The 14th International Conference on
Computer Science & Education (ICCSE 2019)
August 19-21, 2019. Toronto, Canada

A Research on Generative Adversarial Networks Applied to Text Generation

Chao Zhang
School of Computer Science
Hubei University of Technology
Wuhan, China
1509177521@qq.com

Caiquan Xiong*
School of Computer Science
Hubei University of Technology
Wuhan, China
x_cquan@163.com

Lingyun Wang
School of Computer Science
Hubei University of Technology
Wuhan, China
763115717@qq.com

Abstract—**Using deep learning methods to generate text, a sequence-to-sequence model is typically used. This kind of models is very effective in dealing with tasks that have a strong correspondence between input and output, such as machine translation. Generative Adversarial Networks(GAN) is a generation model that has been proposed in recent years, which has achieved good results in generating continuous and divisible data such as images. This paper proposes an improved model based on GAN, specifically using the transformer network structure instead of the original general Convolutional Neural Network or Recurrent Neural Networks as generator, and using the reinforcement learning algorithm Actor-Critic to improve the model training method. By comparing experiments, and selecting the perplexity, the BLEU score, and the percentages of unique n-gram to evaluate the quality of the generated sentences. The results show that the improved model proposed in this paper perform better than comparative models on above three evaluation indexes. This verifies its effectiveness in text generation.**

Keywords ***—Generative adversarial networks; Transformer; Actor-Critic algorithm; Text generation;***

I. INTRODUCTION

Extending GAN training to discrete spaces and discrete sequences has been a very active area. GAN training in the settings of the continuous output environment supports fully divisible calculations, allowing the gradient to pass through the discriminator to the generator. The non-differentiability of discrete sequences hinders the return of the gradient, which lead researchers to avoid this problem either by working in a continuous domain or by considering reinforcement learning methods.

GAN has been applied to conversational generation, the rewards in models proposed by Li[1] are not provided by the discriminator in the confrontational environment, but by the scores of specific tasks, such as the BLEU score. Improvements in the confrontational assessment and good results in the manual assessment were shown compared to the maximum likelihood training baseline. Their approach is to apply reinforcement learning and Monte Carlo sampling to the generator. Zhang[2] proposed a text generation based on GAN. CNN is used as a discriminator in the model, and moment matching is used to solve the problem of error return. Reed[3] proposed using GAN to generate corresponding images based on text descriptions.

The use of efficient gradient approximators instead of non-differentiable sampling operations proposed by Jang[4] has not shown strong results for discrete GAN. Recent unbiased and low variance gradient estimation techniques, such as Tucker et al.[5], may be proven to be more effective.

WGAN-GP was proposed by Gulrajani[6] to generate text one-time by using a one-dimensional convolution network, avoiding the problem of processing backpropagation through discrete nodes. Hjelm[7] proposed a solution that uses a model of boundary search GAN and importance sampling to generate text. In the model proposed by Rajeswar[8], the discriminator directly operates on the continuous probability output of the generator. However, to achieve this, they re-sampled the text with traditional autoregressive sampling because the input to the RNN is predetermined. Che[9] used the output of the discriminator instead of the standard GAN target to optimize the low variance target.

In this paper, a new model based on GAN is proposed. Specifically, using the self-attention mechanism of the transformer network structure as a generator, the advantage is that it can capture the structural relationship within the sequence itself, and can also correlate information at different positions of the sequence. This solves the long sequence dependency problem and allowing parallel computation to speed up the generator training process. The discriminator is a kind of CNN architecture; at the same time, adopting actor-critic algorithm in reinforcement learning to improve the training strategy of GAN.

II. TRANSFORMER-BASED GENERATOR STRUCTURE IMPROVEMENT

A fundamental challenge in generating real text involves the property of the RNN itself. During the training process, the RNN generates words in turn from the previously generated words, but the errors accumulate in proportion to the length of the sequence. The first few words seem reasonable, However, as the length of the sentence increases, the quality of the generated sentence deteriorates rapidly. This phenomenon is called exposure bias. In order to solve this problem, Bengio[10] proposed a scheduled sampling method. However, research by Huszár[11] shows that scheduled sampling is a fundamentally inconsistent training strategy because it produces large, unstable results in practice. In order to overcome the above problems, this paper proposes a transformer-based generator structure, which is used as a generator of the GAN.

The specific transformer-based generator structure belongs to the architecture type of the encoder-decoder. The encoder is made up of 6 identical layers stacked. Each layer contains two sublayers. The first layer is a multi-head self-attention, and the second layer is a feedforward network with a fully-connected position. The two sub-layers each use a residual connection, followed by a layer normalization. Then, the output of each sublayer is LayerNorm(x + Sublayer(x)), where Sublayer(x) is a function implemented by the sublayer itself. To facilitate residual joins, all sublayers and embedding layers in the model have an output dimension of $d_{model} = 512$. The d_{model} here refers to the dimension of embedding. For example, if the input has n words, it will be a matrix of $n \times d_{model}$. The decoder is the same as stacking 6 identical layers. Based on the two sublayers in the encoder layer, the decoder adds a third sublayer that execute multi-head attention on the output of the encoder stack. As with the encoder, each sublayer still uses a residual connection and then layer normalization. At the same time, the self-attention sub-layer in the decoder stack is also modified to prevent the position from being noticed. This combination of masks embeds the output in one position, ensuring that the prediction of the position i can only rely on a known output that is less than i.

In addition to the attention sublayer, each layer in the encoder and decoder also has a fully connected feedforward neural network that is applied identically and individually at each location. It consists essentially of two linear transformations and is activated using the ReLU function.

Since there is no CNN and RNN structure in the generator used in this paper, the structure has no ability to process sequence information. In order to join sequence position information, it is added relative position or absolute position information when processing the input sequence. To this end, "location coding" is added to the input embedding of the encoder and decoder by a flexible calculation method. So that the position coding and the input embedded dimensions d_{model} are the same and can be added.

The generator network architecture used in this paper is intended to overcome the shortcomings of CNN and RNN, namely long dependency problems. Its advantage is the use of multi-head self-attention, the self-attention mechanism makes each word and all other words in the homologous sentence calculate the attention value, so no matter how long they are, the maximum path length is the same. It's just 1. This can capture dependencies between long distances. The use of long heads allows the model to learn about the different information contained in the subspace. In terms of parallel computing, the multi-head attention is similar to CNN which do not depend on the calculation of the previous moment, and the parallel computing can be performed well, even the effect is better than CNN.

For the discriminator of GAN, this paper uses the special CNN architecture proposed in Kim[12]. It consists of a convolutional layer and a pooled layer of the largest pooling operation on the entire sentence that maps each feature. A sentence of length T (filled if necessary) is represented as a matrix $X \in R^{K \times T}$ by concatenating its words into columns, that is, the *t-th* column of X is x_t.

III. GENERATION VS. NETWORK TRAINING STRATEGY BASED ON ACTOR-CRITIC ALGORITHM

The Actor-Critic(AC) is a more classical algorithm in reinforcement learning[13]. While most reinforcement learning algorithms focus on learning value functions, such as value iterations and Temporal Difference learning, or direct learning strategies, such as strategic gradient methods, the AC learns both as a strategy actor and as value. The critics feature.

Based on the AC, this paper improves the training strategy for GAN. The specific method is as follows.

In the model of this paper, the logarithm of the probability estimated by the discriminator is used as the reward value, as in Equation (1).

$$\mathrm{r}_t \equiv \log_\phi D(x) \tag{1}$$

Then the value function of the critic is Equation (2)

$$\mathrm{R}_t = \sum_{s=t}^{T} \gamma^s r_s \tag{2}$$

Where γ is the discount factor for each position in the sequence.

The model in this paper is not completely differentiable due to the sampling operation of generating the probability distribution of the next word. Therefore, in order to train the generator, the gradient with respect to its parameter θ can be estimated by the policy gradient. Here the generator seeks to maximize the cumulative total reward R and through optimizing the generator's parameter θ by performing a gradient rise on $E_{G(\theta)}[R]$.

We can find an unbiased estimator as $\nabla_\theta E_G[R_t] = \mathrm{R}_t \nabla_\theta \log_\theta G_\theta(\hat{\mathrm{x}}_t)$. By using the learned value function as the baseline $b_t = V^{G}(x_{1:t})$ produced by the critic, the variance of the gradient estimator can be reduced. The gradient of the generator of a single marker $\hat{x}_t$ is Equation (3)

$$\nabla_\theta E_G[R_t] = (R_t - b_t) \nabla_\theta \log_\theta G_\theta(\hat{x}_t) \tag{3}$$

In reinforcement learning, the value $(R_t$-$b_t)$ can be interpreted as an estimate of $A(a_t, s_t) = Q(a_t, s_t) - V(s_t)$. Here, the action at is the word selected by the generator $a_t \equiv \hat{x}_t$, and the state st is the current word generated up to the point $s_t \equiv x_1, \ldots, \hat{x}_t$. This method is an AC architecture in which G determines the strategy $\pi(s_t)$.

In a single sequence, the reward for each time step is considered such that the words generated at time step t affect the rewards received at that time step and subsequent time steps. The generator needs to maximize the total return R. The complete generator gradient is given by Equation (4)

$$\nabla_\theta E_G[R] = E_{x_t \sim G}[\sum_{s=t}^{T}(R_t - b_t)\nabla_\theta \log(G_\theta(x_t))]$$

$$= E_{x_t \sim G}[\sum_{t=1}^{T}(\sum_{s=t}^{T}\gamma^s r_s - b_t)\nabla_\theta \log(G_\theta(x_t))] \quad (4)$$

The above equation states that the gradient of the generator associated with the generation will depend on all future discount rewards (s ⩾ t) assigned by the discriminator. For non-zero gamma discount factors, the generation will be penalized for greedily choosing words that alone achieve high returns at that time. For a complete sequence, sum all the generated words for the t=1:T time step.

Finally, like the traditional GAN training, the discriminator of this paper will be updated according to the gradient, as shown in the following

$$\nabla_{\theta_d} \frac{1}{m}\sum_{i=1}^{m}\left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right] \quad (5)$$

As shown in Figure 1, the improved framework is a GAN model consisting of a generator and a discriminator. The generator is a transformer architecture, which is similar to encoder and decoder architecture for text generation. The discriminator uses the special CNN architecture proposed by Kim. This paper name the improved model tranGAN.

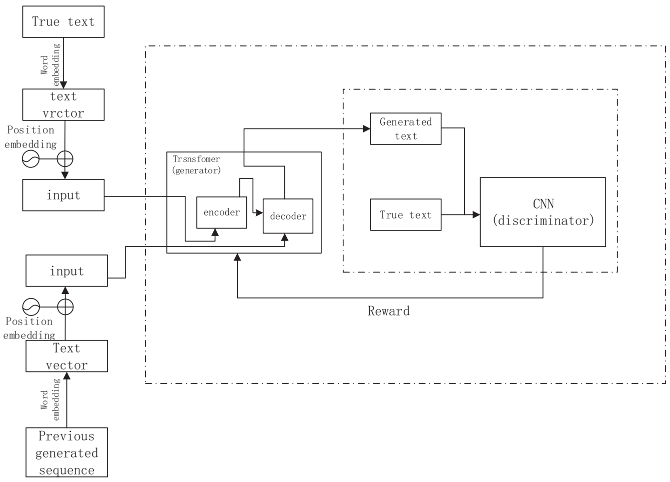

Figure 1. Improved model framework

The generator and discriminator of the entire model require alternating training. For the discriminator network, the method of gradient descent in supervised learning can be used to use the real text data and the text generated by the generator as the training data of the discriminator, and the performance of the discriminator is continuously improved. At the same time, the AC is used to optimize the training instruction of the discriminator to the generator, and the logarithm of the probability estimated by the discriminator is used as the reward value. The critic's value function is calculated by this reward value and passed back to the generator to guide the direction it should optimize. The generator is designed to get the maximum return value. Training ends until the discriminator cannot distinguish between the text generated by the generator and the actual text.

IV. Experiment

The data set used in this experiment is the Penn Treebank dataset with 10,000 unique words, where the training set contains 930,000 words, the validation set contains 74,000 words, and the test set contains 82,000 words.

The assessment of the generated model remains an open research question. So far, in the field of text generation, researchers have not found a unified evaluation index. This article uses the following three evaluation indicators to evaluate the quality of the generated sentences:

(1) Perplexity is the most widely used internal evaluation indicator in the language model. It mainly estimates the probability of occurrence of a sentence based on each word, and uses the length of the sentence for regularization.

(2) Calculate the percentage of the number of unique n-grams generated by the generator, which measures the diversity of the generated sentences. Obviously, the percentage of the number of unique n-grams is relatively low, which proves that there are many repeated words in the sentence or sentence.

(3) The BLEU scoring method proposed by IBM in 2002 is a measure of the similarity between generated sequences and references (training data). This method is more commonly used in machine translation. If the similarity between the two is greater, we think that the effect of the generation is better.

A. *Perplexity*

The results of calculating the perplexity of each model with the number of training epochs are shown in Table 1.

Table 1. Perplexity

model	epochs	50	100	150	200	250
seq2seq		812.56	650.21	290.11	187.34	130.33
seqGAN	perplexity	762.69	510.18	304.58	201.64	107.56
tranGAN		787.82	632.23	224.28	163.88	95.96

The graph drawn by the above table is shown in Figure 2

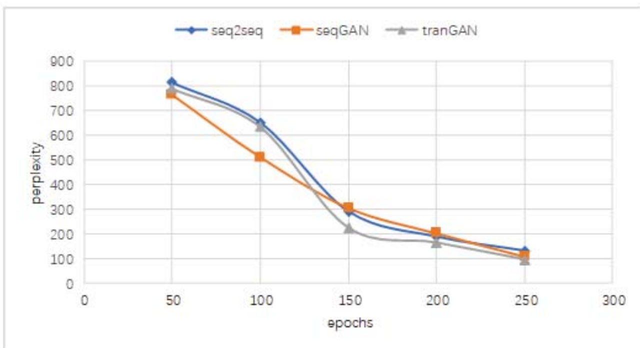

Figure 2. Perplexity varies with the number of training epoch

Through the observation of the graph, we can see more clearly that when the number of training epochs is 50 epochs, the perplexity of the three models is not much different. When the number of training epochs is 100 epochs, the perplexity of seqGAN model is better than seq2seq. Compared with the tranGAN model, the perplexity of the tranGAN model decreased rapidly when the number of training epochs was 150, and it was better than the seq2seq and seqGAN models at the end of the 150 epochs. That is to say, seqGAN can quickly reduce the perplexity in the early stage of training, and the perplexity of tranGAN is slow in the early stage of training, and declines rapidly in the middle of training, and tends to be flat in the later stage. This proves that the improved model can be improved in terms of model perplexity compared to the comparison model.

B. The percentage of the unique n-gram

As mentioned earlier, GAN is more likely to encounter mode-collapse and thus reduce language diversity. Unlike image generation, we can evaluate the mode-collapse index of text by directly calculating the statistics of n-grams. Its calculation method is very simple, which is to count the number of unique n-grams and then divide by the total number of n-grams.

The percentage of the unique n-gram for these three models when the number of epochs is 250, as shown in Table2 below.

Table2. N-gram statistics when the number of epochs is 250

model	% unique 2-gram	% unique 3-gram	% unique 4-gram
seq2seq	44.9	78.2	90.1
seqGAN	48.7	80.4	91.6
tranGAN	49.2	79.8	93.3

The diversity of samples is assessed by n-gram statistics, which is only a rough representation of sample quality. Because of the few samples taken from this model, although the percentage of the unique 4-gram is relatively high, there is still the problem of losing diversity. It is obviously not enough to capture the diversity of natural language by these indicators alone.

Although the mode-collapse problem still exists to some extent, the sample produced by the tranGAN model has improved. The samples generated by the tranGAN model are more realistic than the seq2seq and seqGAN models in the initial model. The tranGAN model training method makes it more robust to the sampling operation.

C. BLEU

BLEU (Bilingual Evaluation understudy) is used in the machine translation evaluation index to analyze the degree of co-occurrence of n-tuples in the candidate translation and the reference translation. In this experiment, the calculation method of BLEU can be changed slightly. The first step of calculation method is as follows

$$CP_n(C,S) = \frac{\sum_i \sum_k \min(h_k(c_i), \max_{j\in m} h_k(s_{ij}))}{\sum_i \sum_k h_k(c_i)} \tag{6}$$

Where c_i represents the text, and $S_i = \{s_{i1}, s_{i2}, \ldots, s_{im}\} \in S$ represents the test set text that is homologous to the training set. n-grams represent a set of phrases of n word lengths, ω_k represents the *k-th* possible n-grams. $h_k(c_i)$ denotes the number of occurrences of ω_k in the generated text c_i, $h_k(s_{ij})$ denotes the number of occurrences of ω_k in the test set text s_{ij}, k denotes the number of possible n-grams, and then calculates the penalty factor,

$$b(C,S) = \begin{cases} 1 & if \quad l_c > l_s \\ e^{1-\frac{l_s}{l_c}} & if \quad l_c \le l_s \end{cases} \tag{7}$$

Where l_c represents the length of the generated text c_i, l_s represents the effective length of the test set text s_{ij} (when there are multiple test set texts, the closest length to l_c is selected), and finally the following Equation(8) is calculated using the calculation results of Equation(6) and Equation(7),

$$BLEU_N(C,S) = b(C,S)\exp(\sum_{n=1}^{N} \omega_n \log CP_n(C,S)) \tag{8}$$

When the number of training epochs is 250, the bleu scores of the three models are shown in Table3 below.

Table3. $BLEU_N$ statistics when the number of epochs is 250

model	BLEU-2	BLEU-3	BLEU-4
seq2seq	0.76	0.43	0.10
seqGAN	0.82	0.45	0.16
tranGAN	0.85	0.52	0.20

As can be seen from Table3, the tranGAN model is generally superior to the seq2seq model and the seqGAN model in BLEU score, which also proves that the improved model generates better text quality than the other two models as the comparison model. In particular, the improved model on the BLEU-4 score has a large improvement compared with the former two.

D. Sample

In the experiment, the samples produced by the three models after training for 250 epochs on the PTB data were taken out, and a part of the samples were extracted and listed in Table4 below to compare the actual effects of the texts generated by them.

Table4. Partial sample comparison

model	sample
seq2seq	sample1： it would would would would be N N foreign foreign or <eos> <eos> <eos> sample2： the the the the on loans at at at large u.s. money <eos> <eos> banks <eos> federal funds N N high N

seqGAN	sample1： N N low N N N near closing bid N N N offered <eos> reserves traded among commercial banks for
	sample2： are a guide to general levels but do n't always represent actual transactions <eos> prime rate N N N <eos>
tranGAN	sample1 ： gloomy forecast south korea has recorded a trade surplus of $ N million so far this year <eos> from January
	sample2：began in N stopped this year because of prolonged labor disputes trade conflicts and sluggish exports <eos> government officials said

From the above table, the samples generated by the tranGAN model are better than the samples generated by the seq2seq model and the seqGAN model. The comparison between the tranGAN model and the other two models shows that. The sentences produced by tranGAN are usually more grammatically and semantically reasonable. It shows the “smoothness” and interpretability of the model. Then, the sentence structure is more reasonable, and the phrase structure is also more used, which proves that the improved model can effectively capture some structural features in the sentence. In contrast to the seq2seq model and the seqGAN model, although the seqGAN model looks better than seq2seq, they all show varying degrees of mode-collapse and are not as good as the improved model in terms of semantics and sentence internal phrase structure.

V. Conclusion

This paper proposes a new model architecture for generating text using confrontational training, called tranGAN, which is a GAN model based on the transformer architecture. Using the multi-head attention of the transformer architecture as well as its excellent performance in capturing long-distance dependencies relationship, internal semantics and phrase structures in sentences, and using the reinforcement learning algorithm actor-critic to improve the training strategy of this model to achieve the purpose of processing discrete sequences, Thus the GAN which is widely used in image generation, can be effectively used in text generation tasks. Through comparative experiments, it is proved that the proposed model and related methods provide better performance, can generate relatively reasonable and realistic sentences.

Acknowledgment

This research is supported by National Key Research and Development Program of China under grant number 2017YFC1405403, and Green Industry Technology Leading Project (product development category) of Hubei University of Technology under grant number CPYF2017008.

References

[1] Li J, Monroe W, Shi T, Jean S, Ritter A, Jurafsky D. Adversarial learning for neural dialogue generation. arXiv preprint arXiv:170106547. 2017.

[2] Zhang Y, Gan Z, Carin L. Generating text via adversarial training. NIPS workshop on Adversarial Training, 2016.

[3] Reed S, Akata Z, Yan X, Logeswaran L, Schiele B, Lee H.Generative adversarial text to image synthesis. arXiv preprint arXiv:160505396. 2016.

[4] Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:161101144. 2016.

[5] Tucker G, Mnih A, Maddison C J, Lawson J, Sohl-Dickstein J. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. Advances in Neural Information Processing Systems, 2017: 2627-2636.

[6] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, CourvilleA C. Improved training of wasserstein gans. Advances in Neural Information Processing Systems, 2017: 5767-5777.

[7] Hjelm R D, Jacob A P, Che T, Trischler A, Cho K, BengioY. Boundary-seeking generative adversarial networks. arXiv preprint arXiv:170208431. 2017.

[8] Rajeswar S, Subramanian S, Dutil F, Pal C, Courville A.Adversarial generation of natural language. arXiv preprint arXiv:170510929. 2017.

[9] Che T, Li Y, Zhang R, Hjelm R D, Li W, Song Y, et al.Maximum-likelihood augmented discrete generative adversarial networks. arXiv preprint arXiv:170207983. 2017.

[10] Bengio S, Vinyals O, Jaitly N, Shazeer N. Scheduledsampling for sequence prediction with recurrent neural networks. Advances in Neural Information Processing Systems, 2015: 1171-1179.

[11] Huszár F. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? arXiv preprint arXiv:151105101. 2015.

[12] Kim Y. Convolutional neural networks for sentenceclassification. arXiv preprint arXiv:14085882. 2014.

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