Sequence to Sequence Chatbot with Policy Gradient

Team
Gayathri Pulagam
Rishitha Bandi
Shreya Goyal
Surabhi Govil

Department of Software Engineering, San Jose State University

Outline

A sequence to sequence attention model which generates text for text inputted. Trained on a large corpus of day to day communication text. Aims at creating a more general purpose dialog agent by integrating policy gradient in decoder output.

- **Environment**: Encoder which takes input calculate attention weights and decoder which outputs action based on a logarithmic distribution of softmax of probable actions.
- **Action**: Infinite number of texts the chatbot can generate for a user input by the agent.
- **Reward**: A weighted sum of simplicity of answer, information flow and semantic coherence scores.
- **State**: Current input text to chatbot agent from user.

For the sequence to sequence simple chatbot: https://pytorch.org/tutorials/beginner/chatbot_tutorial.html

For the rewards: https://github.com/VidhushiniSrinivasan16/EmotionalNLG

Team Members

- Gayathri Pulagam
- Rishitha Bandi
- Shreya Goyal
- Surabhi Govil

Problem Statement

- Improve the performance of a simple retrieval based dialog generation system using Reinforcement Learning.
- Training a seq2seq model with policy gradient to generate a chatbot which can be used to have a day-to-day conversation
- Explore how policy gradient affects performance of a attention seq2seq chatbot.

Motivation

- To create a more general purpose chatbot agent. Not have the agent be restricted to a domain and output not based on maximum reward.
- Designing a automated customer service chatbot which can improve based on the positive and negative emotions of a customer.
- Replicating the general human to human to conversation using reinforcement learning.
- Extending the sequence to sequence attention chatbot by using policy gradient to tune the policy learnt by the model.

Methodology:

- We utilized the DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset to train and evaluate our agent.
- Build an encoder and decoder model.
- Designed a Luong attention layer using dot method to calculate attention.
- Calculated softmax of attention weights from the input in decoder
- Output decoder as log of softmax to get the action with highest probability.
- The probability is determined by the attention mechanism which decides the word based on attention paid to some parts of the sentence.
- Without attention next word is outputted which might not be the best action but only generates max reward.

Model Training

- Initially we experimented with a seq2seq chatbot model with the daily dialog dataset
- We integrated a simple reward into the network to add the reinforcement aspect.
- We added policy gradient to the network and trained the model on an RL loop to see how the performance was affected.
- The agent actions update based on policy gradient.

Results and Demo

- For this use case we observed policy gradient did not improve the loss but moved it to being a more general purpose conversation agent.
- Earlier the chatbot predicted from the question the corresponding answer from the corpus.

```
[ ] # Set dropout layers to eval mode
    forward_encoder.eval()
    forward_decoder.eval()

    # Initialize search module
    searcher = GreedySearchDecoder(forward_encoder, forward_decoder)

    # Begin chatting
    evaluateInput(forward_encoder, forward_decoder, searcher, voc)
```

```
> hi
Bot: hi . .
> hello
Bot: hello . . to to
> my name is jen
Bot: i am .
> well
Bot: i m you to ?
> really
Bot: i don t know . .
> bye
Bot: bye .
> q
```

References

[1] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. IJCNLP 2017. [pdf] [arXiv] [dataset]

Thank you