ラベル付き文章を生成するためのConditional GANsの提案

Miao XIAO† and Qiang MA†

† Graduate School of Informatics, Kyoto University
Yoshida-honmachi, Sakyo-ku, Kyoto, 606–8232, Japan
E-mail: †mxiao@db.soc.i.kyoto-u.ac.jp, ††qiang@i.kyoto-u.ac.jp

Abstract In addition to images generation, Generative Adversarial Nets (GAN) has shown a promising ability to generate plausible text. For instance, SetiGAN used a multi-generator architecture to generate multi-class text. However, the generators of SetiGAN are independent to each other and need to train separately. Therefore, the training of SetiGAN is time-consuming. In this paper, we propose a novel model of conditional GANs with only one generator to generate high semantic text according to various desired labels. We introduce Variational Auto-Encoder(VAE) to encode the labels, and policy gradient to guide the generation. The basic idea is that with different input of conditional labels encoded by VAE, the generator can generate text of different class.

Key words Generative Adeversarial Networks(GAN), Deep Learning, Variational Auto-Encoder(VAE), Natural Language Processing.

1 Introduction

Generative Adversarial Nets (GAN) [1] is a novel model that shows promising abilities to generate both images and texts, such as PGGAN [2] and SeqGAN [3]. Generally, GAN trains two neuron networks simultaneously: one is called discriminator, which recognizes real from fake examples; another network generator imitates real samples to fool the discriminator. When training stops in the perfect equilibrium, the generator will capture the general training data distribution. At the same time, the discriminator would always be careful of whether its inputs are real or not. Though this is an effective way to approximate many intractable probabilistic computations, there are still several limitations when it comes to generating labeled text:

GAN is utilized initially to catch the distribution of images data, and the stochastic gradient descent algorithm is workable due to the consecutiveness of pixels. However in NLP tasks, the training data are usually word vectors without consecutiveness, so we must leverage the power of reinforcement learning to guide the generation process such as policy gradient [4].

In the conventional models of GANs to generate text, a single generator can catch only one distribution, which means no matter how many labels the input data contains, the generator will always regard them as only one class. One practical and thinkable method is to construct a multi-generator-one-discriminator structure, like SentiGAN [5]. On the other hand, GAN is so fragile a model that it is difficult to get both generator and discriminator convergent during adversarial training. Furthermore, mode collapse will also break the antagonism within them. When faced with more than one generator, the discriminator will quickly oscillate again.

Another sapiential way to handle multiple labels is to integrate data with the label vector before inputting them into the networks, like C-GAN [6]. Inspired by these successful models in image generalization, in this paper, we propose a model based on it and use Variational Auto-Encoder(VAE) to encode the label signal and original random noise into a latent space.

Generator in our model is expected to complete two tasks simultaneously: make the text more semantic and consistent with the desired label, so two kinds of loss function should be considered. It is a challenge to weight different loss functions manually.

Our main contributions are as follows:

- We propose a novel conditional GAN with only one generator to generate high semantic text according to various desired labels, and we use policy gradient and Monte Carlo Tree Search to drive the generation during adversarial training;
- We introduce VAE to encode label signals and then integrate it with random noise; furthermore, we construct another classifier to judge whether the label signal is realized or not and then use mutual information to build a loss function for VAE.
- In this way, the diversified training examples from a promising generator can be expected to expand the original dataset, especially for some datasets with high labor-consuming, e.g., fake news classification.

2 Related Work

GAN has shown its ' incomparable power in generating images, but how to make it be also promising in NLP is still a challenge task. Recent researchers focused on how to get more semantic and diversified text.

When facing up with sequences of discrete tokens, GANs has limitations. The primary reason is that discrete outputs from the generative model make it difficult to pass the gradient update from the discriminative model to the generative model. There are several related works that aim to overcome this problem: SeqGAN, and LeakGAN [8] which have been verified to perform very well in generating semantic sentences. SeqGAN utilizes the Policy Gradient algorithm as the optimization method.

Although GANs are at generating new random examples from a given dataset, some datasets have auxiliary information, such as labels, which is also close to the data and should be desirable for GANs to use. C-GAN showed a novel way to use additional information effectively. Their experiments were based on a handwriting data set, and so they first encode the number labels in one hot format and integrate it with the image matrix. Then both generator and discriminator will be conditioned on this signal. The results from this paper are pretty well, but when it comes to processing other composite datasets with complicated features, other encoding methods should be considered.

Context-Aware GANs [16] is a novel model aims to generate images according to the context. They also faced with the problem that the label signal is too sparse to make impact on the GANs, and they intelligently introduced VAE to encode the context as the conditional signals of theri GAN. The dense vector of label signal is also what we want in our model.

Wang et al. proposed a method based on seqGAN aims to generate various text data. They use Policy Gradient as a penalty to optimize the generator, which is contrary to seqGAN, and construct multiples generators for diversified usage but only one discriminator. During adversarial training, generators corresponding to each class are optimized by the discriminator in turn. Due to the instability of GAN, the whole training is time-consuming and intractable to get desired results.

3 Method

We proposed a novel GAN model with only one generator to generate high semantic text according to various desired labels. We introduced VAE to encode the label signal into a latent space, then integrate it with the text matrices and then input it into the network. Furthermore, we construct a new classifier to judge the label signal and send back the rewards to train both generator and VAE network.

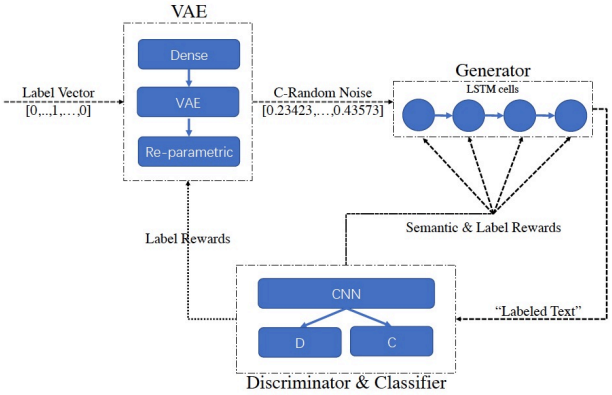

Figure 1 Model Overview

Figure 1 illustrates the whole process during adversarial training. Firstly, we use VAE and another dense layer to encode the label vector with a one-hot format into a latent dense format. The re-parametric layer will make sure the randomness and uniqueness of the input at every timestamp. Then the generator will use this conditional random noise to generate text vectors and input them into discriminator and classifier.

Discriminator in our model is just like the original one who can guide the generation to be more semantic. At the same time, the classifier will judge whether the sentence can be marked as the desired label or not. Then the label rewards along with the real-or-not signals, will be used to optimize both VAE and generator. We proposed a novel GAN model with only one generator to generate high semantic text according to various desired labels.

3 1 Dicriminator and Classifier

GAN trains two neural networks Generator and Discriminator simultaneously. The general discriminator tries to judge whether the input data is from the generator or not, and the generator tries to capture the distribution of the real dataset, then fools discriminator mistakes its inputs as real. The whole process can be formalized as a min-max game with the following loss function between G and D:

$$\begin{aligned} \max_G \min_D V(G,D) = E_{X\sim p_d(x)} log D(X) \\ + E_{Z\sim p_z(z)} log[1 - D(G(z))] \end{aligned} \tag{1}$$

where $p_d(x)$ is the data distribution from training data, $p_z(z)$ is a prior on input noise variables. Based on the framework of advesarial training, both the generator and the discriminator can be improved; but in practice, the discriminator is more likely to gain the advantage of rapid convergence. What we really want is the highly semantic text from generator and discriminator is just expected to suppourt this process. So

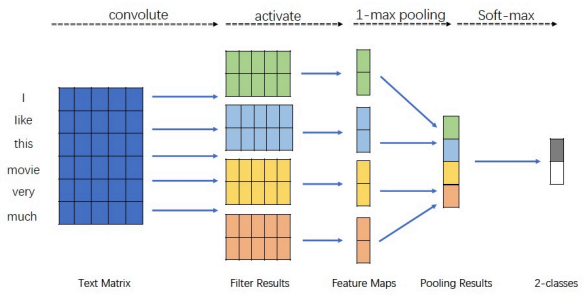

Figure 2 CNN Work Flow

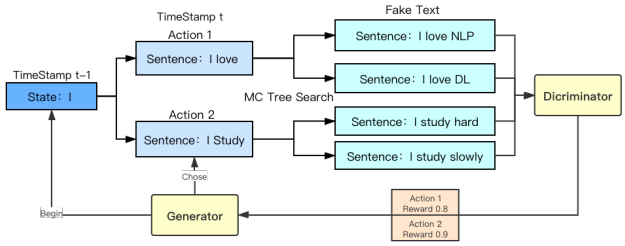

Figure 3 Policy Gradient Process

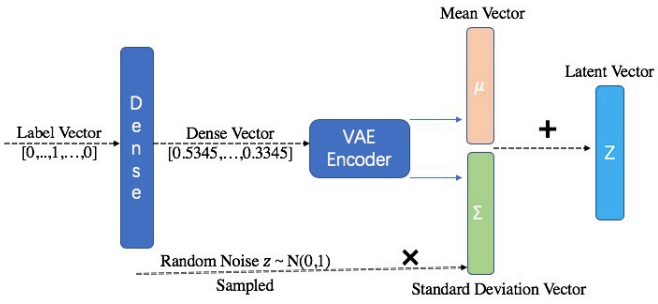

Figure 4 Encoding Layer

here we choose a light CNN-based network [7].

Figure 2 shows the workflow of the CNN network. For instance, if we want to classify whether the text "I like this movie very much" is positive or not, we may embed it firstly. Then input it into CNN. After convolution and pooling, we may get the final results from the softmax layer. It's worth noting that the filter we choose should have the same width as the dimension of the word vectors.

Then based on this structure, we add another classifier to get the label rewards. In consideration of the instability of GAN, if we add this network directly, the adversarial training will be held among three competitors. So the whole process will be harder to get convergent. Referring to another paper InfoGAN [9], the intermediate result of CNN can be expected to be reused, which means we need to construct an additional dense and softmax layer to receive the feature vector from CNN and then do the classification. So given a corpus with k classes, the loss function of our classifier can be formalized as follows:

$$J_{C(\theta)} = -\sum_{i=1}^{k} E_{X \sim P_{r_i}} log D_i(X) \qquad (2)$$

3.2 Generator

Our generator is based on Long Short-Term Memory (LSTM) [10]which is widely used in NLP tasks, such as Machine Translation, Text Classification, QA, and other fields. LSTM is based on RNN but adds memory cells and three control gates to effectively control historical information, especially in long text tasks.

The input gate controls the extent of information from the input vector at the current moment; forget gate controls how much the historical information influences the info at the present moment; the output gate controls the extent to which the value in the cell is used to compute the output. Unlike RNN, the hidden state at the previous moment is not entirely washed away. Thus this kind of structure enhances its ability to process long text sequences and solves the vanishing gradient problem.

The generator will be trained by maximizing equation 1 to be more semantic. On the other hand, as the text is sequential data and consists of word vectors, the generator can only generate words one by one at one specific timestamp. So the best way to train a generator is to give it right away as long as it generated a word. We here use the Policy Gradient to guide this process, as shown in Figure 3.

Firstly, we regard the generator as an agent, word token to be generated by it at every timestamp as action, and all the tokens generated previously as the state. Then the discriminator will judge the sentences from the generator is real or not and send back the possibility as a reward to guide the optimization of the generator. However, only the complete sentences can be judged, so for those incomplete sentences generated at a specific timestamp, Monte Carlo Tree Search is used for completing them. And we can see in figure that the next generation will be generating study.

Furthermore, our generator should also be guided by the label signal, which will be formed by the classifier. But it is too strict about letting every word be conditioned, and the sentences from MC tree search are also not our target. So only when the whole sentence is generated, then classifier will give back the label signal.

3.3 VAE Encoder

During adversarial training, the input of the generator is just random noise sampled from the normal distribution. The label vector is always encoded as a one-hot format. Compared to the dimension of noise, which can be a hundred of, one-hot formats may make little impact on the final

result. Hence, we introduce VAE to encode the label signal. We also use the re-parameter trick to make sure of the randomness of the output vector.

The encoder should also react with the feedback from the classifier. Here, we use the mutual information [11] to catch the loss of encoder.

$$J_{E(\theta)} = \sum_{i=1}^{k} E_{X \sim P_{r_i}} log D_i(X) + H(Z) \qquad (3)$$

As it is hard to get the mutual information directly, we use the lower bound of it, which contains only one more additional term H(Z) than classifier. H(Z) indicates the entropy of the random noise.

4 Experiments

We firstly hold pre-training for generator, discriminator, and classifier respectively to let them obtain the essential ability to do generation and classification.

We use the same movie review dataset as SentiGAN called Stanford Sentiment Treebank [12] and select the sentences which contain at most 15 words. Finally, the whole dataset contains 2133 positive sentences and 2370 negative sentences. To get more diversified sentences from the generator, we also input another IMDB movie review dataset [13] into the generator during pre-training. It is worth mentioning that this dataset doesn't contain labels, and we use it to make our generated text more diversified.

Then during adversarial training, there are two main processes: semantic training and sentiment training. For semantic training, we conducted general adversarial processes for generator and discriminator and make the parameters of VAE and classifier fixed. The following figures show the loss of generator and discriminator.

Another process sentiment training: we will make the generator and classifier fixed, then use the classifier to optimize the VAE layer to encode the label signal. As our network needs to be trained by both positive and negative data, we input them into our model in turn. In detail, in one sentiment train epoch, we conduct 5 epochs for positive training and 5 epochs for negative training orderly.

Furthermore, we set a hypothesis that different label signals should follow different distribution or they are the subdistribution of one major distribution. We set 1000 epochs for semantic training, positive and negative sentiment training respectively.

Finally, we use the trained generator to generate positive and negative reviews according to the label signal. We set two conditions here: fixed generator means whether the generator will be optimized or not during sentiment training; another VAE number means whether the positive and negative signals use the same VAE layer or not. But finally according to the experiments, no matter how these two conditions change, when input positive signal, the generator can only generate positive reviews with a rate at 41% which is classified by our classifier; and when turning to negative signal, the rate is 60%. Obviously, this VAE structure made effects on encoding the label signal.

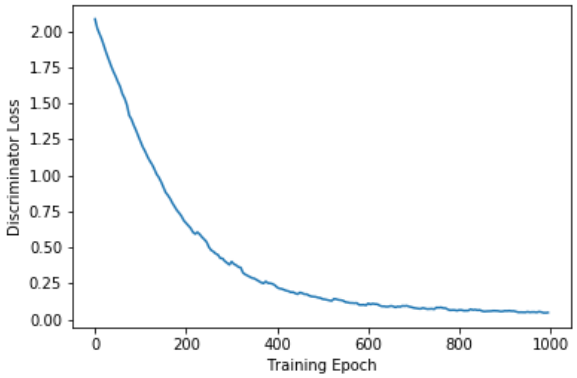

Figure 5 Discriminator Loss

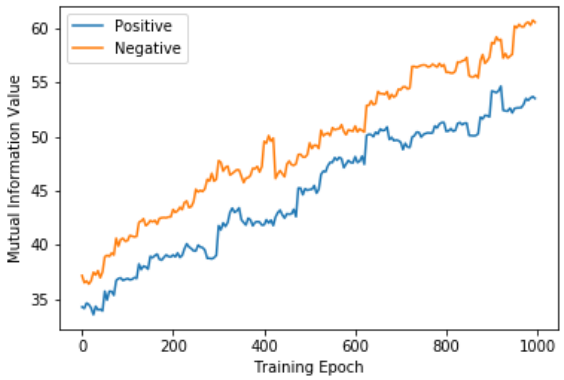

Figure 6 Mutual Information

The following figures indicate the loss of discriminator and the mutual information for positive and negative signals respectively. For classifier, we set it fixed after the pretrain. The loss of generator is weird and up and down repeatedly, so we will not put its' figure here.

The following table shows the positive and negative reviews generated by our model. They are terse and semantic.

5 Discussion and Future Work

In this paper, we proposed a VAE-based conditional GAN to generate various kinds of text with only one generator. To catch the label signal, we construct an additional classifier to return the mutual information to VAE. Our experiments proved that VAE structure made effects on encoding the label signal, on the other hand, We see from the experiments data that the rate of generated negative reviews is more than 50% but the positive is the opposite. As we conducted the

Table 1 Generated Examples

Positive Reviews
Fun and very watchable action movie.
A further extension of the great zombie genre.
An enjoyably dynamic romantic comedy.
Funny and appealing adventures started by an astronaut and an extraterrestrial in a far planet
Negative Reviews
Not as satisfying as i expected and a good scare ...
Complicated doesn't mean intelligent.
Stupid and not fun.
Entertaining but dragged down by heavy flaws.

negative process after the positive, we guess that the negative training covered the positive process. We will change the training order to examine this guess. On the other hand, as our semantic training maybe not enough at all, the generated sentences may be suffered from it and be always classified as negative by classifier. So we will conduct more epochs for semantic training before entering sentiment training.

Then, we will try to optimize the generator. In our model, the generator can be regarded as a black box and we can try other advanced neuron networks,e.g., GPT-2 [14] and BERT [15].

6 Acknowledgements

This work is partly supported by MIC SCOPE(172307001).

References

[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.

[2] Karras T, Aila T, Laine S, et al. Progressive growing of gans for improved quality, stability, and variation[J]. arXiv preprint arXiv:1710.10196, 2017.

[3] Yu L, Zhang W, Wang J, et al. Seqgan: Sequence generative adversarial nets with policy gradient[C]//Thirty-First AAAI Conference on Artificial Intelligence. 2017.

[4] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C]//Advances in neural information processing systems. 2000: 1057-1063.

[5] Wang K, Wan X. SentiGAN: Generating Sentimental Texts via Mixture Adversarial Networks[C]//IJCAI. 2018: 4446-4452.

[6] Mirza M, Osindero S. Conditional generative adversarial nets[J]. arXiv preprint arXiv:1411.1784, 2014.

[7] Kim Y. Convolutional neural networks for sentence classification[J]. arXiv preprint arXiv:1408.5882, 2014.

[8] Guo J, Lu S, Cai H, et al. Long text generation via adversarial training with leaked information[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.

[9] Chen X, Duan Y, Houthooft R, et al. Infogan: Interpretable representation learning by information maximizing generative adversarial nets[C]//Advances in neural information processing systems. 2016: 2172-2180.

[10] Sundermeyer M, Schlüter R, Ney H. LSTM neural networks for language modeling[C]//Thirteenth annual conference of the international speech communication association. 2012.

[11] Church K W, Hanks P. Word association norms, mutual information, and lexicography[J]. Computational linguistics, 1990, 16(1): 22-29.

[12] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. 2013.

[13] IMDb Datasets. https://www.imdb.com/interfaces/

[14] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI Blog, 2019, 1(8).

[15] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[16] Nakamura, Kenki, and Qiang Ma. "Context-Aware GANs for Image Generation from Multimodal Queries." International Conference on Database and Expert Systems Applications. Springer, Cham, 2019.