AI 832 - Reinforcement Learning

Mini Project

Project ID: 1

Members:

Venkat Suprabath Bitra	IMT2019091
Vignesh Bondugula	IMT2019092
Abhinav H Kamath	IMT2019001
Maruti Sriram Rachapudi	IMT2019068

Text Generation using Reinforcement Learning

This problem statement is to generate text that is coherent and relevant to a given context or prompt. The model receives a reward signal from an RL agent based on how well the generated text matches the desired outcome. The main objective is to generate text that is grammatically correct, semantically meaningful, and engaging to read.

For this, TextRL, a customizable Python library that utilizes RL to enhance text generation has been used. It builds upon existing libraries such as Hugging Face's Transformers, PFRL, and OpenAI Gym useful across different text-generation models. TextRL by default supports GPT2 and Google FLAN T5 for text generation and allows user-definable reward function. It uses PPO as the Deep RL agent and provided flexibility for hyperparameter tuning of this agent.

For this mini-project of using RL for text generation/chatbot problem, rigorous analysis of different models and datasets have been carried out using TextRL to evaluate their respective performances. The performance of RL based transfer learning depends on the model, reward function and hyperparameters of the RL agent. Various models and dataset combinations were attempted to understand the RL performance, whose results are presented below.

Objective 1: To generate Semantically Negative Sentences for given Input Prompts

Used Models: Google FLAN T5 and GPT2; Datasets: Custom single- or multi-sentence Prompts, Song Lyrics, News Title

The first analysis was performed using Google FLAN T5 and trained to generate negative response for a single prompt, i.e., “Dogecoin is ”. For this, “cardiffnlp/twitter-roberta-base-sentiment” a sentiment analysis model provided by Hugging Face’s Transformers library was used in the reward generation. The reward was kept in the range of 0 to 10, with a higher reward for more negative response.

The results obtained results are as follows:

Before Training:

```
Dogecoin is

['Dogecoin is a Bitcoin coin.</s>']
```

After Training:

```
{'input': 'Dogecoin is '}

['dog</s>']
```

Similar analysis was also performed using GPT2 and the obtained results are provided below.

```
{'input': 'Dogecoin is '}

['\xa0an \xa0 \xa0 Bitcoin ( Bitcoin ( Bitcoin ( Bitcoin ( Bitcoin ( Bitcoin ( Bitcoin ( Bitcoin
```

It was observed that most of the training instances lead to model converging to a static output: “not you not you not you not” or similar, since the model assumes repeated “not” in the response phrase is more negative in a general sense than expected negative response. Through this analysis it was concluded this sentiment analysis model has an inherent bias of repeating “not” phrases, one can observe that on the official web link: [cardiffnlp/twitter-roberta-base-sentiment · Hugging Face](https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment).

```
{'input': 'Dogecoin is '}

['not you not not you you not not you you </s>']
```

Therefore, another sentiment analysis model “nlptown/ bert-base-multilingual-uncased-sentiment” has been used to study the RL performance.

```
{'input': 'Dogecoin is '}

['\xa0an (anonymous) (anonymous) (anonymous) (anonymous (anonymous) (anonymous)']
```

```
{'input': 'How are you?'}

[' (The first person to mention "an" (anonymous) (anonymous) (anonymous) (anonymous)']
```

It can be observed that there are instances when exactly one of the sentiment analysis models underperforms. In the images provided below, the first sentiment classifier associates the suffix “ dog” is associated with a strong negative sentiment. However, the second sentiment model does not associate negative sentiment in that case.

Similarly, the second sentiment analysis model fails in insensible sentences, as shown above. This can be confirmed using below figure, where the second sentiment model strongly sorts the rating of sentence from 1 to 5 stars and gives a high score to most negative sentiment in its scale. However, the first sentiment classifier realizes it not a meaningful sentence and assigns a neutral sentiment to it.

```
sentiment_1("He is dog.")
✓ 0.1s
[[{'label': 'LABEL_0', 'score': 0.6579106450080872},
  {'label': 'LABEL_1', 'score': 0.31303250789642334},
  {'label': 'LABEL_2', 'score': 0.029056897386908853}]]

sentiment_2("He is dog.")
✓ 0.2s
[[{'label': '3 stars', 'score': 0.26602452993392944},
  {'label': '4 stars', 'score': 0.2262992113828659},
  {'label': '5 stars', 'score': 0.2076987773180008},
  {'label': '1 star', 'score': 0.15164290368556976},
  {'label': '2 stars', 'score': 0.1483345925807953}]]
```

```
sentiment_1("You are dog.")
✓ 0.1s
[[{'label': 'LABEL_0', 'score': 0.798261821269989},
  {'label': 'LABEL_1', 'score': 0.1850828230381012},
  {'label': 'LABEL_2', 'score': 0.016655325889587402}]]

sentiment_2("You are dog.")
✓ 0.1s
[[{'label': '5 stars', 'score': 0.3572203516960144},
  {'label': '4 stars', 'score': 0.1939992904663086},
  {'label': '1 star', 'score': 0.16545647382736206},
  {'label': '3 stars', 'score': 0.16332973539829254},
  {'label': '2 stars', 'score': 0.11999417096376419}]]
```

```python
sentiment_1("\xa0an (anonymous) (anonymous) (anonymous) (anonymous (anonymous) (anonymous)")
```
✓ 0.5s Python

```
[[{'label': 'LABEL_1', 'score': 0.8726766705513},
  {'label': 'LABEL_0', 'score': 0.0813116580247879},
  {'label': 'LABEL_2', 'score': 0.046011634171009064}]]
```

```python
sentiment_2("\xa0an (anonymous) (anonymous) (anonymous) (anonymous (anonymous) (anonymous)")
```
✓ 0.1s

```
[[{'label': '1 star', 'score': 0.5140755772590637},
  {'label': '2 stars', 'score': 0.22503311932086945},
  {'label': '3 stars', 'score': 0.15606173872947693},
  {'label': '4 stars', 'score': 0.0636562630534172},
  {'label': '5 stars', 'score': 0.041173290461301804}]]
```

To account for this inaccuracy when generating the reward function for the model, it was done by taking the mean and min of the reward scores generated by both the sentiment analysis models. The results of the same when using GPT2 and Google FLAN T5 as the base model are provided below.

Result of GPT2:

```
{'input': 'Dogecoin is '}

['urchins just just like it just just just just just just just just just just just
```

```
{'input': 'How are you?'}

[' (Determined) (Determined) (Determined) (Determined (Determined) (Determined)\n (']
```

```
{'input': 'How are you?'}

[' (Laughter) (ancient (ancient) laughter) (ancient) (ancient) (ancient)']
```

Result of Google FLAN T5:

```
Dogecoin is

['!!! !!! And now they have been hacked by a bunch of idiots, who claim to be the most powerful crypto currency,']
```

```
How are you?
["I'm sorry.</s>"]
['I am afraid of you, I know you are.</s>']
```

```
How are you?
['How are you?</s>']
["I'm gonna sleep, but I'm getting a headache and I'm gonna go to the dentist.</s>"]
```

```
How are you?
["How'd you do?</s>"]
["I can't stand the rain, I can't stand it, I can't stand it.</s>"]
```

```
How are you?
['How are you?</s>']
["I'm sorry I can't take a picture with you.</s>"]
```

It could be observed that while there are instances of good convergence after using this reward using Google FLAN T5.

To extend upon this, to be able to generate semantically negative responses in general. To do this a larger dataset in form of news article title dataset and a song lyrics dataset was used. These datasets were taken to extend the problem from just getting semantically negative to getting semantically negative newspaper articles or songs.

```
Fed official says weak data

[' from China will lead to lower interest rates.\n\n\nChina has been increasing its interest rate for the last few months,',
 " would be 'deadly for the government'\n\n\nThe government has announced a $10bn budget for 2015-16",
 ' is to blame for lack of progress\n\n\nMOSCOW (Sputnik) — Russian Foreign Ministry spokesman Sergei',
 ' and sluggish economic growth have helped make the move to buy Chinese bonds.\n\n\n"The market is not ready to give',
 " can be used to limit Russia's nuclear program\n\n\nThe country is building a new nuclear reactor to replace a damaged one"]
```

```
{'input': 'Fed official says weak data', 'output': 'caused by weather, should not slow taper'}

[' has been a factor\n\n\nThere has been a strong rally in support of a proposal to place a $1 trillion tax',
 ' has been found\n\n\nGDP has been about 30% more in recent years\n\n\nGDP has been about',
 ' has been a factor\n\n\nIf this has been a type of type of type of type of type which has been a',
 " has been used to blame Russia\n\n\n'We may have been misled about this type of research which has been conducted in",
 ' has been a factor in some of the worst financial crisis in recent memory\n\n\nMore than a third of about 5 million']
```

When large datasets have been trained using a given model, it is obvious that the model will generate responses with negative sentiment. For such cases, the reward function shoots up towards the maximum reward side. As a result of this, for other cases when the model was trained to generate negative sentiment response, the phrase which is producing the higher rewards has been repeatedly adding to the response and showing the negative sentiment, instead of actual sentiment to the response. In the above example, in the second figure, the model converges to using "has been a factor" or "has been used", etc. which are generally given higher negative scores in a news dataset.

At the same time poor selection of parameters like batch size also lead to generation of repeated phrases to get better rewards as follows:

```
[' in the in-game in-game in-game in-game in-game in-game in in-game in',
 ' in in-game in-game in-game in-game in-game in-game in in-game in in',
 ' in the which-have-a-chance-to-win-the-game in which-the-neighbor',
 ' in in-game in-game in-game in-game in-game in-game in-game in in-',
 ' in the report in which we in the in-game in-game in-game in-game in-game in-']
```

In order to handle this problem, true model comparison strategy has been implemented. The sentiment analysis model compares the responses generated by the optimized and the base model to generate the reward. This reward has been designed such that when the first model generates the negative sentiment then the reward will be considerably low whereas when model makes the positive sentiment to negative sentiment which is true targeted output of the model. Such cases the reward is highest.

Chatbot Extension:

This methodology has been used in text generation application as well as chatbot. For training with chatbot the base model utilized was Facebook Blenderbot with 400 million parameters along with an emotion recognition model, michellejieli/emotion_text_classifier, which classifies an input prompt into seven classes, i.e., anger, disgust, sadness, fear, joy, neutral and surprise.

Initial analysis involved training model to generate responses only for anger, however no significant changes were observed as it was rare to find instances where the model would generate them, making the better reward observation sparse leading to poor model performance. This lead to problem reformulation to generate negative persona or a model which generates pessimistic replies. To do so, the sum of scores for anger, disgust, sadness and fear were combined together along with true model comparison strategy. Dialy-Dialog dataset was used to provide the model with sufficient input prompts.

The results for chatbot are presented as follows. In the pictures attached, the first line indicates the question/prompt given to the chatbot, the second is the original base model reply and the third is the reply of the RL model.

```
Our place is spotlessly clean , compared with others .
That's good to hear. I wish I had a clean place like that.
I can't believe I can't find a clean place like that.
```

```
So Dick , how about getting some coffee for tonight ?
That's a good idea. I think I'll do that. I need some caffeine.
I can't because I have a soccer game, but I can try.
```

```
Cats are beautiful and lovely , aren ’ t they ?
Yes, they are. I have two of them. They are so sweet and playful.
Yes, they are so loving.
I don ’ t think so . I think cats are sweet .
I think they are too, but I'm allergic to them so I can't have one.
```

```
Putting on mascara .
Have you ever tried putting on makeup? It's a form of cosmetic makeup used to enhance or alter the appearance of the human body.
I have never used mascara, but I have seen people doing it.  I don't like the way it looks.
```

```
Okay . Here is the laundry .
I need to wash my clothes, but I don't want to do it. I hate doing it.
I can't believe she doesn't clean it, but I can't find anything she can do.
```

```
What did you say to him ?
I told him to get out of my way.  I didn't want to be rude.
I told him that he is a terrible friend, that he should be more respectful.
```

```
how was school today , Dominic ?
It was good. I had a lot of homework to do. How was your day?
I was sick so I went to my brother's house.  I ate a lot of pizza and drank a lot.
```

```
it's just one exam . Can you re-take the exam ?
I can, but I don't know if I will be able to do well on it.
I can try, but I feel like I can't do very well.  I can't believe that I'm failing this one.
```

```
She's so pretty . I'd like to make friends with her .
That would be nice.  Do you think she would be interested in going out with you?
Me too. I feel bad that I can't make friends with her.
```

```
Yup , that ’ s him . He made millions of dollars selling bonds to people and ended up with a pile of cash when they became useless .
Wow, that's a lot of money. He must have been a very successful businessman.
Yes That I can see why he is famous, but I feel bad for him. I can't believe he was able to do that.
```

Conclusion:

In conclusion, this project aimed to improve text generation and chatbot performance using reinforcement learning (RL). By training our models with RL and a reward system, we achieved higher quality, relevance, and coherence in generated text. The reward system guided the models to produce coherent and contextually relevant responses while discouraging undesirable outputs. Our experiments showed enhancements compared to baseline models, demonstrating the value of RL and transfer learning in text generation and chatbot development. However, challenges like biased content and domain-specific knowledge limitations require further refinement of hyperparameters, making them difficult to address completely.

Code: The code files and best models are attached in the link attached in the README file.

Appendix: Chatbot Design using Policy Gradients (Previous Problem)

This problem involves improving chatbot functionality using reinforcement learning, specifically using policy gradients. Policy gradient algorithms work by updating the chatbot's response policy based on the feedback it receives from users. This feedback can come in the form of a reward signal, such as a user indicating that they are satisfied with the chatbot's response. By continually updating its policy, the chatbot can learn to generate more accurate and engaging responses.

The provided reference paper "Deep Reinforcement Learning for Dialogue Generation" uses a separate policy network on top of the base Seq2Seq model for chatbot. The policy aims to find a suitable response given the pair of previous responses. The rewards used to optimize the model are:

1. Ease of Answering: It aims to penalize dull responses which cause the existing Seq2Seq models to blackhole during the conversation. They have decided to find most probable set of dull responses and then ensure the negative log likelihood of responding those utterances is minimized.

$$r_1 = -\frac{1}{N_{\mathbb{S}}} \sum_{s \in \mathbb{S}} \frac{1}{N_s} \log p_{\text{seq2seq}}(s|a)$$

2. Information Flow: It aims to penalize repetitive sequences thereby penalizing the semantic similarity between consecutive turns for the same agent. This is done by computing negative log of the cosine similarity between.

$$r_2 = -\log \cos(h_{p_i}, h_{p_{i+1}}) = -\log \cos \frac{h_{p_i} \cdot h_{p_{i+1}}}{\|h_{p_i}\| \|h_{p_{i+1}}\|}$$

3. Semantic Coherence: It aims to reduce highly rewarded utterances which are grammatically incorrect or not coherent. The mutual information between the response and previous turn history is computed.

$$r_3 = \frac{1}{N_a} \log p_{\text{seq2seq}}(a|q_i, p_i) + \frac{1}{N_{q_i}} \log p^{\text{backward}}_{\text{seq2seq}}(q_i|a)$$

An affine sum of these rewards is taken as the final reward for any given conversation and its response.

To improve upon this:

1. It can be observed that "Information Flow" reward aims to minimize semantic similarity between responses. By relaxing this reward, we can aim to get a more coherent conversation about a given topic, thereby improving the conversation. This can be done by penalizing utterances that are too similar and too dissimilar.
2. We planned to add scores from pretrained BERT based NLI based transformers to ensure that the sequence of statements is logically coherent by introducing three types of rewards: context, topic and persona consistency. This was inspired from the paper: "RPTCS: A Reinforced Persona-aware Topic-guiding Conversational System" by Zishan et. al.
    a. Context Consistency: For the conversation between two agents, the sequential response provided by a particular agent is compared with previous and current responses of the other agent. Suppose, the conversation is: A1 (previous) -> B1 -> A2 (current) -> B2 (candidate), then the reward is calculated as cosine similarity between [A1, B2] and [A2, B2] and taking the average of these measures.

b. Topic Consistency: We use the conversation history to determine the topic by using a bag of words approach and decoding it with the GPT2Tokenizer. We then check the cosine similarity between the topic and the candidate response.
c. Persona Consistency: We used a pretrained emotion recognition transformer along with a mask NLL loss introduced by the work RTPCS.

These changes should improve the performance of existing model and should also be able to learn generating required personalities, i.e., agent with positive behavioral characteristics and with negative behavioral characteristics.

Tested Solutions:

1. RL-Chatbot by pochih: https://github.com/pochih/RL-Chatbot
The methodology provided in the reference paper has been implemented in this RL Chatbot. It used TensorFlow 1.0.1, which is obsolete and requires very old versions of Ubuntu as well as CUDA drivers making it infeasible to run on both local systems as well as in Google Colab. The required environment and packages have been run in the local system and with the recommended and provided dataset could not provide reasonable chatbot responses using the base Seq2Seq model. Further, by implementing PPO with the reward scheme mentioned above, the obtained results were analyzed. Overall, the model has provided poor performance with occasional improvement in the conversation sentence length. This low performance could be due to the involved poor performance of the base Seq2Seq model before implementing the RL model. Some of the obtained results before and after implementing RL is provided below.

Before Training:

```
dialog 0
A: Where are you going?
B: A friend.
A: A friend of yours and a great woman young you don't mean to.
B: She was real.
A: Yes.
B: She didn't want us out.
A: How.
B: I made it happen. But he said send to me.
A: Are you quite serious.
B: No I'm not.
A: I know because I'm just not.
B: You know you're not a very important person.
A: I'm sure.
B: Oh yes. Yes I wonder where to begin to when when I was alone in that.
A: Yes.
B: And those were your last two.
A: Yes.
B: And your son had them.
A: Why.
B: Your father was a wanting an answer your father was.
A: Son of a bitch who never knew of himself.
```

After Training:

```
dialog 0
A: Where are you going?
B: I must be some. Throw it all up.
A: I'm going to see my girl.
B: Can't. Stay a free will save her friends and they can fix her.
A: And you must go after her she's nervous.
B: And you you get the first one.
A: I always thought you'd do what I tell. What different people won't see it.
B: And what do you think of mrs or does.
A: Maybe I should call him.
B: Well that's what we're here for I got too busy with him.
A: It's a shame he has lots of life.
B: We've been talking about it for the past six years.
A: Would you follow mr black.
B: Yes.
A: Well tell them how can you tell them just by reading.
B: That's an expensive.
A: Walter you're.
B: I'm sorry about last we have another problem with four thousand of my business this afternoon bob and acting never.
A: Jesus not at all then.
B: I know when you'd take the rest of my life when I make it and i'll trust you.
A: Okay give me a minute.
```

2. Deep-Reinforcement-Learning-for-Dialogue-Generation-in-PyTorch by Ls-Dai:
   https://github.com/Ls-Dai/Deep-Reinforcement-Learning-for-Dialogue-Generation-in-PyTorch
   The other implementation of the reference paper is Deep-Reinforcement-Learning-for-Dialogue-Generation-in-PyTorch by Ls-Dai. In this both base model training and the further human feedback analysis has been carried out using Seq2Seq approach. This has been done by integrating the rewards provided into the loss function of the Seq2Seq encoder and decoder instead of using RL. We studied this base implementation with incorporation of RL methodology and the obtained results are studied. Similar to the earlier model, in this case also the base model output is not up to the mark which is be due to limited training within the constraints of time and resources provided by Google Colab. Hence, RL model performance is also very poor. Some of the obtained results of this analysis before and after implementing RL is provided below.

```
> How are you?
Bot: i m fine . . . ?
Bot: i m not sure . . .
Bot: then you ll be right . . .
Bot: i m not gonna tell you . .
Bot: i ll be back when you go .
Bot: you ll be fine . . .
Bot: i m not . . . . . .
Bot: you re a witness . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
Bot: i know . . . . . .
```

```
> How are you?
Bot: fine . . .
Bot: it s fine . .
Bot: oh god . . .
Bot: what is it ? .
Bot: open the door !
Bot: i don t ! !
Bot: it s fine .
Bot: what happened ?
Bot: what s you remember ? .
Bot: what ? ? ?
Bot: she s my wife .
Bot: what ? you re her ? ?
Bot: no i am . .
Bot: are you sure ? ?
Bot: yes . . .
Bot: i you . .
Bot: you . . . .
Bot: you . . . . .
Bot: i . . . . . .
Bot: i . . . . .
Bot: what ? . .
```