



Article

WRGAN: Improvement of RelGAN with Wasserstein Loss for Text Generation

Ziyun Jiao [†]  and Fuji Ren ^{*,†} 

Faculty of Engineering, Tokushima University, Tokushima 770-8506, Japan; c501947010@tokushima-u.ac.jp

* Correspondence: ren@is.tokushima-u.ac.jp

† These authors contributed equally to this work.

Abstract: Generative adversarial networks (GANs) were first proposed in 2014, and have been widely used in computer vision, such as for image generation and other tasks. However, the GANs used for text generation have made slow progress. One of the reasons is that the discriminator's guidance for the generator is too weak, which means that the generator can only get a "true or false" probability in return. Compared with the current loss function, the Wasserstein distance can provide more information to the generator, but RelGAN does not work well with Wasserstein distance in experiments. In this paper, we propose an improved neural network based on RelGAN and Wasserstein loss named WRGAN. Differently from RelGAN, we modified the discriminator network structure with 1D convolution of multiple different kernel sizes. Correspondingly, we also changed the loss function of the network with a gradient penalty Wasserstein loss. Our experiments on multiple public datasets show that WRGAN outperforms most of the existing state-of-the-art methods, and the Bilingual Evaluation Understudy (BLEU) scores are improved with our novel method.

Keywords: GAN; text generation; RelGAN; Wasserstein loss; unsupervised learning; natural language processing



Citation: Jiao, Z.; Ren, F. WRGAN: Improvement of RelGAN with Wasserstein Loss for Text Generation. *Electronics* **2021**, *10*, 275. <https://doi.org/10.3390/electronics10030275>

Academic Editors: Juan M. Corchado, Stefanos Kollias and Javid Taheri
Received: 1 January 2021
Accepted: 20 January 2021
Published: 25 January 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A generative adversarial network (GAN) [1] is an unsupervised learning method that learns by letting two neural networks play against each other. This method was proposed by Ian Goodfellow and his colleagues in 2014. With the improvement of the theory, GANs have gradually shown their great potential. Moreover, GANs have produced many fancy computer vision applications, such as image generation, image conversion, style transfer, and image restoration. GANs have made many computer vision achievements, but their development is relatively slow in natural language processing (NLP). There are two main problems:

1. When a GAN faces discrete data, the discriminator cannot pass the gradient to the generator through backward propagation [2]. For example, for continuous data, if we will output a picture with a pixel value of 1.0, then we can change this value to 1.0001 through backward propagation. However, if the word "penguin" is output for discrete data, the network cannot change it to "penguin + 0.001" through backward propagation because "penguin + 0.001" is not a word.
2. The training process for GANs is unstable. We need to balance the generator and the discriminator carefully. Moreover, the generating task is much more complicated than discrimination. Simultaneously, the discriminator's guidance for the generator is too weak [3], and the direction contains too little information. For the generator, it can only get a "true or false" probability in return. Furthermore, the discriminator may even "cheat". Since the real sample uses one-hot vectors, the discriminator does not need to judge whether the generated data distribution is closer to the real data. It just needs to identify whether only one item of the current data is 1, and the rest are all 0.

For the first problem, there are currently two effective solutions.

The first solution is the combination of a GAN and reinforcement learning (RL) for sequence generation, such as the Sequence GAN (SeqGAN) [4] proposed by Yu et al. with the REINFORCE algorithm. This solution focuses on dealing with non-differentiable problems caused by discrete data by considering RL methods or reformulating problems in continuous space [5]. Using this method will make the GAN more challenging to train and will cause the mode collapse problem.

The second solution is Gumbel–Softmax relaxation [6,7]. The most representative network is RelGAN [5], which was proposed by Nie, W. et al. in 2018. RelGAN can be divided into three parts: a relational-memory-based generator [8], Gumbel–Softmax relaxation [6,7] for training GANs, and multiple representations embedded in the discriminator [5]. This model performs very well on many datasets. However, RelGAN still has disadvantages in the second problem.

For the second problem, Wasserstein GAN (WGAN) and Wasserstein GAN Gradient Penalty (WGAN-GP), proposed by Martin Arjovsky and Ishaan Gulrajani et al. [9,10], respectively, provide a theoretical solution. WGAN converts the discriminator from a binary classification task into an approximately fitting Wasserstein distance. Forcing the discriminator to calculate the distance between the distribution of the generating data and the true data also prevents the discriminator from “cheating”. At the same time, it also provides more accurate guidance information to the generator, not just the probability of being “true or false”. Because the “weight clipping” strategy in WGAN will cause most of the weights to approach two extremes, WGAN-GP was proposed, which uses a “gradient penalty” to replace “weight clipping”. This strategy makes the training more stable and increases the quality of the generated image.

Unfortunately, WGAN is mostly used in computer vision, and there are few applications in text generation. On the other hand, using Wasserstein loss to replace the RSGAN [11] loss in RelGAN does not perform well for real-world data. We found that the gradient would disappear, and the discriminator loss was almost equal to 0 during training. A detailed description of the performance of RelGAN using Wasserstein loss can be found in Section 4.2.2.

In this work, we propose a new architecture based on RelGAN and WGAN-GP named WRGAN. The improved model can effectively solve the two issues identified above. We rebuilt the discriminator architecture with a one-dimensional convolution of multiple different kernel sizes and residual modules [12]. Correspondingly, we also modified the generator and discriminator loss functions of the network with gradient penalty Wasserstein loss. Then, we used the discriminator and the generator with relational memory coordinated by Gumbel–Softmax relaxation to train the GAN model on discrete data. We analyzed and presented the experimental results on multiple datasets and the influence of hyperparameters on the model. The data samples generated from each dataset can be found in Appendix A. Our experiments demonstrate that our model outperforms most current models on real-world data. The rest of the paper is organized as follows: Section 3 presents the methodology of the review. The obtained results are presented in Section 4. Section 5 presents a discussion of the results and the conclusions.

2. Related Works

With the improvement of GAN theory, GANs have also made some progress in text generation. In SeqGAN [4], the error is regarded as a reward for reinforcement learning by training in a feed-forward manner. An exploration model of reinforcement learning is used to update the generator network. SeqGAN created a mode for GAN in the field of text generation. Many subsequent models also rely on reinforcement learning (RL) algorithms. For example, MaliGAN [13] proposed a new loss function of the generator to replace the Monte Carlo tree search [14] and got better results. RankGAN [15] changed the original discriminator from a binary classification model to a sorting model, and changed from a learning to a ranking problem. LeakGAN [16] leaked the characteristic information of

the high-level discriminator to the manager module to guide the generator to generate long texts. MaskGAN [17] used the actor–critic algorithm in reinforcement learning to train the generator and used maximum likelihood and stochastic gradient descent to train the discriminator. DP-GAN (Diversity-Promoting GAN) [18] was proposed with a focus on diversified text generation. The author improved the discriminator based on SeqGAN and proposed a discriminator based on the language model. SentiGAN [19] has multiple generators and a multi-class discriminator. Multiple generators are trained simultaneously, aiming to generate text with different emotion labels without supervision. Moreover, it establishes a penalty-based goal in the generators to force them to produce diverse examples of a specific emotional label. Matt Kusner [6] proposed a new method for dealing with discrete data—Gumbel–Softmax—thus solving the “indifferentiable” problem. RelGAN [5] used relational memory [20] instead of Long Short-Term Memory (LSTM) [21] as the generator to obtain more vital expression ability and better model ability for long texts. At the same time, RelGAN used the Gumbel–Softmax relaxation model to simplify the model and replaced the reinforcement learning heuristic algorithm.

For GAN models that were not designed for text generation, we will first introduce a distance measurement method called “Earth-Mover (also called Wasserstein) distance”, $W(q, p)$, which is informally defined as the minimum cost of transporting mass in order to transform the distribution q into the distribution p (where the cost is the mass times the transport distance). Under mild assumptions, $W(q, p)$ is continuous everywhere and differentiable almost everywhere [10]. The “Earth-Mover distance” $W(q, p)$ can be defined as:

$$W(q, p) = \min_{\gamma} \mathbb{E}_{(q,p) \sim \gamma} \|q - p\|, \quad (1)$$

where γ is a probability distribution that satisfies the constraints:

$$\int \gamma(q, p) dq = P_g(q), \int \gamma(q, p) dp = P_f(p). \quad (2)$$

Instead, WGAN was proposed to use the Wasserstein distance as the loss to solve the unstable problem in GAN training. WGAN theoretically points out the defects of the original GAN. WGAN-GP was used to propose another truncation pruning strategy—“gradient penalty”—to make the training process more stable and achieve higher-quality generation results. The f-GAN [22] used variable dispersion to minimize the training of a generative adversarial network of generative neural samplers. These methods also have reference values for text generation.

Beyond GANs, ARAEs (adversarially regularized autoencoders) [23] represent a new method that can limit the encoded content information to an autoencoder and prevent the autoencoder from learning the identity function. After training, the generator and encoder parts of the autoencoder can be used as a generative model. The PHVM (planning-based hierarchical variational model) [24] first plans a sequence of groups and then realizes each sentence depending on the planning result and the previously generated context, thereby decomposing long text generation into dependent sentence generation sub-tasks. These methods provide us with new ideas for text generation.

In general, there have been many excellent variants of GANs in the field of text generation in recent years. Nevertheless, there are still many problems to be solved in text generation, which also means that there is excellent development potential. Based on this point of view, we research GANs in the text generation field and improve the models.

3. WRGAN

With respect to the problem that the discriminator cannot back-propagate the gradient to the generator after experimental testing, we believe that using the Gumbel–Softmax relaxation technology is better than using reinforcement learning (RL). Therefore, we choose Gumbel–Softmax relaxation. On the other hand, since the LSTM-based generator may lack enough expressive power for text generation, relational memory is employed

as the generator. Because the discriminator gives too little guidance information to the generator, we choose to use Wasserstein loss to enhance the generator’s guidance. At the same time, in order to make the loss work, we carefully design a discriminator network corresponding to Wasserstein loss. The improved model is named WRGAN, which means that it is an improvement of RelGAN with Wasserstein loss.

3.1. Overall Framework

The overall framework of WRGAN is shown in Figure 1. It can be divided into three parts: a relational-memory-based generator, Gumbel–Softmax relaxation, and a one-dimensional convolution-based discriminator. After the generator passes standard maximum likelihood estimation (MLE) training for several epochs, the network starts adversarial training. According to RelGAN, for each M_t at time t , we can get the updated memory \tilde{M}_{t+1} :

$$\tilde{M}_{t+1} = [\tilde{M}_{t+1}^{(1)} : \dots \tilde{M}_{t+1}^{(H)}] \tag{3}$$

$$\tilde{M}_{t+1}^{(h)} = \sigma\left(\frac{Q_t^{(h)}(K_t^{(h)})^T}{\sqrt{d_k}}\right)V_t^{(h)}, \tag{4}$$

where σ is the Softmax function, and Q, K , and V are the query, key, and value vectors. Then, the output of generator o_t is given by:

$$o_t = f_\theta(\tilde{M}_{t+1}, M_t), \tag{5}$$

where f_θ is combinations of skip connections, multi-layer perceptron (MLP), gated operations, and/or pre-Softmax linear transformations [5].

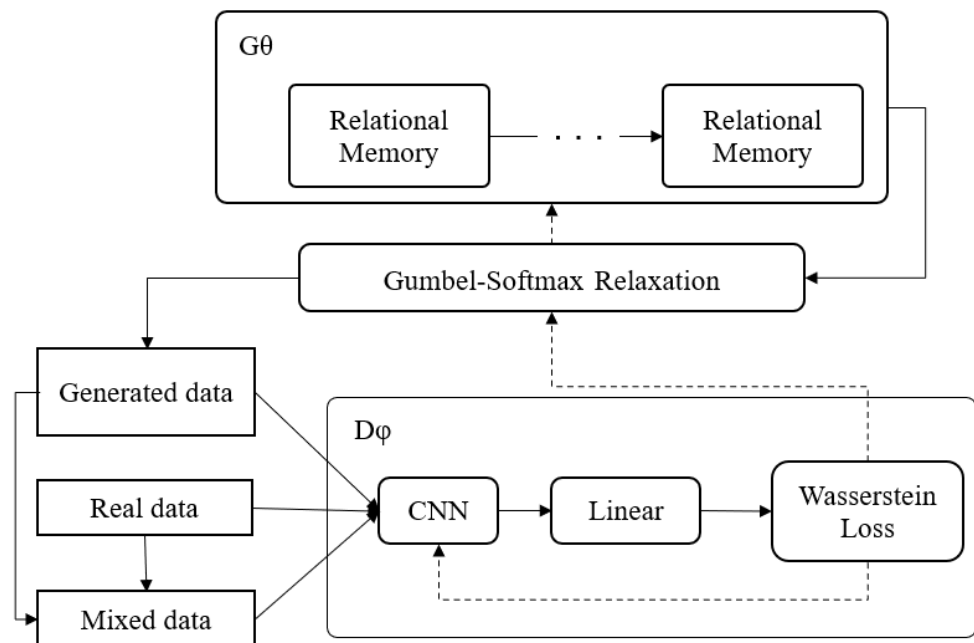


Figure 1. The overall framework of WRGAN.

After getting the generator output o_t , we put o_t into Gumbel–Softmax; then, we can get the generated data \hat{y} , defined as:

$$\hat{y} = \sigma(\beta(o_t + g_t)) \tag{6}$$

where σ is softmax function, and g_t is defined as:

$$g_t = -\log(-\log(1 \times 10^{-10}) + 1 \times 10^{-10}), \tag{7}$$

and β is a tunable parameter; in this work, we set the value of β as 100.

Then, we put the generated data, the real data, and the mixed data into the discriminator to get the loss. The loss of the discriminator represents the relative distance between the distribution of the generated data and the real data. Finally, the model will adjust the network parameters through loss.

3.2. Rebuilding the Discriminator

The proposed discriminator framework is shown in Figure 2, and the parameters are shown in Table 1. We choose one-hot as the input form, and the input shape is [Batch size, Vocabulary size, Max length]. The first layer is a 1D convolutional [25] layer for dimension conversion. The second layer is three groups of Resblock layers with different one-dimensional convolution kernel sizes. The structure of Resblock [12] is shown in Figure 3. The one-dimensional convolution kernel shapes are the same as in the Resblock. The sizes of the three groups of convolution kernels are [1, 3, 5], and the padding is [0, 1, 2]. The Resblock [26] also contains a hyperparameter dimension (Dim). Different data sizes correspond to different dimensions; a detailed analysis of the impact of the parameters on the model can be found in Section 4.6.1. Then, the three channels are concatenated with different convolution kernel sizes. One thing to note is that the two linear layers do not add an activation function.

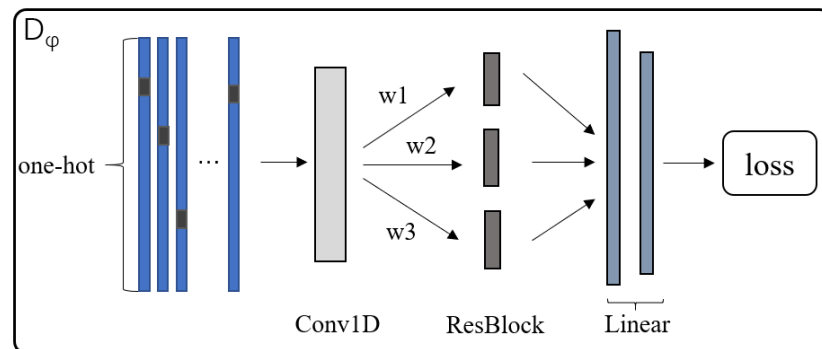


Figure 2. The proposed discriminator framework.

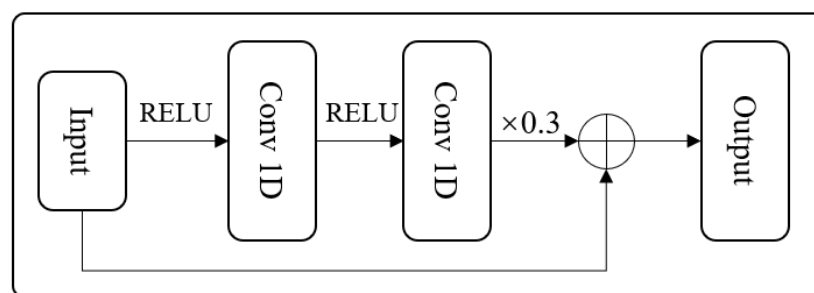


Figure 3. The Resblock.

Table 1. The discriminator parameters.

Layers	Input Shape	Kernel Shape
Conv1D	(Batch size, Vocab size, Max length)	(1, Vocab size)
ResBlock	(Batch size, Dim, Max length)	(1, Dim), (3, Dim), (5, Dim)
Linear1	(Batch size, Dim × Max length × 3)	
Linear2	(Batch size, 1000)	
Output	(Batch size, 1)	

We assume that the weight of the convolutional layer is W_d^t , where $d \in [1, 2, 3]$ either for the real input $[r_1 : \dots : r_T]$ or for the generated input $[\hat{y}_1 : \dots : \hat{y}_T]$. For real data, the distributed representation [27] h_r is:

$$h_r^t = W_d^t r^t. \quad (8)$$

The distributed representation of the generated data is:

$$h_y^t = W_d^t \hat{y}^t. \quad (9)$$

3.3. Network Training

3.3.1. Loss Function

In this work, we use gradient penalty Wasserstein loss. According to WGAN-GP, adding a gradient penalty will avoid all network parameters approaching two extremes so that the weights can be evenly distributed in a specific interval. The gradient penalty ensures the stability of adversarial training. The discriminator loss can be defined as:

$$L_D = \mathbb{E}[D(\hat{x})] - \mathbb{E}[D(x)] + \lambda \mathbb{E}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \quad (10)$$

where \hat{x} is a mixture of the real data and generated data. λ is called the penalty coefficient. All experiments in this paper use $\lambda = 10$. The generator loss is given by:

$$L_G = -\mathbb{E}[D(\hat{x})]. \quad (11)$$

If we put (6) and (7) into (8) and (9), the L_D and L_G will be

$$L_D = \frac{1}{3T} \sum_{d=1}^3 \sum_{t=1}^T D(h_{\hat{y}}^t) - \frac{1}{3T} \sum_{d=1}^3 \sum_{t=1}^T D(h_r^t) + \frac{\lambda}{3T} \sum_{d=1}^3 \sum_{t=1}^T (\|\nabla_{\hat{m}} D(h_{\hat{m}}^t)\|_2 - 1)^2 \quad (12)$$

$$L_G = -\frac{1}{3T} \sum_{d=1}^3 \sum_{t=1}^T D(h_{\hat{y}}^t), \quad (13)$$

where \hat{m} is obtained by randomly mixing real data and generated data.

3.3.2. Training Parameters

As we were limited by hardware equipment, during the training process, we set the batch size to 64. We used the Adam [28] optimizer during the adversarial training process, the learning rate of the generator and the discriminator were both 1×10^{-4} , the L2 regularization weight decay was 0.01 [29], and the dropout of the discriminator was 0.25. The maximum number of iterations was 2000, and the generator embedding [30,31] dimension was 32.

4. Experiments

To evaluate the performance of the model, we tested our model on real-world data, including the Common Objects in Context (COCO) image captions [32], EMNLP2017 WMT News [33], Movie Reviews [34], and Chinese poetry [35] datasets. The specific experimental parameter settings are given in each subsection.

4.1. Evaluation Metrics

Similarly to other models, we used two metrics to evaluate models. The first was the negative log-likelihood (NLL_{gen}) and its counterpart (NLL_{oracle}) [36], defined as:

$$NLL_{gen} = -\mathbb{E}_{Y_{1:T} \sim P_r} \log P_{\theta}(Y_1, \dots, Y_T) \quad (14)$$

$$NLL_{oracle} = -\mathbb{E}_{y_{1:T} \sim P_{\theta}} \log P_r(y_1, \dots, y_T), \quad (15)$$

where P_θ is the generated data distribution and P_r is the real data distribution. We used NLL_{gen} to evaluate the diversity of the generated data.

The other evaluation metric for real data was bilingual evaluation understudy (BLEU) [37]. The BLEU score was used to compare and count the number of commonly occurring n-gram words for the quality evaluation of the generated text. One thing to note is that in order to enable BLEU scores to evaluate the model, we also used test data [38].

4.2. COCO Image Caption Dataset

4.2.1. BLEU and NLL Scores

The Microsoft COCO dataset contains human-generated captions for images. This dataset contains a total of 10,000 captions. After preprocessing [39], we got a dictionary with 4682 unique words, and the maximum sentence length is 37. The BLEU scores compared with those of other models are shown in Table 2. We adopted the same evaluation settings as other models. It can be seen in Table 2 that, except for the BLEU-5 score, which is lower than those of other models, the other scores are better than those of the other models. That means that the improvement of the model is very effective for the COCO dataset. The NLL_{gen} score shows that the improved model also performs well with respect to the generated data diversity.

Figure 4 is a line graph generated after 2000 iterations of training when the hyperparameter dimension was equal to 128. There were no strong fluctuations in the training process, which shows that the model is relatively stable during the training process. The data samples from COCO generated can be found in Appendix A.1.

Table 2. The bilingual evaluation understudy (BLEU) and NLL_{gen} scores on COCO Image Captions where Dimension = 128. For BLEU scores, the higher the better. For NLL_{gen} , the lower the better.

Method	BLEU-2	BLEU-3	BLEU-4	BLEU-5	NLL_{gen}
MLE	0.731	0.497	0.305	0.189	0.718
SeqGAN	0.745	0.498	0.294	0.180	1.082
RankGAN	0.743	0.467	0.264	0.156	1.344
LeakGAN	0.746	0.528	0.355	0.230	0.679
RelGAN (100)	0.849	0.687	0.502	0.331	0.756
RelGAN (1000)	0.814	0.634	0.455	0.303	0.655
WRGAN	0.853	0.687	0.509	0.318	0.673

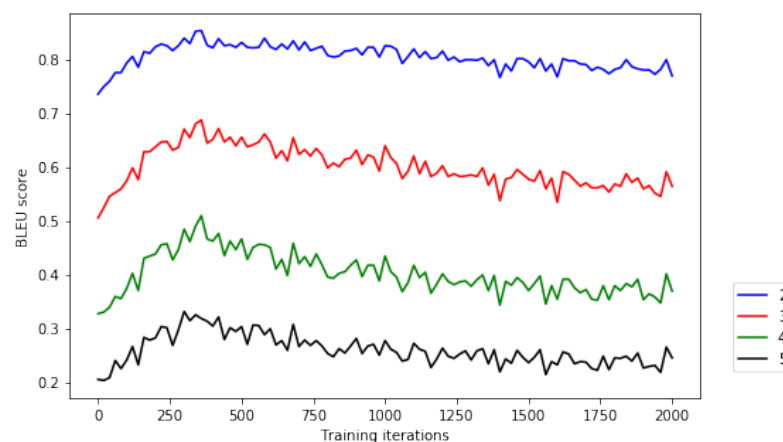


Figure 4. The BLEU scores for Common Objects in Context (COCO) image captions, where Dimension = 128.

4.2.2. Comparison of RelGAN and WRGAN on COCO

The discriminator loss when using Wasserstein loss on RelGAN is shown in Figure 5. We found that the discriminator and generator loss are almost equal to 0 throughout the training process (floating in the range of 0.0005 to 0.005). In this case, the discriminator has no useful guidance information to give to the generator, and the two networks cannot form adversarial learning. That is why it does not perform well on the datasets. Excluding some obvious error causes, we located the problem in the discriminator structure. After many experiments, we made the above modification to the discriminator structure.

```

g_loss: -0.0017, d_loss: 0.0023: 37%
adv_step: 740, nll_gen: 0.6963, bleu2:
g_loss: -0.0018, d_loss: 0.0030: 38%
adv_step: 760, nll_gen: 0.6935, bleu2:
g_loss: -0.0015, d_loss: 0.0014: 39%
adv_step: 780, nll_gen: 0.6893, bleu2:
g_loss: -0.0004, d_loss: 0.0027: 40%
adv_step: 800, nll_gen: 0.6775, bleu2:
g_loss: -0.0019, d_loss: 0.0020: 41%
adv_step: 820, nll_gen: 0.6603, bleu2:
g_loss: -0.0031, d_loss: 0.0008: 42%
adv_step: 840, nll_gen: 0.6377, bleu2:
g_loss: -0.0028, d_loss: 0.0034: 43%

```

Figure 5. The discriminator loss of RelGAN.

Figure 6 shows the BLEU-2 score of the improved model, RelGAN, and RelGAN with Wasserstein loss on the COCO dataset with 1400 iterations. The parameters in the figure are the parameters mentioned in the original paper. We found that RelGAN has strong fluctuations when using Wasserstein loss, and it does not perform well on the COCO dataset.

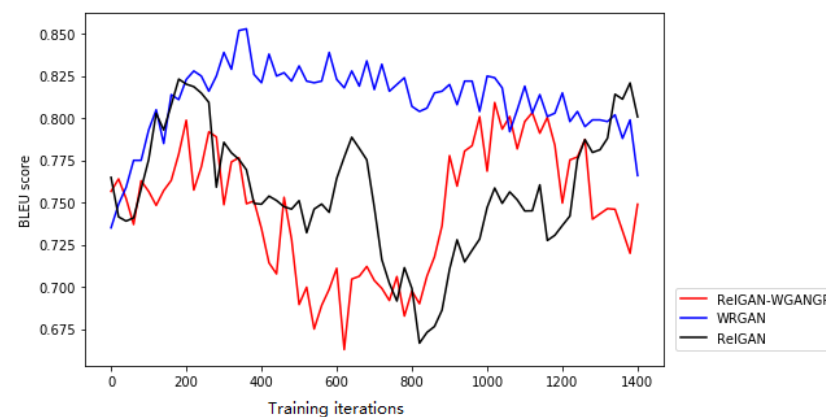


Figure 6. The BLEU-2 scores on the COCO image captions compared with RelGAN.

4.3. EMNLP 2017 WMT News

The BLEU and negative log-likelihood (NLL) scores on EMNLP 2017 WMT News with Dimension = 256 are shown in Table 3. The EMNLP 2017 WMT News dataset contains about 270,000 sentences, and the test data contain 10,000 sentences. After data preprocessing, we got a vocabulary size of 5255, and the maximum sequence length was 51. The training curves of the BLEU scores are shown in Figure 7. The results show that, except in the generated data diversity, which is slightly weaker than that of other models, the improved model is better than other models in the BLEU scores. The figure also shows that the model can have better performance and stable results, even on a large-scale dataset. The data samples generated from EMNLP 2017 WMT News can be found in Appendix A.2.

Table 3. The BLEU and NLL scores on EMNLP 2017 WMT News with Dimension = 256. For the BLEU scores, the higher the better. For NLL_{gen} , the lower the better.

Method	BLEU-2	BLEU-3	BLEU-4	BLEU-5	NLL_{gen}
MLE	0.768	0.473	0.240	0.126	2.382
SeqGAN	0.777	0.491	0.261	0.138	2.773
RankGAN	0.727	0.435	0.209	0.101	3.345
LeakGAN	0.826	0.645	0.437	0.272	2.356
RelGAN(100)	0.881	0.705	0.501	0.319	2.482
RelGAN(1000)	0.837	0.654	0.435	0.265	2.285
WRGAN	0.952	0.782	0.539	0.336	2.812

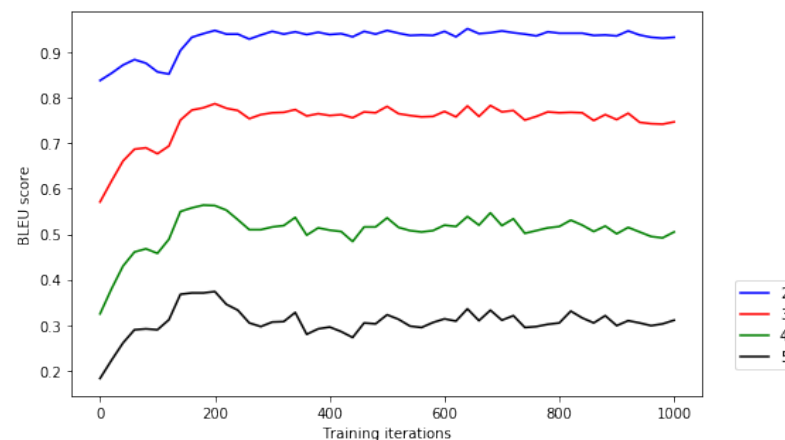


Figure 7. The BLEU scores for EMNLP2017 WMT News with Dimension = 256.

4.4. Chinese Poetry

The Chinese poetry dataset includes a total of 16,394 Tang poems. Each poem contains five Chinese words per sentence. Since we used the BLEU score as the evaluation metric, we randomly selected 8197 poems as the training data. The remaining 8197 poems were the test data. After data preprocessing, we got a vocabulary with a size of 4140. Table 4 shows the BLEU-2 scores for this dataset. The improved model also achieved good results on the Chinese poetry dataset. The data samples generated from the Chinese poetry dataset can be found in Appendix A.3.

Table 4. The BLEU-2 scores on the Chinese poetry dataset with Dimension = 128. For BLEU scores, the higher the better.

Method	SeqGAN	RankGAN	RelGAN	LeakGAN	WRGAN
BLEU-2	0.738	0.812	0.817	0.456	0.835

4.5. Movie Reviews (MR)

The Movie Reviews (MR) dataset has two sentiment classes (negative and positive [40,41]). MR has 4503 samples, including 3152 training samples and 1351 testing samples. After data preprocessing, we got a vocabulary with a size of 6216. Moreover, the maximum sentence length was 15. Table 5 shows the BLEU and NLL scores for the dataset. On MR, the BLEU-5 score of the improved model was also lower than that of other models, and the other items were able to get good scores. The data samples generated from the Movie Reviews dataset can be found in Appendix A.4.

Table 5. The BLEU and NLL scores for the Movie Reviews dataset with Dimension = 128. For BLEU scores and NLL_{div} , the higher the better. For NLL_{gen} , the lower the better.

Method	SentiGAN	CSGAN	CatGAN [42]	WRGAN
BLEU-2	0.532	0.452	0.589	0.623
BLEU-3	0.285	0.204	0.335	0.337
BLEU-4	0.167	0.112	0.194	0.193
BLEU-5	0.143	0.082	0.144	0.128
NLL_{gen}	2.436	2.912	1.619	0.8061
NLL_{div}	0.484	0.254	0.535	0.9097

4.6. Impact of Hyperparameters

4.6.1. Impact of Dimension

In this session, we discuss the impact of the hyperparameter dimension (Dim) on the model. As shown in Figure 8, the BLEU-2 scores of the model on the COCO dataset are plotted when Dim = 128 and Dim = 64, respectively. When Dim = 128, and when the model is around the 1000th iteration, it is not difficult to see that the model is overfitting. However, when Dim = 64, there is no obvious overfitting. After the experiments, we found that the hyperparameter should be proportional to the amount of training data. The training curve of the EMNLP 2017 WMT News dataset can also confirm this view. When Dim = 256 on the EMNLP 2017 WMT News dataset, we did not find obvious overfitting during the training process. In addition, the parameter Dim is not as small as possible. When Dim is too small, the model may not fully get the characteristics of the sentence vector on the current dataset. This will reduce the quality of the generated data. Therefore, we tend to set Dim a little higher.

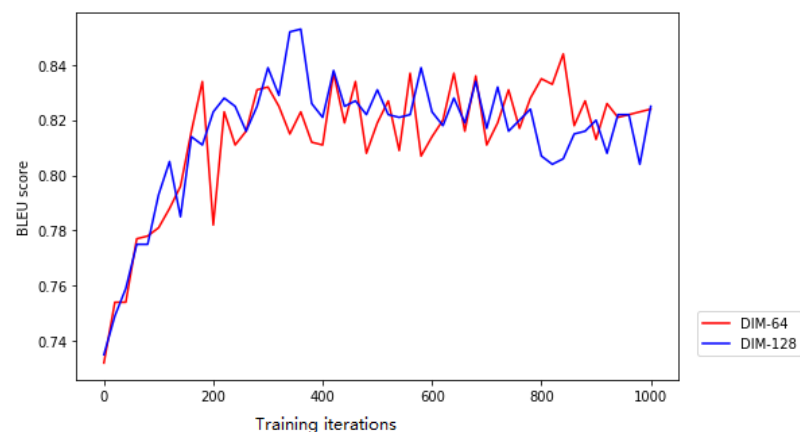


Figure 8. The BLEU-2 scores for the COCO image captions with Dimension = 64 and 128.

4.6.2. Impact of k

The hyperparameter k is the generator step/discriminator step. As shown in Figure 9, as k increases, the model gradually deteriorates and becomes unstable. We think that the reason for this situation is that the time taken for discriminator training is too short, and the generator is too “experienced” for adversarial training. Perhaps, as the amount of training increases, the training will gradually become stable again. Currently, this view is a guess. However, k cannot be as small as possible. A smaller k means more training time for the discriminator. Increased training time for the discriminator will make the model overfit faster. Therefore, after many experiments, we set $k = 1/5$.

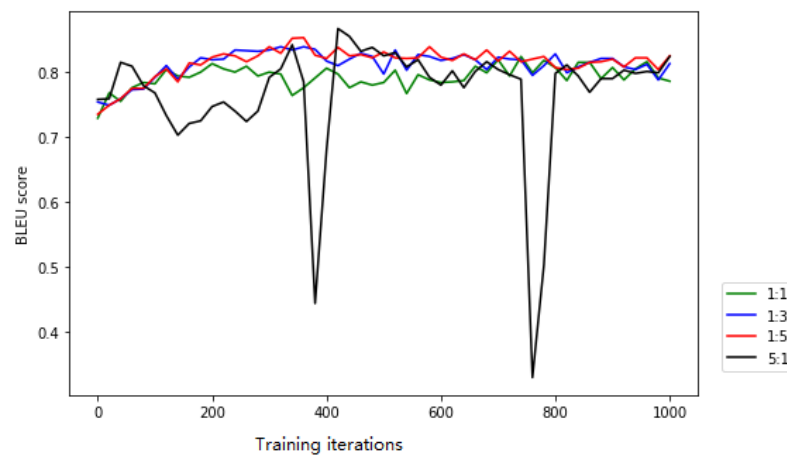


Figure 9. The BLEU-2 scores for the COCO image captions with $k = 5, 1, 1/3,$ and $1/5$.

5. Discussion and Conclusions

5.1. Discussion

Compared with other models, there are still some questions to be solved. The first is about improving the BLEU-5 score. Secondly, the model does not perform well on synthetic data. We still need to change the model's structure to make it better.

5.2. Conclusions

This paper proposes a new and improved model for text generation based on RelGAN and WGAN-GP called WRGAN. We rebuilt the model's structure to allow various modules to operate in coordination. We applied the Wasserstein distance in text generation to provide more useful information for the generator, and we used the relational memory as the generator architecture to reduce the mode collapse. Then, we tested the improved model on multiple real datasets. Compared with other models, our model has a higher evaluation score and sample quality. In the comparative experiment with RelGAN, our model achieved better results. Finally, we analyzed the influence of hyperparameters on the model. We plan to continue to improve this network for future work and to apply this network to more natural language processing applications.

Author Contributions: Conceptualization, all authors; methodology, all authors; software, Z.J.; data preprocessing, Z.J.; data analysis, all authors; supervision, F.R.; writing—original draft preparation, Z.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the NSFC-Shenzhen Joint Foundation (Key Project) (Grant No. U1613217).

Data Availability Statement: Data available in a publicly accessible repository. Publicly available datasets were analyzed in this study. This data can be found here: [32–35].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

GAN	generative adversarial networks
NLP	natural language processing
SeqGAN	Sequence GAN
RL	reinforcement learning
WGAN	Wasserstein GAN
GP	gradient penalty
BLEU	bilingual evaluation understudy
NLL	negative log-likelihood
MLE	maximum likelihood estimation
LSTM	Long Short-Term Memory

Appendix A. The Data Samples Generated from Each Dataset

Appendix A.1. COCO Image Caption Dataset

Table A1. The data samples generated from the COCO image caption dataset.

a yellow bicycle parked next to a red wall.
 a white and blue plane flying in the blue sky.
 a white cat has caught a bird on its tail.
 a woman in the kitchen is holding a dog.
 a man is looking at motorcycle in the road by the building.
 two wet young boys on a table.
 a bathroom with a toilet and a sink.
 many motorcyclists gather in front of a bus.
 a young girl sitting on top of a bike near the ocean.
 some people on the snow covered field.

Appendix A.2. EMNLP 2017 WMT News

Table A2. The data samples generated from the EMNLP 2017 WMT News dataset.

the security services, an independent has already said: "there is a need to follow up the process of the project in 2017.

i have been together with russia, because we just don't know if that is the need to make it," he said.

he will tell the player he didn't want to continue to make his opinion on that.

we are making sure that we will better understand that we need to strengthen our order for the next 18—and we will get ourselves into our future.

in some cases, it is the first time in the past three years, a few of them needed to be with other. 1% of the population is even more within the trump administration.

after the festival, his wife, who was in contact with a police in the UK.

now, it is amazing when people on the court, including a man from the police.

"I'm really concerned, because it was a very careful in my life," she said.

he has had to quit from the hospital after a 13-year-old who had been on three years.

"we are back with this, and we are not about that," he said in a statement for several years.

Appendix A.3. Chinese Poetry

Table A3. The data samples generated from the Chinese poetry dataset.

步履一朝来，乘流处入天。云峰十字巷，月落翠屏中。
 一骑来相应，烟片开林下。散发故人至，色含野水闲。
 北风吹五岭，寒月断寒猿。风光千万阵，霜薄香台中。
 今朝明月居，陇头复何极。仙郎日相见，江树正朝归。
 卧病南国月，孤舟无端倪。浮云过云气，伯牙道为心。
 昔年顾虎丘，恩荣奉旧居。诏书收骥子，才贤气初荣。
 江水将孤帆，楼兰尚郁蒸。清风千里叶，白雪二湖平。
 游此闲门道，双旌到上游。忽入灵洞骨，秋骑发华林。
 画舸下金华，悲歌犹意无。所以终日子，怀山有异方。
 朝日上林隐，苍茫解朝衣。公声何在意，有愧为人乡。

Appendix A.4. Movie Reviews (MR)

Table A4. The data samples generated from the Movie Reviews dataset.

a very good film sits in the place.
 the movie doesn't have to see.
 most fun ride of the place.
 a good documentary on the what begins that never gets.
 the film is smart, sweet and playful point.
 if you're a comic fan.
 a very capable nailbiter.
 could use a little more humanity and delight.
 so boring and meandering.
 a pleasant, but it's also extremely effective.

References

1. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [[CrossRef](#)]
2. Bowman, S.R.; Vilnis, L.; Vinyals, O.; Dai, A.M.; Jozefowicz, R.; Bengio, S. Generating sentences from a continuous space. *arXiv* **2015**, arXiv:1511.06349.
3. Yang, Y.; Dan, X.; Qiu, X.; Gao, Z. FGGAN: Feature-Guiding Generative Adversarial Networks for Text Generation. *IEEE Access* **2020**, *8*, 105217–105225. [[CrossRef](#)]
4. Yu, L.; Zhang, W.; Wang, J.; Yu, Y. Seqgan: Sequence generative adversarial nets with policy gradient. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
5. Nie, W.; Narodytska, N.; Patel, A. Relgan: Relational generative adversarial networks for text generation. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
6. Kusner, M.J.; Hernández-Lobato, J.M. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv* **2016**, arXiv:1611.04051.
7. Maddison, C.J.; Mnih, A.; Teh, Y.W. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv* **2016**, arXiv:1611.00712.
8. Jang, E.; Gu, S.; Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv* **2016**, arXiv:1611.01144.
9. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein gan. *arXiv* **2017**, arXiv:1701.07875.
10. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A.C. Improved training of wasserstein gans. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5767–5777.
11. Natsume, R.; Yatagawa, T.; Morishima, S. Rsgan: Face swapping and editing using face and hair representation in latent spaces. *arXiv* **2018**, arXiv:1804.03447.
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
13. Che, T.; Li, Y.; Zhang, R.; Hjelm, R.D.; Li, W.; Song, Y.; Bengio, Y. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv* **2017**, arXiv:1702.07983.
14. Coulom, R. Efficient selectivity and backup operators in Monte-Carlo tree search. In Proceedings of the International Conference on Computers and Games, Turin, Italy, 29–31 May 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 72–83.

15. Juefei-Xu, F.; Dey, R.; Boddeti, V.N.; Savvides, M. Rankgan: A maximum margin ranking gan for generating faces. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018; Springer: Cham, Switzerland, 2018; pp. 3–18.
16. Guo, J.; Lu, S.; Cai, H.; Zhang, W.; Yu, Y.; Wang, J. Long text generation via adversarial training with leaked information. *arXiv* **2017**, arXiv:1709.08624.
17. Fedus, W.; Goodfellow, I.; Dai, A.M. MaskGAN: Better text generation via filling in the _____. *arXiv* **2018**, arXiv:1801.07736.
18. Xu, J.; Ren, X.; Lin, J.; Sun, X. Diversity-promoting gan: A cross-entropy based generative adversarial network for diversified text generation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3940–3949.
19. Wang, K.; Wan, X. SentiGAN: Generating Sentimental Texts via Mixture Adversarial Networks. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden, 13–19 July 2018; pp. 4446–4452.
20. Santoro, A.; Faulkner, R.; Raposo, D.; Rae, J.; Chrzanowski, M.; Weber, T.; Wierstra, D.; Vinyals, O.; Pascanu, R.; Lillicrap, T. Relational recurrent neural networks. In Proceedings of the Annual Conference on Neural Information Processing Systems 2018, Montreal, QC, Canada, 3–8 December 2018; pp. 7299–7310.
21. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
22. Nowozin, S.; Cseke, B.; Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In Proceedings of the Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 271–279.
23. Zhao, J.; Kim, Y.; Zhang, K.; Rush, A.; LeCun, Y. Adversarially regularized autoencoders. In Proceedings of the International Conference on Machine Learning PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 5902–5911.
24. Shao, Z.; Huang, M.; Wen, J.; Xu, W.; Zhu, X. Long and diverse text generation with planning-based hierarchical variational model. *arXiv* **2019**, arXiv:1908.06605.
25. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
26. Ren, F.; Zhou, Y. CGMVQA: A New Classification and Generative Model for Medical Visual Question Answering. *IEEE Access* **2020**, *8*, 50626–50636. [[CrossRef](#)]
27. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
28. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
29. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press: Cambridge, UK, 2016.
30. Levy, O.; Goldberg, Y. Dependency-based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Baltimore, MD, USA, 22–27 June 2014; pp. 302–308.
31. Ren, F.; Xue, S. Intention Detection Based on Siamese Neural Network with Triplet Loss. *IEEE Access* **2020**, *8*, 82242–82254. [[CrossRef](#)]
32. Chen, X.; Fang, H.; Lin, T.Y.; Vedantam, R.; Gupta, S.; Dollár, P.; Zitnick, C.L. Microsoft coco captions: Data collection and evaluation server. *arXiv* **2015**, arXiv:1504.00325.
33. Proceedings of the Conference on Machine Translation (WMT). Available online: <http://www.statmt.org/wmt17/> (accessed on 1 December 2020).
34. Maas, A.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011), Portland, OR, USA, 19–24 June 2011.
35. Chinese Poems. Available online: <http://www.chinese-poems.com/> (accessed on 1 December 2020).
36. Theis, L.; Oord, A.V.D.; Bethge, M. A note on the evaluation of generative models. *arXiv* **2015**, arXiv:1511.01844.
37. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002.
38. Zhu, Y.; Lu, S.; Zheng, L.; Guo, J.; Zhang, W.; Wang, J.; Yu, Y. Texus: A benchmarking platform for text generation models. *arXiv* **2018**, arXiv:1802.01886.
39. Ren, F.; Deng, J. Background knowledge based multi-stream neural network for text classification. *Appl. Sci.* **2018**, *8*, 2472. [[CrossRef](#)]
40. Han, Z.; Ren, F.; Miao, D. Sentiment analysis method based on an improved modifying-matrix language model. *IEEE Trans. Electr. Electron. Eng.* **2018**, *13*, 1446–1453. [[CrossRef](#)]
41. Quan, C.; Zhang, B.; Sun, X.; Ren, F. A combined cepstral distance method for emotional speech recognition. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1–9. [[CrossRef](#)]
42. Liu, Z.; Wang, J.; Liang, Z. CatGAN: Category-Aware Generative Adversarial Networks with Hierarchical Evolutionary Learning for Category Text Generation. In Proceedings of the AAAI, New York, NY, USA, 7–12 February 2020; pp. 8425–8432.