

Towards Diverse Paraphrase Generation Using Multi-Class Wasserstein GAN

Zhecheng An¹, Sicong Liu^{2,3}

¹Department of Electronic Engineering, Tsinghua University, Beijing, P. R. China.

²Department of Communication Engineering, Xiamen University, Xiamen, P. R. China.

³Key Laboratory of Digital Fujian on IoT Communication, Architecture and Security Technology, Xiamen University, Xiamen, P. R. China.

anzhecheng@gmail.com, liusc@xmu.edu.cn

Abstract

Paraphrase generation is an important and challenging natural language processing (NLP) task. In this work, we propose a deep generative model to generate paraphrase with diversity. Our model is based on an encoder-decoder architecture. An additional transcoder is used to convert a sentence into its paraphrasing latent code. The transcoder takes an explicit pattern embedding variable as condition, so diverse paraphrase can be generated by sampling on the pattern embedding variable. We use a Wasserstein GAN to align the distributions of the real and generated paraphrase samples. We propose a multi-class extension to the Wasserstein GAN, which allows our generative model to learn from both positive and negative samples. The generated paraphrase distribution is forced to get closer to the positive real distribution, and be pushed away from the negative distribution in Wasserstein distance. We test our model in two datasets with both automatic metrics and human evaluation. Results show that our model can generate fluent and reliable paraphrase samples that outperform the state-of-art results, while also provides reasonable variability and diversity.

1 Introduction

Paraphrases are rewritten versions of text with different words or expressions while preserving the original semantic. The automatic paraphrase generation of a given sentence is an important NLP task, which can be applied in many fields such as information retrieval, question answering, text summarization, dialogue system, etc. A paraphrase generator is able to perform text reformulation on these systems to bring variation. Besides, the generated paraphrases can be used as augmented data in many learning tasks such as text identification, classification and inference. Therefore, the generation fidelity, naturalness and diversity play important roles in the evaluation on a paraphrase generator system.

Paraphrase generation is a challenging task due to the complexity of human language. The recent progress of deep learning, especially sequence-to-sequence (Seq2Seq) based models for text generation [Bahdanau *et al.*, 2015;

Bowman *et al.*, 2016], have shown great advantages over the traditional rule-based [McKeown, 1983] and statistic [Quirk *et al.*, 2004] models. Intuitively, a straightforward method to generate paraphrase is to train a Seq2Seq model to convert a sentence into its paraphrasing reference using the maximum likelihood estimation (MLE), where the cross entropy loss is optimized [Prakash *et al.*, 2016; Cao *et al.*, 2017]. This method is further extended in [Ranzato *et al.*, 2016] to use metrics like BLEU [Papineni *et al.*, 2002] and ROUGE [Lin, 2004] as reward function of the reinforcement learning algorithm. To mitigate the gap between lexical similarity and semantic similarity, [Li *et al.*, 2018] replaces the lexical reward function with a trained evaluator and update it using inverse reinforcement learning.

The researches mentioned above focus on converting a sentence into a paraphrasing target optimized with various metrics. However, there may be multiple possible paraphrases for one given sentence. Paraphrases of a certain sentence generated by humans may differ from each other and contain wide linguistic variations, which cannot be captured by single-target transformation models. Moreover, these models tend to generate sentences with high resemblance to the training samples, while other good semantically similar results may be suppressed [Li *et al.*, 2018]. Therefore, in order to further exploit the variability and obtain diverse paraphrases, it is necessary to generatively model the paraphrase distribution instead of a single target sample.

In order to model the distribution of the generated paraphrase, we introduce a random variable as *pattern embedding*. The generated results are explicit conditioning on the pattern embedding variable. Therefore, the model can generate multiple results with diversity by sampling on this variable. To train such a generative model, one existing work is the VAE-SVG [Gupta *et al.*, 2018] that uses a conditional variation auto-encoder (VAE) [Kingma and Welling, 2014] for paraphrase generation. However, in this paper, we exploit the adversarial generative network (GAN) [Goodfellow *et al.*, 2014] to model the paraphrase generation distribution as an alternative approach. Instead of using the KL-divergence to optimize the lower-bound in VAE, the GAN uses adversarial training to directly align the generated distribution with the real distribution, which is able to generate realistic results.

Applying GANs on text generation is non-trivial since text consists of discrete tokens that are non-differentiable. We use the Gumbel-softmax [Jang *et al.*, 2017] as a continuous approximation and use professor-forcing [Lamb *et al.*, 2016] algorithm to match the hidden states of input and paraphrasing sequences. In order to integrate professor-forcing in our model, we design an auto-encoder along with a transcoder as the generator. The transcoder is a feedforward network that takes both the original sentence and the pattern embedding as inputs, and outputs the paraphrase latent code. A shared decoder is used to generate paraphrase sentence and decode the reference sample.

Specifically, we take advantage of the Wasserstein GAN (WGAN) [Arjovsky *et al.*, 2017] to train our paraphrase generation model for better stability and convergence performance. We propose a multi-class extension to WGAN by using multiple critics to measure the generated Wasserstein distance to different classes of samples. The multi-class WGAN enables our model to learn paraphrase generation from both positive and negative samples. The generated paraphrase distribution is forced to get closer to the positive distribution and be pushed away from the negative distribution in Wasserstein distance, which contributes to the generation fluency and relevance.

Overall, the main contributions of this work are summarized as follows: (1) We propose a generative model aiming at generating multiple paraphrases of a given sentence with diversity. (2) With continuous approximation and professor-forcing, the model is trained with GAN to align the generated distribution with the real distribution. (3) We develop the multi-class WGAN that enables our model to learn from both positive and negative samples, which promotes the generation fluency and relevance.

2 Related Work

Neural Paraphrase Generation: [Prakash *et al.*, 2016] proposes a Seq2Seq paraphrase generation model using residual stack LSTM and cross entropy loss. [Cao *et al.*, 2017] introduces an additional copying decoder for keywords extraction from the source. [Xu *et al.*, 2018] uses a fixed vocabulary of rewrite patterns in the decoder to generate diverse paraphrases, and the model is trained using MLE criterion by optimizing on selective patterns. The evaluation of paraphrasing is studied in [Li *et al.*, 2018], where a trained evaluator is used as the reward function to train a paraphrase generation model with inverse reinforcement learning. Besides the above transformation-based model, generative model to formulate the paraphrase generation distribution is also proposed, such as the VAE-based VAE-SVG [Gupta *et al.*, 2018]. In this paper, we use GAN as an alternative generative approach for paraphrase distribution modeling. To the best of the authors’ knowledge, this work is the first in literature that applies GAN in paraphrase generation.

Generative Adversarial Networks: The main idea of GAN [Goodfellow *et al.*, 2014] is to train a generator and a discriminator that compete with each other, forcing the generator to generate realistic outputs to fool the discriminator. In such way, the generated distribution of GAN is

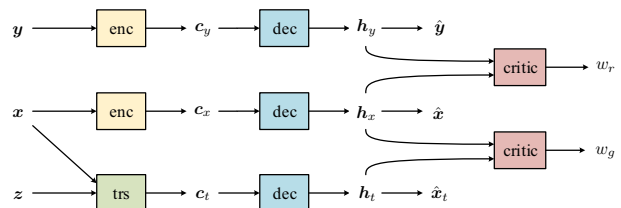


Figure 1: Overview of the proposed paraphrase generation framework.

forced to align with the real distribution. Various extensive algorithms to the vanilla GAN have been proposed to handle different tasks. For example, conditional GAN (CGAN) [Mirza and Osindero, 2014] is used to model conditional distribution by feeding side information to the generator and discriminator. In ACGAN [Odena *et al.*, 2017], an auxiliary classifier is added to the discriminator to tackle the multi-class generation problem. WGAN [Arjovsky *et al.*, 2017] modifies the discriminator as the critic to measure Wasserstein distance instead of Jensen-Shannon (JS) divergence, and achieves better training stability.

GAN-based Text Generation: Since GANs achieve many success in image generation fields, several recent researches focus on applying GAN in text generation. For example, [Hu *et al.*, 2017] combines a discriminator with the VAE model to generate text with controllable attribute. With non-parallel corpus, [Shen *et al.*, 2017] cross-align distributions between two datasets with GAN to perform style transfer. Such adversarial training technique is also used in unsupervised neural machine translation (NMT) [Lample *et al.*, 2018] to match the encoded latent spaces of two languages. For supervised NMT with pairwise samples, [Wu *et al.*, 2018] designs an Adversarial-NMT model using GAN as a probabilistic transformer to process translation on parallel corpus.

3 Method

3.1 Model Framework

The overall framework of our proposed paraphrase generation model is shown in Figure 1. The model consists of an auto-encoder, a transcoder and a critic. The auto-encoder is used to encode and reconstruct the input and reference paraphrasing sentences. The transcoder is a feedforward network that converts a sentence into its paraphrasing latent code, which is then decoded with a decoder shared with the auto-encoder. Finally, the decoded paraphrase result is matched with the recovered real sample using the critic.

Auto-encoder: Consider a pair of paraphrasing sentences (x, y) , where $x = \{x_1, x_2, \dots, x_T\}$ and $y = \{y_1, y_2, \dots, y_T\}$ are sequences of tokens. In the auto-encoder, x and y are encoded into latent code $c_x = \text{enc}_{\theta_e}(x)$ and $c_y = \text{enc}_{\theta_e}(y)$, where θ_e refers to the encoder parameter. c_x and c_y are then decoded by $\text{dec}_{\theta_d}(\cdot)$ with parameter θ_d to recover the original sequences as \hat{x} and \hat{y} . Gated recurrent unit (GRU) based recurrent neural networks (RNN) are used in the encoder and decoder. h_x and h_y denote the decoding

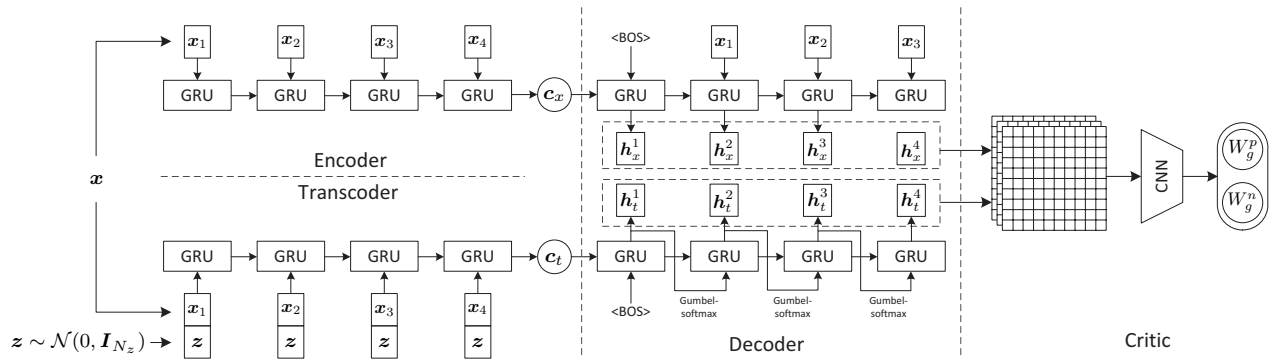


Figure 2: The structure of the proposed auto-encoder, transcoder and critic.

hidden states. The auto-encoder is trained in a teacher-forcing pattern, where ground truth samples are fed into the decoder every time step during training. The training objective of auto-encoder is to minimize the reconstruction loss, which is the sum of token-level cross-entropy loss in this paper, i.e.

$$\mathcal{L}_{\text{AE}}(\theta_e, \theta_d) = \mathbb{E}_{(\mathbf{x}, \mathbf{y})} \left[-\log p_{\text{dec}}(\mathbf{x} | \text{enc}_{\theta_e}(\mathbf{x}); \theta_d) - \log p_{\text{dec}}(\mathbf{y} | \text{enc}_{\theta_e}(\mathbf{y}); \theta_d) \right]. \quad (1)$$

Transcoder: The main purpose of this work is to model the paraphrase distribution of a given sentence $p(\mathbf{x})$ instead of a transformation function $f(\mathbf{x})$. In order to achieve this goal, we introduce a random variable \mathbf{z} as the pattern embedding variable, and perform paraphrase generation conditioning on \mathbf{z} . Therefore, the paraphrase distribution can be derived as $p(\mathbf{x}) = f(\mathbf{x} | \mathbf{z})$, and diverse paraphrase of \mathbf{x} can be generated by sampling on \mathbf{z} . Specifically, we use a feedforward GRU network as the transcoder as shown in Figure 2, which takes \mathbf{x} and \mathbf{z} as inputs, and convert \mathbf{x} into its paraphrasing form in the latent space, i.e. $\mathbf{c}_t = \text{trs}_{\theta_t}(\mathbf{x} | \mathbf{z})$. In particular, \mathbf{z} is a N_z -dimensional random vector sampled from a standard normal distribution $\mathcal{N}(0, \mathbf{I}_{N_z})$, and concatenated with each token in \mathbf{x} for the transcoder input. The latent code \mathbf{c}_t is decoded with a decoder that shares weight with the auto-encoder to output the final paraphrase sequence $\hat{\mathbf{x}}_t$, where \mathbf{h}_t refers to the corresponding decoding hidden states. As shown in Figure 2, \mathbf{c}_t is decoded in a free-run mode, where the output of last state is used as the input of the next state.

Critic: In order to train the output distribution of the paraphrase generation, we apply WGAN in our model, where a critic is implemented aiming at distinguishing generated fake samples from real samples. With a decoded sentence $\hat{\mathbf{x}}$ as condition, the critic is trained to detect whether a sentence is the real paraphrase of $\hat{\mathbf{x}}$. The critic outputs for real and generated samples are denoted as $w_r = f_{\theta_c}(\hat{\mathbf{y}} | \hat{\mathbf{x}})$ and $w_g = f_{\theta_c}(\hat{\mathbf{x}}_t | \hat{\mathbf{x}})$, with parameter θ_c . The structure of the critic is detailed in Section 3.4.

3.2 Multi-class Wasserstein GAN

WGAN [Arjovsky *et al.*, 2017], as an improved GAN algorithm, utilizes the Wasserstein distance instead of JS-divergence to achieve better stability and avoid mode collapse problems. Given the distribution of real and generated samples as \mathbb{P}_r and \mathbb{P}_{θ_g} , the Wasserstein distance between the two

distributions is defined as

$$W(\mathbb{P}_r, \mathbb{P}_{\theta_g}) = \max_{\theta_c \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r} [f_{\theta_c}(x)] - \mathbb{E}_{x \sim \mathbb{P}_{\theta_g}} [f_{\theta_c}(x)], \quad (2)$$

where $\{f_{\theta_c}\}_{\theta_c \in \mathcal{W}}$ is the family of all K -Lipschitz functions $f: \mathcal{X} \mapsto \mathbb{R}$. The critic maps distributions into Wasserstein distance, which acts differently as the discriminator in vanilla GAN. Thus, auxiliary classifier can not be directly integrated into the critic as the AC-GAN [Odena *et al.*, 2017] to handle multi-class generation problem. Therefore, we propose an alternative approach as follows.

We consider N classes in the real samples, each with distribution $\mathbb{P}_r^{(i)}, 1 \leq i \leq N$. For a certain generator with distribution \mathbb{P}_{θ_g} , we use a critic with N outputs to measure the Wasserstein distance between \mathbb{P}_{θ_g} and $\mathbb{P}_r^{(i)}$ respectively, i.e.

$$W(\mathbb{P}_r^{(i)}, \mathbb{P}_{\theta_g}) = \max_{\theta_c \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r^{(i)}} [f_{\theta_c}^{(i)}(x)] - \mathbb{E}_{x \sim \mathbb{P}_{\theta_g}} [f_{\theta_c}^{(i)}(x)]. \quad (3)$$

Suppose we are training a generator to generate samples of class i . The generated distribution should have minimized Wasserstein distance to $\mathbb{P}_r^{(i)}$, while its Wasserstein distance to another class $\mathbb{P}_r^{(j)}$ ($j \neq i$) should exceed a margin in order to be distinguishable across classes. Therefore, we redefine the generator loss as

$$\mathcal{L}_g^{(i)}(\theta_g^{(i)}) = (1 - \beta)W(\mathbb{P}_r^{(i)}, \mathbb{P}_{\theta_g}^{(i)}) + \frac{\beta}{N-1} \sum_{j \neq i} \left[W(\mathbb{P}_r^{(i)}, \mathbb{P}_{\theta_g}^{(i)}) - W(\mathbb{P}_r^{(j)}, \mathbb{P}_{\theta_g}^{(i)}) + \alpha \right]_+, \quad (4)$$

where $[\cdot]_+$ stands for a ReLU. Eqn. (4) contains a term motivated by the triplet loss [Schroff *et al.*, 2015], where we use the Wasserstein distance between distributions to replace the L2 distance between samples. α refers to the enforced margin, and β refers to the weight on the negative loss.

In the paraphrase generation problem, some datasets contain both positive and negative samples. With multi-class WGAN, the generator is able to learn from the positive samples to generate flexible paraphrases, while also exploits from negative sample to improve the generation reliability. Given the real positive and negative distributions as $\mathbb{P}_r^{(p)}$ and $\mathbb{P}_r^{(n)}$,

the paraphrase generator loss is formulated as

$$\mathcal{L}_g(\theta_t, \theta_d) = (1 - \beta)W(\mathbb{P}_r^{(p)}, \mathbb{P}_{\theta_t, \theta_d}) + \beta[W(\mathbb{P}_r^{(p)}, \mathbb{P}_{\theta_t, \theta_d}) - W(\mathbb{P}_r^{(n)}, \mathbb{P}_{\theta_t, \theta_d}) + \alpha]_+. \quad (5)$$

3.3 Continuous Approximation

Applying adversarial training algorithm on RNN based text generator is hard since the generated sequence is discrete and non-differentiable. One approach to tackle this problem is to use REINFORCE [Sutton *et al.*, 2000] algorithm. However, the sampling-based gradient estimation suffers from high variance and unstable training. Instead, we use the Gumbel-softmax [Jang *et al.*, 2017] trick as a continuous approximation to handle the discrete sequence generation problem. In the decoding process of the paraphrase sequence generation, we use the Gumbel-softmax distribution to replace the sampled token feeding to the next RNN step, i.e.

$$y_i = \frac{\exp[(\log \pi_i + g_i)/\tau]}{\sum_{j=1}^V \exp[(\log \pi_j + g_j)/\tau]}, \quad \text{for } 1 \leq i \leq V \quad (6)$$

where $[\pi_1, \dots, \pi_V]$ is the probabilities of decoding tokens, V is the vocabulary size, $\tau > 0$ is a temperature parameter, and $g_i \sim \text{Gumbel}(0, 1)$ distribution. Such reparameterization trick provides a reasonable approximation and makes the generator differentiable that allows gradients to back-propagate in training process.

3.4 Critic Model

Motivated by [Shen *et al.*, 2017], we use professor-forcing [Lamb *et al.*, 2016] to match the decoding hidden states of auto-encoder and paraphrase generator, since they share the same decoder parameters. The hidden states of the paraphrase generator are trained to be indistinguishable from hidden states of the teacher-forced auto-encoder. By using hidden states as critic input, the Wasserstein distance defined in Eqn. (3) is reformulated as

$$W_{\theta_c}(\mathbb{P}_r^{(i)}, \mathbb{P}_{\theta_t, \theta_d}) = \mathbb{E}_{\mathbf{h}_y \sim \mathbb{P}_r^{(i)}} [f_{\theta_c}^{(i)}(\mathbf{h}_y | \mathbf{h}_x)] - \mathbb{E}_{\mathbf{h}_t \sim \mathbb{P}_{\theta_t, \theta_d}} [f_{\theta_c}^{(i)}(\mathbf{h}_t | \mathbf{h}_x)] \quad (7)$$

where $\theta_c = \arg \min \mathcal{L}_c^{(i)}(\theta_c)$. In order to enforce the Lipschitz constraint in the WGAN critic, we use the recently proposed gradient penalty method [Gulrajani *et al.*, 2017]. A penalty term on gradient norm is added to the critic loss, i.e.

$$\mathcal{L}_c^{(i)}(\theta_c) = \mathbb{E}_{\mathbf{h}_t \sim \mathbb{P}_{\theta_t, \theta_d}} [f_{\theta_c}^{(i)}(\mathbf{h}_t | \mathbf{h}_x)] - \mathbb{E}_{\mathbf{h}_y \sim \mathbb{P}_r^{(i)}} [f_{\theta_c}^{(i)}(\mathbf{h}_y | \mathbf{h}_x)] + \lambda \mathbb{E}_{\hat{\mathbf{h}} \sim \mathbb{P}_{\hat{\mathbf{h}}}} [(\|\nabla_{\hat{\mathbf{h}}} f_{\theta_c}^{(i)}(\hat{\mathbf{h}} | \mathbf{h}_x)\|_2 - 1)^2] \quad (8)$$

for $i = p$ or n , and $\hat{\mathbf{h}}$ is sampled randomly from linear interpolation of real and generated samples.

We use a CNN model for the critic. For hidden states \mathbf{h}_x and \mathbf{h} ($\mathbf{h} = \mathbf{h}_y$ or \mathbf{h}_t), we combine the two tensors into a 2-dimensional image like representation. For the i -th hidden state $\mathbf{h}_{x,i}$ in \mathbf{h}_x and j -th hidden state \mathbf{h}_j in \mathbf{h} , the two hidden

Algorithm 1 Proposed paraphrase generation algorithm

Input: Positive and negative paraphrase sentences pair distributions $\mathbb{P}_r^{(p)}$ and $\mathbb{P}_r^{(n)}$, parameters α, β, λ and τ

- 1: Initialize $\theta_e, \theta_d, \theta_t, \theta_c$
- 2: **repeat**
- 3: Sample mini-batches of $(\mathbf{x}^{(p)}, \mathbf{y}^{(p)})$ and $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$ from $\mathbb{P}_r^{(p)}$ and $\mathbb{P}_r^{(n)}$, respectively
- 4: Sample a random pattern embedding $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I}_{N_z})$
- 5: Compute paraphrase latent code $\mathbf{c}_t = \text{trs}_{\theta_t}(\mathbf{x}^{(p)} | \mathbf{z})$
- 6: Compute the paraphrase decoding hidden states in free-run mode with Gumbol-softmax approximation $\mathbf{h}_t = \text{dec}_{\theta_d}(\mathbf{c}_t)$
- 7: **for** $i = p, n$ **do**
- 8: Compute the encoder latent code $\mathbf{c}_x^{(i)} = \text{enc}_{\theta_e}(\mathbf{x}^{(i)})$ and $\mathbf{c}_y^{(i)} = \text{enc}_{\theta_e}(\mathbf{y}^{(i)})$
- 9: Compute decoder hidden states in teacher-forcing mode as $\mathbf{h}_x^{(i)} = \text{dec}_{\theta_d}(\mathbf{c}_x^{(i)})$ and $\mathbf{h}_y^{(i)} = \text{dec}_{\theta_d}(\mathbf{c}_y^{(i)})$
- 10: Compute the auto-encoder loss $\mathcal{L}_{\text{AE}}^{(i)}(\theta_e, \theta_d)$ by Eqn. (1)
- 11: Combine hidden states as Eqn. (9), and compute critic output $f_{\theta_c}^{(i)}(\mathbf{h}_t | \mathbf{h}_x^{(i)})$ and $f_{\theta_c}^{(i)}(\mathbf{h}_y^{(i)} | \mathbf{h}_x^{(i)})$
- 12: Compute Wasserstein distance $W(\mathbb{P}_r^{(i)}, \mathbb{P}_{\theta_t, \theta_d})$ by Eqn. (7) and critic loss $\mathcal{L}_c^{(i)}(\theta_c)$ by Eqn. (8)
- 13: **end for**
- 14: Compute the paraphrase generator loss $\mathcal{L}_g(\theta_t, \theta_d)$ by Eqn. (5)
- 15: Update $\{\theta_e, \theta_d, \theta_t\}$ by gradient descent on loss

$$\mathcal{L}_g + \frac{1}{2}(\mathcal{L}_{\text{AE}}^{(n)} + \mathcal{L}_{\text{AE}}^{(p)})$$

- 16: Update θ_c by gradient descent on loss $\mathcal{L}_c = \mathcal{L}_c^{(p)} + \mathcal{L}_c^{(n)}$
 - 17: **until** convergence
 - 18: **return** paraphrase generation model $\text{dec}_{\theta_d}[\text{trs}_{\theta_g}(\mathbf{x} | \mathbf{z})]$
-

state vectors and their element-wise difference and product are concatenated together forming a feature map as

$$\mathbf{w}_{i,j} = [\mathbf{h}_{x,i}^T, \mathbf{h}_j^T, |\mathbf{h}_{x,i} - \mathbf{h}_j|^T, (\mathbf{h}_{x,i} \odot \mathbf{h}_j)^T]^T \quad (9)$$

The feature map is then fed into a CNN feature extraction network proposed in [Gong *et al.*, 2018], which consists of several DenseNet [Huang *et al.*, 2017] blocks and transition blocks. The extracted features are followed by an MLP to output the final estimation of Wasserstein distances.

The overall training procedure of the proposed paraphrase generation model is detailed in Algorithm 1.

4 Experiments

4.1 Datasets

We train and evaluate our paraphrase generation model on the Quora question pairs¹ dataset and the MSCOCO² dataset. The Quora dataset contains question pairs with human annotations originally aiming for paraphrase identification. Therefore, besides the positive paraphrase examples, Quora dataset also contains non-trivial negative examples, in which a pair of questions may share similar words but have different meanings. These negative examples are helpful for our proposed

¹<https://www.kaggle.com/c/quora-question-pairs>

²<http://cocodataset.org>

Dataset	Generator			Critic	
	#Train	#Test	#Validation	#Positive	#Negative
Quora	126K	4.5K	4.5K	126K	184K
MSCOCO	75K	20K	20K	75K	75K

Table 1: Statistics of datasets.

Models	Quora			
	BLEU	ROUGE-1	ROUGE-2	METEOR
Residual LSTM	29.63	58.89	30.72	31.62
VAE-SVG	26.58	50.92	23.44	26.36
Adversarial NMT	30.57	55.95	31.00	33.56
MC-WGAN (average)	27.54	56.45	27.75	28.14
MC-WGAN (best)	32.33	62.66	36.06	33.16

Table 2: Automatic results on the Quora dataset.

multi-class WGAN model. The Quora dataset consists of over 400K question pairs, after filtering question over 20 words, we get 126K positive samples and 184K negative samples for the training set. For testing and validation, we use two sets each with 4.5K positive samples. The MSCOCO contains an image captioning dataset with about 120K images with each having 5 human annotated captions, which are used by some previous works [Gupta *et al.*, 2018] as a paraphrase dataset. In this paper, we sample 75K pairs of captions to identical images in MSCOCO as the positive training set. We also randomly sample another 75K pairs of captions of different images as negative set. Each of the testing and validation sets we use consists of 20K samples of caption pairs. Since MSCOCO dataset does not contain annotated negative samples, we only use it to demonstrate the fidelity of our model across datasets. The statistics of the two datasets used in this paper is presented in Table 1.

4.2 Training Details

We use the 300-dimensional pre-trained GloVe³ word embeddings in our model. The max length of input and output sentence is set as 20. We implement the encoder, transcoder and decoder using RNNs with GRU cells. The encoder and transcoder are two 2-layers bidirectional GRU networks with inner-attention, and the decoder is a single-layer GRU network. The sizes of all the GRU hidden states are 512. The dimension of pattern embedding is 128. The DenseNet blocks and transition blocks in the critic are implemented the same as [Gong *et al.*, 2018], except all the activation units are replaced by Leaky-ReLU.

Before the adversarial training, we firstly pre-train the auto-encoder and the transcoder with the MLE metric. The auto-encoder RNN is pre-trained in the teacher-forcing mode. However, the transcoder needs to be trained in free-run mode, where the Gumbel-softmax distribution of last state output given by Eqn. (6) is used as the next step input.

Models	MSCOCO			
	BLEU	ROUGE-1	ROUGE-2	METEOR
Residual LSTM	21.90	33.21	11.53	16.27
VAE-SVG	21.92	36.32	10.72	16.05
Adversarial NMT	21.68	36.01	11.75	17.16
MC-WGAN (average)	22.22	35.31	11.52	15.63
MC-WGAN (best)	27.83	48.42	22.93	22.78

Table 3: Automatic results on the MSCOCO dataset.

Models	Relevance	Fluency
VAE-SVG	3.75	4.07
MC-WGAN	4.09	4.22
Reference	4.88	4.95

Table 4: Human evaluation results on Quora dataset.

5 Results and Analysis

5.1 Baselines

We compare the results of our proposed model with several existing paraphrase generation models, i.e. residual LSTM [Prakash *et al.*, 2016] (with two layers), VAE-SVG [Gupta *et al.*, 2018] and Adversarial NMT [Wu *et al.*, 2018]. The reinforcement learning based Adversarial-NMT model is originally used in machine translation. We use it as a paraphrase generation model by sharing vocabulary and word embeddings between the source and target languages. These models represent the typical approaches of neural paraphrase generation, and we use them as baselines to evaluate our proposed model.

5.2 Automatic Evaluation

We first conduct automatic quantitative evaluations to compare the paraphrase generation performance using BLEU-4 [Papineni *et al.*, 2002], ROUGE-1 and ROUGE-2 [Lin, 2004], and METEOR [Denkowski and Lavie, 2014]. These metrics mainly consider the precision and recall of n-grams between the generated sentences and the references. Synonyms from WordNet are also considered in METEOR. However, these lexical metrics are not ideally suitable for the evaluation of paraphrase generation, because good paraphrasing examples may exist besides the given references. This occurs more seriously when a model is aiming at generating diverse paraphrasing samples, since the generation diversity is traded-off to the accuracy on specific references. Therefore, we only use these automatic metrics as part of our evaluation along with human evaluation.

Table 2 and 3 show the performance of models on the Quora and MSCOCO datasets respectively. Since our proposed multi-class WGAN (MC-WGAN) model can generate multiple paraphrases of a given sentence, we list the average and best results separately. Table 2 shows that the best performance of our model outperforms the baseline models in all the considered metrics except for METEOR, which is close to the Adversarial NMT. This indicates our model has the ability

³<https://nlp.stanford.edu/projects/glove/>

Input	Reference	Generated z_1	Generated z_2	Generated z_3
how do you start making money?	what should i do to earn some more money?	how do i make money through youtube?	how do i make money from home?	what are some ways to make money online?
how effective is scrapping 500 and 1000 rupee notes ? will it reduce black money?	how will the ban on 500 and 1000 rupee note stop black money?	how will banning 500 and 1000 rupee notes affect black money?	how will the demonetization of 500 and 1000 rupee notes help indian economy?	how will the ban of 500 and 1000 rupee notes help indian economy?
what are the worst mistakes of your life?	what is the worst thing you did by mistake in your life?	what is the worst mistake you have in your life?	what was the most embarrassing moment of your life?	what has been the worst experience of your life?

Table 5: Some examples of paraphrases generated on Quora dataset.

Input	Reference	Generated z_1	Generated z_2	Generated z_3
a group of kids playing a game of baseball.	the young boys are playing a game of baseball in the park.	a group of young children playing a game of baseball	a group of baseball players playing a game on the playground.	three young children playing baseball on a baseball team.
a man playing tennis going for a low ball	a tennis player with a racket hitting the ball	a man in a tennis court about to hit a tennis ball.	a tennis player in a defensive stance to hit a ball with a racket.	a man in a tennis court gets ready to hit a ball.
small pieces of cake have been arranged on a plate	chocolate dessert bars covered in frosting and sprinkles.	three pieces of cake are on a plate with a cut of syrup.	two pieces of cake are on a plate with strawberries.	three cakes on a plate that have been sliced on top.

Table 6: Some examples of paraphrases generated on MSCOCO dataset.

to generate result close to the reference, i.e. the best results with respect to the ground truth are within our generation distribution. This is also shown by Table 3 on the MSCOCO dataset. Table 2 shows the average performance of our model is no better than the Residual LSTM and Adversarial NMT on Quora dataset, because both Residual LSTM and Adversarial NMT model contain MLE terms in their generator loss and tend to generate samples close to the ground truth. However, with the help of GAN, our model mainly focuses on a distribution perspective. VAE-SVG model is also enabled to generate multiple paraphrases. Table 2 shows the average performance of our model outperforms VAE-SVG on Quora dataset, since the MC-WGAN learns from both positive and negative samples. However, on the MSCOCO dataset, performance gains only show on BLEU and ROUGE-2, because the negative samples are randomly selected.

5.3 Human Evaluation

Table 4 shows the human evaluation performance on Quora dataset, where we mainly compare our model against the VAE-SVG model since both the two are generative models that generate diverse paraphrase results. We randomly choose 200 sentences generated by each model, and assign all the tasks to 3 individual human evaluators to score ranging from 1 to 5 according to the relevance and fluency of each paraphrasing pair. (1 refers to the worst and 5 refers to the best). Results show that our proposed model generates better paraphrasing samples than the VAE-SVG model in both relevance and fluency metrics on Quora dataset. This is partially because our model succeeds in taking advantage of the negative samples to learn better generation distribution.

5.4 Generation Diversity

Table 5 and 6 show some examples of paraphrases generated with our model on Quora and MSCOCO dataset. By

sampling on the pattern embedding vector z , the model is able to generate multiple paraphrases of a given sentence. The shown examples capture the accurate semantic of the input sentences, while provide reasonable variation in the paraphrasing outputs. The results on MSCOCO show greater variation in the paraphrases than the Quora dataset. This is because different captions may describe one image from different aspects, which means the captions may not be strictly semantically identical as the human annotated samples in Quora dataset. Our model is able to capture this feature in the generation phase, which leads the generator to add more variational details in the results.

6 Conclusions

In this paper, we have proposed an alternative deep generative model based on WGAN to generate paraphrase of given text with diversity. We build our model with an auto-encoder along with a transcoder. The transcoder is conditioning on an explicit pattern embedding variable, and transcodes an input sentence into its paraphrasing term in latent space. Consequently, diverse paraphrases can be generated by sampling on the pattern embedding variable. We apply WGAN to force the decoding paraphrase distribution to match the real distribution. By extending WGAN to multiple class generation, the generative model is enabled to learn from both the positive and negative real distributions for better generation quality. The proposed model is evaluated on two datasets with both automatic metrics and human evaluation. Results show that our proposed model can generate fluent and reliable paraphrase samples that outperform the state-of-art results, while also provides reasonable variability and diversity at the same time. Our model provides a new baseline in generative paraphrase modeling. The proposed model with the multi-class WGAN algorithm can be potentially applied in may

other text generation tasks with multiple labels, such as natural language inference generation, in the future works.

References

- [Arjovsky *et al.*, 2017] Martin Arjovsky, Soumith Chintala, and Leon Bottou. Wasserstein Generative Adversarial Network. In *ICML*, 2017.
- [Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.
- [Bowman *et al.*, 2016] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *CoNLL*, 2016.
- [Cao *et al.*, 2017] Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. Joint copying and restricted generation for paraphrase. In *AAAI*, 2017.
- [Denkowski and Lavie, 2014] Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380, 2014.
- [Gong *et al.*, 2018] Yichen Gong, Heng Luo, and Jian Zhang. Natural Language Inference over Interaction Space. In *ICLR*, 2018.
- [Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in NIPS*, pages 2672–2680, 2014.
- [Gulrajani *et al.*, 2017] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in NIPS*, pages 5767–5777, 2017.
- [Gupta *et al.*, 2018] Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. A Deep Generative Framework for Paraphrase Generation. In *AAAI*, 2018.
- [Hu *et al.*, 2017] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *ICML*, pages 1587–1596, 2017.
- [Huang *et al.*, 2017] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on CVPR*, pages 4700–4708, 2017.
- [Jang *et al.*, 2017] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *ICLR*, 2017.
- [Kingma and Welling, 2014] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.
- [Lamb *et al.*, 2016] Alex M Lamb, Anirudh Goyal, Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In NIPS*, pages 4601–4609, 2016.
- [Lample *et al.*, 2018] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc’Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *ICLR*, 2018.
- [Li *et al.*, 2018] Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. Paraphrase generation with deep reinforcement learning. In *Proceedings of the 2018 Conference on EMNLP*, pages 3865–3878, 2018.
- [Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.
- [McKeown, 1983] Kathleen R McKeown. Paraphrasing questions using given and new information. *Computational Linguistics*, 9(1):1–10, 1983.
- [Mirza and Osindero, 2014] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [Odena *et al.*, 2017] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, pages 2642–2651, 2017.
- [Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th ACL*, pages 311–318, 2002.
- [Prakash *et al.*, 2016] Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. Neural paraphrase generation with stacked residual lstm networks. In *Proceedings of COLING 2016*, pages 2923–2934, 2016.
- [Quirk *et al.*, 2004] Chris Quirk, Chris Brockett, and William Dolan. Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 Conference on EMNLP*, 2004.
- [Ranzato *et al.*, 2016] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.
- [Schroff *et al.*, 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of CVPR*, pages 815–823, 2015.
- [Shen *et al.*, 2017] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style Transfer from Non-Parallel Text by Cross-Alignment. In *NIPS*, 2017.
- [Sutton *et al.*, 2000] Richard S Sutton, David Mcallester, Satinder Singh, Yishay Mansour, Park Avenue, and Florham Park. Policy Gradient Methods for Reinforcement Learning with Function Approximation. *NIPS*, 1(5):1057–1063, 2000.
- [Wu *et al.*, 2018] Lijun Wu, Yingce Xia, Fei Tian, Li Zhao, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Adversarial neural machine translation. In *Asian Conference on Machine Learning*, pages 534–549, 2018.

[Xu *et al.*, 2018] Qionkai Xu, Juyan Zhang, Lizhen Qu, Lexing Xie, and Richard Nock. D-PAGE: Diverse paraphrase generation. *arXiv preprint arXiv:1808.04364*, 2018.