RephGAN: Generative Approach for Text Style Transfer

Wangyu Choi, Jongwon Yoon*

Hanyang Univ.

{wangyu92, jongwon}@hanyang.ac.kr

문체 변환을 위한 생성적 접근방법

최완규, 윤종원*
한양대학교

Abstract

With the proliferation of deep learning technologies, natural language generation has experienced considerable performance improvements. However, unlike human text writing, machine text generation has limitations in adapting the style of the generated text to the purpose (e.g., machine translation). In this paper, we introduce RephGAN, a generative text style transfer framework. RephGAN transforms the given text into the other style by learning unpaired, different-style text datasets. We evaluate RephGAN on the YELP dataset by performing a style transfer in both directions between positive and negative reviews. As a result, RephGAN shows that it rephrases in different styles while maintaining the structure of the sentence.

Ⅰ. Introduction

Natural language processing (NLP), such as machine translation, has made tremendous strides in recent years as a result of the availability of enormous datasets and sophisticated computing resources. However, it still has limitations in comparison to human text writing, particularly a lack of style flexibility. In other words, a person adapts their writing style to the context of the text, whereas machine learning models do not.

On the one hand, generative adversarial networks (GANs) have proven their effectiveness in the computer vision field but are still lacking in exploration in the field of natural language processing. Unlike supervised learning, which requires labels (i.e., discriminative approach), GANs that learn the distribution of datasets themselves may be more promising for generating human-friendly, high-dimensional data (e.g., image and natural language).

In this paper, we demonstrate the feasibility for generating natural language using generative adversarial methodologies. For this, we propose RephGAN (RephraseGAN), a generative text style transfer method based on a generative adversarial neural networks (GANs). ReghGAN converts one text style to another and vice versa (i.e., cycle manner). We evaluate RephGAN on YELP reviews [1] dataset. We divide the sentences into positive and negative groups and train the RephGAN to allow text to move between the two groups. As a result, RephGAN successfully transfers between positive and negative while keeping sentence structure.

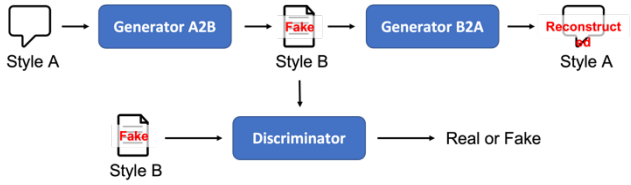

Figure 1: Workflow of RephGAN

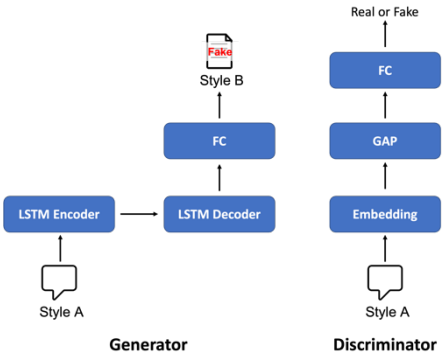

Figure 2: Network architecture

Ⅱ. Method

Overview. RephGAN seeks to change a given text of style A into a text of style B. For this goal, our model includes two mapping functions:

$$G_{A2B}: A \rightarrow B, G_{B2A}: B \rightarrow A$$

In addition, we consider two adversarial discriminators D_A and D_B that distinguish whether a given text is real or fake. The training process is based on CycleGAN [2]. G_{a2b} transforms the text in style A into style B at the

end of training, while G_{B2A} converts the text in style B into style A. Figure 1 shows the workflow of RephGAN.

Network architecture. The generator takes a sequence of words (i.e., sentence) as input and outputs a sequence of words. It is a sequence-to-sequence architecture with its natural language processing efficiency. Specifically, it has two RNNs for an encoder and a decoder, and following the decoder, it generates final output words through a fully connected layer. The discriminator takes a sequence of words and outputs 0 or 1 (i.e., fake or real). It is based on a simple text classifier using global average pooling layer. It consists of embedding layer, global average pooling layer, and fully connected layer. The network architectures of the generator and the discriminator are illustrated in Figure 2.

Training. Our training process consists of two phases: (i) pretraining generator and (ii) CycleGAN training. In the pretraining generator phase, we initialize the parameters of the generator networks to ensure stable GAN training. Then, we train all our networks with generators and discriminators communicating in a cycle manner. To pretrain the generators, we employ identity loss. When G_{A2B} gets a sentence S_B of style B, it preserves the style and outputs S_B as well (and vice versa). The identity loss is as follows:

$$L_{Idt}(G_{A2B}, G_{B2A}) = \| G_{A2B}(S_B) - S_B \|_1 + \| G_{B2A}(S_A) - S_A \|_1 \quad (1)$$

The CycleGAN training phase has three objectives: (i) adversarial loss, (ii) cycle consistency loss, and (iii) identity loss. The identity loss is as in Equation (1), and the adversarial loss L_{Adv} and cycle consistency loss L_{cyc} are as follows:

$$L_{Adv}(G, D, S, \hat{S}) = \log D(\hat{S}) + \log\ (1 - \mathrm{D}(\mathrm{G}(\mathrm{S})) \quad (2)$$

$$L_{cyc}(G_{A2B}, G_{B2A}) = G_{B2A}\big(G_{A2B}(S_A)\big) + G_{A2B}\big(G_{B2A}(S_B)\big) \quad (3)$$

where, $\hat{S}$ is a fake sentence. The full objective of RephGAN is:

$$L(G_{A2B}, G_{B2A}, D_A, D_B) = L_{Adv}\big(G_{A2B}, D_B, S_A, \hat{S}_B\big) + L_{Adv}\big(G_{B2A}, D_A, S_B, \hat{S}_A\big) + L_{cyc}(G_{A2B}, G_{B2A}) + L_{Idt}(G_{A2B}, G_{B2A}) \quad (4)$$

Implementation. We implement RephGAN using PyTorch and use the Adam [3] optimizer for updates with a learning rate of 2E-4. During the pre-training generator phase and the CycleGAN training phase, we train for 15 and 50 epochs, respectively. We set the embedding size to 512 and the hidden dimension of the LSTM to 512 with two layers. In addition, for further performance improvement, we apply a teacher force [4] mechanism to the generator in the pretraining generator phase.

Ⅲ. Evaluation

Dataset. We evaluate RephGAN on YELP reviews dataset. It contains 5.2M reviews, and each review includes review id, user id, stars (i.e., rating), and text. We focus on the rating and text. Specifically, we extract all sentences in the dataset. Then, we divide into two groups: a positive one (i.e., rating of 5) and a negative one (i.e., rating of 1). Finally, we assign the positive group to text style A and the negative group to style B.

Table 1: The result of transferring text style A to B.

#	Real A (positive)	Fake B (negative)
1	this place is awesome.	this place is terrible.
2	it was very delicious.	it was very bland.
3	will definitely be returning.	will not be returning.
4	excellent food!	terrible food!
5	everyone here is fabulous.	everyone here is unacceptable.

Table 2: The result of transferring text style B to A.

#	Real B (negative)	Fake A (positive)
1	we were all disappointed.	we were all impressed.
2	the food was bad.	the food was good.
3	our food was overcooked.	our food was delish.
4	i was so disappointed!	i was so pleased!
5	never will be back.	definitely will be back.

Result. The results of RephGAN transferring style A to style B and style B to style A are shown in Tables 1 and 2. We randomly select five sentences from the validation split dataset and conduct text style transfer. RephGAN rephrases between positive and negative by substituting essential words while maintaining the sentence structure.

Ⅳ. Conclusion

In this paper, we introduce RephGAN, a text style transfer framework for machine text generation. RephGAN learns from an unpaired dataset and maps two styles of text with distinctive features to each other. Consequently, RephGAN successfully transforms the positive and negative sentences into another style in the YELP reviews dataset.

ACKNOWLEDGMENT

This work was supported by Basic Science Research Program through the National Research Foundation of South Korea (NRF) funded by the Ministry of Education NRF-2022R1A2C1008743.

REFERENCES

[1] N. Asghar. "Yelp Dataset Challenge: Review Rating Prediction," arXiv preprint arXiv:1605.05362, 2016.

[2] J. Zhu, T. Park, P. Isola and A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," in IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2223-2232.

[3] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in 3rd International Conference on Learning Representations, 2015.

[4] R. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," Neural Computation, vol. 1, no. 2, pp. 270-280, 1989.