INFO7374 Assignment 4 - Sentiment Analysis

Papers summarization

Our team read four papers before research:

- A Deep Reinforcement Learning Chatbot (Short Version)
- Learning from Dialogue after Deployment: Feed Yourself, Chatbot!
- Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots
- Scalable Sentiment For Sequence-To-Sequence Chatbot Response With Performance Analysis

A Deep Reinforcement Learning Chatbot (Short Version)

1. Dataset

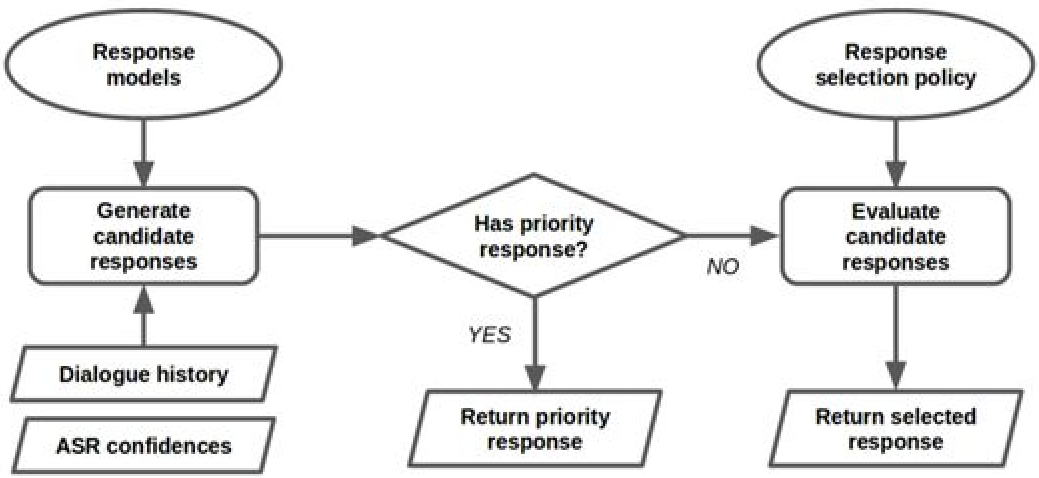
The training data are dialogue responses. There are 22 response models in the system, including neural network based retrieval models, neural network based generative models, knowledge base question answering systems and template-based systems.

2. Model used

a. Supervised MLN using data collected by Amazon Mechanical Turk
b. Reinforcement learning (Off-policy REINFORCE)
c. Linear regression
d. Markov decision process (Q-learning AMT)

3. Methodology

To generate a response, the chatbot follows a three-step procedure.
First, it uses response models to generate a set of responses.
Second, if there exists a priority response in the set of candidate responses, this response will be returned by the system.
Third, if there are no priority responses, the response is selected by the model selection policy.



4. Conclusion

The Q-learning AMT policy achieved an average Alexa user score substantially above the average of the all teams in the Amazon Alexa competition semi-finals. This strongly suggests that learning a policy through simulations in an Abstract Discourse MDP may serve as a fruitful path towards developing open-domain socialbots.
The performance of Off-policy REINFORCE suggests that optimizing the policy directly towards user scores also may serve as a fruitful path. In particular, Off-policy REINFORCE obtained a substantial increase in the average number of turns in the dialogue compared to the average of all teams in the semi-finals, suggesting that the resulting conversations are significantly more interactive and engaging.

Learning from Dialogue after Deployment: Feed Yourself, Chatbot!

1. Dataset

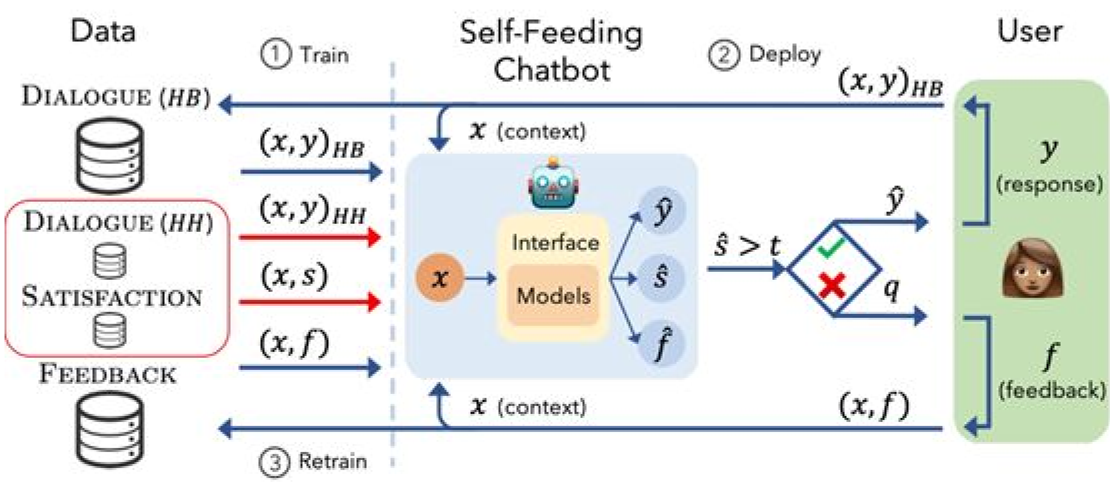
Dialogue dataset generated by the system in “A Deep Reinforcement Learning Chatbot”
Three new datasets to further research in this direction:
(1) deployment chatlogs (513k messages); (2) ratings of user satisfaction (42k); (3) textual feedback on what a bot could have said in a given context (62k)

2. Model used

Transformer architecture

3. Methodology

In the deployment phase, the agent engages in multi-turn conversations with users, extracting new deployment examples of two types. Each turn, the agent observes the context and uses it to predict its next utterance and its partner’s satisfaction. If the satisfaction score is above a specified threshold, the agent extracts a new Human-Bot DIALOGUE example using the previous context and the human’s response and continues the conversation. If, however, the user seems unsatisfied with its previous response, the agent requests feedback with a question, and the resulting feedback response is used to create a new example for the FEEDBACK task. The agent acknowledges receipt of the feedback and the conversation continues.



4. Conclusion

The chatbot has three tasks to achieve self-feeding:
DIALOGUE: carry on a coherent and engaging conversation with a speaking partner;
SATISFACTION: predict whether or not a speaking partner is satisfied with the quality of the current conversation;
FEEDBACK: predict the feedback that will be given by the speaking partner when the agent believes it has made a mistake and asks for help.

By deploying transformer architecture for the three tasks the self-feeding chatbot performs well on human experiment tests.

Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots

Data used:
1.Ubuntu Corpus.
2.Douban Conversation Corpus.
Methodology:
Given an utterance in a certain context, the model is supposed to find the best response among several candidates. The list of candidates is generated based on the top five important key words in the context.

Model used:
1.Sequential Matching Network

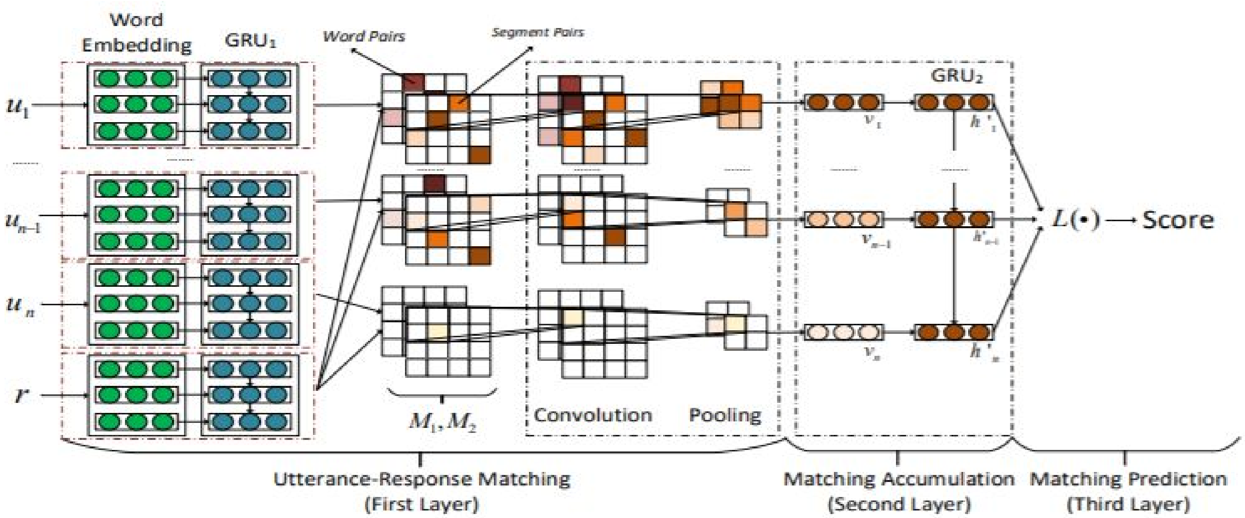


Figure 1: Architecture of SMN

2. Baseline models:
Including, TF-IDF, RNN, CNN, LSTM and BiLSTM, Multi-view, Deep learning to respond(DL2R), and Advanced single-turn matching models.
Key conclusions:
GRU is useful when analyzing corpus with context and even extract grammatical informations, which can be crucial in sentiment analysis.

Scalable Sentiment For Sequence-To-Sequence Chatbot Response With Performance Analysis

Data Used:

Twitter chatting corpus, https://github.com/Marsan-Ma/chat_corpus

Methodology:

Based on a response generator model, these models added sentiment analysis units with different math tricks before or after the base model. In some of the units there is some evaluating metrics that measure the coherence of generated responses. Such a metrics can be used for evaluating the model.

Models Used:

Seq2seq Model (as baseline), persona-based model, reinforcement learning, plug and play model, sentiment transformation network and cycleGAN.

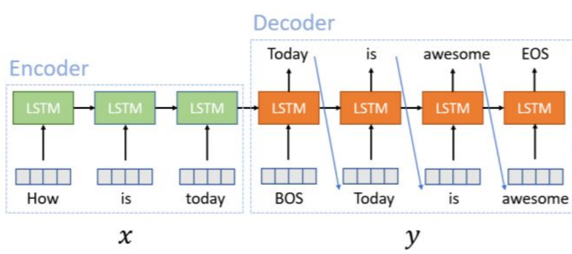


Fig. 1. *Seq2seq model.*

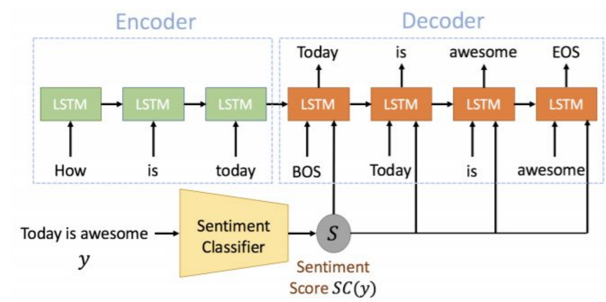


Fig. 2. *Persona-based Seq2seq model*

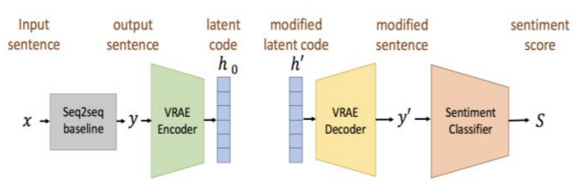


Fig. 3. *Plug and play model. VRAE denotes variational recurrent auto-encoder*

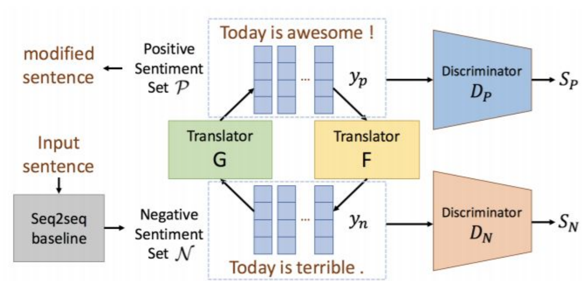


Fig. 4. *CycleGAN Model for sentiment transformation. F and G are*

Key Conclusions:

The reinforcement learning was able to learn properly the different design goals and offer output sentences with good diversity. The cycleGAN model primarily performed word mapping on the original response, so the output sentence quality was more or less preserved. The Plug and Play model and Sentiment Transformation Network were not as successful at the moment, probably because it is not easy to modify the latent code of the sentences while preserving the semantics and sentence quality.

Data Preprocessing

Dataset Preparation

The experiments introduced several datasets listed below with references:
https://github.com/Phylliida/Dialogue-Datasets/blob/master/TwitterLowerAsciiCorpus.txt
https://github.com/marsan-ma/chat_corpus/blob/master/movie_subtitles_en.txt.gz
https://github.com/marsan-ma/chat_corpus/blob/master/twitter_en.txt.gz
https://github.com/Phylliida/Dialogue-Datasets/blob/master/MovieCorpus.txt
TwitterLowerAsciiCorpus.txt

Data Cleansing

The data cleaning process is taking no more than regular steps in nlp processes. Besides, we cannot remove the english stop-words from the corpus for its is also carrying positional information and grammatical information for robots to make response.

1. Split the text by line and remove characters which are not words, digits or common punctuation

```
[17] import re
     tw = open('./TwitterLowerAsciiCorpus.txt')
     twitter = tw.read()
     data = [d for d in twitter.split('\n')]
     data = [d for d in data if d != '']
     #data = eval('[%s]'%repr(data).replace('[', '').replace(']', ''))
     data = list(map(lambda x:re.sub(r'^A-Za-z\d\s\,\.\!\?\'\"\+\-','',x), data))
     print(data[0:5])

["what's up dadyo when did you get back on twitter? haha", "like 2 weeks ag
```

2. Extend " 't " to pure words and add space before punctuation to make sure they can be extracted

```
l1 = ['won' t','won\'t','wouldn' t','wouldn\'t','' m', '' re', '' ve', '' ll', '' s','' d',
l2 = ['will not','will not','would not','would not',' am', ' are', ' have', ' will', ' is',

for i, raw_word in enumerate(data):
    for j, term in enumerate(l1):
        raw_word = raw_word.replace(term, l2[j])

    data[i] = raw_word.lower()
```

3. Add "STA" and "END" as beginning and ending signal for each sentence

```
data = list(map(lambda x:'STA '+x+' END', data))
context = data[::2]
answers = data[1::2]
all = context + answers
```

Word to Index

Tokenize the text and build the vocabulary dictionary.

```
all = ' '.join(all)
tokenized_all = all.split()
tokenized_context = [t.split() for t in context]
tokenized_answers = [t.split() for t in answers]

word_freq = nltk.FreqDist(itertools.chain(tokenized_all))
print ("Found %d unique words tokens." % len(word_freq.items()))
```

```
Found 11854 unique words tokens.
```

```
vocab = pickle.load(open(vocabulary_file, 'rb'))
index_to_word = [x[0] for x in vocab]
index_to_word.append(unknown_token)
word_to_index = dict([(w,i) for i,w in enumerate(index_to_word)])

print ("Using vocabulary of size %d." % vocabulary_size)
print ("The least frequent word in our vocabulary is '%s' and appeared %d times." % (vocab[-1][0], vocab[-1][1]))
```

```
Using vocabulary of size 10000.
The least frequent word in our vocabulary is 'mcdonalds' and appeared 1 times.
```

Generate word to index array

```
for i, sent in enumerate(tokenized_answers):
  tokenized_answers[i] = [w if w in word_to_index else unknown_token for w in sent]

for i, sent in enumerate(tokenized_context):
  tokenized_context[i] = [w if w in word_to_index else unknown_token for w in sent]

# Creating the training data:
X = np.asarray([[word_to_index[w] for w in sent] for sent in tokenized_context])
Y = np.asarray([[word_to_index[w] for w in sent] for sent in tokenized_answers])

Q = sequence.pad_sequences(X, maxlen=maxlen_input, padding='post')
A = sequence.pad_sequences(Y, maxlen=maxlen_output, padding='post')
```

Add Sentiment Weight

We introduced a sentiment analysis package to access straight-forward and convenient sentiment scores. Besides, the AWS sentiment analysis api also provides similar scores in addition.

```
[2]  !pip install vaderSentiment

     Requirement already satisfied: vaderSentiment in /usr/local/lib/python3.6/dist-packages (3.2.1)
```

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

def generate_sentiments(lst):
  analyzer = SentimentIntensityAnalyzer()
  sentiments = list()
  for sentence in lst:
    vs = analyzer.polarity_scores(sentence)
    sentiments.append(vs)
  return sentiments
```

```
[7]  print(generate_sentiments(data[0]))

     [{'neg': 0.0, 'neu': 0.812, 'pos': 0.188, 'compound': 0.4588}, {'neg': 0.08, 'neu': 0.625, 'pos': 0.295, 'compo
```

For implementing sentiment analysis to the chatbot, we introduce the sentiment score as weights multiplied by context array. In this case, we separately use ‘pos’ score and ‘neg’ score to train models with sensitivity to positive or negative input.

Modeling

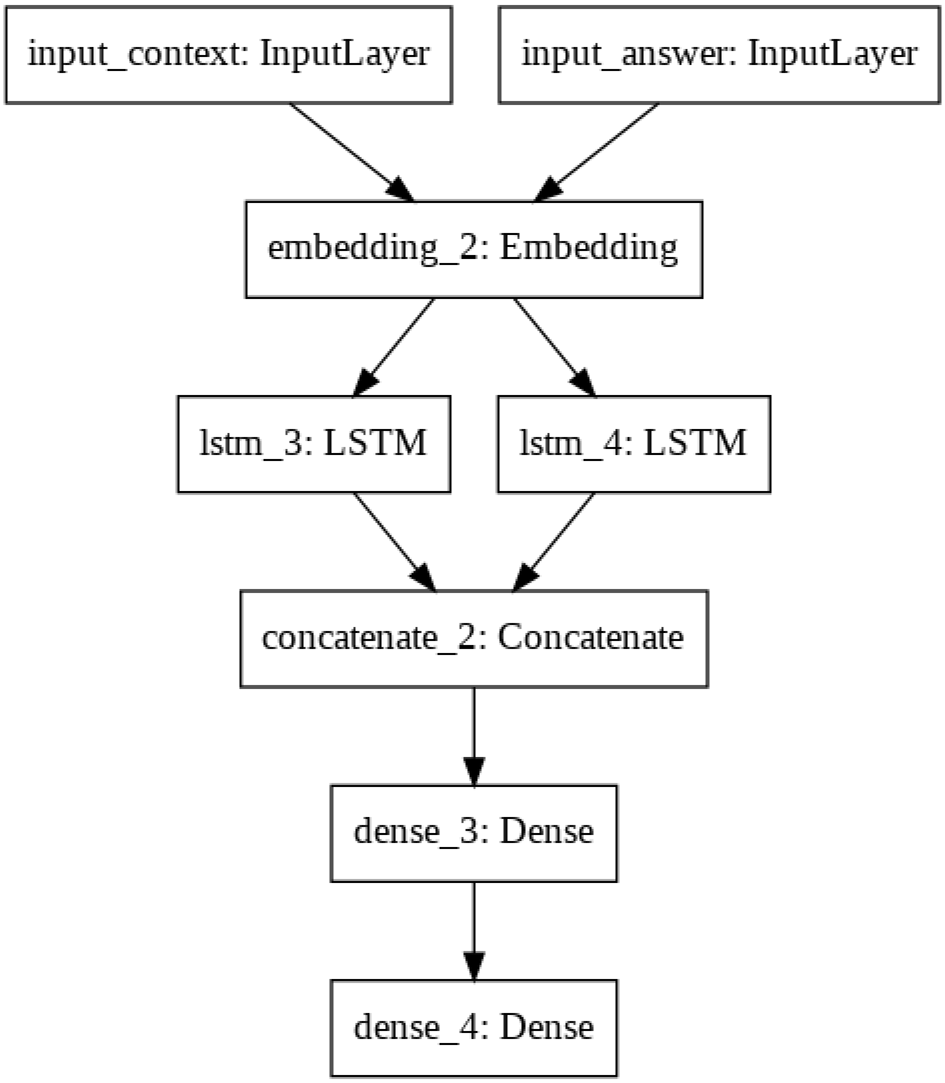
GLOVE Embedding

We use GLOVE embedding matrix to compress the word vector to 100d.

```
embeddings_index = {}
f = open('./glove.6B.zip_files/glove.6B.100d.txt')
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

print('Found %s word vectors.' % len(embeddings_index))
embedding_matrix = np.zeros((dictionary_size, word_embedding_size))

# Loading our vocabulary:
vocabulary = _pickle.load(open(vocabulary_file, 'rb'))

# Using the Glove embedding:
i = 0
for word in vocabulary:
    embedding_vector = embeddings_index.get(word[0])

    if embedding_vector is not None:
        # words not found in embedding index will be all-zeros.
        embedding_matrix[i] = embedding_vector
    i += 1

print(embedding_matrix)
```

The prototype chatbot with a seq2seq model

Seq2seq, as introduced above, is often used as a machine translation base model. It generates a sequence word by word. We use this model to generate plausible response against any inputs. The structure of our model is displayed below:


input_context: InputLayer
input_answer: InputLayer
embedding_2: Embedding
lstm_3: LSTM
lstm_4: LSTM
concatenate_2: Concatenate
dense_3: Dense
dense_4: Dense

```
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
input_context (InputLayer)       (None, 50)            0
____________________________________________________________________________________________________
input_answer (InputLayer)        (None, 50)            0
____________________________________________________________________________________________________
embedding_1 (Embedding)          (None, 50, 100)       1000000     input_context[0][0]
                                                                   input_answer[0][0]
____________________________________________________________________________________________________
lstm_1 (LSTM)                    (None, 300)           481200      embedding_1[0][0]
____________________________________________________________________________________________________
lstm_2 (LSTM)                    (None, 300)           481200      embedding_1[1][0]
____________________________________________________________________________________________________
concatenate_1 (Concatenate)      (None, 600)           0           lstm_1[0][0]
                                                                   lstm_2[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, 5000)          3005000     concatenate_1[0][0]
____________________________________________________________________________________________________
dense_2 (Dense)                  (None, 10000)         50010000    dense_1[0][0]
====================================================================================================
Total params: 54,977,400
Trainable params: 54,977,400
Non-trainable params: 0
```

We trained a model without sentiment weight for 100 epoches as reference.

The first 30 epoches output:

```
input:  and then that mouse had the nerve to try to eat our kibble !  let this be a lesson fur all the other mousies !
output: STA i am so glad for the game , i am ready to y awake . END END END END END END END END END END i have been in the same .
input:  tomorrow
output: STA i am not even fucking END END END END END END END END END END END END END END END END END END END END END END END END
input:  make sure i have a bed and seat saved next to you !
output: STA i am so happy for you END END END END END END END END END i am END END END END END END
input:  wassup shorty .
output: STA i am not even pissed END END END END END END END i am not even do not even get a quick of . END END END END END END
input:  appreciate that shorty , you too .
output: STA i am not rich END END END END END END END END END END END END END END
input:  yea
output: STA i am not a person . . . . . . END END , so , i will not have a fan . END END END END END END END END END END END END
input:  gotchu
output: STA i am not a person . . . . . . END END , so , i will not have a fan . END END END END END END END END END END END END
input:  good  wby
output: STA i am not even a sports fan . . . . . END END , so many of the random of my jaw , and he is so END END END END END END
input:  that is wassup
output: STA i am not even pissed END END END END END END END i am not even mad END END END END END END END END END i am not sure .
```

The final output:

```
input:  gotchu
output: STA i usually think teachers should be better and stuff but hes a piece of shit so idc END END END .
input:  good  wby
output: STA i hope the #indians win today in #ndingrock #nodapl :) END END END but i am not gonna y awake END
input:  that is wassup
output: STA well , i am not talking baseball . i am talking real indians in north dakota .people as mascots =
input:  and the dash for cash races too
output: STA imy END END END do not pee yourself END END END would be good lol END END END END END but not all
input:  do you think this wouldve happened anyway if it was another driver stinkin it up like kyle has recentl
output: STA i do not think there is any shame in being a better than a . . . END a END END END END END END
input:  what did these niggas say
output: STA haha oh damn that sucks . should have rode with us all these damn crazy END END END END END END
input:  kenny is wife is white on top of that . u not gonna tell me u put a flag over the color of your skin .
output: STA damn idk if i want that much . how much mut for it ? END END END END END END END END by 20 END
```

Conversation test:

```
CHAT:

computer: hi ! please type your name.

user: yi
computer: hi , yi ! My name is team5_chatbot.

user: there was a greasy kid at highline who was basically miles dipped in a bucket of ranch
computer:  i am just trying to convince you and wake up having in 7 minutes in the am . . shit do not expect me down  .    (with probability of 0.000001)
user: i killed a mouse today!
computer:  i am gonna call it in the first part of you know you expect you will ? and you can tell them you feel ?     (with probability of 0.000000)
user: so much to do before leaving for d.c. ahhh
computer:  i do not know it was either or so i can not get of the benefits of the week and all defense and keep all . all . all . it ? ?     (with probab
user: whatever, i'm gonna uncork this bottle and have me a glass of wine..
computer:  wyd you know me then lets me get out there ? is where you know not listen to see again ?     (with probability of 0.097071)
user: i don't always drink on tuesday nights/wednesday mornings, but when i do... it's when i just worked from 8am to 2am.
computer:  i am gonna do not even do it like that .but i really want to be much though or try to be perfect of us president not  .    (with probability of
user: it was so hard getting out of bed this morning
computer:  i am gonna do not even do it . like . i know you though , but they do not so much .     (with probability of 0.000009)
user: what is it to do on the tuesday
computer:  i am not gonna call it back there and the shit . . shit i thought you , i would have to wait .     (with probability of 0.000037)
user: bye
computer:  i really do not like this ? i do not get my laundry done #showerthoughts and i ai not no . pharma . getting . is .    (with probability of 0.0
user: exit
computer:  i really do not ? i do not know anymore . and so i can keep them pharma getting so  .    (with probability of 0.019276)
```

With more training epoch the model really made a progress.

A Not-even-sleepy Chatbot with sentiment analysis

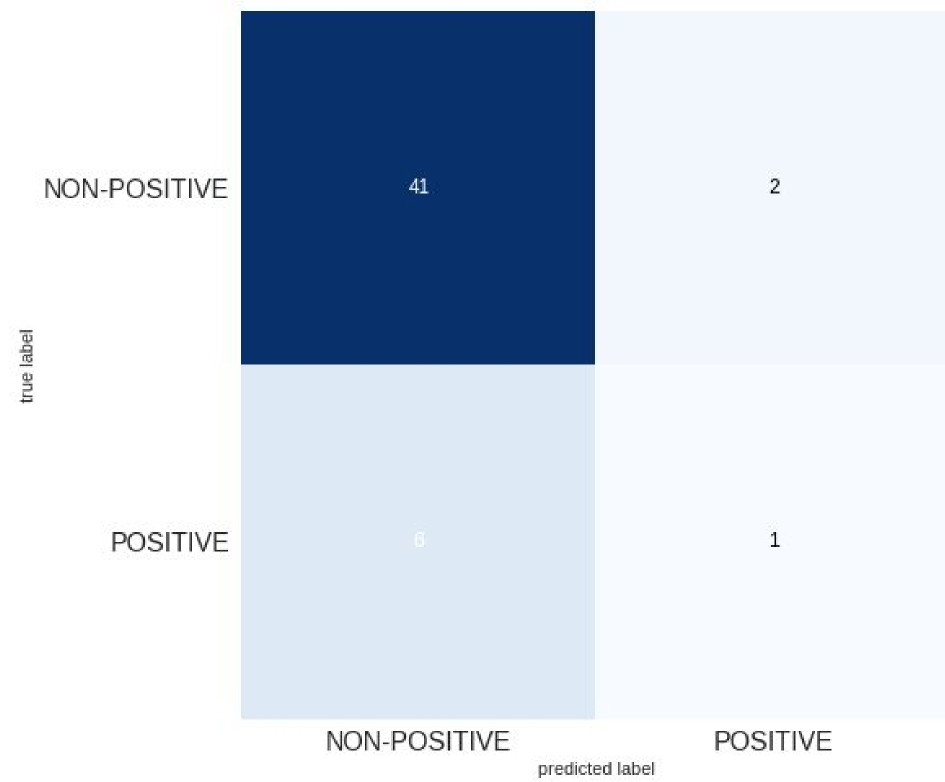
We use two API to gain sentiment scores. One is vaderSentiment, the other is AWS comprehend. We train three models by making the input context array multiplied by vaderSentiment positive score, negative score and AWS comprehend positive score.

vaderSentiment:

The model trained with positive weights input performs as below:

```
input:  what is up dadyo when did you get back on twitter ? haha
output:  i am not , i will not allow it . , there is not much to do at this time of the night .
sentiment: 0
original_output:  like 2 weeks ago and it is going as terribly as i remember , but deg is still hilarious so it is ok
original_sentiment: 0
input:  literally never about that account , love it .
output:  i am ready for them 7 12 is
sentiment: 0
original_output:  answer me this fellow apple peoples : how many times in the past year have you used the escape key ?
original_sentiment: 0
input:  about 50 times today . terminal vim user .
output:  i am not even the basketball fan but this stuff me off its like are they that slow
sentiment: 0
original_output:  seems the major complaints so far are from vim users like yourself . im wondering how force quit is gonna work .
original_sentiment: 0
input:  cmd+opt+esc is good but still available via menubar
output:  i am not sure yet , i might go
sentiment: 0
original_output:  there was a greasy kid at highline who was basically miles dipped in a bucket of ranch
original_sentiment: 0
```

The sentiment analysis confusion matrix:



We can repeat the whole procedure against the negative labeling scores, which means training a robot cares only about negative sentiments.
The outcome is alike.

AWS Comprehend:
Here is how one example of the AWS Comprehend sentiment analysis result looks like:

```
"1": {
        "Sentiment": "NEUTRAL",
        "SentimentScore": {
            "Positive": 0.030794490128755570,
            "Negative": 0.06013264134526253,
            "Neutral": 0.8995440602302551,
            "Mixed": 0.009528853930532932
        },
        "ResponseMetadata": {
            "RequestId": "e370949e-5a20-11e9-8980-03d6e543dc33",
            "HTTPStatusCode": 200,
            "HTTPHeaders": {
                "x-amzn-requestid": "e370949e-5a20-11e9-8980-03d6e543dc33",
                "content-type": "application/x-amz-json-1.1",
                "content-length": "162",
                "date": "Mon, 08 Apr 2019 17:08:10 GMT"
            },
            "RetryAttempts": 0
```

```
        }
    },
```

It gives us four scores and one final sentiment result as positive, negative, neutral or mixed; however, we only focused on positive socore. Therefore, we simplified the labels to two: positive and non-positive (negative, neutral, and mixed).

```python
def get_sentiment(text):
    sentiment = comprehend.detect_sentiment(Text=text, LanguageCode='en')['Sentiment']
    if(sentiment == 'POSITIVE'):
        sentiment = 1
    elif(sentiment == 'NEGATIVE' or 'NEUTRAL' or 'MIXED'):
        sentiment = 0
    return sentiment
```

Then we multiply the positive score with question word vector before training model:

```python
for i in range(row):
    Q[i,:] = Q[i,:]*context_sentiment[i]
```

Result:

We trained for 50 epoches and we starting to get some reasonable output:

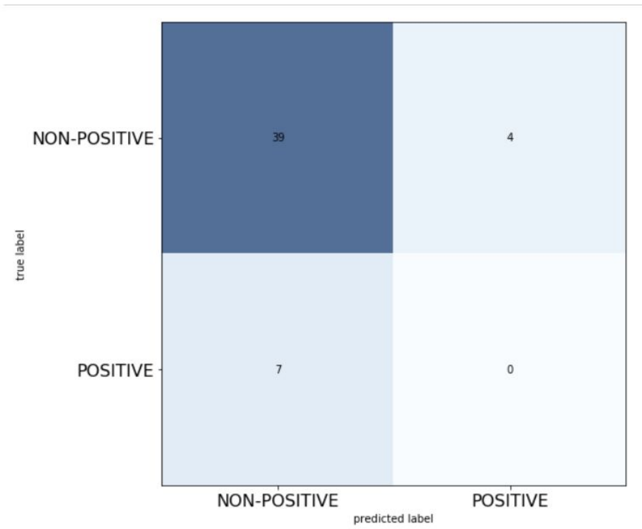
```
Training epoch: 48, training examples: 0 - 2628
Epoch 1/1
36967/36967 [==============================] - 86s 2ms/step - loss: 0.6047
STA i do not know what the mouse was thinking ? ? why come into a house with 2 ferocious felines ? ? END END END END END END
END END END END END END END END END . . . END neato END END END END END lol END
Training epoch: 48, training examples: 2628 - 5256
Epoch 1/1
38361/38361 [==============================] - 89s 2ms/step - loss: 0.6029
STA i am not feelin class tomorrow rn but i definitely will not be feeling it in the morning END END END END END END END END
END END END END END END is gonna be home END END END END END END END END END END . . .
Training epoch: 49, training examples: 0 - 2628
Epoch 1/1
36967/36967 [==============================] - 86s 2ms/step - loss: 0.5964
STA i do not know what the mouse was thinking ? ? why come into a house with 2 ferocious felines ? ? END END END END END END
END END END END END END END END . END . END . END END END END END END END lol
Training epoch: 49, training examples: 2628 - 5256
Epoch 1/1
38361/38361 [==============================] - 89s 2ms/step - loss: 0.5964
STA i am stuck between wanting to cry and rolling a blunt END END i know . i . END END END . END no . END END END END END END
END END END END END END END END END END . END END END END END
```

Here is the comparison with original output with the first 50 input and their sentiments:

```
input:  what is up dadyo when did you get back on twitter ? haha
output:  i do not want any coleslaw with my laynes either
sentiment: 0
original_output:  like 2 weeks ago and it is going as terribly as i remember , but deg is still hilarious so it is ok
original_sentiment: 0
input:  literally never about that account , love it .
output:  i am so busy rn tho d :
sentiment: 0
original_output:  answer me this fellow apple peoples : how many times in the past year have you used the escape key ?
original_sentiment: 0
input:  about 50 times today . terminal vim user .
output:  i am welfare for i am 72 on fixed income mo have not had a ss raise in 3 yrs but get lot
sentiment: 0
original_output:  seems the major complaints so far are from vim users like yourself . im wondering how force quit is gonna w
ork .
original_sentiment: 0
input:  cmd+opt+esc is good but still available via menubar
output:  i am welfare for i am 72 on fixed income mo have not had a ss raise in 3 yrs but get lot
sentiment: 0
original_output:  there was a greasy kid at highline who was basically miles dipped in a bucket of ranch
```

Confusion Matrix:

Finally, we drew a confusion matrix with prediction output sentiment label and original output sentiment label which are both gotten from AWS comprehend sentiment analysis:



As the result, most of non-positive answers are defined correctly, however, all positive original outputs are missing.

AWS Comprehend with relational questions and answers data:
We realized that our input data are not paired correctly, for example: some questions and answers are not related to one topic. So we reorganized the data and get rid off some non relational data and paired the related data together.
We have 4862 lines of Q&A and 11244 unique words tokens in total. Originally, there are 4890 lines and 11854 unique words tokens.

```
4862
4862
context: STA what is up dadyo when did you get back on twitter ? haha END
answer: STA like 2 weeks ago and it is going as terribly as i remember , but deg is still hilarious so it is ok END
context: STA answer me this fellow apple peoples : how many times in the past year have you used the escape key ? END
answer: STA about 50 times today . terminal vim user . END
context: STA seems the major complaints so far are from vim users like yourself . im wondering how force quit is gonna work . E
ND
answer: STA cmd+opt+esc is good but still available via menubar END
context: STA there was a greasy kid at highline who was basically miles dipped in a bucket of ranch END
answer: STA i am disgusted END
context: STA he flashed us then we scored so he sadly put his shirt back on #fuckhighline END
answer: STA what a piece of shit END
context: STA i killed a mouse today !   END
answer: STA yay , you great hunter . ive killed lots of lizards and bugs but never a mouse . END
context: STA i do not know what the mouse was thinking ? ?  why come into a house with 2 ferocious felines ? ? END
answer: STA and then that mouse had the nerve to try to eat our kibble !  let this be a lesson fur all the other mousies !   EN
D
context: STA what day are you coming to effie ? ? END
answer: STA tomorrow END
context: STA im leaving today once i get off work END
answer: STA make sure i have a bed and seat saved next to you ! END
context: STA avi  hey boo END
answer: STA wassup shorty .  END
```

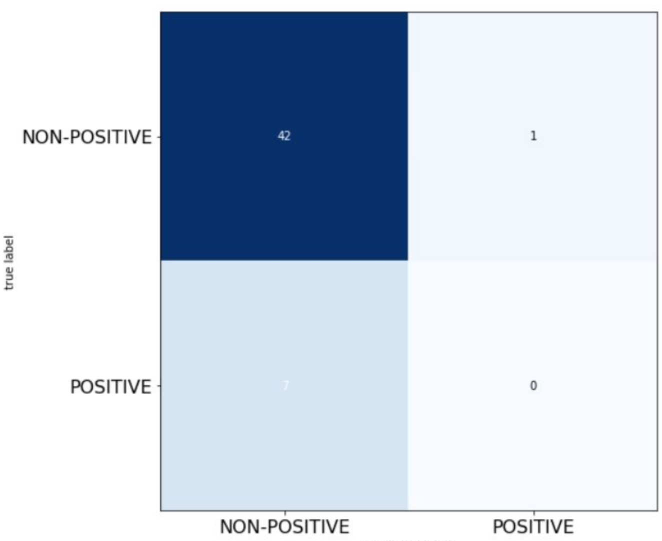
With all the same parameter as the AWS comprehend training above, here is the result after 50 epoches:

```
Training epoch: 48, training examples: 0 - 2431
Epoch 1/1
31559/31559 [==============================] - 45s 1ms/step - loss: 1.0347
STA i agree . . . do not have a little ? END ? END . we are not soft . END . END . END . END . END END END END . END END END
END END END END END END END END END END END END END
Training epoch: 48, training examples: 2431 - 4862
Epoch 1/1
32926/32926 [==============================] - 47s 1ms/step - loss: 1.0715
STA i am fine , i do not want to see anymore scenes with him &amp ; gabi &amp ; the fewer the team END END END END END END EN
D END END END END END END END END END END END END logic positive positive positive struggle END
Training epoch: 49, training examples: 0 - 2431
Epoch 1/1
31559/31559 [==============================] - 45s 1ms/step - loss: 1.0062
STA i am sayinggg but i am not going to see my to see . that . i can not want to see it END END END END END END END END END E
ND END END END END END END END info END END END card END END END END
Training epoch: 49, training examples: 2431 - 4862
Epoch 1/1
32926/32926 [==============================] - 48s 1ms/step - loss: 1.0428
STA i am wide awake cause i am at work , keeping the world safe &amp ; saving lives , lol END END END END END END END END END
END END END END END END END END END END END id everyone END END wont ima seems rim rim
```

The first 50 original input and output with prediction:

```
input:  what is up dadyo when did you get back on twitter ? haha
output:  i am sayinggg but i am , but i am in , but i am in to !
sentiment: 1
original_output:  like 2 weeks ago and it is going as terribly as i remember , but deg is still hilarious so it is ok
original_sentiment: 0
input:  answer me this fellow apple peoples : how many times in the past year have you used the escape key ?
output:  i am not feelin class tomorrow rn but i definitely will not be feeling if it is all
sentiment: 0
original_output:  about 50 times today . terminal vim user .
original_sentiment: 0
input:  seems the major complaints so far are from vim users like yourself . im wondering how force quit is gonna work .
output:  i dvred it . have not finished watching it yet . was at the point where he was interviewing the second guy .
sentiment: 0
original_output:  cmd+opt+esc is good but still available via menubar
original_sentiment: 0
input:  there was a greasy kid at highline who was basically miles dipped in a bucket of ranch
output:  i am wide awake cause i am at work , keeping the world safe &amp ; saving lives , lol
sentiment: 0
original_output:  i am disgusted
original_sentiment: 0
```

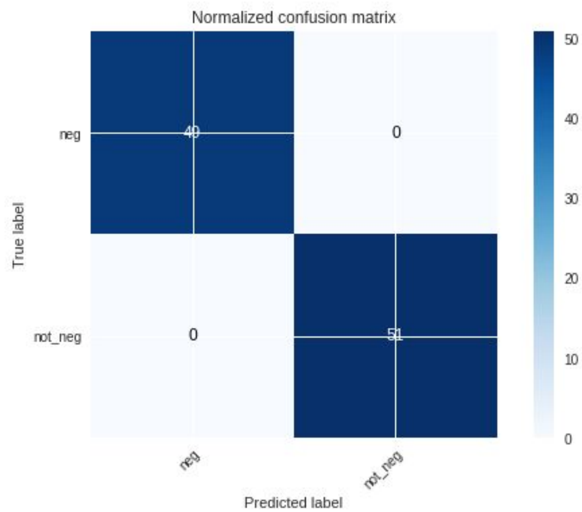
Confusion Matrix:

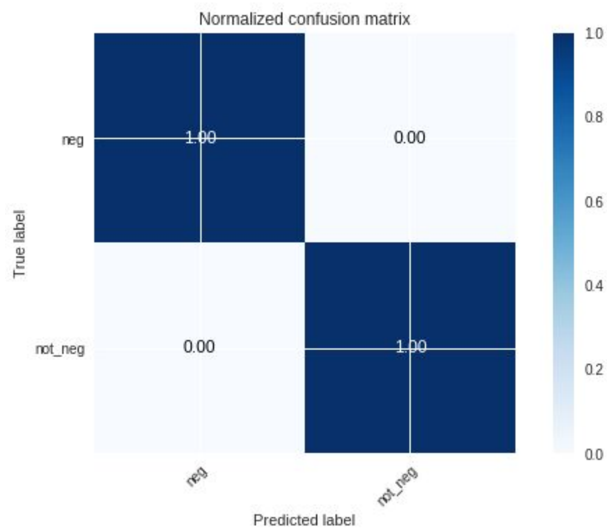


The result is similar with the first one, but only 3 more correct sentiments for non-positive output.

Evaluation and Finalizing

Human evaluation was the major criterion during the training process. We see the responses from chatbot against any user inputs, and evaluate if it is coherent or not. Besides, we introduced a confusion metric on sentiment labels to defined the magnitude of that how well a sentiment analysis will impact the responses from the chatbot. To see how well a robot senses a negative or positive feelings in the context, we define the labels again, by whether there is a score other than zero under a certain label, for example, ‘positive’. Rather than using the true label, such a set of labeling makes more sense when your input sentiment labels are merely a set of scores under a single kind of sentiment. By such a metrics, we can plot a confusion metrics like:





…...with a mean squared error on the sentiment score used for labeling: `0.039804280000000004`, which means the robot can accurately sense the positive or negative feelings expressed, and response with a proper sentiment.

However, when taking a closer exam into the responses, we find that all responses with filtered sentiment (either positive or negative) are forced into a single sentence. Take the negative responses for example:

```
input:   okay so i know we have had our differences on pineapples on pi
output:  i really do not
input:   what in the tits is a coquito
output:  i need to go and pick up my paycheck cause i been broke for th
input:   just riding these ol dirt roads .
output:  i am not a pretty asian girl like you that can just walk in
input:   k babe .
output:  i really do not
input:   hate how y'all try to call me "fake gay" no . . trust &amp ; b
output:  girl me too !
```

Such a result may suggest there can be a more flexible way to introduce sentiment factors to the model.

Conclusion

Based on the observation of this assignment, we find that the coherence of responses are related to the depth of training, whilst the plausibility of responses are based on the corpora fed into the model. As a new corpus introduced into the already weighted model, it may need to be trained as deep as a raw model to produce coherent sentences.
