# DEEP NEURAL LANGUAGE GENERATION WITH EMOTIONAL INTELLIGENCE AND EXTERNAL FEEDBACK

by

Vidhushini Srinivasan

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Computer Science

Charlotte

2018

Approved by:

_____
Dr. Samira Shaikh

_____
Dr. Nicholas Davis

_____
Dr. Minwoo Jake Lee

# ABSTRACT

VIDHUSHINI SRINIVASAN. Deep Neural Language Generation with Emotional Intelligence and External Feedback.
(Under the direction of DR. SAMIRA SHAIKH)

Neural seq2seq models are widely used to generate dialogue using variations of RNN Encoder-Decoder architecture with maximum likelihood estimation as objective function. Advances in this architecture include introducing reinforcement learning to tune network parameters [1, 2]. This research focuses on generating affective responses in an attempt to create conversational agent that cater to emotional context of conversation.

We propose a novel approach using bidirectional RNN Encoder-Decoder seq2seq model with attention mechanism and maximum mutual information as initial objective function combined with Reinforcement Learning (RL). We train our own word2vec embeddings with appended valence, arousal and dominance [3] scores as input. We use an AlphaGo-style strategy by initializing RL system with general response policy learnt from our seq2seq model, which is then tuned using policy gradient method. The internal rewards are Ease of Answering, Semantic Coherence [1] and Emotional Intelligence, incorporated by minimizing affective dissonance [4] between the source and the generated response. We use a two-part training scheme, where we train a separate external reward analyzer to predict the reward using human feedback and then use RL system on predicted rewards to maximize the expected rewards (both internal and external). We use two different datasets namely Cornell Movie Dialog Corpus and Yelp Restaurant Review to train two different models and the affective responses generated by our models are tabulated. We use the standard evaluation metrics like BLEU, Perplexity and ROUGE-L along with the human evaluation to evaluate our models.

# DEDICATION

This work is dedicated to my mother, Dr.P.S. Rathai and my father, R. Srinivasan who constantly motivated and supported me throughout my master's.

# ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Dr. Samira Shaikh, for giving me an opportunity to work with her. My thesis would not have been possible without her constant guidance, valuable input and feedback at each and every step. I would also like to take this opportunity to thank Sashank Santhanam, PhD student, advised by Dr.Samira Shaikh for his excellent suggestions and feedback throughout my thesis. I am very glad to work on one of the most interesting research topics and complete my master's degree with a wonderful research experience.

I would also like to express my gratitude to my committee members, Dr. Nicholas Davis and Dr. Minwoo Jake Lee for showing interest in my research work and accepting to join my thesis committee as panel members. They played a vital role in structuring my thesis by providing ideas and feedback that guided me towards accomplishing my goal. My sincere appreciation to the University of North Carolina at Charlotte for accepting me as a master student and for providing necessary infrastructure and support to successfully complete my master's degree with thesis.

TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

BLEU Bilingual Evaluation Understudy Score

CBOW Continuous Bag of words

EA Ease of Answering

EI Emotional Intelligence

HF Human Feedback

LSTM Long Short Term Memory Network

RL Reinforcement Learning

RNN Recurrent Neural Network

ROUGE Recall-Oriented Understudy for Gisting Evaluation

SC Semantic Coherence

Seq2Seq Sequence-to-Sequence

SVM Support Vector Machine

# CHAPTER 1: INTRODUCTION

The booming growth of machine learning and artificial intelligence has arguably more potential of being transformational than any other innovation we have ever seen. In fact, it has the potential to be so valuable that Mark Cuban predicts the world's first trillionaires will actually be entrepreneurs working with artificial intelligence.

Deep Learning is one of the most widely used tool in building conversational agents. Existing conversational models are prone to being short, dull, off-context, vague and does not capture the affective content. Such a conversational model can be used in variety of applications like healthcare, eCommerce and online counselling.

Many companies already use chatbots for customer service tasks. If such a model lacks emotional intelligence, people could become frustrated or even feel disgusted with a company. Developing such a model, basically involves two tasks on the higher level. First, an analysis of a conversation with a human occurs to detect the sentiment. Next, the chatbot uses that information to come up with a relevant answer that is also emotionally appropriate.

As the chatbots become emotionally in tune, people could have more fun chatting with them. They might still be able to tell they are not communicating with humans in some cases, but the overall conversations should be more engaging.

Incorporating emotional intelligence in the conversational models can improve open-domain conversational prowess and will serve as a crucial component in generating emotionally rich responses suitable for handling different tasks.

## 1.1 Problem Statement

We can see that the chatbots are becoming scary good at mimicking our language. But until they can detect our emotional state and respond accordingly, they might never reach their full potential.

The main goal of my research work is to develop a conversational model that generates emotional, coherent, diverse and interesting dialogues that closely mimics human. There are several problems in building a conversational agents like lack of emotional aspect, short, dull and repetitive responses that motivated us to propose a different approach to resolve the common challenges in developing such models. We mainly focus on affective modelling of natural language conversations as the existing models don't explicitly capture emotional states in the textual conversations. For instance, attention mechanism [5] in sequence-to-sequence model [6] can learn syntactic alignment alignment of words within generated sentences. Similarly, word embedding models like Word2Vec [7] learn word vectors by context using CBOW (Continuous Bag of words) or Skip Gram techniques, preserving low-level word semantics. However, emotional aspects are not captured by existing methods.

We also show that incorporating emotional aspects along with other traditional methods improve the quality of generated responses in open-domain dialog systems.

## 1.2 A Brief Survey on Chatbots

Chatbots, also called Conversational Agents or Dialog Systems, are a hot topic today. Microsoft is making big bets on chatbots, and so are companies like Facebook (M), Apple (Siri), Google, WeChat, and Slack. There is a new wave of startups trying to change how consumers interact with services by building consumer apps like Operator or x.ai, bot platforms like Chatfuel, and bot libraries like Howdy's Botkit. Microsoft recently released their own bot developer framework that has libraries to facilitate building bots with intelligence and cognitive powers quickly and easily.

Users can talk in many places and so should the bot. Azure Bot Service provided by Microsoft can be integrated across multiple channels to increase interactions and reach more customers using your website or app to email, GroupMe, Facebook Messenger, Kik, Skype, Slack, Microsoft Teams, Telegram, text/SMS, Twilio, Cortana, and Skype for Business. Along with this, Cognitive Services enable the bot to see, hear, and interpret in more human ways.

Ebay has it's own chatbot that acts as a shopping assistant and helps customers in purchasing their desired products at desired price. Thus, the chatbots provide more engaging and natural conversations that attract customers in buying products and therefore increase the revenue of the company.

Many companies are hoping to develop bots to have natural conversations indistinguishable from human ones, and many are claiming to be using NLP and Deep Learning techniques to make this possible. But with all the hype around AI it's sometimes difficult to tell fact from fiction.

Emotional Intelligence is one of the core aspects that should be incorporated in these bots to generate emotional response that is relevant to a given situation. Lack of emotional intelligence can have disastrous impacts and can also result in reduction of customer base in case of customer service chatbots.

## 1.3 Taxonomy of Models

There are different varieties of conversational models that can be developed using different techniques. In this section we would discuss about some of the existing models [8] used in building the conversational agents or Dialog systems.

Retrieval-based models, use a repository of predefined responses and some kind of heuristic to pick an appropriate response based on the input and context. The heuristic could be as simple as a rule-based expression match, or as complex as an ensemble of Machine Learning classifiers. They choose response from the fixed set and do not generate new responses. Due to the repository of handcrafted responses, retrieval-

based methods do not make any grammatical mistakes. They cannot handle unseen cases for which no appropriate predefined response exists. These models can't refer back to contextual entity information like names mentioned earlier in the conversation. These models are easier to train.

Generative models, do not depend on pre-defined repository. They generate new responses from scratch. Generative models are typically based on Machine Translation techniques, but instead of translating from one language to another,they translate or map input to the target response. Generative models are smarter. They can refer back to entities in the input and give the impression that we are talking to a human. These models are hard to train as they require huge amount of data. They are prone to make grammatical mistakes unlike retrieval based models.

Most of the models today either use retrieval based models completely owing to high accuracy or they use a combination of retrieval based and generation based models.

Another prominent problem is the generation of long/short conversations. The longer conversations are more difficult to automate. On the other hand, the Short-Text Conversations are easier where the goal is to create a single response to a single input. For example, we may receive a specific greeting message from the user and may respond with a short greeting message in turn. However, long conversations are harder to automate where we have to go through multiple turns and need to keep track of what has been said. Customer support conversations are typically long conversational threads with multiple questions and we may have to track these questions to respond accordingly.

Next, we have an open domain setting where the user can involve in conversation anywhere. Here, we don't have a well-defined goal or intention. Conversations on social media sites like Twitter, Facebook and Reddit are typically open domain and can go into all kinds of directions. There are infinite number of topics and a certain amount of world knowledge is required to create reasonable responses making the

open domain conversation problems hard .

Whereas, in a closed domain setting the number of possible inputs and outputs are limited and we focus on achieving a very specific goal. Technical Customer Support or Shopping Assistants are examples of closed domain problems. These systems don't need to know about politics as they just need to fulfill their specific task as efficiently as possible. The conversation can take place anywhere as in open-domain setting, but the world knowledge is not required in creating responses and the only need for the model is to know about the specific goal that it must achieve.

## 1.4 Common Challenges

There are some obvious challenges [8] when developing such conversational agents that has the capacity to act like humans.

One of the main challenges lies in incorporating context. One needs to incorporate both linguistic and physical context to produce sensible responses. We can incorporate linguistic context by keeping track of information that has been exchanged in long conversations. In general, the conversations are embedded into a vector that is given as input to the neural network. Embedding long conversation into a vector is another challenging problem. We might also need to incorporate other kinds of contextual data such as date/time, location, or information about a user based on the problem that is being addressed by our conversational model.

Another challenge lies in generating responses incorporating coherent personality. When generating responses the model should produce consistent answers to semantically identical prompts. For instance, the inputs like "How old are you?" and "What is your age?" should fetch the same reply. This may sound simple, but incorporating such fixed knowledge or "personality" [9] into models is very difficult and one of the active research problems. Models can learn to generate linguistically plausible responses, but they are not trained to generate semantically consistent ones. This might be due to the fact that they are trained on a lot of data from multiple different

users in different scenarios.

| | |
|---|---|
| *message* | Where do you live now? |
| *response* | I live in Los Angeles. |
| *message* | In which city do you live now? |
| *response* | I live in Madrid. |
| *message* | In which country do you live now? |
| *response* | England, you? |

Figure 1.1: Conversation involving semantically identical prompts

Next step and the most challenging part would be evaluating our model. The ideal way to evaluate a conversational agent is to check whether or not our model is fulfilling its specific task/goal, e.g. the model built to solve a customer support problem, should resolve the issue in the conversation. Obtaining labels that say if the model fulfills a task or not is expensive as they require human judgment and evaluation. For open-domain conversational agents there are no well-defined goals and these agents have no one right answer. Such models are even harder to evaluate. Common metrics such as BLEU [10] that are used for Machine Translation and are based on text matching aren't well suited because sensible responses can contain completely different words or phrases when compared to the actual target. There are other metrics like perplexity, word error rate etc but none of the commonly used metrics really correlate with human judgment.

Understanding the intention and yielding diverse response is another common challenge in developing conversational models. A common problem with generative models is that they tend to produce generic responses like "That's great!" or "I don't know" and even single worded responses like "Yes", "No" that can serve as an answer for most of the prompts. Early versions of Google's Smart Reply tended to respond with "I love you" to almost anything. That's partly a result of how these models are trained, both in terms of data and in terms of actual training objective/algorithm. Some models are trained to artificially promote diversity through various objective/cost functions. The humans give responses that are specific to the

input and that carry an intention/purpose. Open-domain models have no specific goal and it is very difficult to train the model to generate responses that carry an intention. Microsoft's Tay was designed to explore conversational understanding and was hoped to become progressively smarter as it involves in actual conversation. But Tay was persuaded by racists and started posting racist tweets without knowing what racism actually means. The company has to take Tay offline and delete tweets posted by it.Neither Microsoft nor Tay was responsible for posting racist comments or promote racism but the model learnt to post those comments based on its previous conversations with racists. Thus, lack of intention can lead to disaster and creating such models that can generate responses with an intent is extremely difficult and is yet another active research problem.

# CHAPTER 2: BACKGROUND

Deep learning is a subset of machine learning methods based on learning data representations, which falls within the field of artificial intelligence. Learning can be supervised, semi-supervised or even unsupervised. Modern deep learning models are based on an artificial neural network, although they can also include propositional formulas or latent variables organized layer-wise in deep generative models such as the neurons in deep belief networks and deep Boltzmann machines.

In deep learning, each level learns to transform its input data into a more abstract and composite representation. In an image recognition problem, the input may be a matrix of pixels; the first layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize that the image contains a face. Importantly, a deep learning process can learn which features to optimally place in which level on its own.

Let us consider the problem of handwritten digit recognition. To solve this problem, the computer needs to be able to cope with huge variety of data representation. Every digit between 0 and 9 can be written in multiple ways, the size and shape of each handwritten digit can be very different depending on the person who is writing. Coping with the variability of these features and interactions between them, is where deep learning and deep neural networks become useful.

Neural networks are mathematical models whose structure is loosely inspired by that of the brain. Each neuron within a neural network is a mathematical function that takes in data via an input, transforms that data into a more amenable form, and then outputs the result.

All neural networks have an input layer, where the initial data is fed in, and an output layer, that generates the final prediction. But in a deep neural network, there will be multiple "hidden layers" of neurons between these input and output layers, each feeding data into each other.Thus, the term "deep" in "deep learning" and "deep neural networks" refers to the large number of hidden layers that are at the heart of these neural networks. These hidden layers help in recognizing and learning complex patterns and relationships and could help in better recognition of handwritten digits and can be used to solve many other complex real world problems.

## 2.1 Deep RNN and LSTM

Recurrent Neural Networks (RNNs) are popular models that have shown great promise in many NLP tasks. The idea behind RNNs is to make use of sequential information. In a traditional neural network we assume that all inputs and outputs are independent of each other. If we want to predict the next word in a sentence we should better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps. The most commonly used type of RNNs are LSTMs, which are much better at capturing long-term dependencies than vanilla RNNs. LSTMs are essentially the same thing as the RNN, but have a different way of computing the hidden state.

Long Short Term Memory networks [11], called as "LSTMs" are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people. They work tremendously well on a large variety of problems, and are now widely used in many NLP applications. They are predominant in machine translation and natural

language generation.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer. LSTMs have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way Figure(2.1).

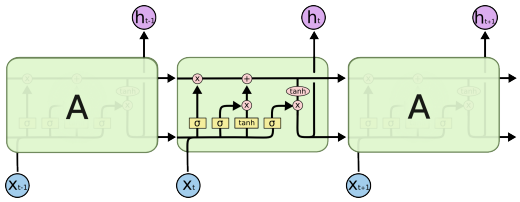


Figure 2.1: LSTM

The LSTMs have the ability to remove or add information to the cell state, carefully regulated by the structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means that nothing should be let through the gate, whereas a value of one means that everything should be let through the gate.

An LSTM has three of these gates, to protect and control the cell state. The input, output and forget gates. For building the proposed conversation agent, we have used a variant of RNN called Deep Bidirectional RNN. Bidirectional RNNs are based on the idea that the output at time t may not only depend on the previous elements in the sequence, but also future elements. For example, to predict a missing word in a

sequence we may want to look at both the left and the right context. Bidirectional RNNs are quite simple. They are just two RNNs stacked on top of each other. The output is then computed based on the hidden state of both RNNs.

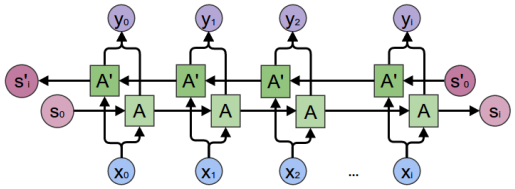


Figure 2.2: Bidirectional RNN

## 2.2 Seq2Seq Encoder Decoder Architecture

A Sequence to Sequence network or seq2seq network or Encoder Decoder network [12], is a model consisting of two RNNs called the encoder and decoder. The encoder reads an input sequence and outputs a single vector known as context vector, and the decoder reads that vector to produce an output sequence. This seq2seq model is most widely used in NLP tasks like Machine Translation, Text Summarization and developing conversational agents etc

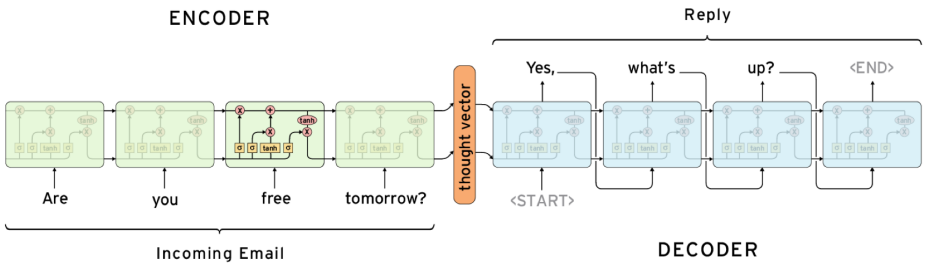


Figure 2.3: Seq2Seq Encoder-Decoder Architecture

Before we begin developing the model, we have to first build a "vocabulary" list containing all the words we want our model to be able to use or read i.e all the words in our corpus. The model inputs will have to be tensors containing the IDs of the words in the sequence.

There are four symbols that we need our vocabulary to contain. Seq2seq vocabularies usually reserve the first four spots for these elements:

<PAD>: During training, we'll need to feed our examples to the network in batches. The inputs in these batches should be the same width for the network to do its calculation. Our examples, however, are not of the same length. That's why we'll need to pad shorter inputs to bring them to the same width of the batch.

<EOS>: This is another necessity of batching as well, but used on the decoder side. It allows us to tell the decoder where a sentence ends, and it allows the decoder to indicate the same thing in its outputs as well.

<UNK>: When training the model on real data, we can find that we can vastly improve the resource efficiency of the model by ignoring words that don't show up often in the vocabulary to warrant consideration. We replace those with <UNK>.

<GO>: This is the input to the first time step of the decoder to let the decoder know when to start generating output.

We can also choose to reverse the order of words in the input sequence so as to get better prediction in the decoding phase. During the preprocessing we do the following:

1. we build our vocabulary of unique words and count the occurrences.
2. we replace words with low frequency with <UNK>
3. create a copy of conversations with the words replaced by their IDs
4. we can choose to add the <GO> and <EOS> word ids to the target dataset now, or do it at training time

An RNN basically contains a number of hidden state vectors each representing information from the previous time steps. For example, the hidden state vector at the 3rd time step will be a function of the first 3 words. By this logic, the final hidden state vector of the encoder RNN can be thought of as a pretty accurate representation of the whole input text and it captures the entire context.

The decoder is another RNN, which takes in the final hidden state vector or context

vector of the encoder and uses it to predict the words of the output reply. The first cell's job is to take in the vector representation v, and decide which word in its vocabulary is the most appropriate for the output response. Mathematically speaking, this means that we compute probabilities for each of the words in the vocabulary, and choose the argmax of the values.

The 2nd cell will be a function of both the vector representation v, as well as the output of the previous cell and this continues till the last LSTM cell where we get the final predicted response as a whole. The goal of the LSTM is to estimate the conditional probability.

$$p(y_1, ..., y_{T'}|x_1, ....., x_T) = \prod_{t=1}^{T'} p(y_t|v, y_1, ...., y_{t-1}) \tag{2.1}$$

We can decipher the equation 2.1 as follows: The left side refers to the probability of the output sequence, conditioned on the given input sequence. The right side contains the term $p(y_t|v, y_1, ..., y_{t-1})$, which is a vector of probabilities of all the words, conditioned on the vector representation and the outputs at the previous time steps. The Pi notation is simply the multiplication equivalent of Sigma (or summation). The right hand side can be reduced to $p(y_1|v) * p(y_2|v, y_1) * p(y_3|v, y_1, y_2)$ ... and so on.

Given the phrase "Are you free tomorrow?", let's think about how most people would answer the question. A majority will start with something along the lines of "Yes", "Yeah", "No", etc. After we're done training our network, the probability $p(y_1|v)$ will be a distribution. The second probability we need to compute, $p(y_2|v, y_1)$, will be a function of the word this distribution $y_1$ as well as the vector representation v. The result of the Pi (product) operation will give us the most likely sequence of words, which we will use as our final response.

One of the most important characteristics of sequence to sequence models is the versatility that it provides. The traditional ML methods (linear regression, SVMs) and deep learning methods like CNNs, require a fixed size input, and produce fixed

size outputs as well. The lengths of the inputs must be known beforehand. This is a significant limitation to tasks such as machine translation, speech recognition, and question answering. These are tasks where we don't know the size of the input phrase, and we'd also like to be able to generate variable length responses, not just be constrained to one particular output representation. Seq2Seq models provide this flexibility.

There are several improvements to this seq2seq model and one such major improvement is using reinforcement learning to fine-tune the seq2seq model.

## 2.3 Reinforcement Learning

Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize cumulative reward. Reinforcement is modeled as a Markov decision process and has the following attributes:

1. a set of environment and agent states, S
2. a set of actions, A, of the agent
3. the probability of transition from state s to state s' under action a is given as $P_a(s, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a)$
4. the immediate reward after transition from s to s' with action a is $R_a(s, s')$.
5. rules that describe what the agent observes

Rules are often stochastic. The observation typically involves the scalar, immediate reward associated with the last transition. The agent is assumed to observe the current environmental state. If not, the agent has partial observation. Sometimes the set of actions available to the agent is restricted based on the task it performs.

A reinforcement learning agent interacts with its environment in discrete time steps. At each time t, the agent receives an observation $o_t$, which typically includes the reward $r_t$. It then chooses an action $a_t$ from the set of available actions, which is subsequently sent to the environment. The environment moves to a new state $s_{t+1}$

and the reward $r_{t+1}$ associated with the transition $(s_t, a_t, s_{t+1})$ is determined. The goal of a reinforcement learning agent is to collect as much reward as possible. The agent can choose any action as a function of the history either randomly or greedily. Reinforcement learning requires clever exploration mechanisms. Randomly selecting actions, without reference to an estimated probability distribution, shows poor performance. The case of finite Markov decision processes are relatively well understood. However, due to the lack of algorithms that scale well with the number of states, simple exploration methods are the most practical.

One such method is $\epsilon$-greedy i.e when the agent chooses the action that it believes has the best long-term effect with probability 1-$\epsilon$ . If no action which satisfies this condition is found, the agent chooses an action uniformly at random. Here, $0 < \epsilon < 1$ is a tuning parameter, which is sometimes changed, either according to a fixed schedule (making the agent explore progressively less), or adaptively based on heuristics.

### 2.3.1 Model-free v.s. Model-based

The model stands for the simulation of the dynamics of the environment. That is, the model learns the transition probability T(s1|(s0, a)) from the pair of current state s0 and action a to the next state s1. If the transition probability is successfully learned, the agent will know how likely to enter a specific state given current state and action. However, model-based algorithms become impractical as the state space and action space grows. On the other hand, model-free algorithms rely on trial-and-error to update its knowledge. As a result, it does not require space to store all the combination of states and actions.

On-policy v.s. Off-policy- An on-policy agent learns the value based on its current action a derived from the current policy, whereas its off-policy counter part learns it based on the action a* obtained from another policy. In Q-learning, such policy is called the greedy policy.

### 2.3.2 Algorithms

Q-Learning - an off-policy, model-free RL algorithm based on the well-known Bellman Equation: $v(s) = E[R_{t+1} + \lambda v(S_{t+1})|S_t = s]$. Here, E refers to the expectation, while $\lambda$ refers to the discount factor. The goal is to maximize the Q-value. The idea of Q Learning is based entirely on value iteration. But to make it clear, value iteration updates all Q values every time, that is, all states and actions. But in fact, in reality, we can't traverse all the states, and all the actions, we can only get a limited series of samples. Therefore, only limited samples can be used for operation. Q-Learning proposes a way to update the Q value:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \lambda \max_a Q(S_{t+1}, a) - Q(S_t, A_t)) \tag{2.2}$$

Although the target Q value is calculated according to the value iteration, this Q value is not directly assigned to the new Q, but the progressive method is similar to the gradient descent, which is a small step toward the target, depending on $\alpha$, this can reduce the impact of estimation errors. Similar to a random gradient drop, it can finally converge to the optimal Q value.

Policy iteration & Policy Gradient - runs an loop between policy evaluation and policy improvement.Policy evaluation estimates the value function V with the greedy policy obtained from the last policy improvement. Policy improvement, on the other hand, updates the policy with the action that maximizes V for each of the state. The update equations are based on Bellman Equation. It keeps iterating till convergence. The goal of reinforcement learning is to find an optimal behavior strategy for the agent to obtain optimal rewards. The policy gradient methods target at modeling and optimizing the policy directly. The policy is usually modeled with a parameterized function respect to $\theta, \pi_\theta(a|s)$. The value of the reward (objective) function depends on this policy and then various algorithms can be applied to optimize $\theta$ for getting the best reward.

It is natural to expect policy-based methods are more useful in the continuous space.

Because there is an infinite number of actions and (or) states to estimate the values for and hence value-based approaches are way too expensive computationally in the continuous space. For example, in generalized policy iteration, the policy improvement step $argmax_{a \epsilon Q} \pi(s, a)$ requires a full scan of the action space, suffering from the curse of dimensionality.

Value Iteration - contains one component. It updates the value function V based on the Optimal Bellman Equation. After the iteration converges, the optimal policy is straight-forwardly derived by applying an argument-max function for all of the states.Model-based algorithm suffers from scalability problem.

SARSA - resembles Q-learning. The key difference between SARSA and Q-learning is that SARSA is an on-policy algorithm. It implies that SARSA learns the Q-value based on the action performed by the current policy instead of the greedy policy.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{2.3}$$

The action $a_{t+1}$ is the action performed in the next state $s_{t+1}$ under current policy.

## 2.4 Related Work

[1] has proposed an advancement in Seq2Seq models using Reinforcement Learning to get diverse coherent responses that could sustain conversations. The authors have designed appropriate reward functions in his RL system to get diverse responses and has overcome some of the challenges in Seq2Seq models by achieving diverse quality responses in his conversational agents. [2] has proposed a way of using external rewards in existing model as a way of fine-tuning the model to get desired results.here, the authors show an approach to effectively solve complex RL tasks without access to the reward function, including Atari games and simulated robot locomotion, while providing feedback on less than one percent of our agent's interactions with the environment. [4] incorporates affective content in neural models by using VAD lexicon [3] as a way to produce emotionally rich, natural, diverse and interesting responses. Here, the authors use Word2vec [7] with VAD appended embeddings with affective

objective functions to achieve affective neural response generation. .[13] have proposed Emotional Chatting Machine (ECM) that can generate appropriate responses not only in content (relevant and grammatical) but also in emotion (emotionally consistent). ECM addresses the factor using three new mechanisms that respectively (1) models the high-level abstraction of emotion expressions by embedding emotion categories, (2) captures the change of implicit internal emotion states, and (3) uses explicit emotion expressions with an external emotion vocabulary. Experiments show that the proposed model can generate responses appropriate not only in content but also in emotion. [14] proposes a general method for improving the structure and quality of sequences generated by a recurrent neural network (RNN), while maintaining information originally learned from data, as well as sample diversity. An RNN is first pre-trained on data using maximum likelihood estimation (MLE), and the probability distribution over the next token in the sequence learned by this model is treated as a prior policy. Another RNN is then trained using reinforcement learning (RL) to generate higher-quality outputs that account for domain-specific incentives while retaining proximity to the prior policy of the MLE RNN. [15] proposed using adversarial training for open-domain dialogue generation. Here, the system is trained to produce sequences that are indistinguishable from human-generated dialogue utterances. They cast the task as a reinforcement learning (RL) problem where they jointly train two systems, a generative model to produce response sequences, and a discriminator,analagous to the human evaluator in the Turing test to distinguish between the human-generated dialogues and the machine-generated ones. The outputs from the discriminator are then used as rewards for the generative model, pushing the system to generate dialogues that mostly resemble human dialogues. [16] provides the first survey of computational models of emotion in reinforcement learning (RL) agents. The survey focuses on agent/robot emotions, and mostly ignores human user emotions. Emotions are recognized as functional in decision-making by influencing

motivation and action selection. Therefore, computational emotion models are usually grounded in the agent's decision making architecture, of which RL is an important subclass. This survey provides background on emotion theory and RL. It systematically addresses 1) from what underlying dimensions (e.g., homeostasis, appraisal) emotions can be derived and how these can be modelled in RL-agents, 2) what types of emotions have been derived from these dimensions, and 3) how these emotions may either influence the learning efficiency of the agent or be useful as social signals. [17] proposed using four basic emotions: joy, sadness, fear, and anger to influence a Qlearning agent. Simulations show that the proposed affective agent requires lesser number of steps to find the optimal path. Here, an imporatnt observation is that the ratio between exploration to exploitation gradually decreases, indicating lower total step count in the long run when the affective agent finds its optimal path. [18] suggested that the traditional objective function, i.e., the likelihood of output (response) given input (message) is unsuited to response generation tasks. Instead they propose using Maximum Mutual Information (MMI) as the objective function in neural models. Experimental results demonstrate that the proposed MMI models produce more diverse, interesting, and appropriate responses, yielding substantive gains in BLEU scores on two conversational datasets and in human evaluations.[19] investigate the use of emotional information in the learning process of autonomous agents. Inspired by four dimensions that are commonly postulated by appraisal theories of emotions, we construct a set of reward features to guide the learning process and behaviour of a reinforcement learning (RL) agent that inhabits an environment of which it has only limited perception. They optimise the relative contributions of each reward feature and the resulting "emotional" RL agents perform better than standard goal-oriented agents, particularly in consideration of their inherent perceptual limitations.

# CHAPTER 3: ARCHITECTURE

My model uses Seq2Seq Encoder-Decoder network that is fine-tuned with Reinforcement Learning System with internal reward functions to promote ease of answering, semantic coherence [1] along with emotional intelligence [4] and external reward from human feedback to generate emotionally rich responses similar to human beings.

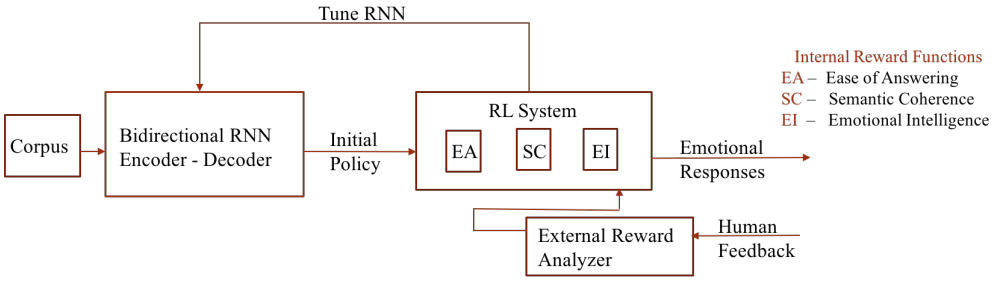


Figure 3.1: Overall Architecture

## 3.1 Dataset & Pre-processing

I have used two different datasets and have created two different models. The Cornell Movie Dialog corpus [20] and the Yelp Restaurant Review dataset. The model developed using Cornell Movie Dialog corpus has all the components except external reward part. The model developed using the second dataset Yelp Restaurant review has this external reward component along with the other internal rewards.

The Cornell Movie-Dialog corpus [20] contains a large metadata-rich collection of fictional conversations extracted from raw movie scripts. There are 220,579 conversational exchanges between 10,292 pairs of movie characters involving 9,035 characters from 617 movies. There are 304,713 utterances in total.

I have made use of two files namely "movie_lines.txt" and "movie_conversations.txt". The "movie_lines.txt" file contains the actual text of utterance and has the following

fields:lineID, characterID (who uttered the phrase), movieID, character name, text of utterance.

The "movie_conversations.txt" has the following fields: characterID of the first character involved in the conversation, characterID of the second character involved in the conversation, movieID of the movie in which the conversation occurred, list of the utterances that make the conversation, in chronological order i.e list of lineID's.

We have to match these lineID's in "movie_conversations.txt" with the "movie_lines.txt" file to get the actual utterance.

All conversations are preprocessed and arranged in actual chronological order matching the lineID's from "movie_conversations.txt" file. After drawing the dialog exchanges from the corpus, we lowercase the text, expand contractions, compress duplicate end punctuation to one symbol (i.e. "!!!!" - "!"), and remove metadata such as XML tags and HTML entities. We then tokenize the text with the SpaCy Python package. We remove approximately 6,000 exchanges for testing and use the rest for training and validation. We take the most common 12,000 words from the training and validation sets as our vocabulary as in [4], and replace any other tokens in these sets with an unknown symbol <UNK>. We partition the training and validation sets such that none of the responses in the training set have <UNK>. This effectively prevents the model from generating the unknown token during inference. This provides us with approximately 160,000 exchanges for training and 40,000 for validation.

The word count is set to maximum threshold of 20. The conversation length is in the range of 2-20.Then, two dictionaries are created that maps words to index and index to word, so that they can used in embedding lookup while building the model.

The yelp restaurant review dataset [21]consits of a list of json files in which we use "yelp_academic_dataset_review.json" file. This file is then converted into csv format for easy reading and understanding of data and its features. The dataset has 5996996 data points. This dataset contains nine features as follows: business_id, cool, date,

funny, review_id, stars, text, useful, user_id.

We would be making use of two features namely "useful" and "text".The "text" is divided into actual input and target. The task of the conversational agent here, is to complete the restaurant review given the initial prompt. The average number of sentences in each review is approximately 6. The word count has maximum threshold of 15. The number of words in a review ranges from 2-70.The input and target are divided into 3 sentences each approximately. Again, we expand contractions, compress duplicate end punctuation to one symbol (i.e. "!!!!"- "!"), and remove metadata such as XML tags and HTML entities. We then tokenize the text with the SpaCy Python package. We partition the training and validation sets.This provides us with approximately 4249026 reviews for training and 1979010 for testing. The "useful" score is used while training the external reward analyzer and while training the reinforcement learning model to make the agent generate responses in sync with the initial prompt.

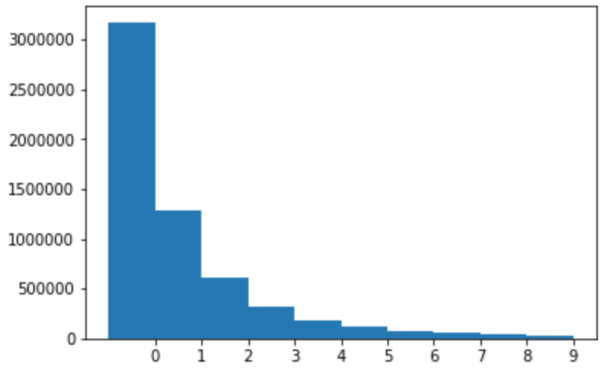


Figure 3.2: Yelp-Useful score distribution

Based on the frequency of the scores we can categorize them into two buckets. I have removed all the records that has no useful score as we are unsure about the usefulness of the review. I have a threshold of Useful score as 59 below which is categorized as "Not Useful" and the other class is "Useful". The external reward analyzer is trained with these two labels.

## 3.2 Word Embeddings

Word embeddings are a type of word representation that allows words with similar meaning to have a similar representation. Word embeddings are in fact a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. Each word is mapped to one vector and we use python's gensim package to create our own word2vec[7] embedding. There are two different techniques to create word embeddings. They are continuous bag of words(CBOW) and skip gram techniques.We have used skip gram technique to create word2vec embedding. This skip gram technique is a shallow neural network with a single hidden layer which works by predicting a word given the context.

When the feature vector assigned to a word cannot be used to accurately predict that word's context, the components of the vector are adjusted. Each word's context in the corpus is the teacher sending error signals back to adjust the feature vector. The vectors of words judged similar by their context are nudged closer together by adjusting the numbers in the vector.

We have also made use of Warriner's lexicon [3] that has valence, arousal and dominance score for each word. The Word2vec [7] model has each word with 1024 dimension. We append Valence(V), Arousal(A) and Dominance(D) score to each word in the word2vec vector and thus our new word2vec VAD appended model has 1027 dimensions for each word. These, words are first directly matched against the lexicon and if not found we consider the lemma of each word to be matched against our dictionary to assign more VAD scores. We use Spacy's lemmatizer to obtain lemma for each word in the corpus. If the lemma also doesn't match we append a neutral vector [5,1,5] similar to [4].Thus, we have our own word2vec-vad embedding that is fed as input to our Bidirectional RNN Encoder-Decoder Seq2Seq model and it is also used in our RL system to model our Emotional Intelligence heuristic.

## 3.3 RL Tuner System

We have used a Bidirectional RNN Encoder-Decoder Seq2Seq model [6] with "bahdanau" [5] attention mechanism. Attention is an extension to the encoder-decoder model that improves the performance of the approach on longer sequences. In Bahdanau mechanism [5], the decoder is given a way to "pay attention" to parts of the input, rather than relying on a single vector. For every step the decoder can select a different part of the input sentence to consider. Attention is calculated with another feedforward layer in the decoder. This layer will use the current input and hidden state to create a new vector, which is the same size as the input sequence (in practice, a fixed maximum length). This vector is processed through softmax to create attention weights, which are multiplied by the encoders' outputs to create a new context vector, which is then used to predict the next output. Then, we also use a greedy decoder in decoding phase to generate the best response at every stage of decoding during decoder training and inference phases. Thus, we fine tune the basic seq2seq generative model with the internal and external rewards in our RL tuner system to generate more interesting, diverse and emotionally appropriate responses.

The standard objective function for Seq2Seq models is the log-likelihood of target T given source S, given as follows:

$$T = argmax_T logp(T|S) \tag{3.1}$$

This formulation leads to generic responses being generated, since it only selects for targets given sources, not the converse. So, we replace this with Maximum Mutual Information (MMI) as the objective function [18]. In MMI, parameters are chosen to maximize (pairwise) mutual information between the source S and the target T:

$$\frac{logp(S,T)}{p(S)p(T)} \tag{3.2}$$

This avoids favoring responses that unconditionally enjoy high probability, and instead biases towards those responses that are specific to the given input.The MMI

objective can written as follows:

$$T = argmax_T logp(T|S) - \lambda logp(T) \tag{3.3}$$

By adjusting the value of $\lambda$ in the above equation we could get a reasonable number of diverse responses. But still we have considerable amount of dull responses in our initial seq2seq model for the given prompt. These responses also lack emotion and are ungrammatical.

In an attempt to overcome the above issues, we have modelled reinforcement learning system with appropriate heuristics. The generated sentences from the Seq2Seq model can be viewed as actions that are taken by the policy defined by the encoder-decoder language model. The parameters of this encoder-decoder network are fine tuned using RL system [1] with policy gradient methods. The components in our RL System are:

Action (a) - dialog utterances to generate i.e action a = gen(S), where gen(S) is the sequence generated by RNN LSTM. The action space is infinite and the sequences of varying length can be generated.

State (S) - dialog is transformed to a vector representation by feeding the current dialog(state) for which the response has to be generated.

Policy - policy takes the form $p_{RL}(p_{i+1}|p_i)$ where $p_{i+1}$ is the dialog to be generated for the given dialog $p_i$. Here, we use a stochastic distribution to represent policy as it is the probability distribution over actions given states, where both state and actions are dialogs. It is difficult to optimize deterministic policy as it leads to discontinuous objective and cannot be further used with gradient-based methods.

Reward (r) - The rewards are designed to overcome the issues we have addressed so far like dull , ungrammatical and unemotional responses. There are three internal rewards and one external reward that our RL system uses to overcome the issues in Seq2Seq architecture[6]. The three intern rewards used in my RL system are Ease of Answering $r_{EA}$, Semantic Coherence $r_{SC}$, Emotional Intelligence $r_{EI}$ [1]and one external reward [2] from human feedback $r_{HF}$.

Ease of Answering(EA) - Here, ease of answering is measured as negative log likelihood of generating a dull response to a dialog. We manually compose a list of 10 dull responses that frequently occur in the Seq2Seq model and penalize them as the model generates those responses. The 10 responses are:

1. I don't know.
2. I don't know what I mean
3. I don't know what you're talking about.
4. You don't know
5. You know what I mean.
6. You know what I'm saying.
7. You don't know anything.
8. I am not sure.
9. I know what you mean.
10. I do not know anything.

Let set S represent a list of dull responses. Then, the reward function can be defined as follows:

$$r_{EA} = -\frac{1}{N_S}\Sigma\frac{1}{N_S}logp_{seq2seq(s|a)} \tag{3.4}$$

$p_{seq2seq}$ represents likelihood output of Seq2Seq model. The RL system is likely to penalize utterances in the above composed list and hence less likely to generate dull responses.$r_{EA}$ is scaled by the length of the target S. Semantic Coherence(SC) - to avoid situations in which the generated responses are highly rewarded but are ungrammatical and not coherent. We consider the mutual information between the action a and the given input to ensure that the responses are coherent and appropriate. This also involves reverse training of the model where we count on the probability of the input prompt given the current generated response.

$$r_{SC} = \frac{1}{N_y}logp_{seq2seq}(y|x_i) + \frac{1}{N_{x_i}}logp_{backward-seq2seq}(x_i|y) \tag{3.5}$$

Emotional Intelligence (EI) - This is incorporated by minimizing affective dissonance

between the prompts and the responses. In a way this approach tries to maintain affective consistency between input and generated response. This heuristic is based on the fact that open-domain textual conversations between humans are affectively similar to each other. So, a happy prompt elicits a happy response and provocation usually results in anger and frustration.Thus, we make an assumption that the affective tone does not fluctuate often in general and we focus on minimizing the dissonance in affective tone between the input prompt and the generated responses.Thus, we would have responses that are emotionally similar to the given input prompts.

$$r_{EI_i} = \lambda p(a) \| \Sigma_{j=1}^{n} \frac{W2AV(x_j)}{|X|} - \Sigma_{k=1}^{i} \frac{W2AV(y_k)}{i} \| \tag{3.6}$$

Here, W2AV in equation 3.6 denotes the word-affect vector of the given sequence and the term $\Sigma_{j=1}^{n} \frac{W2AV(x_j)}{|X|}$ denotes average affect vector of the input prompt and $\Sigma_{k=1}^{i} \frac{W2AV(y_k)}{i}$ denotes average affect vector of the generated response up to the current step i.

In other words, we penalize the distance between the average affective embeddings of the input and the generated sentences.

The Cornell corpus [20] containing the conversations of movie characters are preprocessed and given as input to the RNN.The input is reversed when given as input to the RNN so as to have a good prediction in the decoding phase. The final reward is the weighted sum of the internal rewards alone for the model using Cornell movie dialog corpora. Thus, for cornell corpus,the final reward function just uses the internal reward components of RL system and can be written as follows:

$$r_{Final_Cornell} = \lambda_1 r_{EA} + \lambda_2 r_{SC} + \lambda_3 r_{EI} \tag{3.7}$$

The hyper-parameters for cornell movie dialog corpus 3.1 and yelp restaurant review dataset 3.2 are batch size that denotes the number of sequences in each training batch, gradient clip that squashes or truncates the gradient to avoid exploding gradient issue, learning rate that decides the speed at which the model learns, epochs denote the number of iterations to train the model before testing, dropouts prevent overshooting

and the weights for the rewards are experimented to find the best ones. I have also adopted early stopping technique to prevent the model from continuing to train once the loss becomes constant and there is not much fluctuation. We use Human

Table 3.1: BidirectionalRNN+RL model hyperparameters - Cornell

| **Hyperparameters** | **Value** |
|---|---|
| Batch Size | 128 |
| Gradient Clip | 1.0 |
| Learning Rate | 0.01 |
| Decay Rate | 0.0095 |
| Epochs | 50 |
| Dropout | 0.2 |
| LSTM layers | 2 |
| Encoder RNN Size | 1027 |
| Decoder RNN Size | 1027 |
| $r_{EA}\lambda_1$ | 0.25 |
| $r_{SC}\lambda_2$ | 0.35 |
| $r_{EI}\lambda_3$ | 0.40 |

Feedback for the Yelp restaurant review dataset [21] as it has this feedback in the form of "Useful" Score. The actual dataset is converted into a format suitable for input to our RNN model. The review is divided into input and target response. The first three sentences in each review forms the input and the remaining sentences would be the actual response. Here, the task is to develop a conversational agent that could complete the review given an initial review.

Human Feedback (HF) - The external reward analyzer is pretrained using the Yelp restaurant review dataset. The external reward analyzer is trained with the reviews from the Yelp dataset as the input feature and useful score as the class. Every review falls into two main classes "Useful" denoted by 1 and "Not Useful" denoted by 0. Then, we train a svm classifier which takes as input the "review" of the restaurant and classifies it into either 0 or 1. We have removed all the reviews with the useful score of "0" as it might create noise since, its unsure whether the given review is useful or not.

We use an AlphaGo-style strategy by initializing RL system with general response

policy learnt from our seq2seq model, which is then tuned using policy gradient method. The internal rewards are Ease of Answering, Semantic Coherence [1]and Emotional Intelligence, incorporated by minimizing affective dissonance [4] using the valence, arousal and dominance scores for each word in the response. We use a two-part training scheme, where we train a separate external reward analyzer using svm to predict the reward using human feedback in yelp restaurant review dataset [21] and then use RL system on predicted rewards to maximize the expected rewards (both internal and external). As the response is generated, we use the svm classifier to detect if the generated response is useful or not useful. The svm classifier has an accuracy of 92.35%. The svm classifier has been trained to determine the differences between the two classes "Useful" and "Not Useful". Then, we compare the output of svm on the generated response with the actual class of the input prompt and give the reward accordingly. The class of the generated responses are determined using the svm classifier and we give an absolute reward value of "-10" for generating responses with wrong class that does not match the actual target and "+5" for responses that are generated in sync with the actual target class. In this way, we try to generate responses that are real and humane and also that appears in-line with the context. This synthetic feedback [2] from the external reward analyzer is provided throughout the training phase and a greedy decoder is then used to generate the best response. The final reward function, is the weighted sum of both internal and external rewards for the yelp dataset and can be written as follows:

$$r_{Final_Yelp} = \lambda_1 r_{EA} + \lambda_2 r_{SC} + \lambda_3 r_{EI} + \lambda_4 r_{HF} \tag{3.8}$$

The novelty of our approach lies in the addition of Emotional Intelligence as an internal reward function and combining both internal and external rewards to create an emotionally appropriate model that closely imitates human. Also, the usage of external rewards to generate more sensible and human-like responses is something novel in natural language generation task.Our approach also works for two different tasks,

Table 3.2: BidirectionalRNN+RL model hyperparameters - Yelp

| **Hyperparameters** | **Value** |
|---|---|
| Batch Size | 512 |
| Gradient Clip | 1.0 |
| Learning Rate | 0.15 |
| Decay Rate | 0.01 |
| Epochs | 75 |
| Dropout | 0.3 |
| LSTM layers | 2 |
| Encoder RNN Size | 1027 |
| Decoder RNN Size | 1027 |
| $r_{EA}\lambda_1$ | 0.25 |
| $r_{SC}\lambda_2$ | 0.25 |
| $r_{EI}\lambda_3$ | 0.25 |
| $r_{HF}\lambda_4$ | 0.25 |

one for conversational agent and the other for yelp restaurant review completion.

# CHAPTER 4: EVALUATION & RESULTS

We have used commonly used metrics like BLEU score, Perplexity and ROUGE. As these measures are inappropriate, we are also using human evaluation to evaluate the responses. We would report the metric scores and responses of our models and would compare the seq2seq model with seq2seq+RL model for the two datasets. We would also show the evaluation survey created to facilitate human evaluation of our model in this section.

BLEU - The Bilingual Evaluation Understudy Score, or BLEU is a metric for evaluating a generated sentence to a reference sentence. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.

The score was developed for evaluating the predictions made by automatic machine translation systems. It is not perfect, but does offer compelling benefits:

1. It is quick and inexpensive to calculate.
2. It has been widely adopted.
3. It is language independent.
4. It is easy to understand.

The approach works by counting matching n-grams in the candidate translation to n-grams in the reference text, where 1-gram or unigram would be each token and a bigram comparison would be each word pair. The comparison is made regardless of word order. We use Python Natural Language Toolkit library, or NLTK, that provides an implementation of the BLEU score to evaluate our generated response against a actual target. We use sentence_bleu for calculating this metric against a single response in Cornell corpus and corpus_bleu in NLTK to compute BLEU score against multiple sentences in yelp restaurant review dataset.

Perplexity - defines how a probability model or probability distribution can be used to predict a text. A low perplexity indicates the probability distribution is good at predicting the sample. I have used the perplexity function in NLTK module to compute perplexity for both the cornell and yelp restaurant review datasets.

$$perplexity(text) = 2 * *cross - entropy(text) \tag{4.1}$$

ROUGE - stands for Recall-Oriented Understudy for Gisting Evaluation. It is essentially a set of metrics for evaluating automatic summarization of texts as well as machine translation. It works by comparing generated response against the actual response. We particularly use ROUGE-L, that measures longest matching sequence of words using LCS. An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, we don't need a predefined n-gram length (source: http://research.microsoft.com/en-us/um/people/cyl/download/papers/ rouge-working-note-v1.3.1.pdf)

$$R_{lcs} = LCS(X, Y)/m \tag{4.2}$$

$$P_{lcs} = LCS(X, Y)/n \tag{4.3}$$

$$F_{lcs} = ((1 + beta^2) * R_{lcs} * P_{lcs})/(R_{lcs} + (beta^2) * P_{lcs}) \tag{4.4}$$

## 4.1 Cornell - Model Evaluation & Results

We have two models for the Cornell corpus [20]. The first model is a basic seq2seq model with MMI objective function [18], that does not use RL system. The second model is using seq2seq to choose initial policy and fine-tunes that model to generate more diverse responses. A good model is said to have high bleu score and low perplexity. But these measures are inaccurate as the model could do well though the generated response is not close to the target. For instance, we can convey the same information with different words in a phrase.These scores do not capture this subtle difference and hence they fail to provide a good evaluation of a model. The loss is 4.89 for the model that does not use RL.Though they are inaccurate, we report these

scores as they are the standard metrics used in any language modelling tasks.

Table 4.1: Seq2Seq model evaluation - Cornell

| **Metric Scores** | **without RL - Value** | **with RL - Value** |
|---|---|---|
| BLEU Score | 0.15 | 0.38 |
| Perplexity | 98.96 | 76.65 |
| ROUGE-L | 0.39 | 0.55 |

The table 4.1 gives the evaluation scores for the seq2seq model for cornell corpus. The scores show that the generated response is not similar to the actual target response. Thus, the reported scores tells that the performance of the model is not good enough with low bleu score and high perplexity. I have shown 10 sentences from my test set in this report for comparison between the two models with/without using RL.These 10 sentences are chosen in a way to cover different responses from the two models and forms a good base for comparison between them. Now, let us see the responses from the seq2seq model without using reinforcement learning system for cornell corpus.

The table 4.2 shows that the model generates dull responses like "I do not know" and also single worded responses like "Yes". We could also see that the model hesitates to generate or automate long sentences and ends up in repetition like "I am sorry.I am sorry. I am sorry". Though, the mutual information objective tries to give diverse responses other than the dull ones like "I do not want to hurt you", we could see that the responses are dominated by dull and short sequences.

Next, we compute the evaluation metrics for the seq2seq model using RL technique and tabulate the results obtained.

The table 4.1 gives the evaluation scores for the seq2seq model with RL for cornell corpus. The scores show that the generated responses are not similar to the actual target response similar to the seq2seq model . Thus, the reported scores tells that the performance of the model is not good enough with low bleu score and high perplexity though we could see some improvement in score when compared to the seq2seq model

Table 4.2: Generated Responses without RL- Cornell

| **Prompt** | **Generated Response** |
|---|---|
| have fun tonight ? | Yes |
| it is just the strain again , the stress . we will take a vacation . | i do not know . |
| i must take you to an emergency ward sometime - on a saturday night . that is also fun . i came to see selma , as a matter of fact . | i am sorry. |
| do not tell me what to do ! we are divorced - i am a free woman . you are not my husband and you are not my boss ! and what is more , you are not going to be my boss . | i do not understand . |
| i am just on vacation . | i am sorry . i am sorry . i am sorry . |
| yes . but look at all the fun you had . mind if i try ? | no. |
| do not worry . take it . it will free your head . come on . | i do not want to hurt you . |
| it can not be a coincidence . the woman that owns it has a son . she says he is been on vacation in montana for the last ten days . we are trying to locate him . there 's also a cleaning service that comes in once a week . we are checking that out too . | i know . |
| well , you should - if you look good , you feel good and if you feel good , you project joy into the world . | i do not know . |
| i said i would call you the minute i got free . and i did - and here we are . | i do not know what you are doing . |

without using RL technique.The loss is 3.76 which seems to be a improvement over the technique that does not use RL.

We use the same set of 10 sentences to evaluate our seq2seq model with RL system. We would compare the difference in response between the two models and try to identify the aspects in which they differ and try to reason out the cause for the difference observed in the performance of two models.

From table 4.3 we could see a clear improvement in the response generation with the seq2seq model that uses RL technique. We could see diverse responses like "I appreciate that", "I'm good you are here" and so on. We could see that the dull responses are eliminated as we have penalized the dull responses in our RL system. We could see that the outputs are semantically coherent to the input prompt in most of the

Table 4.3: Generated Responses with RL- Cornell

| **Prompt** | **Generated Response** |
|---|---|
| have fun tonight ? | Yes, you have |
| it is just the strain again , the stress . we will take a vacation . | I appreciate that. |
| i must take you to an emergency ward sometime - on a saturday night . that is also fun . i came to see selma , as a matter of fact . | Never been taken before. |
| do not tell me what to do ! we are divorced - i am a free woman . you are not my husband and you are not my boss ! and what is more , you are not going to be my boss . | Why do you say that? |
| i am just on vacation . | I am not. Are you there. I've caught in the city. |
| yes . but look at all the fun you had . mind if i try ? | I have to get off my cut. Would you take them with me for a good drink. Good. |
| do not worry . take it . it will free your head . come on . | okay . |
| it can not be a coincidence . the woman that owns it has a son . she says he is been on vacation in montana for the last ten days . we are trying to locate him . there 's also a cleaning service that comes in once a week . we are checking that out too . | Jesus. |
| well , you should - if you look good , you feel good and if you feel good , you project joy into the world . | I was back into that church next next -. |
| i said i would call you the minute i got free . and i did - and here we are . | I'm good glad you're here. |

cases. We could see that the RL system has a tendency to end up with questions too like "Why do you say that?". We also notice that sometimes the system starts an irrelevant topic like "I was back into the church..." for an inappropriate prompt. This can be attributed to the lack of dialog history and insufficient amount of conversations from different characters and we are training our model with conversation from different characters that can lead to off-topic response generation occasionally. Overall, we could see that we have eliminated dull responses, added emotional content as responses are emotionally appropriate to the prompt as we minimize affective disso-

nance between the input and the response and get diverse responses though we cannot eliminate ungrammatical responses completely. These ungrammatical responses can also be attributed to the difference in different conversation from different characters in the movie and insufficiency of conversation history.

## 4.2 Yelp - Model Evaluation & Results

We have two models for the Yelp corpus. The first model is a basic seq2seq model with MMI objective function, that does not use RL system. The second model uses seq2seq to choose initial policy and fine-tunes that model to generate more diverse responses. The loss for the system that does not use RL is 9.89.Here, the RL system also uses external feedback by means of useful score in Yelp dataset.

Table 4.4: Seq2Seq model evaluation - Yelp

| **Metric Scores** | **without RL - Value** | **with RL - Value** |
|---|---|---|
| Bleu Score | 0.014 | 0.21 |
| Perplexity | 99.04 | 85.34 |
| ROUGE-L | 0.24 | 0.32 |

The table 4.4 gives the evaluation scores for the seq2seq model for yelp corpus [21]. The scores again emphasize that the generated responses are not similar to the actual target responses. Thus, the reported scores tells that the performance of the model is not good enough with low bleu score and high perplexity. These results are in parallel with the cornell corpus. I have shown few reviews from my test set in the report for comparison between the two models with/without using RL.These reviews are chosen in a way to cover different responses from the two models and form a good base for comparison between them. The conversational agent's task is to complete each review in the test set. Now, let us see the review completion from the seq2seq model without using reinforcement learning system for Yelp corpus.

The table 4.5 shows that the model generates dull responses like "I do not know what

Table 4.5: Generated Responses without RL- Yelp

| **Prompt** | **Generated Response** |
|---|---|
| This has become one of our favorite sushi bars when we visit Vegas. They have been consistent with the freshness and quality of their sushi. We have tried their albacore, maguro, unagi, spicy tuna roll, salmon – and they have never disappointed our taste buds. | It was good. It was good. I don't know |
| I went here with my wife, daughter, son-in-law, and three young grandchildren. This place is excellent for family dining. It was early-dinner time, we had no reservation, and they sat us right away. | i do not know what I mean . |
| the worst experience i have had in my whole life - took my grandson out for his graduation. ordered a martini that came with ice (watered down) martini's dont have ice unless requested, complained to waitress who told bartender, bartender wouldn't do anything, called over the manager who made it himself - here's the kicker, they charged me for 2 martinis (never drank the first) then my meal of ribs and chicken came and the ribs were burnt, showed the manager, he said oh they are burnt and never replaced them. they ruined a happy occasion and i truly hope you print this so they don't ruin other people's special days! | not good i don't think so. |
| Love the margherita pizza, especially during happy hour when it is discounted. Just wish they didn't discontinue their kale and cranberry salad. That was the best! | nice to taste and good food to eat. |
| The service was not what I expected. The girl got our orders mixed up, and I had to go find her to get our beer order that she forgot. For the first time we tried the boneless wings. | i was ok. it was nice. |

i mean" and does not generate long sentences. The loss is 5.89 which seems to be an improvement over the non RL model.We could see that the diversity is limited along with the limited emotional content.

Next, we compute the evaluation metrics for the seq2seq model using RL technique and tabulate the results obtained.

The table 4.4 gives the evaluation scores for the seq2seq model with RL for yelp corpus. The scores show that the generated responses are not similar to the actual target responses similar to the seq2seq model . Thus, the reported scores tells that the performance of the model is not good enough with low bleu score and high perplexity

though we could see some improvement in score when compared to the seq2seq model without using RL technique. This again mirrors the same situation as with cornell corpus. We use the same set of reviews to evaluate our seq2seq model with RL system. Let us examine the responses generated by this model. From table 4.6 we could see a clear improvement in the response generation with the seq2seq model that uses RL technique. We could see diverse responses. We could also see that the outputs are semantically coherent to the input prompt in most of the cases. Overall, we could see that we have eliminated dull responses , added emotional content as responses are emotionally appropriate to the prompt as we minimize affective dissonance between the input and the response and get diverse responses that could complete the given initial review(input) though we cannot eliminate ungrammatical responses completely.

## 4.3 Human Evaluation of Models

We have prepared the survey for human evaluation through Amazon's mTurk. The screenshots show how our survey looks and gives an idea about the type of questions we plan to post in the mTurk Survey.

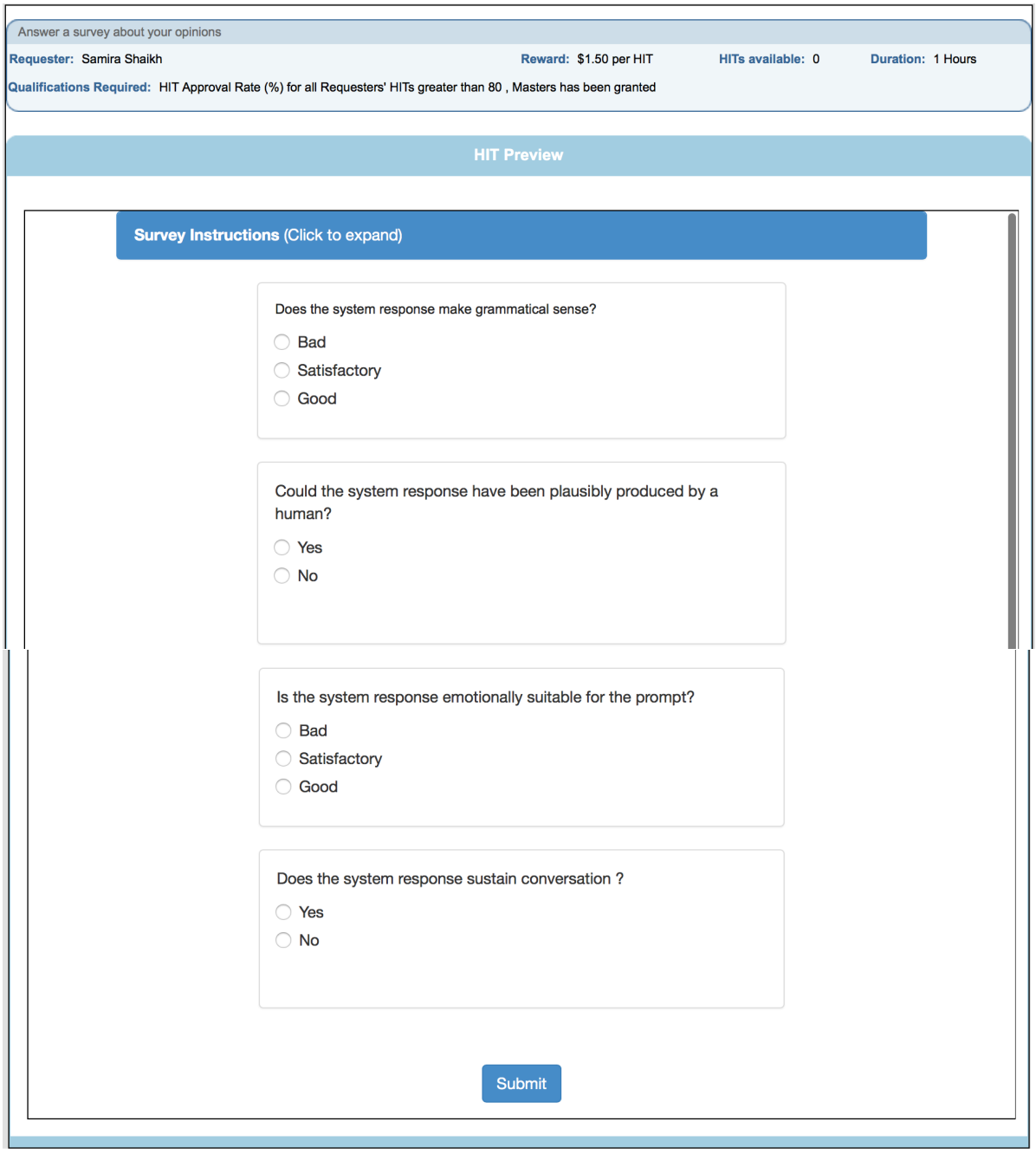


Figure 4.1: mTurk Sample Survey

We have also collected responses through google form surveys Figure 4.2 for cornell and Figure 4.3 for yelp. We have tabulated the measures for syntactic coherence, emotional appropriateness and naturalness for cornell Table 4.7 & yelp restaurant review Table 4.8 datasets.The score ranges from 0-2 where 0 - bad, 1 - satisfactory

and 2 - good.We have considered the average of the scores for each of these metrics in our evaluation so that each measure can have a maximum score of 2.

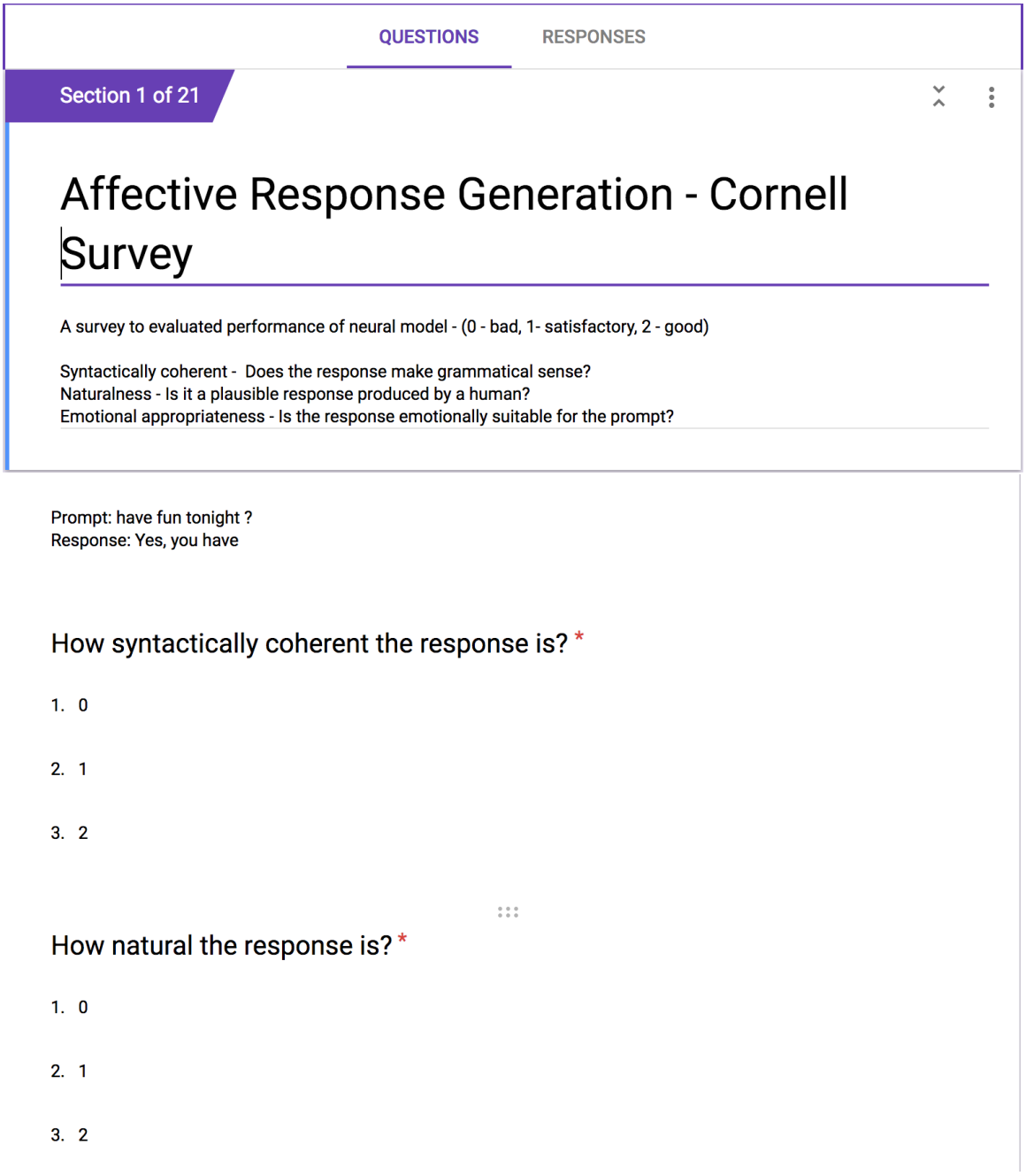


Figure 4.2: Cornell Sample Survey

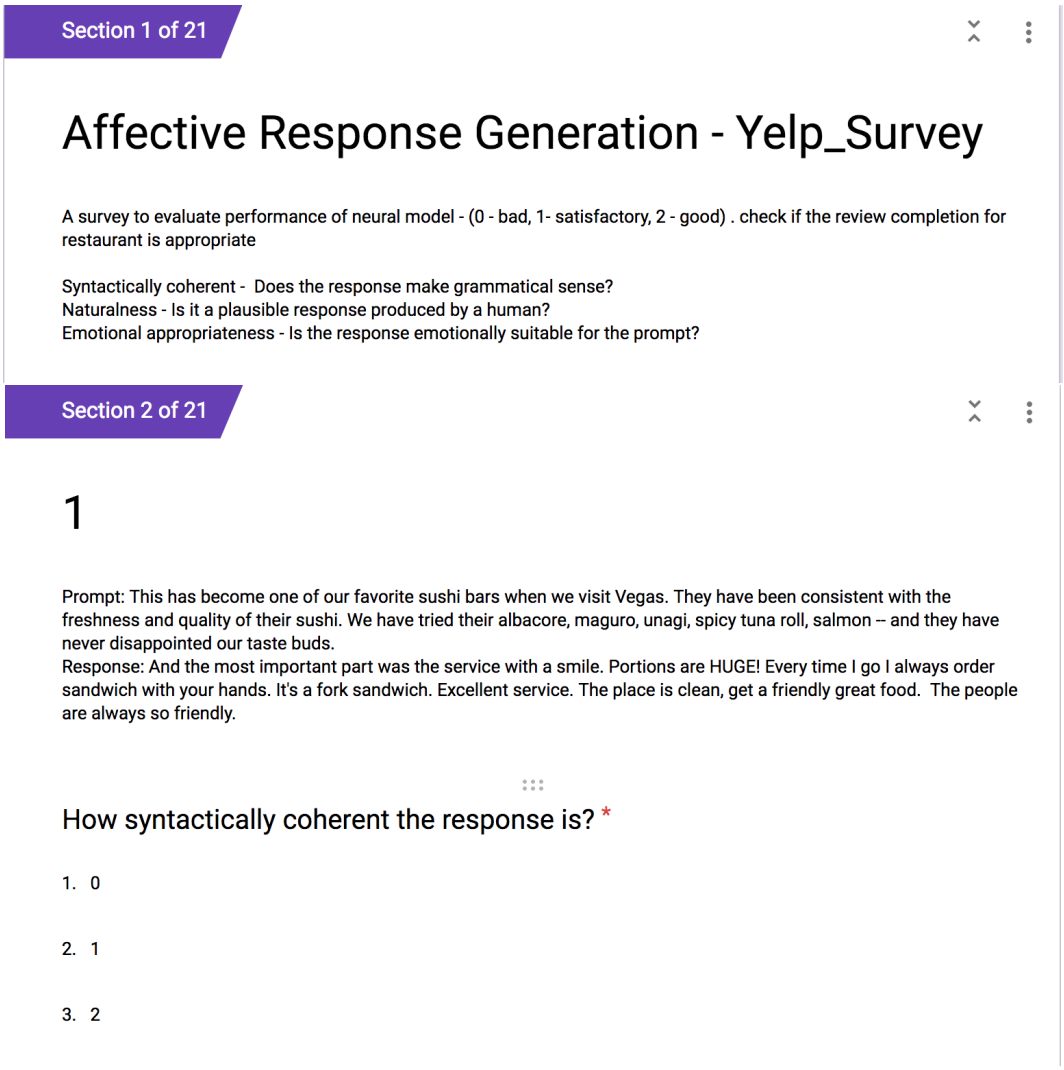


Figure 4.3: Yelp Sample Survey

The figure 4.1 shows a sample survey that we created using Amazon Sandbox application. We would also include questions by portraying our test set responses for our two final models using Ccrnell and yelp datasets and compare the performance of the two models separately.

Table 4.7: Human Evaluation Metrics- Cornell

| **Syntactic Coherence** | **Natural** | **Emotional Appro.** |
|---|---|---|
| 1.45 | 1.42 | 1.44 |

We can see from Table 4.7 for cornell corpus that Syntactic coherence is 1.45, Natural

score that captures how natural the response is to the given prompt is 1.42 and the emotional score that captures the emotional appropriateness of the text to the given prompt is 1.44.

Table 4.8: Human Evaluation Metrics- Yelp

| **Syntactic Coherence** | **Natural** | **Emotional Appro.** |
|---|---|---|
| 1.46 | 1.52 | 1.73 |

We can see from Table 4.8 for yelp restaurant review dataset that Syntactic coherence is 1.46, Natural score that captures how natural the response is to the given prompt is 1.52 and the emotional score that captures the emotional appropriateness of the text to the given prompt is 1.73. In the yelp dataset we could see that the measures are high when compared to the cornell dataset and the emotional score is comparatively good approaching 2.

We could see that the scores in both the cornell and yelp datasets are good.Thus, we could conclude that our model could automate longer sentences capturing coherence, natural flow and emotional content. The cornell corpus has a good overall score on human evaluation so as the review completion in the yelp dataset. Our model outperforms the baseline model [4] in terms of high emotional appropriateness score and it also outperforms the model [1] in terms of diverse and long response generation with emotional appropriateness.

Table 4.6: Generated Responses with RL- Yelp

| **Prompt** | **Generated Response** |
|---|---|
| This has become one of our favorite sushi bars when we visit Vegas. They have been consistent with the freshness and quality of their sushi. We have tried their albacore, maguro, unagi, spicy tuna roll, salmon – and they have never disappointed our taste buds. | And the most important part was the service with a smile. Portions are HUGE! Every time I go I always order sandwich with your hands. It's a fork sandwich. Excellent service. The place is clean, get a friendly great food. The people are always so friendly. |
| I went here with my wife, daughter, son-in-law, and three young grandchildren. This place is excellent for family dining. It was early-dinner time, we had no reservation, and they sat us right away. | fresh and the staff is very friendly. We wish they had seating and it looks like an outside bar as well that's open probably on the busier days.Another favorite of mine. I have enjoyed meals in many and this one is no exception. |
| the worst experience i have had in my whole life - took my grandson out for his graduation. ordered a martini that came with ice (watered down) martini's dont have ice unless requested, complained to waitress who told bartender, bartender wouldn't do anything, called over the manager who made it himself - here's the kicker, they charged me for 2 martinis (never drank the first) then my meal of ribs and chicken came and the ribs were burnt, showed the manager, he said oh they are burnt and never replaced them. they ruined a happy occasion and i truly hope you print this so they don't ruin other people's special days! | Never should go. bad service and food was not good. poor experience and won't go back. |
| Love the margherita pizza, especially during happy hour when it is discounted. Just wish they didn't discontinue their kale cranberry salad. That was the best! | We are regulars, both in the restaurant. I would highly recommend this place.This place is awesome! I had the chicken shwarma and couldn't move for the rest of the day. I advise to eat half at a time (:The food was delicious |
| The service was not what I expected. The girl got our orders mixed up, and I had to go find her to get our beer order that she forgot. For the first time we tried the boneless wings. | Always sit at the bar and placed ourselves in Matt's hands. He made drinks and beers on draft. I was little disappointed with service! |

# CHAPTER 5: CONCLUSION & FUTURE WORK

We could see from the Evaluation & Result section, that the Seq2Seq model with MMI function could give limited diverse responses. Using RL to fine-tune the model produces good results with some diversity and emotional appropriateness for both the Cornell[20] and Yelp [21] datasets. Moreover, the metrics like BLEU, perplexity and ROUGE-L are inaccurate measures of how well the model performs. We have created templates in mTurk to start with our human evaluation procedure. The survey that we collected through google forms shows that our model performs well and could incorporate the proposed emotional content in the conversations. In future, we can try to use different heuristics like maximizing affective dissonance/content as emotional intelligence heuristic in our RL reward system and see the difference in the performance. Here, we have used useful score in the Yelp restaurant review dataset as external feedback. In future, we can try to give more powerful human feedbacks (manual annotation to distinguish human and machine generated ones) that can help the model to generate responses more close to human being. Through this work, we could see the potential of RL in generating diverse responses in language generation task and we could experiment with more powerful heuristics and external feedbacks to exploit its full potential in generating interesting conversations and we could try applying the same technique to similar NLG tasks like Machine Translation, Text Summarization etc to see if the same architecture can improve the performance of those models.

# REFERENCES


[1] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep reinforcement learning for dialogue generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1192–1202, Association for Computational Linguistics, 2016.

[2] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 4299–4307, Curran Associates, Inc., 2017.

[3] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 english lemmas," *Behavior Research Methods*, vol. 45, pp. 1191–1207, Dec 2013.

[4] N. Asghar, P. Poupart, J. Hoey, X. Jiang, and L. Mou, "Affective neural response generation," in *European Conference on Information Retrieval*, pp. 154–166, Springer, 2018.

[5] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[6] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), pp. 3111–3119, Curran Associates, Inc., 2013.

[8] D. Britz, "Deep learning for chatbots." http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction.

[9] H. Chen, X. Liu, D. Yin, and J. Tang, "A survey on dialogue systems: Recent advances and new frontiers," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 2, pp. 25–35, 2017.

[10] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318, Association for Computational Linguistics, 2002.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] A. Deshpande, "How i used deep learning to train a chatbot to talk like me." https://adeshpande3.github.io/adeshpande3.github.io/How-I-Used-Deep-Learning-to-Train-a-Chatbot-to-Talk-Like-Me.

[13] H. Zhou, M. Huang, T. Zhang, X. Zhu, and B. Liu, "Emotional chatting machine: Emotional conversation generation with internal and external memory," *arXiv preprint arXiv:1704.01074*, 2017.

[14] N. Jaques, S. Gu, D. Bahdanau, J. M. H. Lobato, R. E. Turner, and D. Eck, "Tuning recurrent neural networks with reinforcement learning," 2017.

[15] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2157–2169, Association for Computational Linguistics, 2017.

[16] T. M. Moerland, J. Broekens, and C. M. Jonker, "Emotion in reinforcement learning agents and robots: a survey," *Machine Learning*, vol. 107, pp. 443–480, Feb 2018.

[17] W. Badoy and K. Teknomo, "Q-learning with basic emotions," *CoRR*, vol. abs/1609.01468, 2014.

[18] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan, "A diversity-promoting objective function for neural conversation models," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 110–119, Association for Computational Linguistics, 2016.

[19] P. Sequeira, F. S. Melo, and A. Paiva, "Learning by appraising: An emotion-based approach to intrinsic reward design," *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, vol. 22, pp. 330–349, Oct. 2014.

[20] C. Danescu-Niculescu-Mizil and L. Lee, "Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs.," in *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*, 2011.

[21] "Yelp restaurant review dataset." https://www.yelp.com/dataset.

# APPENDIX A: EXAMPLES OF YELP REVIEW COMPLETION

Prompt: I just returned from a 3-night stay at one of the book My Vacay properties in Scottsdale. While my friends and I had a wonderful vacation, the house was a bit of a disappointment. Pros:There was plenty of space, a nice kitchen and master bath, and the location was good.

Response: the orders were messed up and the service was delayed . I would not go to this place.

Prompt: Ahhhh this place is so good! My friend and I came in around 1 pm and it was busy! Luckily there was some tables available so we were seated quickly.

Response: The food was great and the service was spectacular! Entire staff was very friendly. The service was so personable and cashier woman so nice and they were very accommodating to large orders for offices !As good as it gets!

Prompt: When the Grand Lux in our mall in CO closed, we lost our favorite happy hour place. Now we fly to Vegas once a year to enjoy some Asian Nacho goodness and Potato Spring Rolls. We love being able to enjoy a date night dinner of delicious.

Response: Great staff, great portions , good wine! It's a must if and in Vegas for a week and came to this restaurant three times. The pad Thai is amazing! I had the steak kabob and it was tasty and juicy.

Prompt: This place seems AWESOME! Was in town visiting friends and we ran a 3.5 mile Xterra trail race. They weren't kidding when they said trail.

Response: It's tucked away, but not hard to get to.The food was delicious and fresh. prepared exactly the way I stopped here on a whim and had a fine lunch. A great burger and superb popovers.Great food and service

Prompt: Needed a fast bite to eat as I had to hit the hay early and catch a 6 am flight to Chicago. This place was near my hotel and I always hear people rave about it so I thought I would give it a shot. You walk in and there's a counter with a variety of fixins, you can get taco's, burritos, chips, guacamole, and a burrito bowl which is

like a burrito but the tortilla is on the side.

Response: This location is really good too. They have the best pizza on the weekends when Kids Eat FREE! If you have a larger family ( more than enough for 2 the board and sandwiches and I'm bummed I don't live here full-time, every time I come to Phoenix...

Prompt: Very friendly staff and a great range.Their prices may be a few dollars more than a box box store, but I like to support a local business and they definitely earn it in service.

Response: The limited menu helps them keep their focus on the freshness of the meal. This place is always busy and is a testiment to their product. Go. Eat. Enjoy. This place is awesome!!

Prompt: This place was awesome! The French dip with Au Ju is amazing! Great service, great location.

Response: Outside seating and a fine lunch. A great burger and superb popovers. Great food and service. This is the place to be. The chef actually is from Nobu.I had the $50 omakase (a course meal) and every dish was delicious.

Prompt: My food was very good. I can't say the same about my service. No silverware, no water, a couple of mishaps on orders.

Response: The staff is so nice!First time here. I had the best experience. super friendly and helpful.The food was very good and the service was excellent. I would eat here everyday if I could!he food was amazing!

Prompt: Had their pizza yesterday. And I'm excited that I get to have it for breakfast today! My co-worker & I are picky about our pizza pies, so had to toss up a review.

Response: I had the filet mignon with parsley garlic butter and it was delicious!!! Half off wine for happy hour was an added bonus - super impressed!Friendly and fast service. First time for my hubby and second for me.

Prompt: Stopped by this hole in the wall for lunch today and definitely enjoyed the

experience! I ordered the chicken hibachi lunch special (with white rice and miso soup) and gyoza while my partner ordered the chicken teriyaki bowl and spring rolls. Let me start by saying that I left absolutely stuffed for $12.49 plus the tip.

Response: We stopped here on a whim and had a fine lunch. Good food and great service.

Prompt: Me and my family have been going to this place for years to fulfill our sushi and soju fix. Sushi is made fresh from the bar and the prices are reasonable! The staff is very friendly and welcoming, the owner is always willing to sit down and talk with us!

Response: The yummy and flavorful sides. Good wine and beer selection.The food was very good and the service was excellent. I would eat here everyday if I could.