An End-to-End Generative Architecture for Paraphrase Generation

Qian Yang[1*], **Zhouyuan Huo**[2], **Dinghan Shen**[1]
Yong Cheng[3], **Wenlin Wang**[1], **Guoyin Wang**[1], **Lawrence Carin**[1]

[1] Duke University [2] University of Pittsburgh [3] Google AI
qian.yang@duke.edu

Abstract

Generating high-quality paraphrases is a fundamental yet challenging natural language processing task. Despite the effectiveness of previous work based on generative models, there remain problems with exposure bias in recurrent neural networks, and often a failure to generate realistic sentences. To overcome these challenges, we propose the first end-to-end conditional generative architecture for generating paraphrases via adversarial training, which does not depend on extra linguistic information. Extensive experiments on four public datasets demonstrate the proposed method achieves state-of-the-art results, outperforming previous generative architectures on both automatic metrics (BLEU, METEOR, and TER) and human evaluations.

1 Introduction

Paraphrases convey the same meaning as the original sentences or text, but with different expressions in the same language. Paraphrase generation aims to synthesize paraphrases of a given sentence automatically. This is a fundamental natural language processing (NLP) task, and it is important for many downstream applications (Madnani and Dorr, 2010; Yang et al., 2016, 2017; Passonneau et al., 2018). For example, paraphrases can help diversify the response of chatbot engines (Yan et al., 2016), strengthen question answering (Harabagiu and Hickl, 2006; Duboue and Chu-Carroll, 2006; Fader et al., 2014), augment relation extraction (Romano et al., 2006), and extend the coverage of semantic parsers (Berant and Liang, 2014).

Generating accurate and diverse paraphrases automatically is still very challenging. Traditional methods (McKeown, 1983; Bolshakov and Gelbukh, 2004; Zhao et al., 2008) exploit how linguistic knowledge can improve the quality of generated paraphrases, including shallow linguistic features (Zhao et al., 2009), and syntactic and semantic information (Kozlowski et al., 2003; Ellsworth and Janin, 2007). However, they are often domain-specific and hard to scale, or yield inferior results.

With the help of growing large data and neural network models, recent studies have shown promising results. Three families of deep learning architectures have been investigated with the goal of generating high-quality paraphrases. The first is to formulate paraphrase generation as a sequence-to-sequence problem, following experience from machine translation (Bahdanau et al., 2014; Cheng et al., 2016). Prakash et al. (2016) proposes stacked residual long short-term memory networks (LSTMs), while Cao et al. (2017) makes use of the gated recurrent unit (GRU) as the recurrent unit. Shortly afterwards, several works have focused on providing extra information to enhance the encoder-decoder model. Ma et al. (2018) adds distributed word representations, Huang et al. (2018) considers a paraphrased dictionary and

Iyyer et al. (2018) utilizes a syntactic parser. Recently Wang et al. (2018) utilizes a semantic augmented Transformer (Vaswani et al., 2017); however, its accuracy largely depends on extra semantic information. The second family of models employs reinforcement learning (Li et al., 2017b). The third family is the generative architecture that we focus on. Specifically, Gupta et al. (2018) applies the conditional variational autoencoders (CVAEs) (Sohn et al., 2015) with LSTM encoder-decoders in generating paraphrases, expecting RNN's advantage of modeling local dependencies to be a good complement to the CVAE's power at learning global representations. This simple combination, however, has two major challenges.

First, CVAE may fail to generate realistic sen-

[*] Corresponding author

original	How do I improve my English speaking ?
paraphrased	How do I speak English fluently ?
generated	How do I I to ?
original	What is the easiest way to lose weight faster?
paraphrased	What is the best way to loose weight quickly ?
generated	How can I learn lose weight ?

Table 1: Examples for generating paraphrases by CVAE, from which it fails to generate realistic sentences. Original: source sentence. Paraphrased: target sentence. Generated: the sentence generated by CVAE.

tences. When generating synthetic sentences by decoding random samples from the latent space, most regions of the latent space do not necessarily decode to realistic sentences; the example in Table 1 reflects this challenge. In (Bowman et al., 2016), the authors attempted to utilize RNN-based VAE to generate more diverse sentences. However, this ignores the fundamental problem of the posterior distribution over latent variables not covering the latent space appropriately. Second, when learning, the ground-truth words are used to decode, while during inference, the RNN generates words in sequence from previously *generated* words. Bengio et al. (2015) called this phenomenon "Exposure Bias" and tried to solve it by a scheduled sampling method. However, in practice, it produced unstable results, because it is fundamentally an inconsistent training strategy (Huszár, 2015).

We propose the first paraphrase generative model via adversarial training to tackle the above problems, which we believe is a natural answer to the aforementioned challenges. Specifically, we formalize CVAE as a generator of the Generative Adversarial Network (GAN) (Goodfellow et al., 2014) and tailor a discriminator to CVAE. By introducing an adversarial game between a generator and a discriminator, GAN matches the distribution of synthetic data with that of real data. The generator of GAN seeks to map samples from a given prior distribution to *realistic* synthetic data. The discriminator of GAN compares the entire real and synthetic sentences, instead of individual words, which should in principle alleviates the exposure-bias issue (Yu et al., 2017; Li et al., 2017a; Zhang et al., 2017; Guo et al., 2018). The intuition behind our model can be interpreted as using CVAE to generate similar sentences and enhancing CVAE with an extended discriminator, so that the generator and discriminator work effectively together; they provide feedback signals to each other, resulting in a mutual adversarial framework. Overall, our contributions are as follows.

- To the best of our knowledge, this work represents the first to propose an end-to-end paraphrase generation architecture via adversarial training, which does not require extra linguistic information.

- We take advantage of GAN to help choose a better latent-variable distribution. By this, we not only utilize the better latent variables but also strengthen the expressiveness of the generative model.

- Our experiments show that the proposed model is capable of generating plausible paraphrased sentences and outperforms competitive baseline models, with state-of-the-art results.

2 Preliminaries

We first provide preliminaries of variational auto-encoders and conditional variational auto-encoders.

VAE. The variational auto-encoder (VAE) (Kingma and Welling, 2013; Sohn et al., 2015) is a directed graphical model with latent variables. The generative process of VAE employs an

- (Encoder) generating a set of latent variable $\boldsymbol{z}$ from the prior distribution $p_\theta(\boldsymbol{z})$, where θ is the generative parameters;

- (Decoder) then generating the data $\boldsymbol{x}$ from the generative distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ conditioned on $\boldsymbol{z}$.

Although due to intractable posterior inference, parameter estimation of directed graphical models is generally challenging, VAE parameters can be estimated efficiently by a stochastic gradient variational bayes (SGVB) estimator (Kingma and Welling, 2013), and can be optimized straightforwardly using standard stochastic gradient techniques.

SGVB treats the variational lower bound of the log-likelihood as a surrogate objective function, which can be written as:

$$\begin{aligned}\log p_\theta(\boldsymbol{x}) &= -KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z})) \\ &\quad +KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x})) \\ &\quad +\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] \\ &\geq -KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z})) \\ &\quad +\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})], \qquad (1)\end{aligned}$$

where the inequality follows because the second term $KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x}))$ in (1) is nonnegative.

In practice, we can approximate the second term by drawing latent variable samples $\{\boldsymbol{z}^1, \boldsymbol{z}^2, ..., \boldsymbol{z}^L\}$ following $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, where ϕ is the variational parameters, and the empirical objective of VAE with Gaussian latent variables can be represented as:

$$\begin{aligned}\tilde{\mathcal{L}}_{VAE}(\boldsymbol{x};\theta,\phi) = -KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z})) \\ +\frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\boldsymbol{x}|\boldsymbol{z}^l), \qquad (2)\end{aligned}$$

where $\boldsymbol{z}^l = g_\phi(\boldsymbol{x}, \epsilon^l)$, and $\epsilon^l \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. That means $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is reparameterized with a differentiable unbiased function $g_\phi(\boldsymbol{x}, \epsilon^l)$, where $\boldsymbol{x}$ is the data and ϵ is the noise variable. VAE can be trained efficiently by stochastic gradient descent (SGD) because the reparameterization trick allows error backpropagation through the Gaussian latent variables.

CVAE. Sohn et al. (2015) develops a deep conditional generative model (CVAE) for structured output prediction using Gaussian latent variables. Different from the normal VAE, CVAE consists of three variables: input variables $\boldsymbol{x}$, output variables $\boldsymbol{y}$, and latent variables $\boldsymbol{z}$, and its prior distribution is $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. CVAE is trained to maximize the conditional log-likelihood $\log p_\theta(\boldsymbol{y}|\boldsymbol{x})$, and again it is formulated in the framework of SGVB. The variational lower bound of the model is:

$$\begin{aligned}\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) &\geq -KL(q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x})) \\ &\quad +\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})}[\log p_\theta(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{z})] (3)\end{aligned}$$

and the empirical lower bound is represented as:

$$\begin{aligned}\tilde{\mathcal{L}}_{CVAE}(\boldsymbol{x},\boldsymbol{y};\theta,\phi) &= -KL(q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x})) \\ &\quad +\frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{z}^l), \qquad (4)\end{aligned}$$

where $z^l = g_\phi(\boldsymbol{x}, \boldsymbol{y}, \epsilon^l), \epsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and L is the number of samples.

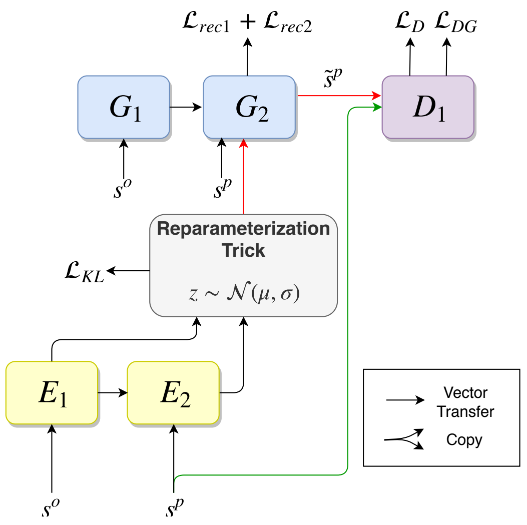

Figure 1: Our generative model. The red arrows denote the process for generating sentences and the green one denotes the transmission of target paraphrased sentences.

3 Model

Our new model, which we call GAP for Generative Adversarial Paraphrase model, targets the goal of generating plausible paraphrased sentences conditioned on the original sentences, without any extra linguistic information. We first propose our end-to-end conditional generative architecture for generating paraphrase via adversarial training. Two training techniques are then described for the proposed model.

3.1 GAP

We consider a dataset with pairs of the original sentence and the paraphrased sentence $\{(s_k^o, s_k^p)\}_{k=1}^K$. For a given sentence s, let w_t denote the t-th word. Each word w_t is embedded into a d-dimensional vector $\boldsymbol{x_t} = \mathbf{W_e}[w_t]$, where $\mathbf{W_e} \in \mathbb{R}^{d \times V}$ is a learned word embedding matrix, V is the vocabulary size, and the v-th column of $\mathbf{W_e}$ is $\mathbf{W_e}[v]$. All columns of $\mathbf{W_e}$ are normalized to have unit ℓ_2-norm, *i.e.*, $\|\mathbf{W_e}[v]\|_2 = 1, \forall v$, by dividing each column with its ℓ_2-norm. After embedding, a sentence of length T (padded when necessary) is represented as $\mathbf{X} \in \mathbb{R}^{d \times T}$, by simply concatenating its word embeddings, *i.e.*, $\boldsymbol{x_t}$ is the t-th column of $\mathbf{X}$.

The training set is $\mathcal{S} = \{(s_k^o, s_k^p)\}_{k=1}^K$ of pairwise data, where $s_k^o \in \mathcal{S}^o$ denotes the original sentence sequence, and $s_k^p \in \mathcal{S}^p$ denotes the reference paraphrased sentence sequence. The goal of our model is to learn a function f: $\mathcal{S}^o \mapsto \mathcal{S}^p$. The overall model structure of the proposed method is shown in Figure 1. The components can be divided into three modules: encoder in yellow, decoder in blue (which is also the generator), and the discriminator in purple. The connections between them are drawn in arrows.

Encoder E. The encoder E consists of two LSTM networks E_1 and E_2, and E generates latent variable $\boldsymbol{z}$ from the input sentences. There are two different paths generating latent variable $\mathbf{z}$. In the first path, we input sampled s^o and s^p into E_1 and E_2, respectively, and generate sufficient statistics of a Gaussian distribution, mean μ and standard deviation σ. Therefore, the distribution of the latent variable $\mathbf{z}$ can be represented as $q(\mathbf{z}|s^o, s^p)$.

However, we cannot know the paraphrased sentence s^p when we evaluate or apply the trained model in real applications. The training framework is not fit for the circumstance at testing time, and only E_1 is used during the inference. Thus, we also compute the other set of sufficient statistics, mean μ' and standard deviation σ' by inputting s^o into encoder E_1. The sampled latent variable $\mathbf{z}$ is only dependent on s^o, and we can represent the distribution of $\mathbf{z}$ as $p(\mathbf{z}|s^o)$.

Decoder/Generator G. We again combine two LSTM decoders G_1 and G_2 as the generator G. At first, we feed the original sentence s^o into decoder G_1 and obtain final state $\boldsymbol{c}_T^{G_1}$, $\boldsymbol{h}_T^{G_1}$. The paraphrased sentence $\tilde{s}^p$ is predicted word by word. At each step, we concatenate the latent variable $\boldsymbol{z}$ and the previously predicted (ground truth) word, and then input it into LSTM decoder G_2 with hidden state from the last step. At timestep 0, the input word is BOS ("begin of sentence" symbol), and the hidden state is the output from G_1. Therefore, the probability of the predicted paraphrased sentence $\tilde{s}^p$, given the encoded feature vector $\boldsymbol{z}$, is defined as:

$$p(\tilde{s}^p|\boldsymbol{z}, s^o) \quad = \quad \prod_{t=1}^{T} p(\tilde{w}_t^p|\tilde{w}_{<t}^p, \boldsymbol{z}, s^o), \qquad (5)$$

where $\tilde{w}_t^p$ is the t-th generated token, $< t \triangleq \{0, \cdots, t-1\}$. w_0^p denotes BOS. All the words in $\tilde{s}^p$ are generated sequentially using the LSTM, based on previously generated words, until the end-of-sentence symbol is generated. The t-th word $\tilde{w}_t^p$ is generated as $\tilde{w}_t^p = \operatorname{argmax}(\mathbf{V}\boldsymbol{h}_t)$. Note that the hidden units $\boldsymbol{h}_t$ are updated through $\mathcal{E}(\tilde{\boldsymbol{x}}_{t-1}^p, \boldsymbol{h}_{t-1}, \boldsymbol{z})$, where $\mathcal{E}$ denotes the transition function in the second LSTM cell related to the updates. The transition function $\mathcal{E}(\cdot)$ is implemented with an LSTM. $\mathbf{V}$ is a weight matrix used to compute a distribution over words. $\tilde{\boldsymbol{x}}_{t-1}^p$ is the embedding vector of the previously generated word $\tilde{w}_{t-1}^p$, *i.e.*,

$$\tilde{\boldsymbol{x}}_{t-1}^p = \mathbf{W}_{\boldsymbol{e}}[\tilde{w}_{t-1}^p], \qquad (6)$$

and is the input for t-th step. Consequently, the generated sentence $\tilde{s}^p = [\tilde{w}_1^p, \cdots, \tilde{w}_T^p]$ is obtained given $\boldsymbol{z}$, by simply concatenating the generated words.

Two-Path Loss. Thus far, we can compute the loss from two paths of the encoder. In the first path, we minimize the KL loss and MLE loss:

$$\begin{aligned}\min_E \quad & \mathrm{KL}(q(\boldsymbol{z}|s^o, s^p)||p(z|s^o)) \\ & -\mathbb{E}_{q(\boldsymbol{z}|s^o,s^p)}\left[\log p(\tilde{s}^p|\boldsymbol{z}, s^o)\right]. \quad (7)\end{aligned}$$

In the second path, the loss is presented as follows:

$$\min_E \quad -\mathbb{E}_{q(\boldsymbol{z}|s^o)}\left[\log p(\tilde{s}^p|\boldsymbol{z}, s^o)\right]. \qquad (8)$$

The proposed two-path reconstruction loss diminishes the gap between prediction pipelines at training and testing, and helps to generate diverse but realistic predictions.

The objective function of optimizing E and G can be written as follows:

$$\begin{aligned}\min_{E,G} \quad & \lambda_1 \mathrm{KL}(q(\boldsymbol{z}|s^o, s^p)||p(z|s^o)) \\ & -\lambda_1 \mathbb{E}_{q(\boldsymbol{z}|s^o,s^p)}\left[\log p(\tilde{s}^p|\boldsymbol{z}, s^o)\right] \\ & -\lambda_2 \mathbb{E}_{q(\boldsymbol{z}|s^o)}\left[\log p(\tilde{s}^p|\boldsymbol{z}, s^o)\right] \\ & -\lambda_3 \mathbb{E}_{\tilde{s}^p \sim p(\tilde{s}^p|\boldsymbol{z}', s^o)} \log(D(\tilde{s}^p)), \quad (9)\end{aligned}$$

where D denotes the discriminator (described below) and $\boldsymbol{z}' \sim q(\boldsymbol{z}|s^o)$. We use λ_1, λ_2 and λ_3 to balance the weights of the different losses.

Discriminator D. We use one LSTM as the discriminator. The goal of the discriminator is to distinguish real paraphrased sentences from generated sentences. Given the embeddings of true paraphrased sentence s^p and the fake generated sentence $\tilde{s}^p$, the discriminator loss is defined as:

$$\begin{aligned}\min_D \quad & -\mathbb{E}_{\tilde{s}^p \sim p(\tilde{s}^p|\boldsymbol{z}', s^o)} \log(1 - D(\tilde{s}^p)) \\ & -\mathbb{E}_{s^p \sim p(s^p)}[\log D(s^p)], \qquad (10)\end{aligned}$$

where $\boldsymbol{z}' \sim q(\boldsymbol{z}|s^o)$.

Algorithm 1 Training Pipeline

```
Initialize: Gradients of corresponding parameters: Encoder $E$ parameters $g_E = [g_{E1}, g_{E2}]$, Decoder/Generator $G$ parameters $g_G = [g_{G1}, g_{G2}]$, Discriminator $D$ parameters $g_D$;
1: while $G$ has not converged do
2:     Sample a batch of $\{s^o, s^p\}$ from dataset;
3:     $\boldsymbol{z}_1 = E(s^o, s^p)$;
4:     $\mathcal{L}_{KL} = KL(q(\boldsymbol{z}_1|s^o, s^p)||p(\boldsymbol{z}|s^o))$;
5:     $\boldsymbol{z}_2 = E_1(s^o)$;
6:     Predict $\tilde{s}^p_1$ following: $\tilde{s}^p_1 = G(s^o, \boldsymbol{z}_1)$;
7:     Predict $\tilde{s}^p_2$ following: $\tilde{s}^p_2 = G(s^o, \boldsymbol{z}_2)$;
8:     $\mathcal{L}_{rec1} = -\log p(\tilde{s}^p_1|\boldsymbol{z}_1, s^o)$;
9:     $\mathcal{L}_{rec2} = -\log p(\tilde{s}^p_2|\boldsymbol{z}_2, s^o)$;
10:    Input $s^p$ and $\tilde{s}^p_2$ into Discriminator $D$;
11:    $\mathcal{L}_D = -\log(1 - D(\tilde{s}^p_2)) - \log D(s^p)$;
12:    $\mathcal{L}_{DG} = -\log(D(\tilde{s}^p_2))$;
13:    $g_{E_1} = \nabla_{E_1}(\lambda_1\mathcal{L}_{KL} + \lambda_1\mathcal{L}_{rec1} + \lambda_2\mathcal{L}_{rec2} + \lambda_3\mathcal{L}_{DG})$;
14:    $g_{E_2} = \nabla_{E_2}(\mathcal{L}_{KL} + \mathcal{L}_{rec1})$;
15:    $g_G = \nabla_G(\lambda_1\mathcal{L}_{rec1} + \lambda_2\mathcal{L}_{rec2} + \lambda_3\mathcal{L}_{DG})$;
16:    $g_D = \nabla\mathcal{L}_D$;
17:    Update model parameters using Adam optimizer with gradients $g_{E1}$, $g_{E2}$, $g_G$ and $g_D$.
18: end while
```

3.2 Training Techniques

We summarize the training pipeline in Algorithm 1. At each iteration, a pair of original and paraphrased sentences are fed into two paths for sentence reconstruction. A discriminator is also used to distinguish between real and generated sentences, which is helpful to the exposure bias problem.

Policy Gradient. Backpropagation of gradients from the discriminative model to the generative model is difficult for sequence generative models. Following Yu et al. (2017), we use the REINFORCE algorithm (Williams, 1992) to approximate gradients with respect to the generator and encoder. In the experiments, we regard the generator G as the stochastic policy and the output of discriminator D as its reward. In this way, we can propagate the gradients from the discriminator to both the generator models and encoder models.

Warm-up Training. It is difficult to train GAN using gradient-based methods (Goodfellow et al., 2014). Previous generative models (Zhang et al., 2017) often pre-train the generator using a supervised model, like an auto-encoder. In this paper, we propose a warm-up technique to train our generative model. Following the notation in Algorithm 1, we suppose a warm-up step t_{wp}. In the training process, we gradually increase the weight of $\mathcal{L}_{DG}$ to a fixed value λ_3. From step $t = 0$ to t_{wp}, the update of λ^t_3 can be represented as

$$\lambda^t_3 = \frac{t}{t_{wp}} \cdot \lambda_3. \tag{11}$$

When $t \geq t_{wp}$, we fix $\lambda^t_3 = \lambda_3$ as a constant.

We also need to balance the losses between (7) and (8). The second loss (8) represents that during inference, z is generated only depending on x^o. We let $\lambda_1 = 1 - \lambda^t_2$ and increase the value of λ^t_2 gradually until t_{wp}. Similarly, λ^t_2 is updated from $t = 0$ to t_{wp} through:

$$\lambda^t_2 = \frac{t}{t_{wp}} \cdot \lambda_2. \tag{12}$$

If $t \geq t_{wp}$, we fix $\lambda^t_2 = \lambda_2$.

4 Experiments

We assess the performance of our GAP model and compare it with previous methods. We first describe the datasets we use, then present the details of experimental setup, and finally analyze both quantitative and qualitative results.

4.1 Datasets

Following previous work (Prakash et al., 2016), we conduct experiments on the same four standard datasets that are used widely for paraphrase generation. Their content is specified below.

MSCOCO. MSCOCO was originally an image-captions dataset, containing over 120K images, with five different captions from five different annotators per image. All the annotations toward one image describe the most prominent object or action in this image, which is suitable for the paraphrase generation task. Specifically, the dataset has two divisions: training dataset ("Train2014", over 82K images) and validation dataset ("Val2014", over 40K images). We follow the operations of previous papers, randomly choosing four captions out of five as two source-reference pairs and limit the length of sentence to be 15 words (removing the words beyond the first 15), so that we can compare our results with published work.

Quora. Quora consists of over 400K lines of potential question pairs which are duplicate to each other if a particular position of this line is annotated with 1. Again, we follow the operations of previous work and filter out those pairs annotated with 1. There are 155K such question pairs in total, among which three sub-datasets are created, *i.e.*, training dataset 50K, 100K, 150K and test dataset 4K. The goal of using three sub-datasets is to show how the size of dataset can affect the results of paraphrase generation.

4.2 Experimental Setup

Encoders E_1, E_2 and generators G_1, G_2 are constructed using 2-layer LSTMs (Hochreiter and Schmidhuber, 1997). For discriminator D, it is also 2-layer LSTM. At the beginning, we map each word to a 300-dimensional feature vector and it is initialized with 300-dimensional GloVe vectors (Pennington et al., 2014). Therefore, a sentence of length N can be represented by a matrix of size $N \times 300$. Before inputting the embedding vector into LSTM models, we pre-process these vectors using a two-layer highway network (Srivastava et al., 2015). We set the dimension of latent variables $\boldsymbol{z}$ to be 300. To balance these losses, we set $\lambda_2 = 0.5$ and $\lambda_3 = 0.01$. The warm-up step is 1×10^4 in all experiments.

We use Adam optimizer (Kingma and Ba, 2014) with learning rate 1×10^{-4} and other parameters as default, for all models. Following (Sutskever et al., 2014), we clip gradients of the encoder and generator parameters if the norm of the parameter vector exceeds 10, and we clip the gradients of the discriminator if the norm of the parameter vector exceeds 5. Greedy search is employed to generate paraphrases in the experiment.

4.3 Automatic Evaluation

We follow previous papers to choose the same three well-known automatic evaluation metrics: BLEU (Papineni et al., 2002), METEOR (Lavie and Agarwal, 2007), and TER (Snover et al., 2006). As studied by (Wubben et al., 2010), human judgments on generated paraphrases correlate well with these metrics. We then compare our results on these metrics with previous baselines below.

Baselines. Our GAP model contains LSTM and VAE components, so we compare it with (*i*) the basic attentive sequence-to-sequence model from Machine Translation (seq2seq) (Bahdanau et al., 2014), (*ii*) the stacked residual LSTM model doing paraphrase generation (Residual LSTM) (Prakash et al., 2016), and (*iii*) a VAE model based on LSTM together with its variants (VAE-S, VAE-SVG, VAE-SVG-eq) (Gupta et al., 2018). Beyond these, we also evaluate other baselines including Transformer (Vaswani et al., 2017), semantic augmented Transformer which requires extra linguistic information (Transformer-PB) (Wang et al., 2018), and a deep reinforcement learning approach (RbM-SL) (Li et al., 2017b).

Model	Measure	MSCOCO		
		BLEU↑	METEOR↑	TER↓
Seq2Seq + Att	-	18.60	16.80	63.00
Residual LSTM	-	37.00	27.00	51.60
Transformer	-	41.00	32.80	40.50
Transformer-PB	-	44.00	34.70	37.10
VAE-SVG	best-B	41.70	30.80	41.70
VAE-SVG	best-M	41.30	31.00	41.60
VAE-SVG	best-T	41.30	30.90	40.80
VAE∗	avg	42.53	32.77	41.38
Ours	best-B	**45.60**	35.72	39.47
Ours	best-M	41.32	**36.17**	40.52
Ours	best-T	42.46	34.79	**36.83**

Table 2: Test accuracy on MSCOCO dataset, in percentage. VAE∗ is our implementation of VAE. "best-B", "best-M" and "best-T" represent the scores with the best BLEU, METEOR and TER respectively. "avg" denotes an average over "best-B", "best-M", and "best-T". All other results are directly cited from the respective papers. For different model variants exist in one paper, we only show the one with highest scores here. For the arrows ↑ of BLEU and TER, a higher score is better. ↓ of TER represents a lower score is better. The best results are in boldface.

Results and Analysis. The results on MSCOCO are shown in Table 2. We find that our GAP model outperforms the baseline models on all metrics. For example, comparing our scores on MSCOCO dataset when we achieve the best BLEU with the ones from VAE-SVG, we improve the results about 4 BLEU, 5 METEOR, and 4 TER, respectively.

Table 3 shows the performance on Quora dataset. We notice that our accuracy increases with the increasing size of data. Meanwhile, our model is more robust for a relatively smaller dataset such as Quora-50K, leveraging its advantage in learning with fewer data. For example, we achieve much better results than VAE and its variants on the smaller Quora-50k.

Ablation Study. VAE∗ in Table 3 extends the pre-

Model	Measure	Quora-50K			Quora-100K			Quora-150K		
		BLEU↑	METEOR↑	TER↓	BLEU↑	METEOR↑	TER↓	BLEU↑	METEOR↑	TER↓
Seq2Seq + Att	-	26.06	20.35	-	36.55	26.28	-	-	-	-
Residual LSTM	-	27.32	22.37	-	37.38	28.17	-	-	-	-
RbM-SL	-	35.81	**28.12**	-	43.54	**32.84**	-	-	-	-
VAE-S	best-B	15.80	20.10	69.40	17.50	21.60	67.10	19.80	22.60	63.90
VAE-S	best-M	15.60	21.10	71.50	17.50	22.70	69.50	19.70	23.80	66.90
VAE-SVG	best-B	17.10	21.30	63.10	22.50	24.60	55.70	30.30	28.50	47.30
VAE-SVG	best-M	17.10	22.20	63.80	22.40	25.50	55.60	30.30	29.20	47.10
VAE-SVG-eq	best-B	17.40	21.40	61.90	22.90	24.70	55.00	31.40	29.00	46.80
VAE-SVG-eq	best-M	17.30	22.20	62.60	22.90	25.50	54.90	32.00	30.00	46.10
VAE∗	avg	33.54	22.36	60.45	41.33	28.46	58.24	43.31	28.25	42.36
Ours	best-B	**38.23**	22.51	56.66	**45.05**	31.49	55.54	**47.06**	30.94	41.32
Ours	best-M	37.14	**24.09**	61.43	45.05	**31.49**	55.54	45.51	**34.31**	40.49
Ours	best-T	36.19	22.22	**50.50**	44.41	27.73	**46.41**	42.30	29.60	**39.47**

Table 3: Test accuracy on Quora dataset, in percentage. VAE∗ is our implementation of VAE. "best-B", "best-M" and "best-T" represent the scores with the best BLEU, METEOR and TER respectively. "avg" denotes an average over "best-B", "best-M", and "best-T". All other results are directly cited from the respective papers, and "-" means no such results reported. For the arrows ↑ of BLEU and TER, a higher score is better. ↓ of TER represents a lower score is better. The best results are in boldface.

vious VAE-SVG in (Gupta et al., 2018) by using our proposed two-path loss. We observe that our VAE∗ usually outperforms previous VAE-based models. It demonstrates the proposed two-path reconstruction loss can improve the quality of generated paraphrases. Moreover, our proposed method via adversary training, denoted as "Ours", also has superior performance than VAE∗. Therefore, our proposed adversarial training and two-path loss take an apparently positive effect on alleviating the exposure bias problem and generating diverse but realistic predictions.

4.4 Human Evaluation

Data Preparation. The accurate evaluation of paraphrases is an open problem. We believe that automatic evaluation is not enough for evaluating paraphrases from a fine-grained perspective, in terms of three attributes: grammar correctness (plausibility), equivalence to the original sentence (equivalence), diversity expression (diversity). For both MSCOCO and Quora datasets, we randomly sample 100 sentence pairs (original sentence, paraphrased sentence) from the test corpus, and apply the generative architectures, including both the VAE∗ model and our GAP model, to generating paraphrases. Thus, we obtain three different sentence pairs: (original sentence, "generated" sentence) by the reference, (original sentence, generated paraphrase) by VAE∗, and by our GAP model. To make the analysis fair, we randomly shuffled all of them. We then partitioned them into ten buckets.

Process. We set up an Amazon Mechanical Turk experiment; ten human judges are asked to evaluate the quality of paraphrases. We hope our judges play similar roles to the discriminator in our model, to make a true/fake judgment. It is easier for them to make a binary choice than score how good the paraphrased sentences are from a wide score range. These ten human judges are finally involved in evaluating the total 600 sentence pairs. Each pair is judged by two different judges, and the average score is the final judgment. The agreement between judges is moderate (kappa=0.42). These ten human judges confirmed that they were proficient in English and they had understood the goals of the annotation process very well. They were trained by means of instructions and examples. If the meaning of a generated sentence contains a grammatical error or does not express the same meaning as the original sentence, or lacks diversity, we asked the annotators to score 0 for the corresponding attributes, and 1 otherwise.

Results and Analysis. We report the results in Table 4. Our model generates paraphrases with higher scores in terms of plausibility, equivalence, diversity than the previous VAE model, and their differences are statistically significant (paired t-test, p-value < 0.01). The failure cases we observed include implausible sentences, inequiva-

Model	MSCOCO			Quora		
	Plausibility	Equivalence	Diversity	Plausibility	Equivalence	Diversity
Reference	0.79	0.68	0.59	0.87	0.75	0.66
VAE$*$	0.30	0.26	0.22	0.35	0.31	0.26
Ours	0.43	0.38	0.35	0.46	0.45	0.37

Table 4: Human judgments for paraphrase generation on different models.

original (MSCOCO)	a group of kids are eating pizza and drinking soda
paraphrased	a person sits at a table eating pizza
generated-VAE	a man sitting at a table with a pizza
generated-Ours	a man sitting at a table eating a plate of pizza and soda
original (Quora-50K)	What is the fastest possible way to lose weight?
paraphrased	What are some ways to lose weight fast?
generated-VAE	How can I lose weight weight?
generate-Ours	How can I lose weight in a month?
original (Quora-100K)	What were the immediate and most important causes that led to World War 1?
paraphrased	What were the causes of World War I?
generated-VAE	What were the main causes of World War I?
generated-Ours	What were the direct and main causes of World War 1?
original (Quora-150K)	What could be the reason behind Arnab Goswami quitting Times Now?
paraphrased	Is Arnab Goswami quitting from Times now?
generated-VAE	Why did Arnab Goswami quit?
generated-Ours	Why did Arnab Goswami resign from Times Now?

Table 5: Samples of generated paraphrases from MSCOCO and Quora. Note that for the last two examples, our generated result is even better than the ground truth where our model paraphrasing "immediate" using "direct", and paraphrasing "quit" using "resign".

lent, and inaccurate expressions. It is believed that the reason for this is related to the input training data. It contains noise caused by the length limitation of $\leq$ 15 words. But note that even for the reference paraphrase, the accuracy cannot be as high as 100%. Here is an example from the real data: for the original sentence "children are playing soccer on a field with several adults observing nearby", the reference paraphrase is "soccer player hits another player in the face". Considering this, the results show that our GAP generates relatively more plausible and diverse paraphrase sentence, compared to the baseline model.

Case Study. In Table 5, we show examples sampled from both MSCOCO and Quora. We observe that our model generates paraphrases with higher diversity than VAE, with no loss of information. What's more, we discover some results where our model is even better than the ground truth, which are shown in the last two examples in the Table 5.

5 Related Work

Generative models or conditional generative models have experienced remarkable progress in the visual domain, such as VAEs in (Kingma et al., 2014; Sohn et al., 2015) and InfoGAN (Chen et al., 2016). Recent work (Larsen et al., 2015; Chen et al., 2017) also considers combining autoencoders or variational autoencoders with GAN to demonstrate superior performance on image generation. In the area of NLP, generation can be summarized from two perspectives: *text generation* with the goal of yielding diverse and plausible sentences given an original sentence, and *conditional text generation* aiming to generate new sentences conditioned on an original sentence. Considering the latter, examples include generating a new sentence similar to an original sentence (paraphrase generation), in a different style from the original sentence (style transfer), or dependent on dialogue history (dialogue generation). Attempts at using VAEs (Bowman et al., 2016; Wang et al., 2019), GANs (Yu et al., 2017; Zhang et al., 2016), and both (Hu et al., 2017) have been made to address generic text generation. However, all of them are not suitable for conditional text generation. Recent work in (Gupta et al., 2018) tries to handle paraphrase generation using VAEs. However, it suffers from challenges common to VAEs. Our method addresses these challenges with the help of GAN. To the best of our knowledge, this is the first work on using GAN with VAEs for paraphrase generation task.

6 Conclusions

We propose the first deep conditional generative architecture for generating paraphrases via adversarial training, in the hope of combining advantages of CVAE to generate similar distributions with the advantage of GAN to generate plausible sentences. Experimental results evaluated on automatic metrics demonstrate the advantages of our model, with human evaluations also verifying effectiveness. In future work, we intend to accelerate the training of our encoders and decoders with the techniques in (Huo et al., 2018) and apply our architecture and training techniques to other NLP tasks. Overall, we believe that our research makes an important step for using generative models in NLP, especially in conditional text generation.

Acknowledgments

This research was supported in part by DARPA, DOE, NIH, ONR and NSF.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1415–1425.

Igor A Bolshakov and Alexander Gelbukh. 2004. Synonymous paraphrasing using wordnet and internet. In *International Conference on Application of Natural Language to Information Systems*, pages 312–323. Springer.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. *CoNLL 2016*, page 10.

Ziqiang Cao, Chuwei Luo, Wenjie Li, and Sujian Li. 2017. Joint copying and restricted generation for paraphrase. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Liqun Chen, Shuyang Dai, Yunchen Pu, Chunyuan Li, Qinliang Su, and Lawrence Carin. 2017. Symmetric variational autoencoder and connections to adversarial learning. *arXiv preprint arXiv:1709.01846*.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180.

Yong Cheng, Yang Liu, Qian Yang, Maosong Sun, and Wei Xu. 2016. Neural machine translation with pivot languages. *arXiv preprint arXiv:1611.04928*.

Pablo Duboue and Jennifer Chu-Carroll. 2006. Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*.

Michael Ellsworth and Adam Janin. 2007. Mutaphrase: Paraphrasing with framenet. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 143–150. Association for Computational Linguistics.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1156–1165. ACM.

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

Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 905–912. Association for Computational Linguistics.

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

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Shaohan Huang, Yu Wu, Furu Wei, and Ming Zhou. 2018. Dictionary-guided editing networks for paraphrase generation. *arXiv preprint arXiv:1806.08077*.

Zhouyuan Huo, Bin Gu, Heng Huang, et al. 2018. Decoupled parallel backpropagation with convergence guarantee. In *International Conference on Machine Learning*, pages 2103–2111.

Ferenc Huszár. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *stat*, 1050:16.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589.

Raymond Kozlowski, Kathleen F McCoy, and K Vijay-Shanker. 2003. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 1–8. Association for Computational Linguistics.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2015. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231. Association for Computational Linguistics.

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

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2017b. Paraphrase generation with deep reinforcement learning. *arXiv preprint arXiv:1711.00279*.

Shuming Ma, Xu Sun, Wei Li, Sujian Li, Wenjie Li, and Xuancheng Ren. 2018. Query and output: Generating words by querying distributed word representations for paraphrase generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 196–206.

Nitin Madnani and Bonnie J Dorr. 2010. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

Kathleen R McKeown. 1983. Paraphrasing questions using given and new information. *Computational Linguistics*, 9(1):1–10.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Rebecca J Passonneau, Ananya Poddar, Gaurav Gite, Alisa Krivokapic, Qian Yang, and Dolores Perin. 2018. Wise crowd content assessment and educational rubrics. *International Journal of Artificial Intelligence in Education*, 28(1):29–55.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Aaditya Prakash, Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual lstm networks. *arXiv preprint arXiv:1610.03098*.

Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Su Wang, Rahul Gupta, Nancy Chang, and Jason Baldridge. 2018. A task in a suit and a tie: paraphrase generation with semantic augmentation. *arXiv preprint arXiv:1811.00119*.

Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Topic-guided variational autoencoders for text generation. *arXiv preprint arXiv:1903.07137*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2010. Paraphrase generation as monolingual translation: Data and evaluation. In *Proceedings of the 6th International Natural Language Generation Conference*, pages 203–207. Association for Computational Linguistics.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. 2016. Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 516–525.

Qian Yang, Gerard de Melo, Yong Cheng, and Sen Wang. 2017. Hitext: text reading with dynamic salience marking. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 311–319. International World Wide Web Conferences Steering Committee.

Qian Yang, Rebecca J Passonneau, and Gerard De Melo. 2016. Peak: Pyramid evaluation via automated knowledge extraction. In *Thirtieth AAAI Conference on Artificial Intelligence*.

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

Yizhe Zhang, Zhe Gan, and Lawrence Carin. 2016. Generating text via adversarial training. In *NIPS workshop on Adversarial Training*, volume 21.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. 2017. Adversarial feature matching for text generation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 4006–4015. JMLR. org.

Shiqi Zhao, Xiang Lan, Ting Liu, and Sheng Li. 2009. Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 834–842. Association for Computational Linguistics.

Shiqi Zhao, Cheng Niu, Ming Zhou, Ting Liu, and Sheng Li. 2008. Combining multiple resources to improve smt-based paraphrasing model. *Proceedings of ACL-08: HLT*, pages 1021–1029.