Text Style Transfer: Leveraging a Style Classifier on Entangled Latent Representations

Xiaoyan Li
Department of Computer Science
University of Toronto
Toronto, Canada
xiaoy.li@mail.utoronto.ca

Sun Sun **Yunli Wang**
Digital Technologies Research Centre
National Research Council Canada
Waterloo, Ottawa, Canada
Sun.Sun,Yunli.Wang@nrc.gc.ca

Abstract

Learning a good latent representation is essential for text style transfer, which generates a new sentence by changing the attributes of a given sentence while preserving its content. Most previous work adopt disentangled latent representation learning to realize style transfer. We propose a novel text style transfer algorithm with entangled latent representation, and introduce a style classifier that can regulate the latent structure and transfer style. Moreover, our algorithm for style transfer applies to both single-attribute and multi-attribute transfer. Extensive experimental results show that our method generally outperforms state-of-the-art approaches.

1 Introduction

Text generation, which leverages knowledge in computational linguistics and artificial intelligence for automatically generating natural language texts, is the core problem for a number of Natural Language Processing (NLP) applications such as speech to text, conversational/dialogue system (Banchs and Li, 2012; Kim et al., 2007), and text summarization (Ozsoy et al., 2011; Liu et al., 2018). Text style transfer can be thought of as a controllable text generation task, which aims to restyle a given sentence by changing specific attributes (sentiment, tense, formality, or politeness) while preserving the remaining attributes and the content. Successful applications of text style transfer include paraphrasing (Han et al., 2017), formality transfer (Rao and Tetreault, 2018), and text simplification (Cao et al., 2020).

A good latent representation is essential to the performance of text style transfer. Regarding the structure of the latent representation, the current work for text style transfer can be generally categorized into the disentangled representation and the entangled representation. In particular, the former method aims to learn disentangled latent representations by separating the style information from the content, while the latter method learns latent representations that entangle the style with the content. Disentangled representations are often interpretable and consequently most of the current work adopts this method (Hu et al., 2017; Yang et al., 2018; Zhao et al., 2018; John et al., 2019; Bao et al., 2019). However, learning disentangled representations is often challenging; and multiple attribute-specific decoders are commonly required for text generation, which is undesirable especially when transferring multiple attributes. The entangled representations, on the other hand, has been shown to achieve promising performance on the content preservation and to produce fluent sentences with a much less complicated architecture (Lample et al., 2019; Wang et al., 2019; Liu et al., 2020).

Although existing models achieve adequate performance on text style transfer, most of them are designed specifically for style transfer (Hu et al., 2017; Shen et al., 2017; Yang et al., 2018; Lample et al., 2019; John et al., 2019; Bao et al., 2019; Wang et al., 2019), and meanwhile lack of explicit modeling of the latent space. We argue that the quality of latent representations is crucial for text generation. In this study, we focus on building a generative model that supports both text style transfer and text generation with regularized entangled latent representations.

Our contributions can be summarized as follows: (1) We extend the framework of adversarial autoencoder by including a classifier for both the regularization of the latent space and text style transfer. We show that the classifier can divide sentences with different attributes into different regions in the latent space and thus greatly improve the performance of style transfer. (2) We provide algorithms for both single-attribute and multi-attribute style transfer. We empirically compare with sev-

Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), pages 72–82
Bangkok, Thailand (Online), August 6, 2021.

eral state-of-the-art baselines and show that our proposed method achieves promising results.

2 Related Work

In this section, we first introduce several Generative Adversarial Network (GAN)-regularized autoencoders, which have been successfully used for text manipulation. Then we review some recent methods for text style transfer focusing on the latent representation.

2.1 Probabilistic Generative Autoencoders

GANs (Goodfellow et al., 2014) are popular generative models consisting of two basic components: a generator for generating new samples and a discriminator for distinguishing real samples from generated samples. Makhzani et al. (2015) introduce adversarial autoencoders (AAEs), which turn basic autoencoders into probabilistic models. The encoder $\mathcal{E}_\phi$ maps the input $\boldsymbol{x}$ to a latent representation $\boldsymbol{z}$, $\boldsymbol{z} = \mathcal{E}_\phi(\boldsymbol{x})$. The decoder $\mathcal{D}_\theta$ reconstructs the input from $\boldsymbol{z}$ as $\hat{\boldsymbol{x}} = \mathcal{D}_\theta(\boldsymbol{z})$. The discriminator $\mathcal{D}_w$ is introduced to distinguish between $\boldsymbol{z}$ and samples from a prior distribution $\mathrm{P}_{\boldsymbol{z}}$. The objective of AAEs is formulated below:

$$\min_{\phi,\theta} \max_{w} \mathcal{L}_{rec}(\phi,\theta) - \lambda \mathcal{L}_{adv}(\phi, w),$$
$$\mathcal{L}_{rec}(\phi,\theta) = \mathrm{E}_{\boldsymbol{x}\sim \mathrm{P}_{\boldsymbol{x}}}[-\log p_\theta(\boldsymbol{x}|\mathcal{E}_\phi(\boldsymbol{x}))],$$
$$\mathcal{L}_{adv}(\phi, w) = \mathrm{E}_{\boldsymbol{z}\sim \mathrm{P}_{\boldsymbol{z}}}[-\log \mathcal{D}_w(\boldsymbol{z})] + \mathrm{E}_{\boldsymbol{x}\sim \mathrm{P}_{\boldsymbol{x}}}[-\log(1 - \mathcal{D}_w(\mathcal{E}_\phi(\boldsymbol{x})))].$$

Shen et al. (2020) adopt AAEs and introduce denoising adversarial autoencoders (DAAEs) with a smoother structure of latent space. Based on the Wasserstein autoencoders (WAEs) (Tolstikhin et al., 2018; Zhao et al., 2018) propose adversarially regularized autoencoders (ARAE) by extending AAEs for discrete sequences. Unlike AAEs which use a fixed prior distribution, Zhao et al. (2018) adopt a learnable prior parameterized by the generator of a GAN . Particularly, the discriminator and the generator are first learned by using the latent representation from the encoder. The discriminator is then used to adversarially train the encoder by minimizing the discrepancy between the posterior and the prior.

2.2 Methods for Text Style Transfer

Disentangled latent representation Most work on text style transfer is based on learning disentangled latent representations (Hu et al., 2017; Shen et al., 2017; Yang et al., 2018; John et al., 2019; Bao et al., 2019), where the attributes are separated from the content. For example, to generate a sentence with desired attributes, the decoder in (Hu et al., 2017) takes the style-independent latent representation and the desired style as the input. Shen et al. (2017) adopts adversarial training by using a binary CNN-based discriminator to determine whether a generated sentence is successfully transferred or not. Yang et al. (2018) use a target domain language model instead of a conventional binary classifier as the discriminator.

Entangled latent representation On the contrary, some recent work proposes to learn latent representations that entangle the style with the content. Although the learning of disentangled latent representations is unnecessary, other mechanisms are needed to guide the style transfer. For example, Lample et al. (2019) apply the back-translation mechanism (referred to as BTDAE) and the algorithm achieves the state-of-the-art performance on the content preservation. Wang et al. (2019) transfer text style by updating the latent representation (referred to as TAE) based on the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015). Liu et al. (2020) also use a gradient-based optimization to update the latent representation.

Style classifier A classifier is commonly used in a style discriminator to enforce the desired style (Hu et al., 2017; Yang et al., 2018; Tian et al., 2018). For example, ARAE (Zhao et al., 2018) imposes a style classifier on the latent representation of a sentence to ensure the transferred sentence containing the target attribute.

The main differences between our method and ARAE (Zhao et al., 2018) can be summarized as follows: 1) Representation structure: the latent representation of ARAE can be considered as disentangled while ours is entangled. 2) Style classifier: the style classifier we adopt helps the clustering of latent representations based on attributes while the classifier in ARAE only enforces the target contribute on transferred sentences. 3) To realize style transfer, ARAE uses the classifier to train the encoder net adversarially, such that the latent representation of the given text could contain the information of the target attribute. However, in our method, the style transfer is realized by directly modifying the latent representation. 4) Due to the adversarial training process ARAE requires multiple decoders to perform style transfer, while our

method can use only one decoder.

3 Our Method

We first briefly explain text style transfer as follows. Generally, a source or an input sentence includes both the content and the attribute. In Figure 1, we use (x, y) to include both the input x and the attribute y. As a concrete example, a sentence "this place is a great place to live !" has two types of attribute: the positive sentiment and the present tense. The positive sentiment is reflected by the word "great" and the present tense is reflected by the word "is". All remaining words in this sentence are considered as the content. For single-attribute style transfer, only one attribute (e.g. sentiment) will be transferred, and the other attributes and the content will be kept the same. In the above example with the sentiment style transfer, the sentence will be converted into a negative sentence: "this place is a terrible place to live !" by flipping the sentiment label y from positive to negative. In contrast, for multi-attribute style transfer, two or more attributes will be transferred simultaneously while the rest will be preserved. Again, in the above example the sentence will be converted into "this place was a terrible place to live !" by transferring both the sentiment and the tense.

3.1 Network Architecture

We propose a generative model that can be used for both text style transfer and text generation. The network architecture of our method is illustrated in Figure 1, where the top one is for training and the bottom one is for style transfer. The network for training includes three parts: an autoencoder, a GAN, and a style classifier. Specifically, the autoencoder is learned to reconstruct the input sentences. The discriminator of the GAN is to distinguish the aggregated posterior of the encoder from the prior, which is modeled by the generator of the GAN. The style classifier uses the latent representation as the input, and classifies latent representations based on their attributes. The classifier can also be used in style transfer, which will be explained later.

3.2 Objective Function for Training

The overall objective function for training includes three parts: the reconstruction loss, the adversarial loss induced by the GAN, and the classification loss. Similar to ARAE (Zhao et al., 2018), we use the Wasserstein distance to measure the discrepancy between two distributions. Denote the parameters of the encoder, the decoder, the discriminator, the generator, and the classifier as ϕ, θ, w, ψ, and ϕ_c, respectively. The overall objective function is defined as follows.

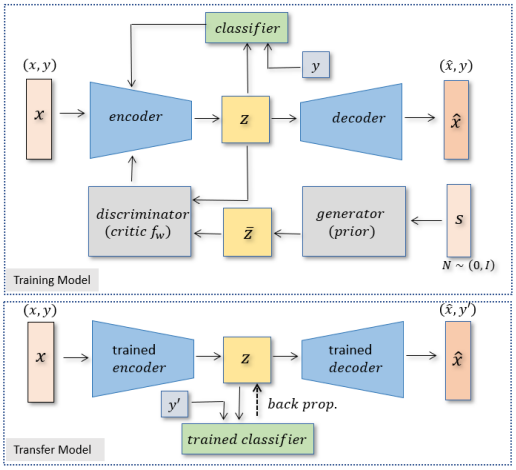


Figure 1: Network architecture: training (top) and text style transfer (bottom).

$$\mathcal{L}(\phi, \theta, \phi_c) = \mathcal{L}_{rec}(\phi, \theta) + \lambda_w \mathcal{L}_{crit}(\phi) + \lambda_c \mathcal{L}_{clas}(\phi, \phi_c),$$

where

$$\mathcal{L}_{rec}(\phi, \theta) = \mathrm{E}_{\boldsymbol{x} \sim \mathrm{P}_{\boldsymbol{x}}}[-\log p_\theta(\boldsymbol{x}|\mathcal{E}_\phi(\boldsymbol{x}))],$$
$$\mathcal{L}_{crit}(\phi) = \mathrm{E}_{\tilde{\boldsymbol{z}} \sim \mathrm{P}_{\boldsymbol{z}}}[f_w(\tilde{\boldsymbol{z}})] - \mathrm{E}_{\boldsymbol{x} \sim \mathrm{P}_{\boldsymbol{x}}}[f_w(\mathcal{E}_\phi(\boldsymbol{x}))],$$
$$\mathcal{L}_{clas}(\phi, \phi_c) = \mathrm{E}_{\boldsymbol{x} \sim \mathrm{P}_{\boldsymbol{x}}}[-\log p_{\phi_c}(y|\mathcal{E}_\phi(\boldsymbol{x}))].$$

In the above expressions, the variable $\tilde{\boldsymbol{z}}$ is the output of the generator $\mathcal{G}_\psi(\boldsymbol{s})$, where the noise $\boldsymbol{s} \in \mathcal{N}(0, \boldsymbol{I})$; y is the label of the source attribute; and the critic function f_w of the discriminator is obtained by a min-max optimization:

$$\min_\psi \max_w \mathcal{L}_{crit}(\psi, w) = \mathrm{E}_{\boldsymbol{x} \sim \mathrm{P}_{\boldsymbol{x}}}[f_w(\mathcal{E}_\phi(\boldsymbol{x}))] - \mathrm{E}_{\tilde{\boldsymbol{z}} \sim \mathrm{P}_{\boldsymbol{z}}}[f_w(\tilde{\boldsymbol{z}})].$$

We summarize the training algorithm in Algorithm 1. First, the autoencoder is trained by minimizing the reconstruction loss, *i.e.*, $\min_{\phi,\theta} \mathcal{L}_{rec}(\phi, \theta)$. Next, based on the latent representation from the encoder, the encoder and the style classifier are jointly trained by minimizing the classification loss, *i.e.*, $\min_{\phi,\phi_c} \mathcal{L}_{clas}(\phi, \phi_c)$. Meanwhile, the critic function f_w and the generator of the GAN are learned via the min-max optimization $\min_\psi \max_w \mathcal{L}_{crit}(\psi, w)$. Finally, the critic function f_w is utilized to adversarially train the encoder, *i.e.*, $\min_\phi \mathcal{L}_{crit}(\phi)$. We

emphasize that we do not explicitly disentangle the attributes from the content in the latent representation. Therefore, to implement style transfer, the style classifier is crucial, which guides the clustering of the entangled latent representations based on their attributes.

Algorithm 1: Training Algorithm.

Inputs: $\mathrm{P}_{\boldsymbol{x}}$ input distribution; $\mathcal{E}_\phi$ encoder; $\mathcal{G}_\psi$ generator; f_w discriminator/critic function

for *each training iteration* **do**

// Train the encoder and decoder for reconstruction(ϕ, θ).

Sample $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim \mathrm{P}_{\boldsymbol{x}}$ and compute $\boldsymbol{z}^{(i)} = \mathcal{E}_\phi(\boldsymbol{x}^{(i)})$;

Backprop loss: $\mathcal{L}_{rec}(\phi, \theta) = -\frac{1}{m}\sum_{i=1}^m \log p_\theta(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i)})$;

// Train the attribute classifier(ϕ_c) and optimize the encoder using the classifier regularisation(ϕ).

Sample $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim \mathrm{P}_{\boldsymbol{x}}$ and save attribute $y^{(i)}$;

Backprop loss: $\mathcal{L}_{clas}(\phi_c, \phi) = -\frac{1}{m}\sum_{i=1}^m \log p_{\phi_c}(y^{(i)}|\mathcal{E}_\phi(\boldsymbol{x}^{(i)}))$;

// Train the discriminator/critic function(w).

Sample $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim \mathrm{P}_{\boldsymbol{x}}$ and $\{\boldsymbol{s}^{(i)}\}_{i=1}^m \sim \mathcal{N}(0, \boldsymbol{I})$;

Compute $\boldsymbol{z}^{(i)} = \mathcal{E}_\phi(\boldsymbol{x}^{(i)})$;

Backprop loss: $\min_\psi \max_w \mathcal{L}_{crit}(w, \psi) = \frac{1}{m}\sum_{i=1}^m f_w(\boldsymbol{z}^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(\mathcal{G}_\psi(\boldsymbol{s}^{(i)}))$;

// Train the encoder adversarially(ϕ).

Sample $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim \mathrm{P}_{\boldsymbol{x}}$;

compute $\tilde{\boldsymbol{z}}^{(i)} = \mathcal{G}_\psi(\boldsymbol{s}^{(i)})$;

Backprop loss: $\mathcal{L}_{crit}(\phi) = \frac{1}{m}\sum_{i=1}^m f_w(\mathcal{E}_\phi(\boldsymbol{x}^{(i)})) - \frac{1}{m}\sum_{i=1}^m f_w(\tilde{\boldsymbol{z}}^{(i)})$;

end

3.3 Style Transfer

After training the network, we can implement text style transfer (as shown at the bottom of Figure 1). We summarize the algorithm for style transfer in Algorithm 2, which works for both single-attribute and multi-attribute style transfer. Normally for multi-attribute style transfer, multiple style classifiers are required: each corresponding to an attribute. To make the network scalable, we instead use a single style classifier by combining the labels of attributes. In this case, each attribute label corresponds to an attribute-combination (such as present-positive, past-positive, present-negative, and past-negative with both the tense and the sentiment as the attributes for transferring).

Algorithm 2: Transfer Algorithm.

Inputs: input distribution $\mathrm{P}_{\boldsymbol{x}}$; encoder $\mathcal{E}_\phi$; well-trained classifier $\mathcal{C}_{\phi_c}$; the initial weights $\boldsymbol{w} = \{w_j\}$; decay coefficient λ; target attribute y'; threshold t; maximal iterations I; attribute vector $\boldsymbol{v}$; the weight of attribute vector k.

Result: A target latent representation $\hat{\boldsymbol{z}_{\boldsymbol{f}}}^{(i)}$ or $\hat{\boldsymbol{z}_{\boldsymbol{v}}}^{(i)}$.

Sample $\{\boldsymbol{x}^{(i)}\}_{i=1}^m \sim \mathrm{P}_{\boldsymbol{x}}$ and compute $\boldsymbol{z}^{(i)} = \mathcal{E}_\phi(\boldsymbol{x}^{(i)})$;

// Method 1: based on Fast Gradient Sign Method.

for *each* $w_j \in \boldsymbol{w}$ **do**

$\hat{\boldsymbol{z}}^{(i)} = \boldsymbol{z}^{(i)} - w \bigtriangledown_z * \mathcal{L}_{clas}(C_{\phi_c}(\boldsymbol{z}^{(i)}), y'^{(i)})$;

for $|\mathcal{C}_{\phi_c}(\boldsymbol{z}^{(i)}) - y'^{(i)}| > t$ **do**

$it++$;

$w_j = \lambda w_j$;

$\hat{\boldsymbol{z}_{\boldsymbol{f}}}^{(i)} = \boldsymbol{z}^{(i)} - w \bigtriangledown_z * \mathcal{L}_{clas}(\mathcal{C}_{\phi_c}(\boldsymbol{z}^{(i)}), y'^{(i)})$;

if $it > I$ **then**

break ;

end

end

end

// Method 2: based on vector arithmetic. $\boldsymbol{x}_s^{(i)}$ are the samples with source attribute and $\boldsymbol{x}_t^{(i)}$ are the samples with target attribute.

Sample $\{\boldsymbol{x}_s^{(i)}\}_{i=1}^n \sim \mathrm{P}_{\boldsymbol{x}}$ and $\{\boldsymbol{x}_t^{(i)}\}_{i=1}^n \sim \mathrm{P}_{\boldsymbol{x}}$;

compute $\boldsymbol{z}_s^{(i)} = \mathcal{E}_\phi(\boldsymbol{x}_s^{(i)})$ and compute $\boldsymbol{z}_t^{(i)} = \mathcal{E}_\phi(\boldsymbol{x}_t^{(i)})$;

Calculate the attribute vector $\boldsymbol{v} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_s^{(i)} - \frac{1}{n}\sum_{i=1}^n \boldsymbol{z}_t^{(i)}$;

$\hat{\boldsymbol{z}}_v^{(i)} = \boldsymbol{z}^{(i)} \pm k * \boldsymbol{v}$

To perform style transfer, given an input sentence, we first get its latent representation as the output of the encoder. With the entangled latent representation, the key of style transfer is how to update the latent representation of the source sentence. To achieve that, we adopt two different but commonly used updates: the fast gradient based and the vector arithmetic based. To obtain the target sentence with the desired attribute, we then feed the updated latent representation to the decoder.

Fast gradient based: FGSM is employed by Wang et al. (2019) to update the latent representation for style transfer. Concretely, the latent representation is updated along the gradient of the classification loss with the step size w. A set $\boldsymbol{w}$ contains a few step sizes with an increasing order. We sequentially test these step sizes until obtaining

the desired latent representation. This is to maximally preserve the content of the sentence and also to prevent the modification of the latent presentation from falling into a local optimum. In each iteration, the updated latent representation $\hat{\boldsymbol{z}}_f$ is given as follows:

$$\hat{\boldsymbol{z}}_f = \boldsymbol{z} - w \bigtriangledown_z * \mathcal{L}_{clas}(\mathcal{C}_{\phi_c}(\boldsymbol{z}), y'),$$

where $\mathcal{L}_{clas}$ represents a style classifier loss, $\mathcal{C}_{\phi_c}$ is a well-trained classifier, and y' represents the target label. The detailed algorithm is displayed in Method 1 of Algorithm 2.

Vector arithmetic based: In several studies *e.g.*, Zhao et al. (2018); Shen et al. (2020), the latent vector arithmetic based method is employed in text style transfer or text interpolation. Specifically, the latent representation $\boldsymbol{z}$ of the source sentence is modified by an attribute vector "$\boldsymbol{v}$". For example, assume that the source attribute is positive. When transferring the attribute from positive to negative we can update $\boldsymbol{z}$ by $\boldsymbol{z} - \boldsymbol{v}$; and when transferring from negative to positive we can update $\boldsymbol{z}$ by $\boldsymbol{z} + \boldsymbol{v}$. The same as Shen et al. (2020), the attribute vector "$\boldsymbol{v}$" uses the mean of the latent representations of 100 samples with the source attribute and 100 samples with the target attribute from the validation set. For multi-attribute transfer, the attribute vector v is computed in the same way. The only difference is that the label of the source attribute and the target attribute corresponds to an attribute-combination as explained before. The updated latent representation $\hat{\boldsymbol{z}}_v$ can be formulated as follows:

$$\hat{\boldsymbol{z}}_v = \boldsymbol{z} \pm k * \boldsymbol{v},$$

where k is a hyperparameter denoting the weight associated with the attribute vector.

4 Experiments

In this section, we first visualize the latent representation of our method, and then compare our method with several baselines, namely, TAE (Wang et al., 2019), ARAE (Zhao et al., 2018), and DAAE (Shen et al., 2020) for single-attribute and multiple-attribute text style transfer. We then evaluate our model on text generation and compare it with ARAE.

4.1 Datasets

We use Yelp and Amazon datasets for evaluation.

Yelp: This dataset consists of Yelp restaurant and business reviews (Li et al., 2018), which includes $444K$ training samples, $4K$ validation samples, and $1K$ test samples.

Amazon: This dataset includes product reviews from Amazon (He and McAuley, 2016), which includes $555K$ training samples, $2K$ validation samples, and $1K$ test samples.

On both Yelp and Amazon datasets, reviews with a rating score above three are considered as positive samples, otherwise are considered as negative samples.

4.2 Experimental setups

In our experiment, similar to ARAE (Zhao et al., 2018), we use one layer LSTM with 200 hidden units for both the encoder and the decoder. Both the generator and the discriminator in the GAN use simple MLP networks. The style classifier is built by a shallow MLP network with two hidden layers, as our experiment indicates that too many layers can degrade the performance of the classifier.

The weighting parameters λ_w and λ_c are set to 0.1 on Yelp and 1 on Amazon. In the fast gradient method, the set of the initial weights $\boldsymbol{w}$ is set to $\{0.005, 0.006, 0.007, 0.008, 0.009, 0.01\}$, where the weights are ordered increasingly.

4.3 Evaluation

Following previous studies, for both automatic and human evaluations, we assess the performance of style transfer from three perspectives: transfer control, content preservation, and fluency. In automatic evaluation, three commonly used metrics are adopted: the transfer rate, the BLEU score, and the Perplexity (PPL) score.

Transfer control: It evaluates whether the style of the source sentences is correctly flipped. The transfer rate is the percentage of the corrected transferred sentences, and we use a *fastText* classifier (Joulin et al., 2017) to determine that.

Content preservation: It evaluates how the content is preserved in the transferred sentences. We use n-gram statistics (4-gram) of the BLEU score (Papineni et al., 2002) to quantify the content preservation against the references (Li et al., 2018).

Fluency: It evaluates the grammatical structure and the naturalness of the generated (or transferred) text sentences. We use a language model KenLM (Heafield, 2011) to calculate the PPL score of text sentences for evaluating fluency.

4.4 Evaluation on Latent Representation

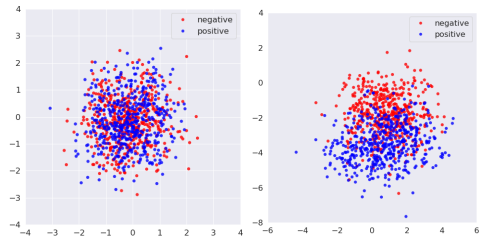
4.4.1 Visualization

We show the projected latent representation of both the source and the target sentences and compare it with TAE (Wang et al., 2019). To better show the structure of the latent representation, we use the visualization tool t-SNE (van der Maaten and Hinton, 2008) in 2-dimension. Figure 2(a) shows the latent representations of 1000 source samples with positive and negative labels. Figure 2(b) and Figure 2(c) show the latent representations of six target samples, which are updated by FGSM. $w[i]$ in these two figures denotes the i-th step size in the set $\boldsymbol{w}$, and a larger value of i indicates a larger value of $w[i]$.

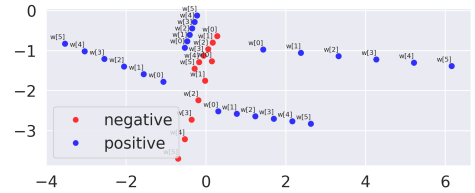
In our framework, both the GAN and the style classifier help the regularization of the latent representation. Figure 2(a) indicates that in our method, the latent representations tend to form two clusters. Specifically, the positive samples tend to locate at the bottom while the negative samples tend to locate on the top. In contrast, the positive and the negative samples in TAE are generally mixed together. In Figure 2(c), as the value of the step size w increases in our method, the latent representation of the positive samples tends to move towards the bottom, which corresponds to the position of the positive cluster. In contrast, the latent representation of the negative samples tends to move towards the top, which corresponds to the position of the negative cluster. This observation clearly shows the guidance of the style classifier on clustering the latent representations. In comparison, in Figure 2(b), without the GAN and the style classifier in TAE, the latent representation of each target sample needs to be updated along different directions.

4.4.2 Evaluation of Latent Representation via K-nearest-neighbours

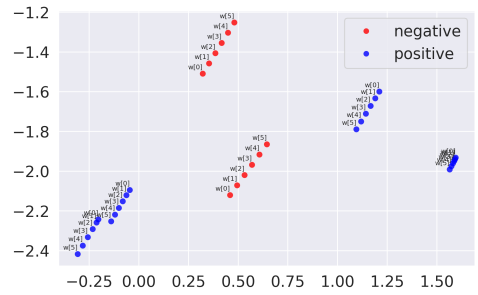
It is desirable that close latent representations lead to semantically similar sentences after feeding latent representations to the decoder. Such property indicates the smoothness of the latent space. In this experiment, we find $k = 9$ nearest neighbours of the latent representation of a sentence "*service is terrible and won't return.*", and then generate sentences by feeding these latent representations to the decoder. It is expected that the generated sentences are close to the source sentence in terms of the sentiment attribute and the content. For comparison, we consider four differ network architectures and



(a) Projected latent representations in TAE (left) and in our method (right).



(b) Projected target latent representations with FGSM update in TAE.



(c) Projected target latent representations with FGSM update in our method.

Figure 2: Comparison between our method and TAE on projected latent representations.

show the generated sentences in Table 1.

- **TAE**: The generated sentences contain both the positive and the negative samples. Moreover, many of them have different contents that are related to "place" or "location" instead of the source content "service" or "return".

- **TAE+GAN**: We regularize the latent representation in TAE by a GAN. Although all sentences are related to the source content "service" their sentiment attributes are largely different.

- **TAE + classifier**: We add a style classifier in TAE. Different from **TAE+GAN**, the generated sentences have the same negative attribute but some sentences deviate from the source content "service" or "return".

- **Our method**: Our network architecture includes both the GAN and a style classifier.

All generated sentences have the same negative attribute and are related to the source content "service" or "return".

Table 1: Evaluation of the smoothness of the latent space via k-nearest-neighbours. The source sentence is "*service is terrible and won't return* ".

TAE	TAE + GAN
service was terrible . their service is terrible . service is great and friendly . this *place* is terrible ! service is slow and horrible . this *location* is terrible . service is always good . service was bad . everything was great and i will return !	service is great and friendly . service isn't that great either . service is mediocre and slow . service is slow and horrible . service is lacking and food is mediocre . the service is friendly and fast . prices are good and the service is great . service is quick and friendly . the service is always friendly and good .
TAE + classifier guidance	**Our method**
service was terrible . their service is terrible . *food* was terrible . this *place* is terrible . service is slow and horrible . the *waiter* was terrible . i wo n't be back . service is mediocre and slow . terrible service .	service is n't that great either . i wo n't be back . the service is n't frequent enough . will not return to this place . needless to say we wo n't be back ! service was n't too bad - nice people . the service was not that professional ! the service did n't get any better . service is n't too bad , but could be better .

4.5 Evaluation of Style Transfer

4.5.1 Single-Attribute Style Transfer

We compare our method with TAE by using FGSM to update the latent representation since TAE is only designed with FGSM (Table 2). TAE has a very low transfer rate in experiments. This however leads to high BLEU scores and low PPL scores as most target sentences are the same as the source sentences. By contrast, our method can successfully transfer most sentences, and leads to decent BLEU and PPL scores.

Table 2: Comparison between our method and TAE for sentiment transfer on Yelp.

Methods	Transfer ↑	BLEU ↑	PPL ↓
TAE			
$\boldsymbol{w} = 2$	0.24	**37.98**	**42.08**
$\boldsymbol{w} = 4$	0.25	35.70	48.73
$\boldsymbol{w} = 6$	0.25	33.33	56.50
Our method:			
$\boldsymbol{w} = 0.005$	0.76	25.74	**70.16**
$\boldsymbol{w} = 0.007$	0.80	25.16	72.70
$\boldsymbol{w} = 0.01$	**0.87**	23.90	75.46

We also compare the performance of our methods with ARAE and DAAE using the vector arithmetic based update on latent representations for style transfer. As mentioned before, the vector arithmetic based update can be used to evaluate the smoothness of the latent space. In Table 3, we compare with two baselines on both Yelp and Amazon and display the results that achieve the best trade-off among the three evaluation metrics. The hyperparameter k of vector arithmetic method is chosen based on the performance in the validation set. Our method achieves the highest transfer rate and a comparable BLEU score with ARAE on Yelp, while ARAE achieves the lowest PPL score. On Amazon, our method obtains the best performance on the transfer rate and the BLEU score with a slightly higher PPL score than ARAE. By contrast, DAAE does not perform well on both datasets especially on Amazon.

Table 3: Evaluation results of style transfer based on the vector arithmetic based update on Yelp and Amazon.

	Methods	Transfer ↑	BLEU ↑	PPL ↓
Yelp	**ARAE** $\pm 1.5\boldsymbol{v}$	0.536	**20.08**	**64.75**
	DAAE $\pm 2.0\boldsymbol{v}$	0.461	18.55	114.59
	Our method $\pm 1.5\boldsymbol{v}$	**0.792**	19.90	78.63
Amazon	**ARAE** $\pm 2.5\boldsymbol{v}$	0.513	14.71	**31.37**
	DAAE $\pm 1.0\boldsymbol{v}$	0.473	3.50	–
	Our method $\pm 2.0\boldsymbol{v}$	**0.884**	**14.73**	33.86

FGSM and vector arithmetic method for style transfer have their pros and cons. Table 4 shows the evaluation results of both FGSM based and vector arithmetic based methods on Yelp data. Generally, for both methods, as the step size $\boldsymbol{w}$ or $\boldsymbol{v}$ increases, the transfer rate is improving, while the performance of BLEU and PPL are decreasing. In the case of the FGSM based method, the best trade-off is when w is set as 0.007, while the vector arithmetic based method has the best trade-off when v is set as 1.5. With $w = 0.007$ and $v = 1.5$, FGSM based method achieves better transfer rate and BLEU score but the lower performance of PPL than vector arithmetic based method. From our experiment, we also observe that FGSM based style transfer needs much longer updating time in testing than the vector arithmetic based method.

Table 4: Comparison results between FGSM based and vector arithmetic based style transfer on Yelp.

Methods	$\boldsymbol{w}/\boldsymbol{v}$	Transfer ↑	BLEU ↑	PPL ↓
FGSM based	$\boldsymbol{w} = 0.005$	0.76	25.74	70.16
	$\boldsymbol{w} = 0.007$	*0.80*	*25.16*	*72.70*
	$\boldsymbol{w} = 0.01$	0.87	23.90	75.46
Vector arithmetic based	$\pm 1.0\boldsymbol{v}$	0.49	**30.00**	**49.42**
	$\pm 1.5\boldsymbol{v}$	*0.79*	*19.90*	*78.63*
	$\pm 2.0\boldsymbol{v}$	**0.94**	11.38	113.52

Although the automatic evaluation metrics *e.g.*, the BLEU score, are widely used, they sometimes

do not well align with the human judgement (Ma et al., 2018). Therefore, to fully evaluate the performance we also carried out the human evaluation on sentiment transfer and compare it with ARAE on Yelp. We use the vector arithmetic based update of the latent representation on the first 200 positive and 200 negative sentences in the test set. Three annotators were recruited and provided scores in the range of 1∼5 regarding the transfer control, the content preservation and the fluency. The Kappa statistic of the agreement between raters in the human evaluation is 0.657. The average scores over three annotators are shown in Table 5, and our method generally outperforms ARAE.

Table 5: Human evaluation results of sentiment transfer on Yelp.

Methods	Transfer ↑ Control	Content ↑ Preservation	Fluency ↑
ARAE	3.388	2.671	3.439
Our method	**3.468**	**3.018**	**3.612**

4.5.2 Multi-Attribute Style Transfer

We also evaluate our model for multi-attribute transfer and compare with ARAE on Yelp. The goal of multi-attribute style transfer is to transform multiple attributes in a sentence at once while preserving the main content of the sentence. Using the same example sentence "this place is a great place to live !" with positive sentiment and present tense, multi-attribute transfer converts it into a sentence with the negative sentiment and the past tense "this place was a terrible place to live !".

In the training phase, the style classifier uses both the latent vector of a given sentence and the original attribute label as the inputs; while in style transfer, the style classifier uses both the latent vector and the desired attribute label as the inputs. In pre-processing of single-attribute style transfer, the attribute is labelled as either "0" or "1". In multi-attribute style transfer (e.g. tense and sentiment), each attribute combination will be defined as an individual class (e.g. present-positive: "0", past-positive: "1", present-negative: "2", and past-negative: "3").

In particular, we use the Stanford Parser to extract the main verb of a sentence from Yelp and then determine the tense of a sentence based on its part-of-speech tag (POS tags) (Klein and Manning, 2003). Table 6 shows the evaluation results of style transfer for two attributes: sentiment and tense transfer. In our model, we test on two network variants, one consisting of two style classifiers each corresponding to one attribute, and the other consists of only one style classifier that combines both attributes into one label as described before. The results show that our method is superior to ARAE even with one style classifier. When using two classifiers to realize multiple attributes transfer, for example, tense and sentiment transfer, each classifier is responsible for transferring one attribute. Specifically, after the sentiment-classifier transferred sentiment attribute, the transferred sentences will be the inputs of the tense-classifier for transferring tense attribute. As using two classifiers, each classifier only needs to transfer one attribute, having less pressure of transferring two attributes together, it achieves higher accuracy than one classifier case. However, since it requires two steps transfer, the self BLEU score decreases slightly.

Table 6: Results of multiple attributes (sentiment and tense) transfer on Yelp.

Methods	Transfer ↑	Self BLEU ↑	PPL ↓
ARAE ($\pm 2.0v$)	0.663	11.57	86.99
Our method:			
2 classifiers; $\pm 1.5v$	**0.750**	13.86	85.47
1 classifier; $\pm 1.5v$	0.733	**14.34**	**85.41**

4.6 Evaluation of Text Generation

Unlike most of the current models for style transfer, our model can also be used to generate new text sentences owing to the introduction of the latent prior distribution. To generate a new sentence, we first take the noise $\boldsymbol{s}$ as the input to the generator of the GAN and get a latent representation. Then we feed the latent representation to the decoder and obtain the new sentence.

In previous work, both LSTM (Zhao et al., 2018; Lample et al., 2019; Shen et al., 2020) and transformer (Wang et al., 2019) have been used as the base network in the encoder and decoder architecture. Hence, we further evaluate the performance of our method based on these two networks on Yelp and show the results in Table 7. Experimental results indicate that both the transformer-based networks and the LSTM-based networks in our method achieve a similar trade-off among the three evaluation metrics. The transformer-based method achieves higher BLEU scores but lower transfer rates and higher PPL scores, while the LSTM-based method leads to a better performance on

the transfer rate and the PPL score, but a worse performance on the BLEU score.

Table 7: Comparison of sentiment transfer between the transformer-based autoencoder and the LSTM-based autoencoder in our method on Yelp.

Methods	Transfer ↑	BLEU ↑	PPL ↓
Transformer-based:			
w=0.7	0.73	**31.14**	71.58
w=0.8	0.79	27.42	84.13
w=0.9	0.82	24.48	92.65
w=1.0	0.85	21.40	97.21
LSTM-based:			
w=0.005	0.76	25.74	**70.16**
w=0.007	0.80	25.16	72.70
w=0.009	0.84	24.50	74.32
w=0.01	**0.87**	23.90	75.46

We evaluate the quality of the text generation using the LSTM-based network in our method by comparing $10,000$ generated sentences with sentences generated by ARAE on Yelp. The experimental results in Table 8 show that our method achieves a better degree of fluency. In particular, the PPL score of our method is lower than that of ARAE by around 10% (the PPL score of our method and ARAE is 76.83 and 86.98, respectively).

Table 8: Generated sentences of our method and ARAE.

Our method
the woman who could give up the store says you are very picky .
the wait staff is great but overall i did n't like the customer.
i will not recommend this place to any women in future .
the man was always great and the service was really helpful .
do not waste a star from the older man this place is overpriced .
the store experience is awesome the salesman it was very nice.
oh ok and the man in the service looked nice .
kind of really nice man 's walking the restaurant that they 're very delish .
ARAE
the gentleman i left inside the kitchen was a rather nice follow up .
this woman has gotten me .
the woman in a little job of perfect !
the man was not that amazing if i tried to order it .
this woman has a cake must me in the burgh .
all was excellent by the salesman we had to do .
there is a friendly man and the crowd of bacon in your face .
their woman was under staffed as very polite and how talented .

Text samples of ARAE are from Zhao et al. (2018).

4.7 Style Classifier in Other Models

Through the above experiments, we have illustrated the effect of the style classifier in our method on clustering the latent representations based on the attributes. We also perform an ablation study regarding the style classifier on two other advanced models: DAAE and BTDAE (Lample et al., 2019). In DAAE, we implement style transfer by the vector arithmetic based update, and in BTDAE the back-translation algorithm is used for style transfer. From Table 9, we observe that with the inclusion of a style classifier in DAAE the performance on all evaluation metrics is improved. For BTDAE, with a style classifier the BLEU and the PPL scores are improved. These results again demonstrate the effectiveness of a style classifier on style transfer.

Table 9: Comparison between the models and the models with a style classifier on Yelp.

Methods	Transfer ↑	BLEU ↑	PPL ↓
DAAE ($\pm 2.0v$)	0.461	18.55	114.59
DAAE + Class.($\pm 1.5v$)	**0.646**	**22.02**	**112.50**
BTDAE	**0.87**	38.41	36.42
BTDAE + Class.	0.86	**39.87**	**34.39**

As the official code of BTDAE is unavailable, we implemented the algorithm based on the description in Lample et al. (2019).

5 Conclusion and Future Work

In this paper, we proposed a new approach for text style transfer with entangled latent representations. We added a classifier to regularize the distribution of latent sentences in a probabilistic autoencoder. Extensive experiments show that this regularized latent structure significantly improves the downstream text manipulation tasks. Compared with benchmarks our method achieves impressive results on both single-attribute and multi-attribute text style transfer. Moreover, both approaches of fast gradient and vector arithmetic style transfer outperform baselines on style transfer tasks. In addition, we demonstrated that the classifier regularization also improves other style transfer models.

In the future, we would like to explore other methods to regularize latent representation in controllable text generation. Moreover, text generation models have a wide range of applications in NLP tasks. Besides style transfer, we will apply our model to other tasks such as text simplification and examine the latent structure in these applications.

References

Rafael E. Banchs and Haizhou Li. 2012. IRIS: a chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL 2012 System Demonstrations*, pages 37–42, Jeju Island, Korea. Association for Computational Linguistics.

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou,

Olga Vechtomova, Xin-yu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019, Florence, Italy. Association for Computational Linguistics.

Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. 2020. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples.

Mengqiao Han, Ou Wu, and Zhendong Niu. 2017. Unsupervised automatic text style transfer using LSTM. In *NLPCC*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

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, International Convention Centre, Sydney, Australia. PMLR.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. *arXiv:1607.01759*.

Seokhwan Kim, Cheongjae Lee, Sangkeun Jung, and Gary Geunbae Lee. 2007. A spoken dialogue system for electronic program guide information access. In *IEEE*.

Dan Klein and Christopher D Manning. 2003. Fast exact inference with a factored model for natural language parsing. In *Advances in neural information processing systems*, pages 3–10.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Dayiheng Liu, Jie Fu, Yidan Zhang, Chris Pal, and Jiancheng Lv. 2020. Revision in continuous space: Fine-grained control of text style transfer.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2018. Generative adversarial network for abstractive text summarization.

Qingsong Ma, Ondřej Bojar, and Yvette Graham. 2018. Results of the wmt18 metrics shared task: Both characters and embeddings achieve good performance. In *Proceedings of the third conference on machine translation: shared task papers*, pages 671–688.

Laurens van der Maaten and Geoffrey Hinton. 2008. Viualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv:1511.05644*.

Makbule Ozsoy, Ferda Alpaslan, and Ilyas Cicekli. 2011. Text summarization using latent semantic analysis. *J. Information Science*, 37:405–417.

Kishore Papineni, S. Roukos, T. Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Tianxiao Shen, Jonas Mueller, Regina Barzilay, and Tommi S. Jaakkola. 2020. Educating text autoencoders: Latent representation guidance via denoising.

Youzhi Tian, Zhiting Hu, and Zhou Yu. 2018. Structured content preservation for unsupervised text style transfer. *arXiv*, arXiv:1810.06526.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. 2018. Wasserstein auto-encoders. In *ICLR 2018*.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Advances in Neural Information Processing Systems*, pages 11036–11046.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems*, pages 7287–7298.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5902–5911. PMLR.