[Final Paper] Deep Reinforcement Learning with Hierarchical Recurrent Encoder-Decoder for Conversation

Heejin Jeong* and **Xiao Ling**

Final Project Paper for CIS700-007 Spring 2017

Introduction

In recent years, numerous neural network based methods have been introduced for automatic dialogue generation. (Ilya Sutskever 2014) presented a sequence to sequence learning model with deep neural networks (SEQ2SEQ) consisting of two multilayered Long Short-Term Memory (LSTM) and showed that it outperformed a standard SMT-based system on an English to French translation task. (Vinyals and Le 2015) applied this sequence to sequence framework to conversational modeling and their model predicted the next sentence given the previous sentence. Unlike other models that had been used widely, this model achieved better performance requiring much fewer hand-crafted rules. However, since the SEQ2SEQ model was originally proposed for machine translation tasks, it does not capture previous conversations when it is applied to a conversation task. Being able to address previously mentioned topics or information is essential in conversation. In order to solve this issue, (Serban 2015) extended the hierarchical recurrent encoder-decoder (HRED), proposed by (Sordoni 2015) to the dialogue domain. Although they used only triple utterances for implementation, they were able to show their proposed model outperformed both *n*-gram based modelas and baseline neural network models. Another issue of the SEQ2SEQ model is that it tends to be short-sighted since the model does not consider its future outcomes. Addressing this issue, (Li 2016) proposed a novel approach for a dialogue generation task combining a policy gradient method, one of deep reinforcement learning methods, and the SEQ2SEQ model (DRL-SEQ2SEQ). The goal of the learner in reinforcement learning (RL) is maximizing her expected discounted future total reward. Thus, we can expect a model trained using RL would generate an outcome more likely continuing the current conversation.

While the HRED model does not consider its future outcomes, the DRL-SEQ2SEQ model cannot address its conversation history other than a few immediate previous utterances. In this project, we studied the HRED model for dialogue generation in open domain proposed by (Serban 2015) and DRL-SEQ2SEQ. We initially aimed to replace SEQ2SEQ model in DRL-SEQ2SEQ with HRED in order to utilizing the advantage of each model, but due to the time constraint, we were not able to combine both models but tried implementing them on OpenSubtitle Dataset. This paper mainly discuss DRL part of the project. Implementation of the HRED model is studied by Xiao Ling.

Deep Reinforcement Learning for Dialogue Generation

In the RL framework, there is a learner in a certain state s and the learner executes an action a to an environment based on her action policy $\pi^{(a)}$. Then, the environment gives a feedback in a form of a reward, r, and the learner updates her optimal policy π^* in order to maximize her expected discounted future total reward:

$$\mathbf{E}[r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t} r_T | s_t, \pi^*]$$

where T is time horizon and γ is a discount factor which controls the relative importance of the immediate reward for the learner. If $\pi^{(a)}$ differs from π^*, the learning method is categorized to *off-policy*, and if $\pi^{(a)} = \pi^*$, the method is *on-policy*. In order to apply RL to a certain learning task, we need to define an appropriate learning system first. In this paper, we followed the learning system defined in (Li 2016).

Learning System for Dialogue Generation

The learning system consists of two agents where they take turns talking with each other, $p_1, q_1, p_2, q_2, \cdots, p_t, q_t$. p_i and q_i represent *i*-th utterance of the first and the second agent in a conversation, respectively. Unlike the paper of (Li 2016), we will use u_i as a notation *i*-th utterance of a conversation regardless which agent spoke the utterance in order to make sure that the learning model is trained with both agents.

A state is defined by the previous two dialogue turns, $s_t = [u_{t-1}, u_t]$. Specifically, the concatenated vector of the two utterance vectors is used as an input of the SEQ2SEQ encoder. An action a is defined as a dialogue utterance to generate. The action space, $\mathscr{A}$ is considered as an infinite space, but we constrained the maximum length of the generated utterance, L, and the number of available vocabularies, $|V|$ (V is a set of available vocabulary). Therefore, $|\mathscr{A}| = L \times |V|$ in our model. The RL policy, p_{RL} use the form of a SEQ2SEQ encoder-decoder defined by its parameters.

Reward

(Li 2016) defined three reward functions and used the weighted sum of the rewards as a total reward. The first reward function measures the ease of answering a generated turn (or action) using the log likelihood of responding to that action with a dull response. They defined dull responses with 8 turns including "I don't know what you are talking about", "I have no idea", etc. The function is:

$$r_{1,t} = -\frac{1}{N_{\mathscr{D}}} \sum_{d \in \mathscr{D}} \frac{1}{|d|} \log p_{seq2seq}(d|a_t) \tag{1}$$

where $\mathscr{D}$ is a set of dull responses and $p_{seq2seq}$ is a probability distribution with parameters learned by SEQ2SEQ. $p_{seq2seq}$ is different from p_{RL}.

The second reward function penalizes semantic similarity between consecutive turns from the same agent. They used cosine similarity to measure:

$$r_{2,t} = -\log \cos(h_{u_{t-1}}, h_{a_t}) = -\log \cos \frac{h_{u_{t-1}} \cdot h_{a_t}}{||h_{u_{t-1}}||_2 ||h_{a_t}||_2} \tag{2}$$

where h_{u_t} is an SEQ2SEQ encoder output with the input u_t. The last reward function addresses semantic coherence considering mutual information:

$$r_{3,t} = \frac{1}{|a_t|} \log p_{seq2seq}(a_t|u_{t-1}, u_t) + \frac{1}{|u_t|} \log p^{backward}_{seq2seq}(u_t|a_t) \tag{3}$$

where $p^{backward}_{seq2seq}$ the probability distribution with parameters learned by SEQ2SEQ but with sources and targets swapped. Therefore, this backward SEQ2SEQ model has to be seperately trained before using the RL rewards.

Model

Any RL method falls into one of policy-based, value-based, or actor-critic RL areas. Among policy-based RL methods, a policy gradient optimization (PGO) using REINFORCE algorithm (Williams 1992) has been widely used. (Li 2016) also used the REINFORCE algorithm for learning the RL model. The trainig procedure of the RL model is divided into three steps as shown in the Fig.**??**. First, we train a SEQ2SEQ model and a SEQ2SEQ backward model. Next, we train a mutual information model learned with PGO using the semantic coherence (Eq.3) as an only reward. As mentioned earlier, the two pre-trained models are used to compute the reward. The policy parameters of the model is initialized by the parameters of the pre-trained SEQ2SEQ model. The objective function to maximize is:

$$J(\theta) = \mathbf{E}[m(\hat{a}, [u_{t-1}, u_t])] \tag{4}$$

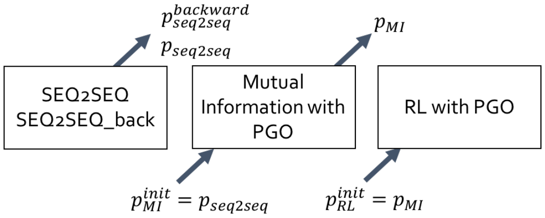

Figure 1: Training Procedure of the proposed Reinforcement Learning Model

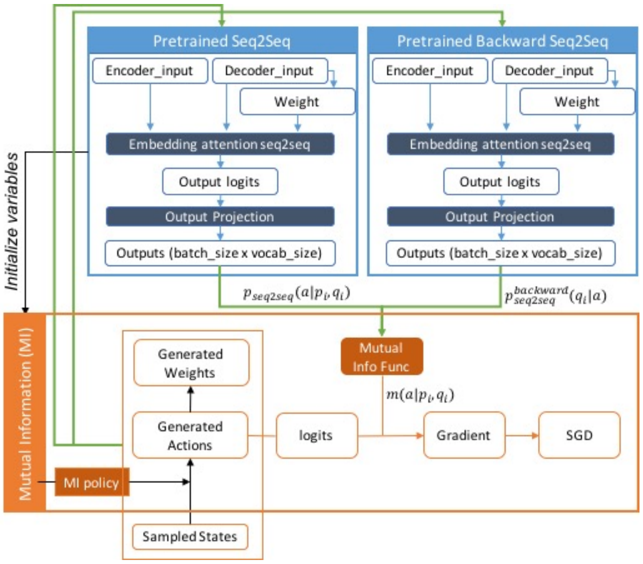

Figure 2: SEQ2SEQ and MI model Code Architecture

where $m(\hat{a}, [u_{t-1}, u_t])$ is equal to $r_{3,t}$ in the Eq.3. The gradient is estimated using the likelihood ratio trick in PGO:

$$\nabla J(\theta) = m(\hat{a}, [u_{t-1}, u_t]) \nabla \log p_{RL}(\hat{a}, [u_{t-1}, u_t]) \tag{5}$$

Then, we update the parameters in the encoder-decoder model using stochastic gradient descent (SGD). In addition to the standard PGO method, (Li 2016) adopted a curriculum learning strategy proposed by (Marc' Aurelio Ranzato 2016) and a baseline strategy. The baseline strategy is common in implementing PGO method since PGO tends to have high variance, and the strategy helps to reduce the variance.

Experiment

Dataset

We first trained the modified SEQ2SEQ model on the CALLHOME American English Speech corpus (LDC97S42), consisting of 120 of 30-minute phone conversations between native English speakers.The size of vocabulary we used is 8038. However, in most of the conversation datasets, one speaker talks a long sentence or several sentences in a row and the other speaker answers with one or two words such as *"mhm"*,*"okay"*, and *"yeah"* as shown in Table.1. In fact, about 20% of the utterances of the dataset is *"mhm"*. As a results, the traiend model after 44200 steps generated a short answer for long input utterances and a long answer for short input utterances (Table.**??**). Therefore, we concluded that this dataset may not have been appropriate for learning a dialogue generation model, and used OpenSubtitles dataset (http://opus.lingfil.uu.se/OpenSubtitles.php) with 50005 voabulary size. The DRL paper also used this dataset, but unlike the paper, we didn't extract *"i don't know what you are talking about"*. Instead, we refined abbreviated words such as *"i'm"* → *"i am"*, "you're" → *"you are"*, "i've" → *"i have"*, "wanna" → *"want to"*, "gonna" → *"going to"*,

Table 1: Example Conversation in the CALLHOME corpus

A:	who is in um someone not that they have problems but someone who's like an okay student but kind of on the borderline you know like maybe not a great homelife and we would ha i got paired up
B:	uh-huh
A:	with someone at um lipsmack i forget the school it was actually in port richmond um breath i forget the name of the school
B:	really
A:	hensfiel no it was in the philadelphia school system and it it was a middle school
B:	mhm

"don't" → *"do not"*, etc. The tokenization part was mostly done by Xiao Ling.

Implementation

For the SEQ2SEQ model, we applied Sequence to sequence with attention mechanism and used GRU cells instead of LSTM cells. We implemented models in tensorflow version 1.0.1. We used `embedding_attenion_seq2seq()` function in `seq2seq.py` and related fucntions/codes written in tensorflow. We also utilized `seq2seq_model.py` in tensorflow version 0.12.x as a bottom line of model codes.

The `seq2seq_model.py` is structured as follow (see Fig.2): when the model is initialized, three main lists of placeholders are constructed - encoder, decoder, and weight. The length of the lists are encoder size, decoder size, and decoder size, respectively. Each element of a list is a tensor with a shape [batch size, None]. Variables are also built and initialized. In each step of training, batch size data are sampled in a given training dataset and converted into a proper format in the function `get_batch()` by operations such as padding, adding GO id, sizing, etc. Then, these are feeded to the placeholders. Output logits are computed through `embedding_attenion_seq2seq()` function and losses are computed using a softmax loss function. Then, variables in the encoder-decoder recurrent neural network (RNN) are updated.

The mutual information model requires pre-trained SEQ2SEQ model and backward SEQ2SEQ model for its initialization and for computing mutual information score. The model first builds three placeholder lists - concatenated states (e.g. u_{t-1}, u_t), the most recent state (e.g. u_t) and weights for the backward model. Unlike SEQ2SEQ, only batch size number of states are sampled in each learning step. Then, actions are generated from the current RL policy given the sampled states. These states, actions, weights are input to the pretrained SEQ2SEQ and backward SEQ2SEQ model, and we obtain log probabilities in each model. Then, the mutual information score is computed using the probabilities and the gradient in Eq.5 is computed. Variables of the encoder-decoder RNN are updated.

Table 2: Backward Seq2Seq Model Examples

Good	**human:**	i don't know man
	bot:	what do you think
	human:	that s amazing
	bot:	wow
	human:	that works best for me
	bot:	i il be right back
	human:	I love you too
	bot:	i love you
	human:	i know i'm hot
	bot:	i m a little nervous
	human:	i'm done with you
	bot:	i m a fucking bitch
Bad	**human:**	i m gonna go to the bathroom
	bot:	i m gonna go to the bathroom
	human:	this is weird
	bot:	i m gonna be a great man
	human:	okay
	bot:	i m gonna go to the bathroom
	human:	you look gorgeous
	bot:	i m gonna go see the kitchen
	human:	fine
	bot:	i m fine

Results

The training results of the backward SEQ2SEQ model after 109000 steps are presentsed in Table.2. Since it is a backward model, the bot generates the most likely utterance that will have the input utterance as its response. The examples in the Good section show good performance of the model. The generated responses in the Bad section do not make sense. The model tends to generate certain responses a lot such as "i m gonna go to the bathroom", or to repeat all or some words in an input utterance. The training results of SEQ2SEQ model with concatenated utterances after 107000 steps are presented in Table.3. Each row of the table is one conversation, since the model consider two previous sentences. However, from the results, it was not hard to see that the model responses considering both two previous utterances. Also, it generates "i am sorry" or "no" many times. For example, for almost any human input utterance starting with "can you", it generates "no".

We wanted to see if the RL model were able to improve such dull response issues. We implemented only mutual information model without the curriculum strategy. However, during the decoding step after training, we found out that there might be an initialization problem due to the variable name feature since we use three models with a similar structure. Due to the time constraint, we were unable to solve the issue and present results in this paper. We hope we can solve the issue before the final presentation and will be able to add results in the final version of our final paper.

References

Ilya Sutskever, Oriol Vinyals, Q. V. L. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112. Association for Computational Linguistics.

Table 3: Seq2Seq Model Examples

human:	this is bad
bot:	this is a joke
human:	what joke
bot:	i do not know
human:	what are you doing
bot:	nothing
human:	you are a loser
bot:	you are a real joke you know
human:	what is a real joke
bot:	i do not know
human:	how old are you
bot:	eleven
human:	you are so young boy
bot:	i am sorry
human:	it's totally okay
bot:	i am sorry
human:	what are you up to this weekend
bot:	i am sorry
human:	where are you from
bot:	where are you going
human:	i am going home
bot:	i am going to get you out of here
human:	thank you but where
bot:	what is this
human:	nothing
bot:	i am sorry

Li, J. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1192–1202. Association for Computational Linguistics.

Marc' Aurelio Ranzato, Sumit Chopra, M. A. 2016. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*.

Serban, I. V. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Association for the Advancement of Artificial Intelligence*.

Sordoni, A. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Conference of Information Knowledge and Management*.

Vinyals, O., and Le, Q. V. 2015. A neural conversational model. In *Proceedings of the 31st International Conference on Machine Learning*.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3-4):229–256.