NLP Project: Dialogue Simulator

Annelot Pruyn
i6170151

Barbara Futyma
i6170086

1 Introduction

For this project a natural language processing task had to be implemented. Various options were considered but finally a dialogue generator was decided upon. Hence, the goal set for this project was to create a dialogue generator which can create understandable and human-like dialogues. This topic was chosen because the dynamics of human speech are quite complex hence research in this topic seems quite interesting.

The dialogue simulator will produce a text word by word. When selecting the next word in a sentence, various strategies can be applied. The strategies implemented are ”Sampling” which chooses a next word according to its probability, ”Greedy” which always chooses the word with the highest probability and ”BeamSearch” which creates a shortlist of the most probable words and then recalculates the probability once a few more words have been added and chooses the optimal one. The question to be answered is which of these approaches generates the best dialogues according to the following criteria: correct grammar, understandability and sentence diversity.

1.1 Hypothesis

Greedy will perform well since it will always produce the most probable sentence hence it will predominantly create grammatically correct sentences, however Greedy will repeatedly produce the same phrases when given a specific sequence causing it to be repetitive and unoriginal. The sampling strategy will create more diverse sentences because it considers every possibility, however the bot’s grammar might be sacrificed. BeamSearch includes a deeper search and considers multiple possibilities without considering all of them, this will ensure correct grammar for the most part without getting repetitive. Hence BeamSearch will most likely perform the best.

2 Prior Literature

Alexander Vlasblom in a report ”Text prediction using n-grams” uses n-grams to predict the next word based on already entered text. The author firstly explains how the data was cleaned, tokenized and filtered. Then the n-gram tables are built, processed and split into the predictor and the predicted word - (n-1)gram and the final word. Then the maximum likelihood formula is used to compute the conditional probabilities. The simple backoff approach combined with weights is used to generate a list of next probable words.[3]

In his article, Neil Yager discusses the fundamentals of neural text generation. Yager firstly describes the need for a language model and touches upon two in particular. Firstly a simpler approach is mentioned: using n-grams as a language model, despite the simplicity of the model it can be surprisingly powerful. Further he discusses RNN’s (recurrent neural networks) such as LSTM (Long Short Term Memory) and GRU (Gated Recurrent Unit), which have the advantage that sequences of abritrary lengths can be used as input. But the main advantage is that RNN’s can learn long term dependencies, hence longer text can be generated without loss of context or grammatical consistency. Yager then goes on to introduce the three strategies of text generation which have also been implemented in this project: Greedy selection, Sampling and BeamSearch.[2]

Ziang Xie discusses different approaches to handling neural text generation in his paper but mostly immerses into the difficulties neural text generation brings along. Xie states that within text generation models, decoders can behave undesirably by generating generic responses, repetitive outputs or even struggling to produce grammatical sentences. [4]

3 Data

Multiple datasets were found which comply to the task, in the end the ”Movie Dialogue corpus” was chosen. This dataset contains conversations extracted from different movie scripts, keeps track of which character said what in which movies and further details of the movie like its title, genre and production year. However only text of the dialogs was used in the project. [1]

4 Model

For the language model a custom model was implemented which uses a combination of several n-grams. The model calculates five separate probabilities for each word i when given a sequence w of length 4 : $P(i|[w_0, w_1, w_2, w_3]), P(i|[w_1, w_2, w_3])$, $P(i|[w_2, w_3]), P(i|[w_3])$ and $P(i)$. Once it has calculated these values it takes a combination of them to determine the ’score’ of a word, which represents the likeliness of a word following the sequence w. Three approaches were implemented to select the next word.

4.1 Greedy selection

The greedy selection always chooses the word which has gained the highest score, hence the most probable word to occur in the sequence. This will have as a consequence that the dialogues produced by the greedy algorithm are almost always completely grammatically correct. However, the greedy algorithm has one downfall, it is deterministic and hence when given the same sequence w, it will always produce the same sentence. To prevent the algorithm from repeatedly writing the same sentence, since it would choose to begin with the most likely word to start a sentence, the beginning word of the sentence is decided via the sampling algorithm explained below.

4.2 Sampling selection

The sampling algorithm takes a random float and chooses the corresponding value based on the cumulative probability. In other words, it chooses the next word i with the same probability as the word has of occurring, i.e. a word with a 70% chance of being the next word will have a 70% chance to be chosen by the Sampling method as the next word.

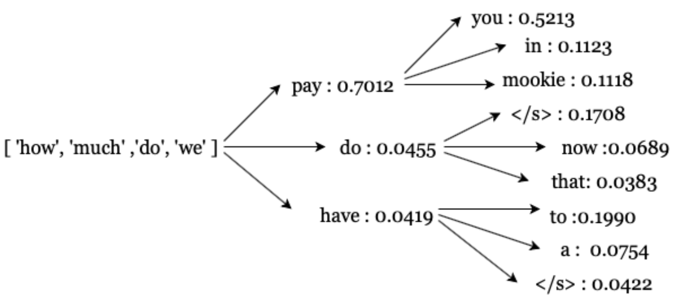

Figure 1: Beam Search Inner workings

4.3 Beam Search

The beam search selection function uses a depth first tree search in order to find the best next word. Firstly, the algorithm finds the three most probable words using the same metrics as the Greedy selection. Once it finds these words it finds the next three probable words again for each of the first three words. It then uses a heuristic function which is calculated using the ’fivegramCounts’, ’fourgramCounts’ and ’trigramCounts’ for all paths down the tree.

5 Results

In this section some results of experiments that were performed.

5.1 Greedy example 1:

Starting from ”i want to get”
Speaker 1: i want to get out of here
Speaker 2: i dont know
Speaker 1: i dont know
Speaker 2: i dont know
Speaker 1: i dont know
Speaker 2: i dont know

Greedy always chooses the word with the highest probability, thus every sentence starts with the same word and the same sentence is generated every time. However, generated sentences are grammatically correct and understandable. The same problem occurred in Ziang Xie paper and the author says that repeating outputs are a common issue. [4]

5.2 Greedy example 2:

Starting from ”i love you and”
Speaker 1: i love you and i want to be a good one

Speaker 2: shut up
Speaker 1: you know what i mean
Speaker 2: the only thing i know is that i cant go to the bathroom
Speaker 1: did you see that
Speaker 2: i dont know
Speaker 1: you know what i mean
Speaker 2: hey
Speaker 1: yes
Speaker 2: big deal
Speaker 1: i don't know
Speaker 2: im not sure i can do this
Speaker 1: what
Speaker 2: we have to get out of here
Speaker 1: and you know what i mean
Speaker 2: id like to see you
Speaker 1: thanks

When while choosing the first word of a sentence the probability is taken into account, greedy generates different sentences and stops repeating itself as often as before.

5.3 Sampling Example 1:

Starting from "i want to get"
Speaker 1: i want to get too much attention theres you and youre entitled to worry
Speaker 2: him wid one fierce flash of me eyes mister can you visit part and an audience - theres no company no fortune you owe me uri one last push
Speaker 1:
Speaker 2: the bellboy looks puzzled
Speaker 1: chance of the old days the sand pit in back of the south latrine
Speaker 2: pain on this his office back of st mikes hes our psychiatric type of your business thats for sure
Speaker 1: i you okay
Speaker 2: because that is what
Speaker 1: to get out

Sentences generated with sampling are most of the time not understandable and not grammatically correct, but the diversity is high.

5.4 Beam Search example 1:

Starting from "how are you doing" Speaker 1: how are you doing
Speaker 2: take it easy now dont want me to go back to the hospital
Speaker 1: sure i dont know
Speaker 2: gustafsons what do you think
Speaker 1: umm i dont know what youre talking to me
Speaker 2: he is a good idea
Speaker 1: no no no no i dont want you to be happy
Speaker 2: thank god for you faith i think i know what you mean
Speaker 1: you know what i want to talk about it

Sentences generated with Beam search mostly are grammatically correct. Separate sentences are understandable with rare exceptions.

5.5 Beam Search example 2:

Starting from "i wonder how the"
Speaker 1: i wonder how the wars going
Speaker 2: have to do is get in my bed and then the big papa bear he roared and somebodys been sleeping in my bed and then the big papa bear he roared and somebodys been sleeping in my bed and then the big papa bear he

In this case beam search generated a very long sentence, which is more repetitive than in the previous example.

6 Analysis

Greedy strategy is the best one of the three in terms of correct grammar, all sentences generated by greedy are grammatically correct. The sampling strategy performs the worst in most cases. The sentences produced are for the most part grammatically incorrect. These results can be explained due to the fact that all words which have some probability to be the next word can be considered. Hence if we have ten thousand words which all have a really small chance of being the next word but collectively have a big chance q then the chance of one of those improbable words being picked is q. If the word being picked has a small probability the chance of the sentence being grammatically correct decreases. As for beam search, despite the grammar produced by beam search being slightly less good than that of greedy, it produces more interesting and diverse sentences. It also tends to produce longer and more complex phrases.

7 Conclusion

The greedy algorithm, when given a first word decided by sampling selection, produces well understandable conversations. However it lacks diversity and falls into repetitiveness after a certain while. The sampling strategy performs poorly because of its struggle to produce grammatically sound sentences. Although the beam search algorithm performs the best overall; when considering grammar, uniqueness and consistency; it performs worse than initially expected. Further improvements to the program could be implementing seq2seq, however given the time restrictions this was not possible.

8 References

[1] Movie dialogue corpus:
https://www.kaggle.com/Cornell-University/movie-dialog-corpus
(Accessed on 22/05/2019)

[2] Yager N (2018), ”Neural text generation: How to generate text using conditional language models”

[3] Vlasblom A (2015), ”Text Prediction Using N-Grams”

[4] Xie Z (2018),”Neural Text Generation: A Practical Guide”