Chatbot DAISY: Dialog Analogous Intellectual System

Liu, Ting-Wei,
B03901170@ntu.edu.tw

Wen, Ming-Hao
B03901179@ntu.edu.tw

Department of Electrical Engineering, National Taiwan University

ABSTRACT

In our work, we further leverage the success of the seqGAN [1] model in generating meaningful and human-like dialogs, we present our results of training a conversational Chatbot named DAISY: Dialog Analogous Intellectual System in full. She is a conditional sequence generative adversarial network that is trained with policy gradient [2]. In our work, we focus on the task of single turn dialog generation to produce more intellectual and meaningful conversations using generative adversarial training [3], our model aims to generate sequence of words that are indistinguishable from human generated sentences given the previous dialog history. The task is formulated as a Reinforcement Learning (RL) problem where we jointly train two models, a generative model *G* that defines the policy of generating a dialog sequence given the dialog history, and a discriminative model *D* that strives to distinguish machine-generated dialogs and human-generated dialogs. The generative model takes the form of a Seq2seq [4] model with attention mechanism [5], whereas the discriminative model consists of a hierarchical encoder [6] that encodes input dialogs into a vector representation, and does a binary classification based on the encoded vector. We further use Monte Carlo Search as a rollout mechanism to calculate reward for every generation step, and apply the technique of teacher forcing, which greatly aids the generative model in training. We train our model on the Cornell Movie-Dialogs Corpus [7], and shows that compared with ordinary Seq2seq [4] Chatbots that can only generate dull, generic, and repetitive response, our model DAISY can generate dialog responses that are much more interactive, interesting, and human-like.

Index Terms— seqGAN, Chatbot, Policy Gradient, Dialog Generation, Reinforcement Learning, Generative model, Discriminative model, Seq2seq, attention mechanism, Cornell Movie-Dialogs Corpus.

1. INTRODUCTION

In recent works, generative adversarial networks (GAN) [3] have shown promising results in generating realistic real-valued data like images. However, it has limitations in generating discrete data like dialogs, where dialogs are unlike images, is composed of sequences of discrete tokens. As mentioned in "*Adversarial Learning for Neural Dialogue Generation*" [1], one of the main reasons why the approach of GAN fails at dialog generation is that sampling discrete tokens at the Generator output makes it difficult for gradients to back-propagate from the Discriminator to Generator. Also, the discriminative model can only assess a complete generated sequence, incapable of grading partially generated sequence. In other word, credits are given to the whole sequence of actions, but not every action at each generative step. This is a problem because the reward associated with a sequence of actions is used for all actions in the sequence. Consider the following example: given input history *where are you from*, the machine generated response is *I don't know*, whereas the ideal response would be something like *I am from Taiwan*. The model then assigns the same negative reward to all tokens in the machine generated sequence *I don't know*. However, the first token *I* should receive a neutral reward since it can also be the starting token to the ideal response *I am from Taiwan*.

The above two problems were solved in the seqGAN paper [1], sequence generative adversarial network is trained through policy gradient, where the generative network is modeled as a stochastic policy in reinforcement learning (RL). The token sampling process in the generative model is considered as a policy that chooses a token given the previous chosen tokens. This policy is directly trained via policy gradient, which naturally avoids the differentiation problem in discrete sampling. Furthermore, a Monte Carlo rollout searching mechanism is employed to approximate the state-action value, which is used to compute intermediate step rewards. This allows the model to learn the reward of partially decoded sequences, and is necessary in generating high quality response.

Moving from dialog generation to conditional dialog generation, the paper "*Adversarial Learning for Neural Dialogue Generation*" [2] achieved promising results in such a task while employing policy gradient training and Monte Carlo rollout on generative adversarial networks as in [1]. Our implementation is based on [2], where we set our training objective resembling the idea of Turing test [8]: for machines to generate dialogs that are indistinguishable from human generated ones. As in the framework of all generative adversarial networks [3], our model consists of a generator

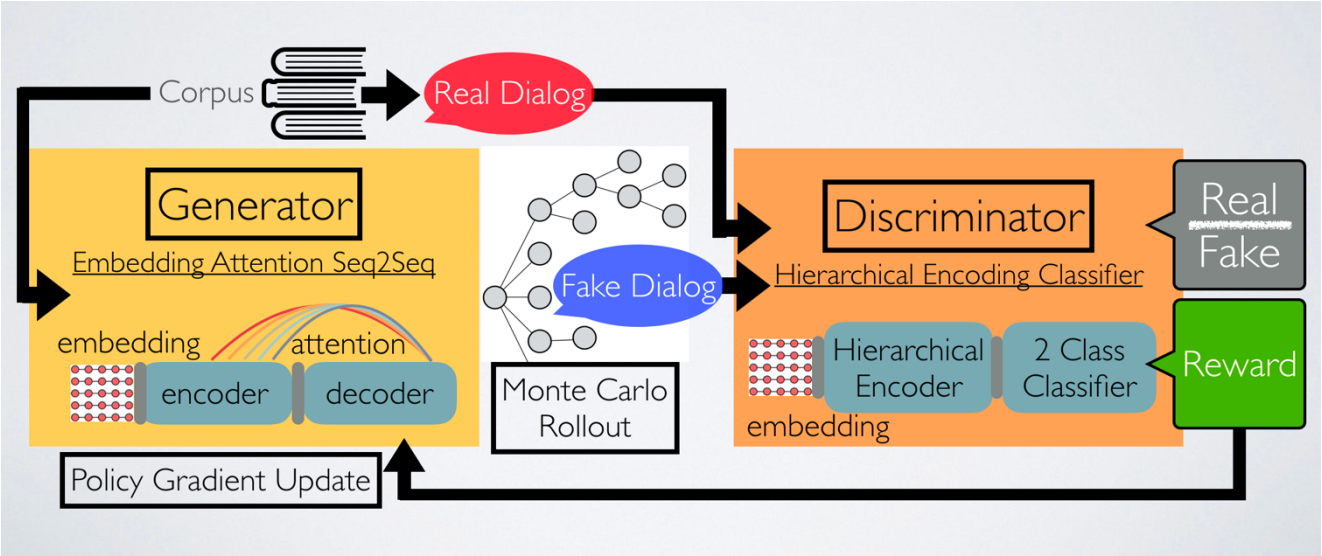


Figure 1. Training Framework of Our ChatBot as Described in Section 2.

that is responsible for defining the probability of generating a dialog sequence given the previous dialog sequence history, and a discriminator that distinguishes whether the given dialog is machine generated or human generated. The discriminator is then analogical to the human evaluator in Turing test [8]. This training objective is formulated into a Reinforcement Learning problem, where the output from the discriminator is used as a reward to the generator, pushing the generator to gain a higher reward by tuning its output towards a more human-like generated dialog.

In our work, we train our own Chatbot DAISY, a conditional sequence generative adversarial network with policy gradient, while using Monte Carlo rollout and teacher forcing mechanism to aid our training, similar to the architecture proposed in [2]. We show that compared to standard Seq2seq [4] models, Chatbot trained in this manner possess a higher ability to generate more intriguing, interactive, and non-repetitive responses.

2. ADVERSARIAL TRAINING WITH POLICY GRADIENT

In this section, we discuss in detail how we formulate the task of conditional dialog generation into a generative adversarial training problem using reinforcement learning. The task can be framed as follows: given a dialog history x, the model needs to generate a response $y = \{y_1, y_2, \dots, y_T\}$, where dialog history x is approximated by using the concatenation of two preceding sequences, and the response y is a sequence of word tokens sampled by our model.

Generative model

The generative model G takes the form of an encoder-decoder recurrent neural network, similar to standard Seq2seq [4] models. The role of the generative model is to generate dialogs that can fool the discriminator, it defines the policy that generates response y given the dialog history x. Hence we can view the sequence generating process as a sequence of actions being taken according to the policy defined by the generator. The encoder encodes the dialog history x into a vector representation, then the decoder computers the probability of generating each token at every time step through softmax function. Moreover, we use attention mechanism [5] on the encoder-decoder hierarchy to aid the decoding process.

Discriminative model

The discriminative model D takes the form of a binary classifier, similar to standard GAN [3] models. The role of the discriminative model is to judge whether the given dialog is machine generated or human generated. It takes the concatenation of {x, y} as input and outputs a scalar value (0 to 1) indicating whether the input pair is generated by machine (0) or human (1). The input sequences are encoded into a vector representation using a hierarchical encoder [6], in which each sentence is mapped to a vector representation through LSTM, then another LSTM runs on sentence level that encodes the dialog sequences into a single vector representation. This vector is then used to compute linear regression for 2-class classification, returning the probability of the input dialog being machine generated or human generated.

Policy Gradient Training

Policy Gradient is used to encourage the generator to generate dialogs that are indistinguishable from human conversations, in which the score of the current sequence assigned by the discriminator is used as a reward for the generator. More specifically, given a input dialog history x, the generator

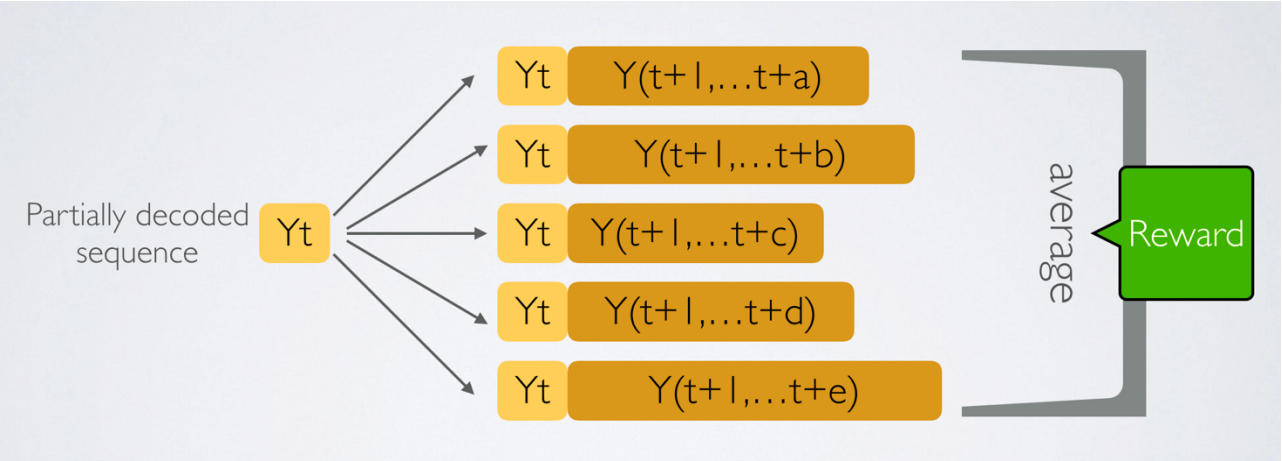


Figure 2. Monte Carlo Rollout Mechanism Illustration as Described in Section 3.

generates a sequence of words $y = \{y_1, y_2, \dots, y_T\}$ by sampling from its policy, the concatenation of $\{x, y\}$ is then fed to the discriminator, generating a score $D(\{x, y\})$. The generator is then trained to maximize the expected reward of generated sequence:

$$J(\theta) = \mathbb{E}_{y\sim p(y|x)}(D(\{x, y\})|\theta)$$

The gradient of $J(\theta)$ can then be approximated by the likelihood ratio trick [9]:

$$\nabla J(\theta) \approx [D(\{x, y\}) - b]\, \nabla \log \pi(y|x)$$
$$= [D(\{x, y\}) - b]\, \nabla \sum_t \log p(y_t|x, y_{1:t-1})$$

where π is the policy of the generator and b is the baseline value to reduce the variance of the estimate while keeping it unbiased. While the generator updates through policy gradient, the discriminator simultaneously updates with both positive and negative examples, that is, human generated dialogue and machine-generated dialogue respectively.

3. TRAINING MECHANISMS

As mentioned in Section 1 we utilize two additional training mechanisms to train our model: First, we use a Monte Carlo rollout searching mechanism to approximate the state-action value, which is used to compute reward for every intermediate generation steps. Second, a technique called teacher forcing is applied to the training process of the generator. Adding these two training mechanisms greatly increase the performance of our model, we discuss the details in this section.

Reward for every generation step

Since the discriminator can only assign rewards to fully generated sequences, we compute the intermediate reward by using the Monte Carlo (MC) search. Given a partially decoded sequence Y_t at time step t, the model keeps sampling from the policy until the decoding finishes. This process is repeated N times (N=5), resulting in N fully generated sequences that starts with Y_t. These N sequences are fed to the discriminator, the average reward of these N sequences thus represents the intermediate reward at time step t, and is used as a reward of Y_t. For each partially generated sequence $Y_t = y_{1:t}$ the discriminator gives a reward $D(\{x, Y_t\})$. The gradient update of the generator can then be rewritten as:

$$\nabla J(\theta) \approx \sum_t [D(\{x, Y_t\}) - b]\, \nabla \log p(y_t|x, Y_{1:t-1})$$

where the average reward $D(\{x, Y_t\})$ is given by:

$$D(\{x, Y_t\}) = \frac{1}{N}\sum_{i=1}^{N} D(\{x, Y_t^i\})$$

as Y_t^i denotes the i^{th} fully generated sequence generated from the partially decoded sequence Y_t. The values for rewards are then different among each generated words in a single response, whereas in the absent of MC search, all words in a response are assigned to the same reward $D(\{x, y\})$. With MC rollout, our generator can learn the different values for each separate token, and demonstrates a higher performance and generation quality. However, the downfall of applying MC rollout is that it is significantly time consuming, as it repetitively samples N complete sequence for each partial sequence.

Teacher Forcing

Updating the generative model using policy gradient and Monte Carlo Search leads to unstable training, even when the generator is initialized with a pre-trained Seq2seq [4] model. This is because in the process of generative adversarial training with policy gradient, the generative model is indirectly exposed to the golden sequences through the

reward passed back from discriminator. This training strategy is fragile since once the generator deteriorates on some misguiding gradients in some batch, and the discriminator constantly does a good job in recognizing sequences from the generator, the generator immediately gets lost, as mentioned in [2]. Training may suddenly breakdown after a few hours, the generator knows that the generated sequences are bad based on the reward, but it doesn't know how to produce a good sequence due to the vast size of space of possible sequences, it needs to get some positive rewards to push itself to generate good sequences. To solve this problem, we give the generator direct access to the ground truth by also feeding sequences generated by humans to the generator. At every generating step, we feed the generator the ground truth instead of the token generated by itself, and allow the discriminator to assigns rewards on these complete sequences. We use this reward on the teacher forcing sequences only if the reward is larger than the baseline value *b*, the generator thus updates on these positive examples. This is analogical to having a teacher to intervene with the student's work in a test for some fraction of time, this allows the generator to constantly stay on track and continues to generate good sequences. Overall, the teacher forcing model can be viewed as a regularizer to regulate the generator once it starts deviating from the training dataset, and helps stabilizing the training process.

4. TRAINING DETAILS

We train our Chatbot DAISY using the Cornell Movie-Dialogs Corpus [7] dataset, this dataset contains 220,579 conversational exchanges between 10,292 pairs of movie characters, which involves 9,035 characters from 617 movies, and a total 304,713 utterances. This dataset is particularly perfect for training a conversational Chatbot because it contains multi-turn conversations, in which several lines are exchanged between characters. In our implementation, we use buckets to hold training sequences with different length, in which we found that sequences with length smaller than 10 are more suitable for our task (ChatBot) since normal human conversations usually consists of sentences with less than 10 words. Moreover, we utilize sampled softmax during training to reduce computation complexity of the regular softmax over a large vocabulary size. The training procedure is illustrated in Figure 3, in which G, D, C represent the generator, discriminator, and corpus respectively. We first pre-train the generator on real data for 300+ epochs, then we initialize generative adversarial training with the pre-trained model. In the discriminator update steps, we train the discriminator to distinguish human dialog and machine generated dialogs. In the generator update steps, we use Monte Carlo Rollout to calculate reward for every time-step, then we update the generator with the reward given by the discriminator. At last we use Teacher Forcing to make sure the generator stays on track, we update the generator with MLE (Maximum Likelihood Estimation) training for every generator step. For more training details, we use a learning rate of 1e-4 for both Generator and Discriminator, and a batch size of 64 with RMSProp optimizer, the number of D steps is set to 3 and number of G steps is set to 1 in our training. The pre-train loss curve is shown in Figure 4.

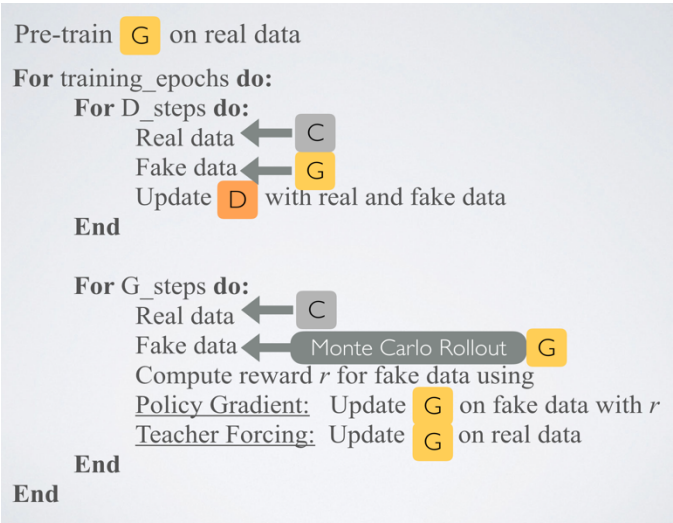
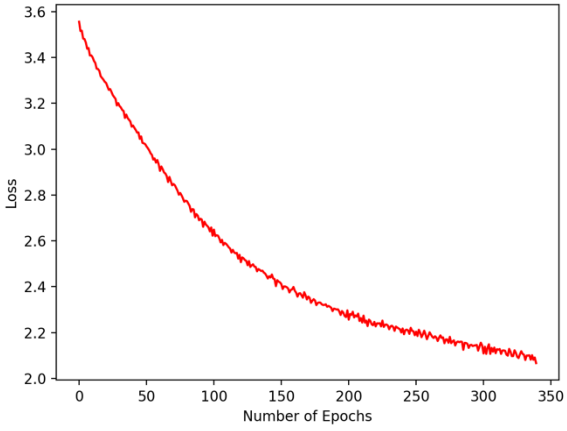


Figure 3. Training Procedure as mentioned in Section 4.

Figure 4. Pre-Train Loss Curve.



5. RESULT

We evaluate our ChatBot Daisy's grammar and semantic fluency with BLEU score, in which the similarity between two sentences are calculated. We let the model generate sequences for all inputs in the testing set, then for each generated sequence we calculate its' BLEU score with all human sequences in the testing set and obtain an average BLEU score for that particular generated response. The average of all generated responses thus represents the BLEU score for that model. The pre-trained seq2seq [4] model obtained a score of **0.395** and the DAISY model obtained a score of **0.352**. It is reasonable for the generative adversarial

model to obtain a lower score, since the BLEU score is calculated with the testing set, the MLE trained model will surely generate sequences that are more alike with the testing set. Finally, we show some dialogs generated by our DAISY model and the seq2seq [4] baseline model given the same user input. In comparison, it is obvious that our model can generate dialog responses that are much more interactive, interesting, and human-like:

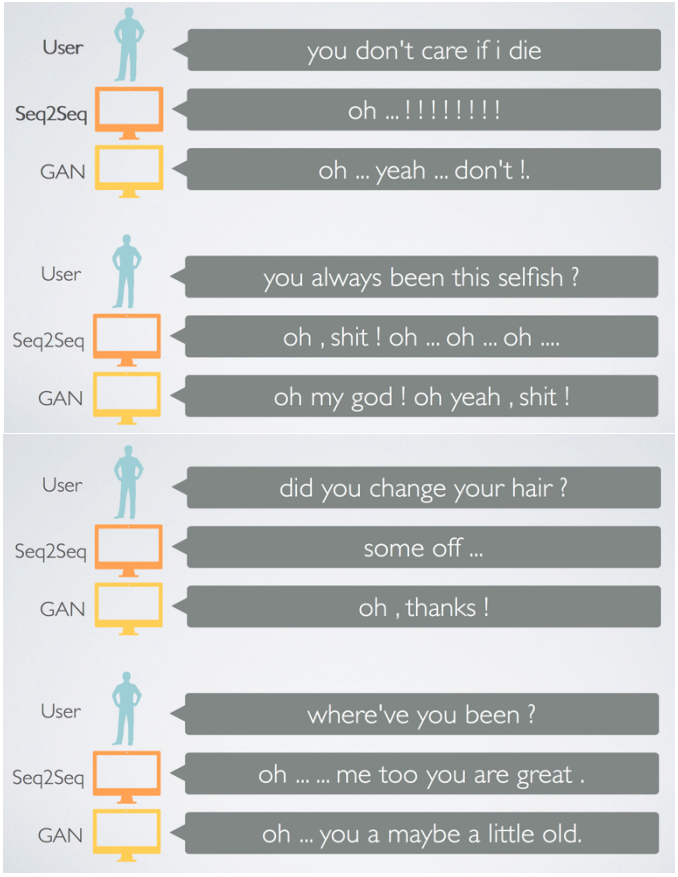


Figure 5. Dialog Generation Comparison

6. CONCLUSION

We built a ChatBot DAISY, which is a conditional sequence generative adversarial network that is trained with policy gradient updates. Our model consists of a generator and discriminator framework, as in general seqGAN[1] networks. In our work, we used several techniques including generator pre-train, Monte Carlo Rollout, Teacher Forcing, and Sampled Softmax to aid our training. We found that the DAISY model has similar grammar ability as the seq2seq [4] baseline model (with similar BLEU scores). In addition, the DAISY model was capable of generating more intriguing and interesting responses than the baseline model, which we verified through the dialog generation comparison section. In future work, we hope to move towards multi-turn dialog generation with more advanced GAN variant models. The implementation of this work can be found on github: https://github.com/b03901170/machine-learning-tensorflow.

7. REFERENCES

[1] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In arXiv:1609.05473v6

[2] Jiwei Li, Will Monroe, Tianlin Shi, Se´bastien Jean, Alan Ritter and Dan Jurafsky. 2017. Adversarial Learning for Neural Dialogue Generation. In arXiv:1701.06547v5.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. pages 2672–2680.

[4] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Proc. of ICLR.

[6] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. ACL

[7] Cornell University. 2011. Cornell Movie-Dialogs Corpus. The original site of Cornell university is currently not working, but the dataset can still be found here: https://www.kaggle.com/Cornell-University/movie-dialog-corpus

[8] Alan M Turing. 1950. Computing machinery and intelligence. Mind 59(236):433–460.

[9] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning 8(3-4):229–256.