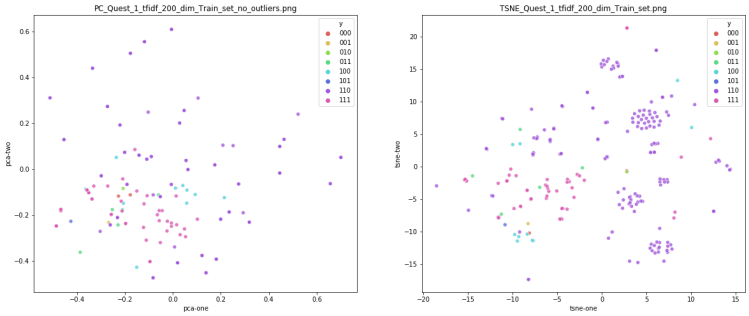

Figure 21: Left: Principal Component Plot for 200 dimension tfidf for dataset 1 without generated data Right: t-SNE breakdown of 200 dimension tfidf without generated data

The plots of the TFIDF representation without generated data show clearer label separation. The label 110 is clearly clustered and separated from most of the data in class 111, which is the second most prominent label.

The datasets with the worst classifier performance were 3 and 9. The plots for dataset 3 with generated data are shown below. Figure 22 shows the 200

dimension TFIDF representation. Figure 23 shows the 200 dimension Glove word vector representation.

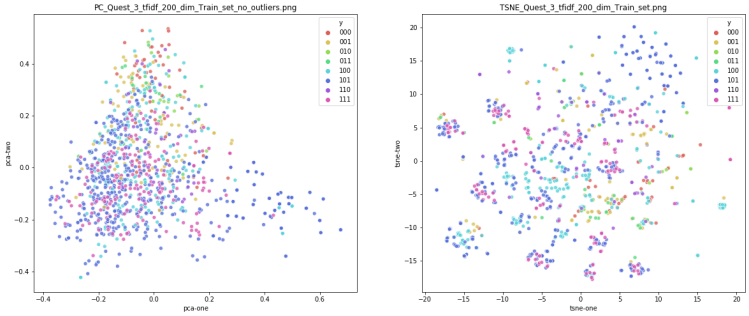

Figure 22: Left: Principal Component Plot for 200 dimension TFIDF for dataset 3 with generated data Right: t-SNE breakdown of 200 dimension TFIDF with generated data

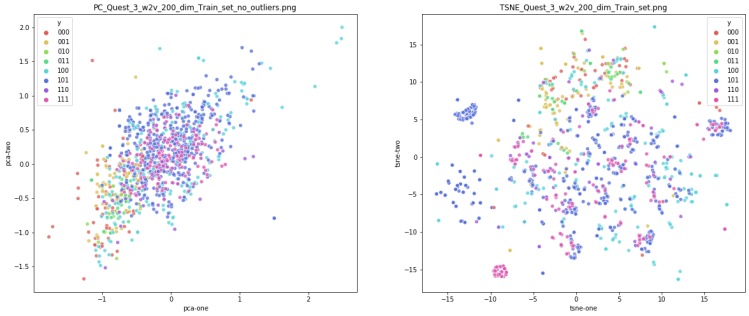

Figure 23: Left: Principal Component Plot for 200 dimension Glove word vectors for dataset 3 with generated data Right: t-SNE breakdown of 200 dimension tfidf with generated data

In both plots, there is not very clear separation between data with different labels. There are also several labels with a large number of samples, so this is a more difficult classification problem than many of the other classification problems in the analysis. The figures 24 and 25 show the same plots for this dataset without generated data:

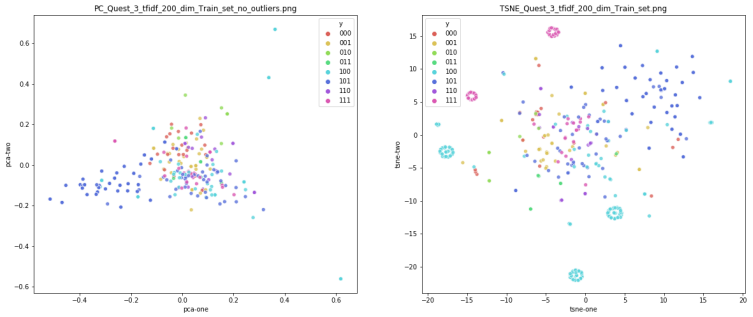

Figure 24: Left: Principal Component Plot for 200 dimension tfidf for dataset 3 without generated data Right: t-SNE breakdown of 200 dimension tfidf without generated data

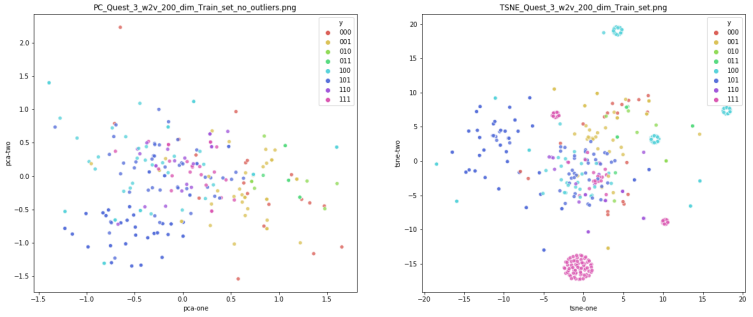

Figure 25: Left: Principal Component Plot for 200 dimension Glove word vectors for dataset 3 without generated data Right: t-SNE breakdown of 200 dimension Glove word vectors without generated data

Although there appears to be slightly more clustering in the t-SNE plots, the same issue with data points that are not sufficiently separated is still present in both graphs.

The graphs for dataset 9 are shown below. Figure 26 shows the 200 dimension tfidf representation. Figure 27 shows the 200 dimension Glove word vector average.

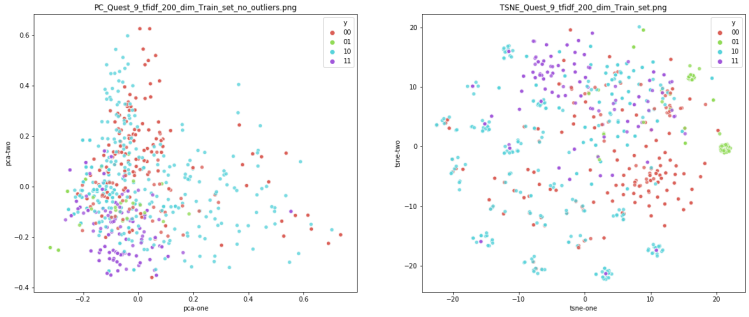

Figure 26: Left: Principal Component Plot for 200 dimension tfidf for dtaset 9 with generated data Right: t-SNE breakdown of 200 dimension tfidf with generated data

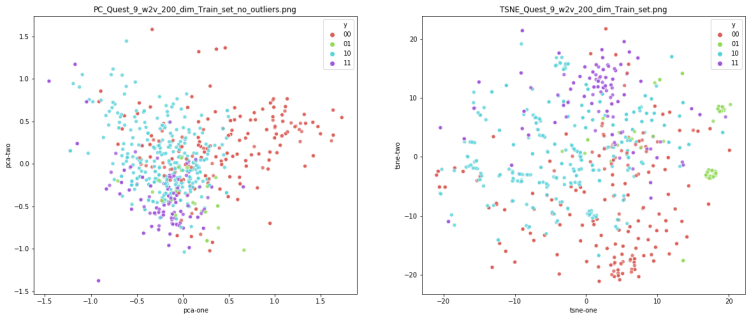

Figure 27: Left: Principal Component Plot for 200 dimension Glove word vectors for dataset 9 with generated data Right: t-SNE breakdown of 200 dimension Glove word vectors with generated data

There are clear clusters of data for each label except 01, but there is a considerable amount of overlap between the classes in all of the plots. Figures 28 and 29 show the TFIDF and Glove word vector average plots for dataset 9 without generated data.

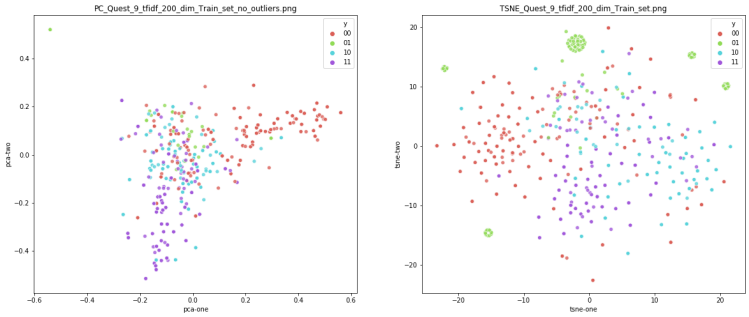

Figure 28: Left: Principal Component Plot for 200 dimension tfidf for dataset 9 without generated data Right: t-SNE breakdown of 200 dimension tfidf without generated data

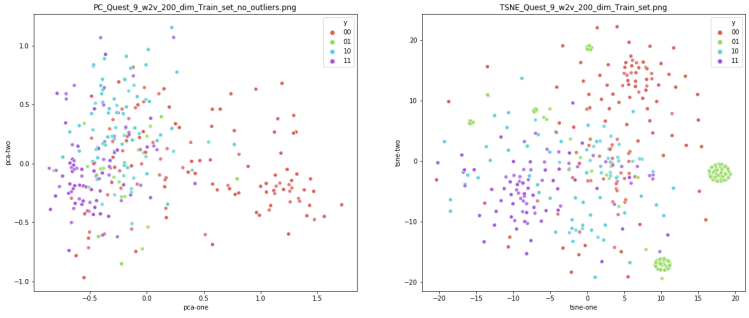

Figure 29: Left: Principal Component Plot for 200 dimension Glove word vectors for dataset 9 without generated data Right: t-SNE breakdown of 200 dimension Glove word vectors without generated data

In this case, removing the generated data does not make a big difference in the separability of the labels. No classes are dropped as were in the plots for dataset 4.

5.3.2 Unique Words:

Another way to visualize the data is by the percent of words in the data for each label that are unique to that label. For example, consider the plot of the percentage of unique words per label in dataset 1 shown in Figure 30 and the plot of the distribution of labels in the context data for dataset 1 in figure 31.

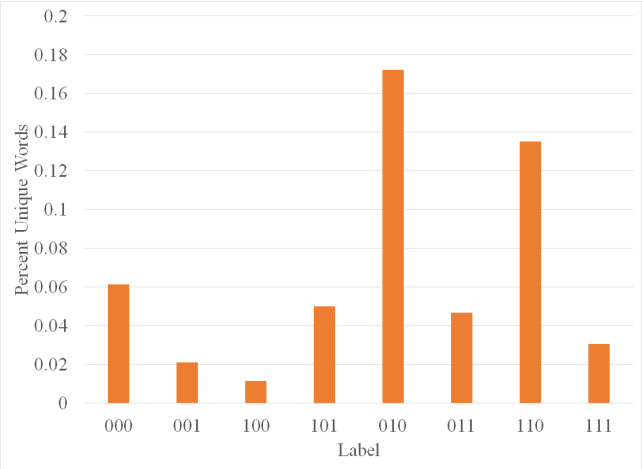

Figure 30: Percent of Words that are Unique to Each Label in Dataset 1

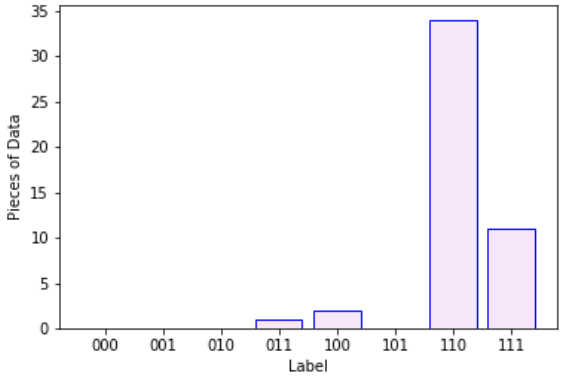

Figure 31: Label Distribution for Context Data in Dataset 1

Most of the labels do not have a very high percentage of unique words, but the label with the most samples in the context set has the second highest percentage of unique words. This may help the classifier perform well on this label in the dataset.

For dataset 2, the distribution of unique words shown in Figure 32 and the distribution of labels in the context set shown in Figure 33 are similar.

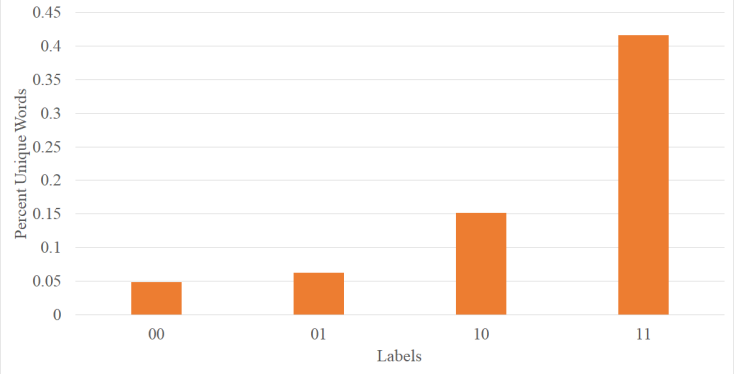

Figure 32: Percent of Words that are Unique to Each Label in Classifier 2

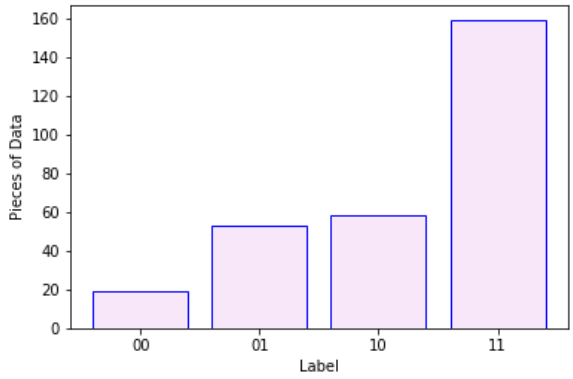

Figure 33: Label Distribution for Context Data in Dataset 2

This may make it easier for the classifier to find ways to distinguish data with different labels because there are more features for labels with more samples.

For the dataset for classifier 9, the distribution of unique words per label and the distribution of labels in the context dataset are different as is shown in figures 34 and 35. Label 00 has the most data points in the context set, but the other classes are also well represented. This makes it more difficult for the

classifier to distinguish between labels.

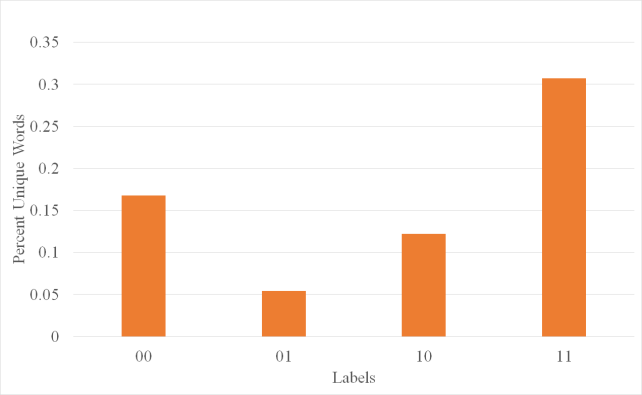

Figure 34: Percent of Words that are Unique to Each Label in Dataset 9

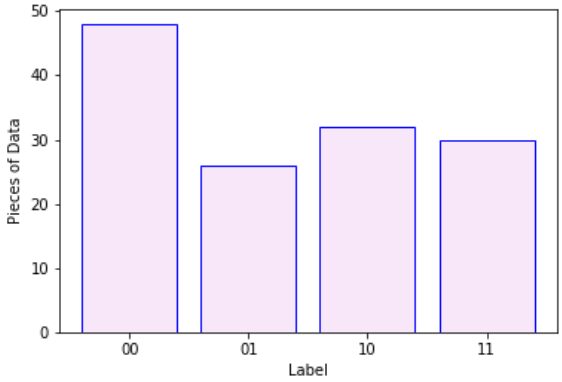

Figure 35: Label Distribution for Context Data in Dataset 9

5.4 Conclusion and Future Work:

Based on the modeling results, the text classification performance of this framework looks promising The advantages of using this framework include:

1. The binary labeling system makes it simple to model on new text features by adding additional labels to existing data, so the dialogue models can be updated to classify more specific test features.

2. The binary labeling system also makes it possible to generate a large training data set with any label distribution.

3. This framework can be used to create classification models that are able to capture non-intent based features like sentiment or politeness.

4. The framework can be used to replace a multiple-choice dialogue system without altering the structure of the simulation.

Future work could focus on:

1. Integrating this dialogue classification framework with some of the past work on creating free-input question answering educational simulations to create a dialogue system that is robust for both sentiment and intent based classification.

2. Expanding the data generation algorithm to include synonym replacement using word vector similarity to increase the lexical diversity

3. Comparative testing of multiple-choice and free-input dialogue systems

References

[1] Sheridan, Martha et al. Investigating the effectiveness of virtual reality for cross-cultural competency training SIEDS 2018. 2018 Systems and Information Engineering Design Symposium (SIEDS) (2018): 53-57.

[2] Elder, Lindsey. (November 2017). U.S., China participate in Disaster Management Exchange in Portland". Retrieved from `https://www.army.mil/article/196967/us_china_participate_in_disaster_management_exchange_in_portland`

[3] Kron FW, Fetters MD et al. "Using a computer simulation for teaching communication skills: A blinded multisite mixed methods randomized controlled trial." Patient Educ Couns (2016)

[4] Zielke M. A., Evans et al. Serious Games for Immersive Cultural Training: Creating a Living World. IEEE Computer Graphics and Applications, vol. 29, no. 2, Mar./Apr. (2009) pp. 49-60

[5] M. Kim, Julia et al. "BiLAT: A game-based environment for practicing negotiation in a cultural context." International Journal of Artificial Intelligence in Education (2009)

[6] Newble, DI et. al. "A comparison of multiple choice and free response tests in examinations of clinical competence. Medical Education." (1979). 13. 263 - 268

[7] Stankous, Nina, "Constructive response Vs Multiple-Choice Tests In Math: American Experience and Discussion" European Scientific Journal (2016)

[8] Eskenazi, Maxine. An overview of spoken language technology for education, Speech Communication vol 51, (2009) p. 832-844

[9] Bernstein, Jared & Najmi, Amir & Ehsani, Farzad. "Subarashii: Encounters in Japanese Spoken Language Education." CALICO Journal. 16. (1999).

[10] Ehsani, Farzad & Bernstein, Jared & Najmi, Amir. "An interactive dialog system for learning Japanese." Speech Communication. 30. (2000). 167-177.

[11] Seneff, Stephanie et al. Spoken Dialogue Systems for Language Learning. HLAACL (2007).

[12] Seneff, Stephanie. Web-based dialogue and translation games for spoken language learning. SLaTE (2007).

[13] Jia, Jiyou. The study of the application of a web-based chatbot system on the teaching of foreign languages." Proceedings of SITE 04, AACE Press, USA. 2004. 1201-1207

[14] Wik, Preben & Hjalmarsson, Anna Embodied conversational agents in computer assisted language learning. Speech Commun., Volume 51, Issue 10, October 2009 pp. 10241037.

[15] Johnson, W.L. et al. "Tactical Language Training System: Supporting the rapid acquisition of foreign language and cultural skills." InSTIL/ICALL Symposium, Venice, Italy. (2004).

[16] Mateas, Michael & Andrew Stern. Structuring Content in the Faade Interactive Drama Architecture. AIIDE (2005).

[17] Mateas, Michael & Andrew Stern. Natural Language Understanding in Faade: Surface-Text Processing. TIDSE (2004).

[18] Mateas, Michael & Andrew Stern. "Natural Language Understanding in Faade: Surface-Text Processing." Gbel S. et al. (eds) Technologies for Interactive Digital Storytelling and Entertainment. TIDSE (2004). Lecture Notes in Computer Science, vol 3105. Springer, Berlin, Heidelberg

[19] Leuski, Anton & Patel, Ronakkumar & Traum, David & Kennedy, Brandon. "Building effective question answering characters." Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue (2006). 18-27.

[20] Swartout, William R. et al. Virtual Humans for Learning. AI Magazine 34 (2013): 9-.

[21] Leuski, Anton and David R. Traum. NPCEditor: A Tool for Building Question-Answering Characters. LREC (2010).

[22] Kang, Yiping et al. Data Collection for Dialogue System: A Startup Perspective. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers) pp. 33-40

[23] Serban, Iulian V. et al. A Deep Reinforcement Learning Chatbot. (2017) ArXiv e-prints

[24] Danescu-Niculescu-Mizil, Cristian et al. "A computational approach to politeness with application to social factors." Proceedings of ACL, (2013)

[25] Navlani, Avinash (August 2018) "KNN Classification using Scikit-learn" Retrieved from `https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn`

[26] Mathur, Pankaj (May 2016) A Simple Multilayer Perceptron with TensorFlow. Retreived from `https://medium.com/pankajmathur/a-simple-multilayer-perceptron-with-tensorflow-3effe7bf3466`

[27] Saito, Yosuke. (January 2017) Easy Implementation of Convolution and Pooling Layer in Deep Learning. Retrieved from `https://saitoxu.io/2017/01/01/convolution-and-pooling.html`

[28] Koizumi, R. Relationships Between Text Length and Lexical Diversity Measures: Can We Use Short Texts of Less than 100 Tokens? Vocabulary Learning and Instruction Volume 1, Issue 1 (2012) pp.60-69

Appendices

A Multiple-choice Questions and Feedback:

Table 22 gives the multiple-choice options and corresponding feedback for question 1.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. You approach him to introduce yourself.

Table 22: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
Good Morning Captain Wang, I am honored to have you join us.	Good choice. This is an appropriate introductory greeting within Chinese culture. **Score: Best**
Good Morning Captain Wang, how are you doing today?	This is an acceptable answer, but it is not ideal to ask about the other person's welfare on a first meeting. **Score: Second Best**
Good morning, Captain Wang.	This answer is curt for a first meeting. It would be better to offer a slightly more formal greeting to a senior officer. **Score: Worst**

Table 23 gives the multiple-choice options and corresponding feedback for question 2.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. Upon meeting the commander of the Chinese team for the first time, he looks slightly discouraged when he sees your rank. He asks why your commander isn't present.

Table 23: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
He couldn't make it today because he had unexpected business come up. I will be in charge in his absence.	This is the best answer because it provides an excuse for the absence of your boss. Failing to do so would result in a loss of face for the Chinese Captain, because it would demonstrate that your officer does not have time to meet Captain Wang. **Score: Best**
No, he is not here. I am the officer in charge of the U.S. component of the coalition.	This is an acceptable answer because it clearly expresses that you have taken charge of the U.S. element. **Score: Second Best**
No, he is not here. I am taking his place.	This answer is too curt and could be insulting to a Chinese officer senior in rank to you because you provide no explanation for the absence of your superior officer. **Score: Worst**

Table 24 gives the multiple-choice options and corresponding feedback for question 3. This question only comes up if the player does not explain why their commander could not come to the meeting in question 2.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. He asks why your commander isn't present. Your commander can't make it because he had unexpected business.

Table 24: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
Unfortunately, he was assigned another mission at the last minute and cannot make it today.	This answer does a good job of providing a reasonable excuse without revealing any internal information. **Score: Best**
He is busy today	This answer is palatable because it offers an excuse for your senior officer's absence, but also offensive because it implies that your superior officer's time is more important than a Chinese officer of the same rank. **Score: Second Best**
Id rather not say.	This answer is the worst because it is unnecessarily ambiguous. It only serves to increase suspicion between U.S. and Chinese. **Score: Worst**

Table 27 gives the multiple-choice options and corresponding feedback for question 4.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. He asks you for information on the type of supplies that you have brought for the mission. You have not decided what type of supplies you are going to use and will be covering this in a meeting this afternoon.

Table 25: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
I appreciate that you are looking out for your team, but you will get all of it soon when we have the brief this afternoon.	This answer is appropriate because it displays understanding and empathy towards the officer and his need to know, but is direct in telling him he will not comply. **Score: Best**
Is it alright if this waits until the actual mission brief? I'm busy getting ready for it now.	Has an air of politeness to it that implies if the Chinese officer really needed the information, that you will give it, but it would be inconvenient to do so. **Score: Second Best**
Not all of the details have been finalized so I'm not sure if that it would be of any help to you now.	The response is curt and dismissive. **Score: Worst**

Table 26 gives the multiple-choice options and corresponding feedback for question 5.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. The Chinese commander asks you for information on the type of supplies that you have brought for the mission. You tell him, you want to wait until the meeting this afternoon to talk about it. He tells you that knowing some basic information now would be helpful.

Table 26: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
We have crates of supplies like food, water, and hospital equipment available. As of right now that will be primarily what we are giving the locals.	The response is direct and does not seem to hold ulterior motives. **Score: Best**
There really is not much to tell. It's going to be our basic loadout for Humanitarian aid missions.	Deliberately makes assumptions about the Chinese officer knowing what a basic loadout is for an American aid mission which is not entirely empathetic. **Score: Second Best**
I'd really prefer just to wait until the brief, if that is ok.	This answer is very direct and centered on the speaker's discomfort with sharing information rather than make up a reason or simply share. Depending on the tone could display distrust. **Score: Worst**

Table **??** gives the multiple-choice options and corresponding feedback for question 6.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. You have just heard that planes carrying equipment for the Chinese team have been delayed. You want to give a response that will indicate how you will adapt to the changes.

Table 27: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
I'm sorry to interrupt, it sounded important. Anything I can do to help? With your plane being delayed I'd like to do my best to make the mission go smoothly.	You are offering a solution to their problem while keeping it at the lowest level. This helps save face for the officer by solving their problem for them. Good for long term orientation as well. **Score: Best**
Oh, is it about the plane being delayed? We have plenty of supplies here to compensate so we can accomplish the mission together until it arrives. I'm sure we can figure something out.	More of a mutual solution rather than offering something first. **Score: Second Best**
I hope everything was resolved. If it is alright, I have a few questions about the mission since I heard you had a flight delay and that will affect how many Chinese soldiers are on the ground.	Addresses their problem but does not offer solutions. Focuses on your problem due to their shortcomings. **Score: Third Best**
Well, I wanted to come over and share some of what we are planning for the mission and try and collaborate more closely since I just found out your plane was delayed.	Worst option since the tone is quite casual and seems to be flippant of the Chinese Officers situation. **Score: Worst**

Table 28 gives the multiple-choice options and corresponding feedback for question 9.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss joint aid mission. He has just told you that planes carrying equipment for the Chinese team have been delayed, and unless the planes arrive, his team may not be able to help. You try to convince him to help you and have told him that you will go to a higher ranked officer if he does not help, but he says it is out of his control. You want to convince him that it is better to cooperate than to make you go to a higher ranked officer.

Table 28: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
Please work with me here. There are plenty of supplies to go around.	Shows the frustration of the American officer and show how he feels trapped just like the Chinese Officer. Shows how they need to accomplish it together. Overall best option. **Score: Best**
I saw you pointing at the containers here. Would the issue have to do with that? We can make this work, we have to.	Ties the two parties together and points out a potential reason why the issue exists with the supplies. **Score: Second Best**
It's not the ideal solution, I know, but unless we come to a compromise with manning or supplies, I will have no choice but to go to your commanding officer.	Makes the American look like he is only concerned about himself and is using the same desire to stay under the radar to get the mission accomplished. **Score: Worst**

Table 29 gives the multiple-choice options and corresponding feedback for question 10.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. There has been a mix up, but the Chinese commander has proposed an alternative plan. You have a few problems with the plan. You want to discuss it with him further.

Table 29: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
Can I talk with you in private once questions are over?	Good job! This answer both demonstrates respect for the team leader's image in front of his subordinates and accomplishes your intent of discussing the plan. **Score: Best**
Yes, I have a few problems with your plan I'd like to discuss in private.	Although asking to talk in private is the appropriate response, you addressed your interests in a tactful and respectful manner, and the team leader will most likely not find fault with your wording. He will, however, be flustered that you addressed a potential flaw in front of his subordinates. **Score: Second Best**
I don't have any questions right now.	Although waiting to speak to the team leader when he is in private is culturally appropriate, this response is too passive and does not even arrange for a later discussion. **Score: Third Best**
Can you explain to me why we are setting up two individual aid stations?	Asking the team leader to explain to you will likely make him angry that he has to justify himself to a peer, and doing this in public will most likely embarrass him. **Score: Worst**

Table 30 gives the multiple-choice options and corresponding feedback for question 12.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. You originally planned to have one aid station, but he has proposed an alternative plan with two aid stations. You are now meeting to discuss his plan.

Table 30: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
Thank you for meeting with me. I just had some questions about the part of the plan concerning the aid stations? Why are you doing it that way?	Good job expressing thanks for meeting with him. Phrasing your issues with the plans as questions will also reaffirm his status as the leader of his team and prompt him to be more gracious in discussing the plans. **Score: Best**
Ok. Now that we're in private, why did you say that we would set up two aid stations?	This answer isn't bad, but in demonstrating eagerness to discuss the problem right off the bat it may be conveyed as disrespect. **Score: Worst**

Table 31 gives the multiple-choice options and corresponding feedback for question 13.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. He has proposed to set up two aid stations to help refugees, one American and one Chinese. You think it would be better to have one large American aid station with American medics.

Table 31: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
What benefit does your way have? If we split up the supplies, we won't be able to give every person what they need at a single station.	By asking for the team leader's reasoning instead of calling him out, you create a nonthreatening basis for discussion and also offer a valid point of your own. **Score: Best**
The plan was to use one large aid station with American medics running it. How is this way better?	You simply restated the original plan, which the team leader elected to change. This raises the probability of an impasse since you are not superior to the Chinese team leader. **Score: Worst**

Table 32 gives the multiple-choice options and corresponding feedback for question 14.

Context for this question: You are an American soldier who is meeting with the commander of a Chinese army platoon to discuss a joint aid mission. You are discussing how to divide the responsibilities between the two teams. You think it would be better to have American medics running the aid station because you feel that they have better training. You want to Chinese team to work site security because they have more training in that area.

Table 32: Multiple-Choice and Feedback Table Question 1

Multiple-Choice Text	Feedback and Score
When I made the plan, I believed that securing the site and protecting the victims was the most important task. Your men have good experience in security, so I wanted them to make sure everyone is safe. Our supplies are better suited for running the aid station.	Good job! In addressing the Chinese team's strengths and demonstrating respect for them by aligning them with important tasks, you will appeal to the team leader's inherent pride in his group's abilities. **Score: Best**
Our medics are better at doing this sort of thing, and your men are better at site security.	You mentioned each team's relative strengths, but unless you incentivize their job's importance and appeal to the leader's pride, they will keep doing what they were doing. **Score: Second Best**
Our American medics are trained at running aid stations and treating patients. They have much more experience than your medics. Our medics should handle the flow of patients.	This might be true, but mentioning this to the team leader will make him feel offended and angry. **Score: Worst**

B Label Sub-categories:

Table 33: Sub-categories for each dataset and multiple-question part 1

Dataset:	Corresponding Multiple-choice Question(s)	Label sub-category 1	Label sub category 2	Label sub category 3
1	1	Greets officer (e.g., "Good morning Captain Wang" or "Good morning sir")	Doesn't ask about officer's welfare, (e.g., does not say "How are you doing today?")	Uses an honorific (e.g., "I'm honored to meet you" or "I'm thrilled to meet you")
2	2,3	States a reason for boss's absence or implies that the boss's absence is out of his control (i.e., uses language like "couldn't" or "can't" or states "He would have liked to be here")	Expresses that you are leading the mission (e.g., "I am taking his place" or "I am leading")	N/A
3	4	Gives the time when he will get the information (e.g., "We will give that information in the brief this afternoon")	Displays understanding and empathy towards the officer and his need to know. Does not dismiss the concern (e.g, "I appreciate that you are looking out for your team")	Explains that you are not ready (e.g. Not all details have been finalized or "I am preparing for the brief')
4	5	Answers the question (e.g. does not say "I'd really just prefer to wait until the brief')	Gives detailed information (e.g. "We have food, water, etc." rather than "We have the basic loadout for a humanitarian aid mission")	N/A

Table 34: Sub-categories for each dataset and multiple-question part 2

Classifier:	Corresponding Multiple-choice Prompt(s)	Label sub-category 1	Label sub category 2	Label sub category 3
5	6	Addresses the problem (e.g., "I heard you had a plane delay" or "I head not as many Chinese soldiers will be on ground")	Gives possible solution or offers to find a solution (e.g., "We have enough supplies for your team" or "We will work together to find a solution")	
6	7	Implies that they should resolve the situation together	Does not mention going to a higher officer (e.g., "we may have to go to the higher officer if we don't get a solution" or "I will have to go to a higher officer")	N/A
7	8	Asks to speak about it in private or meet later (e.g., "I would like to talk to you in private later. I have a few questions.")	Does not publicly challenge his plan or imply that you disagree with his plan (e.g., does not say "I have an issue with your plan")	N/A

Table 35: Sub-categories for each dataset and multiple-question part 3

Classifier:	Corresponding Multiple-choice Prompt(s)	Label sub-category 1	Label sub category 2	Label sub category 3
8	9, 10	Asks for the reason why he chose his plan (e.g., "Would you explain why your plan calls for setting up two aid stations?")	Presents the reasoning for your own plan (without directly restating your plan) (e.g., "If we split up the supplies, we won't be able to give every person what they need at a single station.")	N/A
9	11	Demonstrates respect for the Chinese team's strengths (e.g., "Your men have good experience" or "Your men have better training in site security")	Does not praise the American medics (e.g., does not say "American medics are better trained for running aid stations")	N/A