RESPONSE GENERATION IN DISCRETE SPACE USING GENERATIVE ADVERSARIAL NETWORKS

Mengjie Zhao
Ecole polytechnique fédérale de Lausanne
mengjie.zhao@epfl.ch

ABSTRACT

In this work, we propose a novel architecture which can be employed in question answering and response generation tasks. The architecture consists of two major modules, namely the primary network and the generative refining network. The primary network focuses on generating candidate answers in response to an input question from users. The role of the primary network can be played by any existing response generation model such as sequence to sequence model or memory networks. Furthermore, the generative refining network focuses on refining the outputs from the primary network through a new generating process, powered by adversarial training. The generative refining network can be trained in an on-line manner, i.e., for each input, it provides a refined answer. Depending on the type of the primary network, the entire architecture can be trained either end-to-end without handcrafted techniques or trained on-line. Experimental results show that the newly proposed architecture generate acceptable responses to the given input question. More importantly, this architecture is capable to be further developed to fit to various response generation tasks.

1 INTRODUCTION

With the booming and advances of deep artificial neural networks, dialog systems based on natural language understanding have been increasingly popular. One crucial task of these systems is the Question Answering (QA) task. Question answering is incredibly broad and more or less any task one can think of can be cast into this setup, as shown by Weston et al. (2015). Dialog generation problems can be view as multi-round QA tasks, but with emphases on generating relevant and fluent response to an input utterance.

One existing popular model applied in resolving the QA tasks is the memory network proposed by Weston et al. (2014). For a given question, the model reasons with inference components combined with a long-term memory component that can be read and written to. The empirical results show that memory networks perform remarkably and achieved more than 90% of test accuracy. On the other hand, several models are proposed to leverage the dialog generation tasks. For example, sequence-to-sequence (seq2seq) model proposed by Sutskever et al. (2014) studies exiting dialogs in training dataset then generate appropriate responses. Many real world artificial intelligence applications have started to utilize these dialog generation models, such as *Google Now* by Guha et al. (2015) and *Xiaoice* by Microsoft.

The models applied in the dialog generation tasks belong to the generative models as shown by Neal (1985); Salakhutdinov (2009) which can be categorized as unsupervised learning techniques. The generative models take a large amount of data from some specific domains then try to produce outputs similar the inputs. Training a generative model is a sophisticated task, as a result, many techniques such as sampling, likelihood and adversary are introduced as shown by Huszár (2015). Currently, the generative adversarial networks (GAN) proposed by Goodfellow et al. (2014) and Variational Auto-Encoder (VAE) proposed by Kingma & Welling (2013) draw a significant attention from the research community.

Most state-of-the-art neural QA models like memory networks and seq2seq model can be trained in an end-to-end manner through minimizing a single cost function with minimum reliance on feature engineering or other handcrafted techniques over the training dataset. In this case, the utterance is generated from the model directly in response to a given question or sentence. However, because these models belong to neural generative models which focus on analyzing and understanding distributions of real data, there is no strong guarantee that the generated responsive utterances would be correct, relevant and fluent, even though the defined cost was reduced to minimum through large scale gradient descent based algorithms introduced in Bottou (2010) during the training procedure.

In this work, we propose a new generative neural network that tries to refine the results from pre-trained neural dialog models (primary networks). The new generative neural network architecture utilizes the adversarial training idea from GAN networks and can be trained in an on-line manner because of its small size and fewer parameters. More importantly, it will not break the end-to-end property of the primary networks. It takes outputs, e.g. candidate answers or generated utterances, from end-to-end pretrained models like memory networks or seq2seq model, then try to refine the outputs through a new generating process. To the best of authors' knowledge, there is no existing models functioning as refining the responsive utterances using a new generative process. The ultimate goal of this new generative refining network is to combine responses from different type of pretrained primary networks then generate more relevant and fluent utterances. In this work, we focus on refining outputs from a pretrained seq2seq model.[1]

2 Related Works

Researches of applying deep neural networks in the fields of QA and dialog generation have saw significant improvements in recent years. The QA problem, due to its tractability in evaluation, becomes an appealing research topic. There are formalized tasks such as Facebook bAbI tasks proposed by Weston et al. (2015) available to determine whether a system, e.g. a chatbot, is capable of communicating with human.

For dialog generation tasks, Ritter et al. (2011) propose to use statistical machine translation procedure to model the generation of dialogs. They compare the proposed statistically based model with standard natural language processing (NLP) tools and achieved a doubled performance on status messages from social medias like Twitter. Since the statistical model focuses on generating one utterance at a time, Li et al. (2016b) propose a deep reinforcement learning framework to enhance the predictive ability of conversing agents and obtained more relevant and coherent results through modeling the potential future direction of the dialog. This work shares a similar goal with our work which is generating improved utterances. One crucial problem of existing seq2seq models is the inconsistent responses. For a set of identical questions, it is possible for the model to generate inconsistent responses such as different ages for one same person. This problem is tested by Bordes & Weston (2016) and Li et al. (2016a) try to address the problem through taking consideration of background information and speaking style.

Currently, several popular generative models achieved remarkable performance in understanding and imitating real world data. The generative adversarial networks proposed by Goodfellow et al. (2014) draw significant attentions from the research community. Within this model, a discriminator is introduced to guide the generative model to produce better results such as higher resolution images than other vanilla models as shown by Lotter et al. (2015). Recently there are two methodologies in applying GAN in text generation tasks which are finished in discrete and continuous space respectively. Zhang et al. (2017) propose the textGAN model to finish text generation tasks where Gamble-softmax and other smooth approximation methods are applied to leverage the problem that GAN is originally defined on continuously distributed datasets. On the other hand, Yu et al. (2017) proposed SeqGAN to circumvent the intractability of applying GAN on text generation. In this model, text generation are guided by the optimal policy (policy that has maximum rewards) which is obtained through policy iteration. This idea is closely related to our proposed framework, but the goal is significantly different. They try to generate more vivid sentences while our goal is to refine

[1] All codes are available at `https://github.com/joemzhao/nnMiscs`

the responsive utterances. The neural dialog generation procedure shown by Li et al. (2017) focuses on generating dialogs that are as close to human dialog as possible instead of trying to refine the output utterances.

3 Task Description and the Model

Within this section we firstly show our motivation and illustrate the tasks that are targeted to be resolved, after that, we demonstrate the detailed architecture of the proposed generative refining network which is a small scale neural network that can be trained on-line.

As described in section 1 and 2, existing neural dialog generative models are mostly trained end-to-end. Since the results are delivered by the model directly, there is no strong guarantee that the responsive utterances are relevant and coherent to the input utterance, such as the inconsistent problem emerged in the seq2seq model. As a result, we propose the generative refining network which takes in the generated utterances from primary network and try to refine them. To be more concrete, considering the generated utterances from different pretrained models. It is possible that some of these utterances are close to the ground truth while others are not. As a result, we utilize the idea of adversarial training suggested by Goodfellow et al. (2014) to carry out a new generative process trying to combine and refine the utterances.

Following figure displays a bird view of the entire architecture we propose and the generative refining network is circled in the right hand side rectangle:

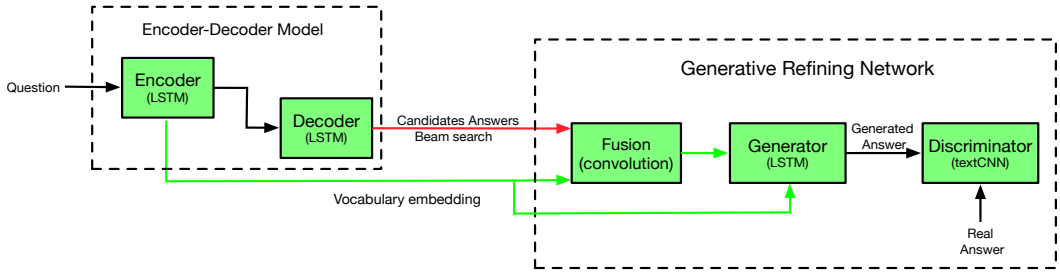


Figure 1: A bird view of the entire applied architecture

The entire architecture consists of two major modules, namely the Encoder-Decoder Model (i.e. seq2seq model in this case) acting as a primary network to generate candidate utterances, and the generative refining network serving in refining the utterances from the seq2seq module. The two major modules are composed of several small elements like the long short term memory (LSTM) by Hochreiter & Schmidhuber (1997) and convolutional neural networks by Krizhevsky et al. (2012):

- Sequence to Sequence Model
  - Encoder (LSTM)
  - Decoder (LSTM)
- Generative Refining Network
  - Fusion (convolutional operation)
  - Generator (LSTM)
  - Discriminator (textCNN)

The whole architecture takes a question as input and response with utterances from the generative refining network. It should be noticed that the seq2seq model is pretrained end-to-end while the generative refining network is trained with an on-line manner. One can observe that it is the beam search operation (red arrow) applied to generate candidate answers that makes the entire architecture not end-to-end trainable. However, this problem can be resolved by employing other differentiable

generative models or approximation methods. Within such cases, the gradients from the generative refining network can also be backpropagationed to update parameters in the primary networks. Due to the significant complexity of the architecture, it is crucial to evaluate every module to confirm its correctness before going further. Following section provides a module-wised analysis and training details are explained.

4 TRAINING AND EXPERIMENTS

In this section, we firstly provide some practical statistics of the dataset we used to train the network. After that, we demonstrate the training procedure of the entire architecture and confirm the correctness of each module step by step.

The dataset we use is the Big Bang Theory (TBBT) dataset. It contains the subtitles of this famous TV series. We parsed the entire word token stream to (Q, A) sets such that we can feed the data to our model. There are 20525 unique tokens result in a vocabulary size of 20525 defined in the model. There are in total 524339 tokens which means that this dataset is relatively small but dense and more tractable, compared with existing datasets like the OpenSubtitle dataset by Tiedemann (2009).

4.1 SEQ2SEQ MODEL

The deep neural seq2seq model proposed by Sutskever et al. (2014) uses a multi-layered LSTM to map the input sequence to a fixed sized vector representation, after that, another LSTM is applied to decode the target sequence from the learned vector representation. This model is widely used in machine translation and other language understanding tasks. In our proposed architecture, we use the seq2seq model as the primary network to generate candidate utterances. The training procedure is to maximize the log probability of a correct response *A*, given the source utterance *Q*:

$$\frac{1}{|S|} \sum_{(Q,A)\in S} log\ p(A|Q)$$

where S is the training dataset. After the training, the candidate answers that maximize the probability $p(A|Q)$ are obtained by a simple beam search algorithm. When we apply the seq2seq model in our architecture, one of the most important hyper-parameters is the size of word embedding which is also used in the further generative refining network. As a result, we evaluate how the size of embedding could affect the model performance, as shown in following Figure 2:

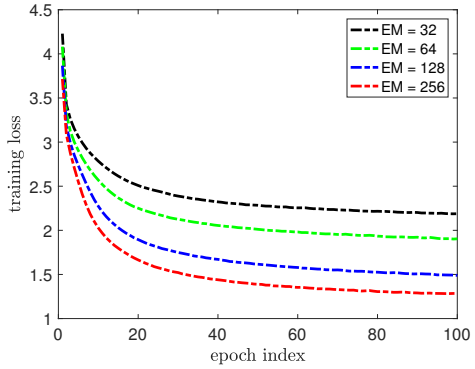


Figure 2: Larger embedding size reduces loss

Q: what is your name ?
A: my name is siri .

Q: do you know switzerland ?
A: oh , of course .

Q: i think you are correct .
A: so why are you saying that ?

Q: can you drive me home ?
A: uh , sure .

Table 1: sample generated (Q, A) set

From above figure we can observe that different embedding sizes result in significantly different model performance. The performance gap between $EM = 64$ and $EM = 128$ is more significant than that of other adjacent curves. Word embedding that has size of 256 gives the best performance compared with other sizes. This result meets our expectation since higher dimension mean higher number of freedom to represent a word. It can be predicted that the loss could be reduced

further if the dimensionality is increased. However, from the consideration of performance improvement and computational tractability, we will choose word embedding size as 256 in following works.

Above Table 1 provides some generated utterance to input questions (not shown in training dataset). It can be conclude that the responsive utterances is relevant and coherent to given questions, and the bilingual evaluation understudy (BLEU) can be used to give quantified scores if necessary. After the correctness of the seq2seq model was confirmed, we utilize the trained seq2seq model to generate candidate answers for future use in the generative refining network. The detailed procedures are:

- For each existing (Q, A) set, we take the Q as an input to the seq2seq model.
- The seq2seq will generate utterances in response to the input Q. Now, instead of only taking the utterance with largest score rated by the beam search algorithm, we pick out the top 20 candidates. These candidates are sorted according to their scores (from beam search algorithm). It is possible that some pairs do not have as many as 20 candidates. In this case, we pad them with `[end]` for future use.
- Finally, we obtained tuples shaped as $(Q, A, \vec{A}')$ where Q, A and $\vec{A}'$ are question, real answer (ground truth) and candidate answers respectively.

After obtaining the $(Q, A, \vec{A}')$ tuples, we feed the tuples together with the word embedding from the encoder LSTM to the generative refining network to start a new generating process.

4.2 Generative Refining Network

This section is dedicated to demonstrate and explain the generative refining network which serves in refining the utterances generated from the primary network. There are three main components of the generative refining network, namely the fusion, generator and discriminator and they are implemented as a convolutional operation, LSTM and textCNN respectively. The fusing operation is going to be introduced independently while the generator and discriminator will be explained together since they are similar to the SeqGAN model proposed by Yu et al. (2017).

4.2.1 Fusion

The fusion takes candidate utterances and their word embedding as inputs to produce new representations in another latent space. After that, the new representations are summed up with different weights. The weights can either be proportional to the scores from the beam search algorithm or be set to trainable parameters (in our implementation they are trainable).

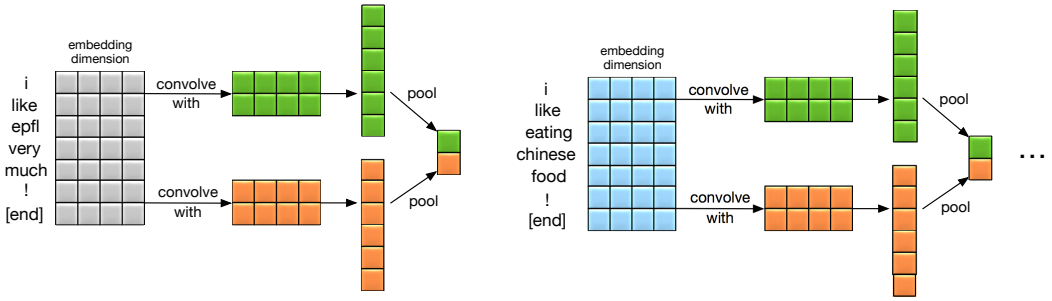


Figure 3: Illustration of the fusing operation

Above figure illustrates how the fusing operation is carried out. Consider two generated candidate utterances *i like epfl very much ! [end]* and *i like eating chinese food ! [end]* which are generated from the primary network. The word embedding of a sentence is firstly convolved with two filters which have size of $2 \times embedding\_size$, followed by a max pooling operation. The same convolutional operation is implemented over every candidate utterance. Finally, a weighted sum is carried out to combine the new representation of all candidates.

As one can easily tell that the fusing operation will lose time dependency information within an utterance. However, as the goal here is to find the averaged *approximate position* of the candidate utterances in the new latent space, the time dependency here is less important. It will be introduced that the time dependency information can be learned in the adversarial training procedure shown in following section.

4.2.2 ADVERSARIAL TRAINING

The novel GAN network proposed by Goodfellow et al. (2014) is a generative model which takes some noises as inputs, and a discriminator is introduced to act as a guider to force the generator to produce outputs that are close to the real data. GAN achieves impressive performance and are further developed such as DCGAN by Radford et al. (2015) which is capable to generate more realistic images. The WGAN introduced by Arjovsky et al. (2017) resolved the instability and diverge problem when training GAN over datasets defined in continuous space. To be more formal, we can denote:

- $p_z(z)$ as probability distribution function (PDF) of the input noises
- $p_x(x)$ as PDF of the real data
- $G(z;\theta_g)$ as outputs from the generative network parameterized by θ_g
- $D(x;\theta_d)$ as outputs from the discriminative network parameterized by θ_d

where x represents the *real* data and outputs from *G* are considered as *fake* data. *D* outputs a scalar representing the probability of inputs following the real data distribution. Hence, *G* and *D* are playing a two-play minmax game with value function $V(G,D)$:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))]$$

we can observe that this minmax game has a global optimum when $p_g = p_{data}$. In such a case, the generated data follow a distribution which is identical to that of the real data, and D outputs $\frac{1}{2}$ since it is confused by outputs from G. To better understand how GAN works, we implemented some experiments using a simple GAN to run on the MNIST dataset by LeCun et al. (1998), where both G and D are simple multilayer perceptions (MLP):

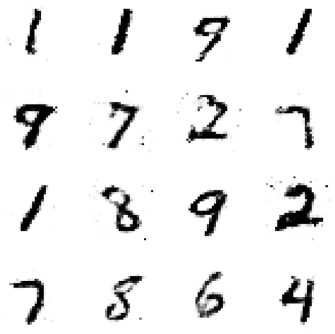


Figure 4: Sample generated digits

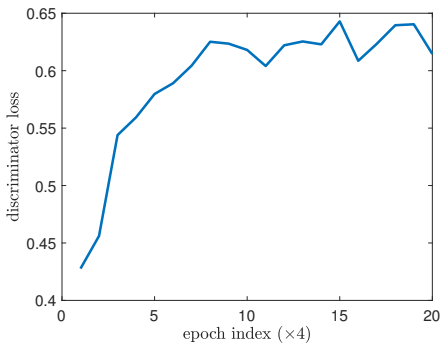


Figure 5: Discriminative loss

we can observe that the generated sample digits are already close to real world handwritten digits. The loss of D converges to around 0.62, which is not strictly $\frac{1}{2}$. This may due to the simple network structure of D and G which does not prevent issues like overfitting. It can be predicted that using more sophisticated networks is capable to provide better results.

It is well known that GAN is difficult to train because it requires appropriate pretrainings. In addition, G and D are required to be trained in a harmonious pace. Experimental results show that the discriminative model D is easier to train than the generative model G. This meets our expectation because discriminative models learn the boundaries between different classes while

generative models study the entire distributions. There are papers published trying to provide pragmatic suggestions for training GAN and its derivative networks shown in Gulrajani et al. (2017).

Although GAN has achieved impressive performance in generating data defined in continuous space such as images and videos shown by Mathieu et al. (2015), applying GAN on text and response generation is a more difficult task. A simple explanation is that words are represented by one-hot vectors, as a result, small amount of updates on these discrete numbers cannot bring changes on the words. To circumvent this problem, researchers propose SeqGAN which employs reinforcement learning to apply GAN in text generation tasks. This idea is adopted in our generative refining network, hence here we demonstrate how we implement and run it over the TBBT dataset.

We consider the generating process of a responsive utterance as a sequential decision making process. G is treated as an agent in reinforcement learning. The state is defined as the generated words till now and the action can be considered as the word to be generated next. D is responsible to provide reward to G via evaluating its generated utterance. In this case, we can search for the optimal policy that offers highest rewards using Monte Carlo search.

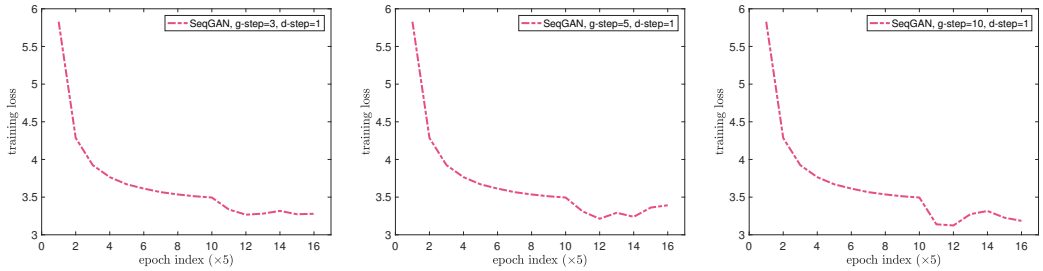


Figure 6: Training loss of SeqGAN, adversarial training phase starts at epoch 51

Above Figure 6 shows the training loss of a SeqGAN, with different number of G and D steps in adversarial training phase. We firstly pretrain the model with 50 epochs via reducing the negative log-likelihood (NLL) loss. Starting from epoch 51, we start adversarial training. One can observe that the adversarial training further reduces the loss in all three cases. The same experiment is repeated many times and results show that this reduction in loss is statistically significant.

We can also observe how different G steps and D steps within each adversarial training phase can affect the model performance. $Step_G = 3, Step_D = 1$ gives the most stable performance. When $Step_G = 5, Step_D = 1$, the model simply diverges and the loss is increasing. $Step_G = 10, Step_D = 1$ provides a more variant loss reduction than the other two, although the amplitude of loss reduction is also the largest. This result meet our expectation since it is difficult to control the adversarial training phase in GAN. To the authors' best knowledge, new efficient techniques for stabilizing the training for applying GAN on text haven't emerged and this is still a open research topic.

4.3 Using the Generative Refining Network

After demonstrating the entire architecture and how the generative refining network is organized, we now show the training algorithm and other essential details. In addition, some sample generated results from the network are provided and comparisons are made accordingly.

4.3.1 Training Procedure

To start with, we firstly introduce the *teacher forcing* technique. This technique is used by Li et al. (2017) which is close to the *professor forcing* method proposed by Lamb et al. (2016). This technique is introduced because during the adversarial training phase, G cannot be exposed directly to the ground truth answer, as can be observed from the network architecture. Consider a training batch within which G fails to learn while D is well trained, as a result, G is not clear what to generate

next, resulting in the divergence of entire training. The *teacher forcing* technique forces G to learn correctly via feeding ground truth into G with reward value 1. Hence, this technique forces G to generate utterances closing to the ground truth.

For training the generative refining network, the inputs are word embedding of the entire vocabulary and the generated $(Q, A, \vec{A}')$ tuples from the primary network (seq2seq model). The output is denoted as $\tilde{A}$ in response to the input question Q. Following algorithm illustrates how the network is trained:

Algorithm 1: Training procedure for the generative refining network

Input : vocabulary word embedding Σ, tuple $(Q, A, \vec{A}')$
Output : refined answer $\tilde{A}$ in response of Q
Fusion : Fusing the embedding of $\vec{A}'$ to obtain a initial state h_0, set it as the initial state of G
Pretrain: Pretrain generator G with candidate answers $\vec{A}'$

```
Call Pretrain;
for number of adversarial training rounds do
    for i=1, G-steps do
        Call Fusion;
        generate Ã from G given Q;
        compute reward r from D using Ã;
        update G on Ã using r;
        teacher forcing: update G on (Q, A);
    end
    for i=1, D-steps do
        generate Ã from G given Q;
        require (Q, A) from data;
        feed ← A as positive example, Ã as negative example;
        update D with feed
    end
end
```

Above algorithm shows the procedure of training the generative refining network which tries to improve answers from a primary network. To do this, we firstly use the fuse operation to obtain a initial state h_0 for the generator (LSTM) in adversarial training. After that, the adversarial training procedure is carried out trying to produce $\tilde{A}$ that is closer to the ground truth A. Because this entire training procedure is run for each (Q, A) set, the model can be trained in an on-line manner as the network size of G and D are manageable and the input dataset – candidate set $\vec{A}'$ – is also small.

4.3.2 Results Comparison

In this section, we provide some generated utterances $\tilde{A}$ from the generative refining network, given the question Q. In addition, we also included utterances generated by other models in response to same Q for comparisons. Comparisons are made among:

- GT, the ground truth answer.
- Vanilla, G without adversarial training.
- GRN-Fusion, the generative refining network with h_0 from fusion.
- GRN-Random, the generative refining network with randomly (Gaussian) initialized h_0.
- BS-1st, the candidate generated with largest score from beam search.
- BS-Random, the candidate picked randomly from results of beam search.

Table 2 in next page illustrates some results generated from each model corresponding to the same input question Q.

Input	leonard i don 't think i can do this
GT	what , are you kidding ? you 're a semi pro .
Vanilla	mishegas soundsational ferdinand october reinforce tip unlucky indulging
GRN-Fusion	jeez, what are you
GRN-Random	sweetie ,
BS-1st	why not ?
BS-Random	oh ,
Input	we don 't mean to interrupt we live across the hall
GT	oh , that 's nice .
Vanilla	to a
GRN-Fusion	oh , that 's nice .
GRN-Random	no
BS-1st	oh , that 's nice .
BS-Random	uh,
Input	i don 't know i 've never reneged on a proffer of X before
GT	let 's try just walking out .
Vanilla	shackles elliott asleep hippos
GRN-Fusion	smiling wocket lasso let 's try just
GRN-Random	let 's play
BS-1st	let 's try just story out .
BS-Random	let is
Input	oh no that 's probably not such a good idea civil' servants have a documented
GT	okay , well , thank you , again .
Vanilla	transitive hoi
GRN-Fusion	why would they
GRN-Random	but penny
BS-1st	no ,
BS-Random	let is

Table 2: Generated utterances from different models in responsive to Q

As we can observe from above table, the **vanilla** model performed poorly in all cases. This meets our expectation because the vanilla model only experienced the pretraining but does not experience the adversarial training, meaning that it does not know the ground truth which can result in bad responses. We can also observe that **BS-Random** performed a bit better than vanilla, e.g., utterances from BS-Random makes more sense than that of vanilla. This is reasonable because it is picked out by the beam search algorithm instead of generated randomly.

We can observe that the **GRN-Fusion** performed correlatively with the **BS-1st**. For example, in the second case, both two models give response which is the same to the ground truth. This result can be explained as follows, because the fusion takes weighted representation of candidate answers as initial state to G, when the first candidate is exactly the ground truth, the weight allocated to this candidate is increased significantly, combined with the effects of teacher forcing, GRN-Fusion produces utterance which is the same to the ground truth. In the fourth case, both two model fail and the generated utterances are irrelevant. Considering the remaining two cases, both models give utterances where some words are related to the ground truth. Comparing **GRN-Fusion** and **GRN-Random**, we see that the first model gives better performance, as shown in the first three cases. In the last case, all models fail to generate relevant results.

We experienced significant difficulty when training this model. For example, when $Step_D$ and $Step_G$ are not coherent, the training collapses then G begin to generate `[end], [end], [end], [end] ...` or `[oh], [oh], [oh], [oh] ...`. This problem can happen as shown in related publications, but by well tuning the pretraining and setting coherent steps for G and D, we can get resealable results. Hence we conclude that GRN-Fusion will not deteriorate results from the primary network and sometimes and generate better utterances. More experiments could be carried out to obtain conclusions having more statistical analysis and some quantitative measure such BLEU could be applied if necessary.

5 Conclusion and Future Works

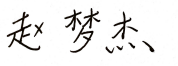
In this work we propose a novel architecture that can be employed in question answering problem and dialog generation tasks. It consists of two major submodules, namely the primary network and the generative refining network. The primary network can be any response generation models such as seq2seq model or more complex architectures like memory networks. The primary network is responsible to generate some candidate answers for a given input utterance. After that, the generative refining network comes in charge and tries to refine the outputs from the primary network through an adversarial training process. Depending on how the candidate answers are generated from the primary network, the entire architecture can be trained in manner of either end-to-end or on-line. However, the generative refining network can always be trained on-line if the size of models is manageable. Experimental results show that the new architecture can generate comparable results with the outputs from the primary network and are capable to generate better utterances.

Currently there is no similar architecture that can be considered as a reference, hence there are some pragmatic problems to be explored and addressed within the newly proposed generative refining network. For example in real world applications, the length of users' questions are unexpected hence the candidate answers also have unexpected length. However, in the new network architecture this length is required to define the model and our current solution is to pad all candidate utterances to the largest length. More scalable solutions can be proposed. In addition, the tuning of hyper-parameters is still a difficult task. For instance, the performance of the generative refining network is significantly determined by the number of $Step_G$ and $Step_D$. Tuning these parameters is more art than science. Overall, although there are shortages within this architecture, we still believe the generative refining network can be further developed to be more pragmatic and to improve responses generated from existing response generation models.

Acknowledgments

The author would like to thank Fei Mi and Prof. Faltings for offering this opportunity to start this project at LIA. The consistent supports and valuable advices are indispensable for conducting the project. The author also would like to thank all colleagues in the chatbot group for valuable and helpful discussions and suggestions.

Hereby I confirm all works are finished faithfully and independently without any plagiarism, adhere to the EPFL honor code.



Mengjie Zhao

June 11, 2017

REFERENCES

Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Bordes, Antoine and Weston, Jason. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*, 2016.

Bottou, Léon. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Guha, Ramanathan, Gupta, Vineet, Raghunathan, Vivek, and Srikant, Ramakrishnan. User modeling for a personal assistant. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pp. 275–284. ACM, 2015.

Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent, and Courville, Aaron. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Huszár, Ferenc. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Lamb, Alex M, GOYAL, Anirudh Goyal ALIAS PARTH, Zhang, Ying, Zhang, Saizheng, Courville, Aaron C, and Bengio, Yoshua. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pp. 4601–4609, 2016.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, Jiwei, Galley, Michel, Brockett, Chris, Spithourakis, Georgios P, Gao, Jianfeng, and Dolan, Bill. A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155*, 2016a.

Li, Jiwei, Monroe, Will, Ritter, Alan, Galley, Michel, Gao, Jianfeng, and Jurafsky, Dan. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016b.

Li, Jiwei, Monroe, Will, Shi, Tianlin, Ritter, Alan, and Jurafsky, Dan. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*, 2017.

Lotter, William, Kreiman, Gabriel, and Cox, David. Unsupervised learning of visual structure using predictive generative networks. *arXiv preprint arXiv:1511.06380*, 2015.

Mathieu, Michael, Couprie, Camille, and LeCun, Yann. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

Neal, Radford M. Geoffrey e. hinton,* peter dayan, brendan j. frey. *J. Mol. Spectrosc*, 111:403, 1985.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ritter, Alan, Cherry, Colin, and Dolan, William B. Data-driven response generation in social media. In *Proceedings of the conference on empirical methods in natural language processing*, pp. 583–593. Association for Computational Linguistics, 2011.

Salakhutdinov, Ruslan. *Learning deep generative models*. PhD thesis, University of Toronto, 2009.

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

Tiedemann, Jörg. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In Nicolov, N., Bontcheva, K., Angelova, G., and Mitkov, R. (eds.), *Recent Advances in Natural Language Processing*, volume V, pp. 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, 2009. ISBN 978 90 272 4825 1.

Weston, Jason, Chopra, Sumit, and Bordes, Antoine. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

Weston, Jason, Bordes, Antoine, Chopra, Sumit, Rush, Alexander M, van Merriënboer, Bart, Joulin, Armand, and Mikolov, Tomas. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

Yu, Lantao, Zhang, Weinan, Wang, Jun, and Yu, Yong. Seqgan: sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Zhang, Yizhe, Gan, Zhe, and Carin, Lawrence. Generating text via adversarial training. 2017.