Seinfeld TV Script Generator

Project Report

05-13-2021

Joyce Lu (yl6211), Miaozhi Yu (my1251)

CSCI-GA-3033: Introduction to Deep Learning System

Project Summary

The goal of this project is to design and train neural networks that generate TV scripts for the show *Seinfeld*. The script generator should be able not only to generate TV script for individual characters, but also to replicate dialogue between main characters.

In this project, we implement two Recurrent Neural Network (RNN) models and one seq2seq model, and evaluate their corresponding performance.

Problem Motivation

Natural language generation is a popular but also difficult problem to model as human language is extremely complex. There are a lot of factors and features to consider when using models to simulate how humans speak. This is even more difficult in simulating human dialogues. Since it is difficult to capture the sentiment of previous generated text, dialogue generated by NN models often suffered from incomprehension, and incoherence from a human reader perspective.

In this project we trained a dialogue generator specifically for a US situational comedy show, Seinfeld, which was aired between 1989 to 1998. There are four leading characters in the show, each of whom has distinctive personality traits. We would like to explore how well different models can help improve the sentence structure as well as coherence of human dialogues.

Technical challenges

There are mainly four technical challenges. First of all, to train on a large text dataset proper vocabulary needs to be built out to handle different special characters and mark conversation breaks. The raw scripts contain data from 9 seasons total 180 episodes with more than 39K lines from leading characters.

In addition, it is difficult or even impossible to select the model with best performance in advance. In this project we experimented with both RNN as well as seq2seq models.

Moreover, we are lacking proper evaluation metrics for generated dialogue. There are different evaluation metrics existing for text generation tasks however most of them are focusing on how well the generated text aligns with the golden reference. In the dialogue generation scenario, we might have a different judgement standard. Instead of looking for prediction accuracy, any dialogue that is semantically correct, that captures the role's unique characteristics, and proper sentiment of the previous sentences, will be considered of high quality.

Last but not least, different from regular text generation tasks, dialogue generation has more specific requirements on how the model captures the speaker's unique characteristics. The model needs to properly switch the tone and vocabulary based on the speaker in a certain dialogue. And stronger coherence between lines is needed as the dialogue is commonly carried out in the form of questions and responses.

Approach

GRU

We start by training a character based GRU model. Here is the model summary:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
gru_6 (GRU)                  (None, 128)               83712
_________________________________________________________________
dense_6 (Dense)              (None, 88)                11352
_________________________________________________________________
activation_6 (Activation)    (None, 88)                0
```

It consists of three layers: 1. A GRU layer with input size 128. 2. A fully connected dense layer which contains the 88 different characters including letters and marks etc. 3. The last layer is an activation layer where we use softmax to predict the likelihood of the next possible character.

However, the performance of the GRU is very poor. It is sometimes not able to generate correct English words, for example,“ jerry: yeirfies”. Even though sometimes it's able to capture the word structure, it is not able to capture the sentence structure. For example, “jerry: with all do respect for this my face any”. The model is able to spell every word correctly, however, when putting the words together into a sentence, it does not make any sense.

LSTM

To address the incomprehension issue experienced by GRU model, we further explore a word-based bi-diractional LSTM model.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_1 (Bidirection (None, 512)               44347392
```

```
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 21397)             10976661
_________________________________________________________________
activation_1 (Activation)    (None, 21397)             0
=================================================================
Total params: 55,324,053
Trainable params: 55,324,053
Non-trainable params: 0
_________________________________________________________________
```

The bi-directional model allows us to capture not only the sentiment from previous lines but also recording the information presented in the latter part of the dialogue. The generated sentences have proper sentence structure and capture the person's specific characteristic.

With this first phase success, we collectively train the model with all script lines from the four leading characters, trying to see if that can produce any sensible dialogue between characters. However as each person speaks in different ways, it is very hard or even impossible to use one model to perfectly replicate all characters. The result shows a lack of coherence between the conversations.

NMT

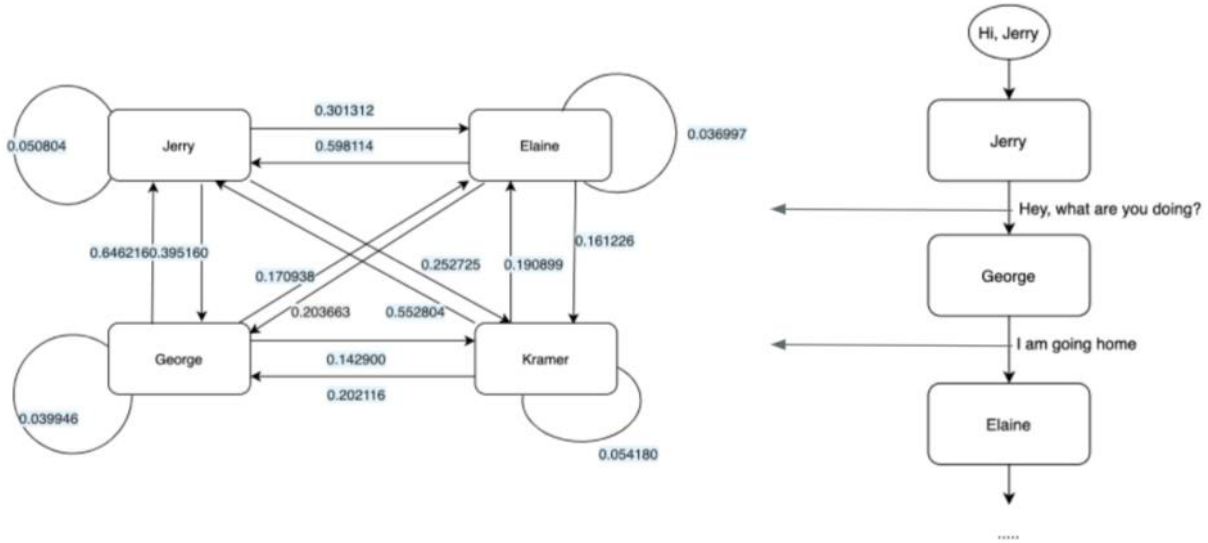
As shown above, LSTM models are able to generate somewhat sensible text based on the lines of each main character. However, it is not enough for TV scripts as it is not able to generate sensible dialogue between characters and switch characters. So we move on to explore the NMT model.

The NMT model has two parts, encoder and decoder. For the encoder part, we first use a character-based convolutional encoder to derive word embedding. Then an RNN layer to encode lines of dialogue as a sequence of encoder hidden states. For the decoder part, we use RNN to produce decoder hidden states based on target sentences using attention to the encoder hidden states. Finally, we put a softmax layer to predict the most likely target word.

NMT models are usually used in translation. It needs a source sentence in one language and a target sentence in another. When training the NMT model for each character, take jerry for example, we split the training data into two parts, other characters' lines as the source sentence and jerry's responding lines as the target sentence. The hypothesis is that using the encoder and decoder are able to establish semantic relation between the source and the target sentence, therefore generate

sensible dialogue. Also the NMT is trained for each character, in this way, we hope to capture the characteristics of each role.

NMT Solution Structure



There are two parts in this NMT solution structure, a probabilistic graph on the left and NMT models on the right. We derive the probability of each character talking to other characters from the raw script. For example, when Jerry initiates a dialogue, it has 0.3 probability that he speaks to Elaine, 0.39 probability that he speaks to George and 0.25 probability that he speaks to Kramer. And of course, he has a 0.05 probability talking to himself. Let's say, we start the dialogue by sending jerry a greet message, eg. "Hi, Jerry". Using "Hi, Jerry" as the input for the NMT model we trained for Jerry, the model produces a response, for example "Hey what are you doing?", then we pick the next responder according to Jerry's probability of talking to other characters and call the corresponding model. In this way, we can generate dialogues among the leading roles.

Implementation details

	Character	Dialogue	EpisodeNo	SEID	Season	char_dial
0	JERRY	Do you know what this is all about? Do you kno...	1.0	S01E01	1.0	jerry: Do you know what this is all about? Do ...
1	JERRY	(pointing at Georges shirt) See, to me, that b...	1.0	S01E01	1.0	jerry: (pointing at Georges shirt) See, to me,...
2	GEORGE	Are you through?	1.0	S01E01	1.0	george: Are you through?
3	JERRY	You do of course try on, when you buy?	1.0	S01E01	1.0	jerry: You do of course try on, when you buy?
4	GEORGE	Yes, it was purple, I liked it, I dont actuall...	1.0	S01E01	1.0	george: Yes, it was purple, I liked it, I dont...

Above is an example of our dataset. We split the dataset into training, validation and test dataset. We put the dialogues in episode 1-15 for any season to the training dataset and the ones in episode 15-19 to test dataset and the rest into validation dataset.

We trained the LSTM and GRU models using keras framework on GCP V100 GPU resources and trained the NMT model using Pytorch on GCP K80 GPU resources.

Experimental evaluation

Model Generated Text:

- Scripts for individual characters generated based on character specific mode.
    - **GRU:**
      jerry: *What do you mean you want to get the same thing is the bottom. You know what you think that doesn't make it out of....*
      george: *Well, I don't know what they want to the postrouse and the many to the point. I can't see that me.*
      elaine: *(exciter) You're the connation of the stopped me.*
      elaine: *Well, I can'*

    - **LSTM:**
      jerry: well, theres this uh, woman might be comin in.

      jerry: you know, maybe if maybe we could start a little more.

      elaine: oh, lighten up. we're going home. i'm going to hell.
- Dialogue generated based on all leading character scripts
    - **LSTM**:

      jerry: i don't know what you do.

      george: i can't believe this guy.

      jerry: i can't believe this guy. i can't believe he hasn't called me to see.

      elaine: what are you doing?

      jerry: i don't know.

      elaine: what?

    - **NMT**

      ELAINE: Hi.
      GEORGE: Kramer?
      ELAINE: I know!
      GEORGE: All right, I think I may have to go back back.
      ELAINE: She said you're getting together Saturday night!
      GEORGE: Oh no no no no no no no.
      ELAINE: Yeah!
      GEORGE: I think I don't understand.

Evaluation metrics:

- BLEU

$$BLEU = \prod_{n=1}^{N} p_n^{w_n}$$

$$p_n \; is \; the \; modified \; precision \; for \; ngram$$

$$w_n \; is \; the \; weight \; assigned \; to \; ngram$$

BLEU is a precision focused metrics calculating n-gram overlap between the reference and prediction text. BLEU's output is always a number between 0 and 1.This value indicates how similar the candidate text is to the reference texts, with values closer to 1 representing more similar texts.

- Unigram Perplexity:

$$p(w) = {count(w)}/{count(vcab)}$$

$$\frac{1}{n}\log(\big(p(s)\big) = \frac{1}{n}(\log(p(w_1)) + \log(p(w_2)) + \cdots + \log(p(w_n)))$$

$$PP(s) = 2^{-\frac{1}{n}(\log(p(s)))}$$

Since perplexity is taking the inverse of the conditional probability of predicting certain words based on the previous words. Thus, the smaller the perplexity score, the better the performance is.

- Human Judgement
  The above metrics compare the generated text with the golden reference. However it is not able to measure qualities like “humor”, “personality” etc. They are also not able to measure the coherence of the dialogue. To measure these qualities, we decided to use human judgement.

One can notice that NMT model scores are quite different from the two RNN scores. Let’s dive deep into the comparison of these three models:

	BLEU	Perplexity
GRU	0.262481981275824	4.48217140309848e-05
LSTM	0.4553999999999997	0.00019151019468558275
NMT	1.575391024288e-77	9907.62

One reason for the NMT model having high perplexity is that we randomly pick a word from the candidate hypothesis set. If we always pick the one with the highest probability,

sometimes it always picks the same word. For example, it will generate sentences like “no no no no no no no no…”. It does not help in creating a meaningful dialogue.

For RNN models, we can see that LSTM is better in both BLEU and perplexity. This aligns with the experiment result. However, for NMT, it is able to generate the most sensible dialogue among the three but it has the worst evaluation metric score. One reason for this is that NMT has different structure and purpose. It is trying to find the best target word. BLEU and perplexity, on the other hand, measures how well the model is able to predict the next possible word. They do not care about the semantic relation between lines and the overall quality in the dialogue. So we decided to use human judgment to measure the readability of the generated script.

Conclusion

In this project, we explored three different models in producing plausible Seinfeld scripts and evaluated the model performance accordingly. Although we did not achieve state of art results for the script dialogue generation, the models we experimented with met the basic requirements of generating sentences with proper structure and context.

For future directions, we would like to bring in topic analysis methods like LDA to improve the semantic relation in the dialogue and overall quality of the script. It would also be very beneficial and helpful for us to continue exploring other academic work on related topics.

Reference

- Paper:

  Papineni, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting of the Association for Computational Linguistics. 2002.

- GitHub Repo:

  Natural Language generation using Neural Machine Translation Context (https://github.com/samjkwong/NLG-NMT)

- Technical blogs:
  - https://www.geeksforgeeks.org/ml-text-generation-using-gated-recurrent-unit-networks/
  - https://medium.com/@david.campion/text-generation-using-bidirectional-lstm-and-doc2vec-models-2-3-f0fc07ee7b30
  - https://towardsdatascience.com/perplexity-in-language-models-87a196019a94