Plug-and-play textual style transfer

Shangquan Sun
Department of Statistics
sunsean@umich.edu

Jian Zhu
Department of Linguistics
lingjzhu@umich.edu

1 Introduction

Stylistic variations are intrinsic in human language and reflects the social dynamics of language use (Bell, 1984). Generally, linguistic style is a matter of word choice, whether a speaker choose to use one linguistic elements instead of another (Coupland, 2011). The task of textual style transfer is defined as rendering a source text into another linguistic style without changing its underlying semantic content (Garbacea and Mei, 2020), such as converting a negative review to positive or turning an informal sentence into a formal one. Such a task lends itself to a variety of practical applications, including but not limited to automatic review generation and assisting systems for literary creation.

While great progress has been made towards fluent and content-preserving style control, the current studies still suffer from the following shortcomings. Since most studies focus on binary styles (positive/negative or formal/informal), achieving finer-grain style transfer remains challenging, both in terms of dataset creation and modeling. Secondly, generated texts are mostly short sentences as long as 15 to 20 words. Generating long and coherent texts with various controllable styles remains far from a solved problem.

In this work, we plan to adopt the framework of *Plug and Play Language Model* (Dathathri et al., 2020) to text style transfer. We utilized the pre-trained transformer to overcome the length limit in language generation and the attribute classifiers to control and guide the generation process.

2 Related Works

2.1 Textual style transfer

Textual style transfer can be formulated either as a supervised task when paralleled text pairs are sufficient, or as an unsupervised task when only coarse style labels are available (Garbacea and Mei, 2020). Due to the lack of parallel data, only a few studies focus on training on paralleled data (Zhang et al., 2020), while the majority of research opt for unparalleled style transfer.

The mainstream approach seeks to learning (disentangled) representations of various textual attributes in the latent space to achieve fine-grained control of text transfer (Shen et al., 2017; John et al., 2019; Lample et al., 2018; Zhou et al., 2020).

While disentanglement might not always be achievable and neither is disentanglement necessary for generative high quality outputs (Lample et al., 2018), some studies opt for a different line of text transfer research. Adversarial training has also been adopted to perform style transfer with the use of an adversarial discriminator to aid learning implicit representations of style (Fu et al., 2018; Dai et al., 2019). Moreover, reinforcement learning algorithms have also been proposed to enhance the quality of style transfer (Luo et al., 2019a,b; Wu et al., 2019), demonstrating promising results.

With recent advancement, converting the styles of short sentences is quite within the reach. However, transferring styles of long texts still remains challenging, as most approaches train the generative decoder from scratch with limited data, without utilizing the more powerful pre-trained generative models such as BERT (Devlin et al., 2019) or GPT-2 (Radford et al., 2019). Secondly, most models are not easily extensible, as extending existing models to cover more stylistic attributes may entail re-trained the whole model.

2.2 The plug-and-play approach

While the large pre-trained language models have achieved impressive performance in language generation, few textual style transfer models attempt to utilize these pre-trained models to improve language generation. One potential reason is that even fine-tuning large models can be computationally

intensive. The plug-and-play approach is the latest solution to harness the generative power of large pre-trained models with light-weight models.

Currently there are two approaches. One is to guide the large-pre-trained model to generate texts with certain attributes using a lightweight classifier. The *Plug and Play Language Models* (PPLM) (Dathathri et al., 2020), a framework for language generation that combines the powerful pre-trained language model (GPT-2) and attribute classifiers to guide the generation of language. This framework harnesses the massive knowledge learned pre-trained language models while also maintains flexibility and extensibility with the plug-in attribute classifiers, which modifies the hidden states of GPT-2 on the fly through propagating gradients. The attribute classifier can be easily re-trained without adjusting the language generator and can be extended to include more attributes. While the generation speed of PPLM is still slow, the *GeDi* model (Krause et al., 2020) improves the generation speed by incorporating the Bayes rule and contrast probabilities to empirically estimate the conditional probability of attribute classifier. It borrows the architecture of *CTRL* (Keskar et al., 2019) for the light-weight guiding language model. It further improves sequence generation performance by utilizing *Nucleus Sampling* (Holtzman et al., 2020).

Alternatively, it is also possible to perform style transfer in the latent space. Wang et al. (2019) jointly trained an autoencoder and an attribute classifier that classifies the sentiment of the encoded latent vector. Then sentiment transfer is the modification of the latent vector through the gradient propagation from the attribute classifier. *Emb2Emb* (Mai et al., 2020) reduces the sequence-to-sequence task of generating text in a specific style into a task of regressing embedding-to-embedding with only linear layers. The framework possesses the ability of preserving original attribute-independent contents by adopting a cosine similarity loss no matter input data are paralleled or not.

3 Method

Currently almost no studies have utilized pre-trained generative models to perform textual style transfer. Training from scratch often result in lower fluency as the data for style transfer are relatively small. We believe that the plug-and-play approach is a promising method to harness the pre-trained models in a computationally efficient way. We have worked on the following two models. The first model is still not strictly a plug-and-play model but we will change it in subsequent experiments.

3.1 Modifying the latent vector through gradients propagation

At first, we planed to adopt the method proposed by Wang et al. (2019). In this approach, an encoder-decoder model is trained to reconstruct the language. In this model, given a set of texts and their stylistic attributes $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, the encoder E_{θ_e} maps the input text $\mathbf{x}_i$ to a low dimensional vector $\mathbf{z}_i$ in the latent space, whereas the decoder D_{θ_d} learns to reconstruct the original text $\mathbf{x}_i$ based on the latent vector $\mathbf{z}_i$. Meanwhile, the stylistic C_{θ_c} predicts the style label y based on the latent vector $\mathbf{z}$.

$$\mathbf{z}_i = E_{\theta_e}(\mathbf{x}_i); \hat{\mathbf{x}}_\mathbf{i} = D_{\theta_d}(\mathbf{z}_i); y = C_{\theta_c}(\mathbf{z}_i) \quad (1)$$

The model is trained to both optimize the reconstruction loss $\mathcal{L}_{ae}$ (log likelihood) and the classification loss $\mathcal{L}_c$ (cross-entropy).

$$\mathcal{L} = \mathcal{L}_{ae} + \mathcal{L}_c \quad (2)$$

During the inference time, the input text $\mathbf{x}_i$ is first encoded by the encoder to produce a latent vector $\mathbf{z}_i$. Then the style transfer is performed by iteratively modifying the latent vector $\mathbf{z}_i$ using the gradients produced by the style classifier C_{θ_c}, as shown in the following formula.

$$\mathbf{z}^* = \mathbf{z} - w_i \nabla \mathcal{L}_c(C_{\theta_c}(\mathbf{z}), y) \quad (3)$$

It is assumed that the modified $\mathbf{z}^*$ will move in the direction of the opposite style in the latent space guided by the style classifier.

Our modification. In their original implementation, Wang et al. (2019) use a shallow transformer for both encoder and decoder. However, our inspection reveals that sentences generated by their model, though stylistically diverse, were not fluent enough. We inferred that this may be caused by insufficient training, as the training sentences were mostly short sentences with an average length of 15 words. Given such simple input, the model might not have learned enough linguistic knowledge to generate fluent language.

Addition model. Initially, we planned to improve the quality of generated texts by harnessing the large pre-trained generative models. Bart

(Lewis et al., 2019) is a pre-trianed generative model with the encoder and the decoder architecture, which suits the current framework. Fine-tuning on Bart could enable the model to utilize the language knowledge learned by Bart. Since the encoder of Bart produces a sequence of hidden states, we average-pooled the hidden states across sequence length and use a linear layer to transform it into the latent vector $\mathbf{z}$. The decoder of Bart will reconstruct the input texts based on $\mathbf{z}$, otherwise the overall design is the same as that in Wang et al. (2019).

However, simply pooling the hidden states negatively impacted the decoder's ability to reconstruct the text as much context has been lost. We also modified the method by summing the latent to the original hidden states before feeding it to the decoder.

$$\hat{z} = \sigma(\frac{1}{T}\sum_{i=0}^{T} h_i) \tag{4}$$

$$\mathbf{z} = \mathbf{W}\hat{z} + b \tag{5}$$

$$\hat{h}_i = h_i + \mathbf{z} \tag{6}$$

So that we end up with the modified states $\hat{h}_{1,\ldots,T} = [\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_T]$, which are used by the decoder to reconstruct the original texts.

Gating model. However, only using a single vector of hidden state might not be enough to represent the original sentence, as shown in our pilot experiments. We decided to using the gating mechanism to preserve more original hidden states to achieve better reconstruction during training and inference. For the gating model, an extra gating operation was added to modified the hidden states. Following Equation 6,

$$\hat{h}_i = h_i \cdot \sigma(W_z\hat{h}_i + b_z) \tag{7}$$

We found that the gating mechanism can help the model better learn to generate fluent language.

With the new hidden states $\hat{h}_i$, the output can be generated by the decoder:

$$\hat{\mathbf{x}}_i = D_{\theta_d}(\hat{h}_i) \tag{8}$$

3.2 Transforming the latent vector through non-linear mappings

Compared to the previous method, *Emb2Emb* (Mai et al., 2020) chooses to add a mapping module to transfer input embedding generated by encoder to another desired output embedding in the same manifold recognizable for the decoder, instead of directly changing the parameters of the encoder. The architecture of *Emb2Emb* is as shown in Figure 1. In this way, the previous problem of sequence-to-sequence is reduced into embedding-to-embedding problem.

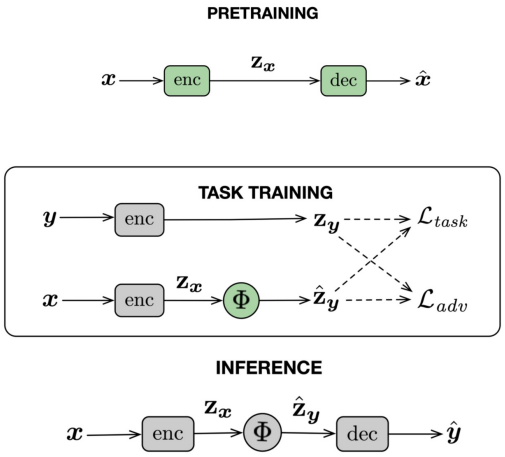

Figure 1: The architecture of *Emb2Emb* (Mai et al., 2020).

The first part of the model is the same as the previous method in 3.1, which is to pretrain an autoencoder consisting of encoder and decoder. Then in the second part is to freeze the pretrained autoencoder and add a mapping module, Φ, to transfer embedding z_x from encoder to z_y. z_y is then fed to decoder and elicits output sequence. The last part is inference by feed input sequence x through encoder, mapping module, and decoder.

The training objective of mapping module Φ is as Eq. 9

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{task} + \lambda_{adv}\mathcal{L}_{adv} \\ \mathcal{L}_{task} &= \lambda_{sty}\mathcal{L}_{sty}(\hat{z}_x) + (1-\lambda_{sty})\mathcal{L}_{cont}(\hat{z}_x, z_x) \\ \mathcal{L}_{adv} &= -\log(disc(\Phi(z_x))), \end{aligned} \tag{9}$$

where $\Phi(z_x) = \hat{z}_x$, $\mathcal{L}_{sty} = -\log(c(\Phi(z_x)))$, $\mathcal{L}_{cont}$ is a cosine similarity loss, and c as well as $disc$ are both a classifier, except that c is an externally pretrained and frozen one while $disc$ is trained during training Φ. The purpose of c is to evaluate how likely the output sequence is fitting the desired style of sentiment. The purpose of $disc$ is to discriminate against the embedding produced

by Φ by the following objective:

$$\max_{disc} \sum_{i=1}^{N} \log(disc(z_{x_i})) + \log(\overline{disc}(\hat{z}_{x_i})) \quad (10)$$

where $\overline{disc}(z_{x_i}) = 1 - disc(z_{x_i})$ and $disc(z_{x_i}$ is the probability that z_{x_i} is generated by the encoder.

Our Modification

- 1. We noticed that there is a dropout layer immediately after input layer in the external classifier c, which will cause many useful numeric values of input embedding to be zero. Therefore, we reduced the dropout layer. Besides, we deepen the classifier composed of only one layer neural network to a network of two layers.

  2. We added a batch normalization layer before activation layer in the mapping module. Previously, Φ is constructed by a Offsetnet (Mai et al., 2020) as shown in the left of Figure 2. Our modified net is shown in the right of Figure 2. The batch normalization is helpful for convergence and performance (Ioffe and Szegedy, 2015).

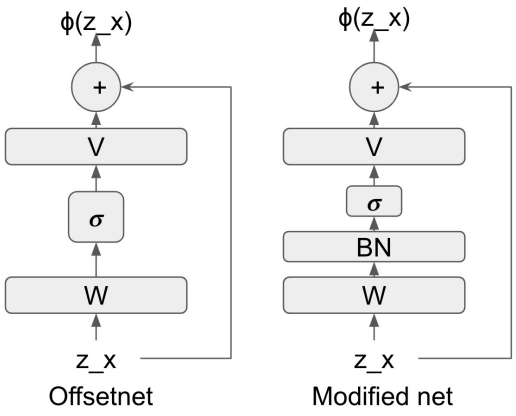

Figure 2: Left: Structure of Offsenet. Right: Structure of our modified network.

4 Dataset

We'll use the Amazon review dataset[1] for training and evaluating our model. This is the dataset containing product reviews on Amazon with human rated sentiments (Ni et al., 2019). The dataset contains 558,000 documents with the average length of documents is around 15 words. The partition and processing of dataset is inherited from Wang et al. (2019). The statistics of the data is as shown in Table 1. Note that there are no human written gold standard for training and validation set, except test set. Therefore, the dataset is basically nonparallel. The gold standard in test set enables the evaluation of BLEU.

	Train	Val	Test
Negative	277,000	1,015	500
Positive	278,000	985	500

Table 1: Statstics of Amazon dataset

5 Experiments

5.1 FGIM

We implemented the model discussed in Section 3.1. To facilitate replication, we used part of the original implementation [2] by (Wang et al., 2019). The Bart model was accessed through the Python package `transformers`.

We only used the Amazon review dataset. As the data had already been cleaned, no further preprocessing steps were carried out. All sentences were BPE segmented using the same Bart tokenizer. The model was trained using the Adam optimizer with a learning rate of 1e-5. Warm-up was used in the first 2000 iterations. The effective batch size was 64 after gradient accumulation. For comparison, we also trained the original model for 50 epochs using the exact same settings. All training was run on a Nvidia 2080Ti GPU and the training took about 10 hours to complete for all models.

5.2 Emb2emb

We also implemented the model discussed in Section 3.2. To facilitate replication, we used part of the original implementation [3] by (Mai et al., 2020). The code is written in Python and Pytorch. The data partition has been discussed in Section 4. The data processing and tokenization are discussed in Section 5.1.

The model was trained using the Adam optimizer with a learning rate of 1e-4. The final model is early stopped in 10 epochs. The batch size was set to 64. For comparison, we also trained the original model for 10 epochs using the exact same settings.

[1] http://deepyeti.ucsd.edu/jianmo/amazon/index.html

[2] https://github.com/Nrgeup/controllable-text-attribute-transfer

[3] https://github.com/florianmai/emb2emb

All training was run on the GreatLakes[4] cluster (ARC-TS) with 4 cores and 10 GB per core and the training took about 12 hours to complete.

6 Evaluation

Since the task of style transfer is newly emerging (Garbacea and Mei, 2020), it faces a bottleneck of lacking standard evaluation methods(Mir et al., 2019). Because of the possible lack of parallel data and ground truth for style transfer, some common metrics, e.g. ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002), may not be adopted as the only evaluation methods in the task. So multiple metrics should be used together for evaluation.

6.1 Automated evaluation

- **Transfer Strength:** Classification accuracy has been used as a metric indicating transfer strength for style transfer (Shen et al., 2017; Fu et al., 2018). We have trained a binary sentiment classifier based on the training set, and it achieved 79% accuracy in the test set. This model was used to evaluate the performance of all sentiment transfer.

  For the generated texts, we generated sentiment labels by flipping the labels of the text sentences. (If the original label is 1, we flipped the label to 0. This is because the generated sentences are supposed to be the style-transferred versions of the original sentences. So they should have reverse labels.)

- **Content:** BLEU (Papineni et al., 2002) is commonly used for measuring content preserving.

7 Results and discussions

7.1 Evaluations on FGIM

The evaluation results are presented in Table 2. The results show that our modification of FGIM did not improve the quality of the generated texts. For the addition model, the model simply produced repetition of random words or words from the original sentence (see Table 3). Its average BLEU is very close to zero, which implies that the output is meaningless and ungrammatical. FGIM+BART+Gating, while it does generate slightly more fluent texts, the overall quality is still unsatisfying.

While the quality of the original texts and the human written ground truth was not great, they are still readable texts. However, none of the models can generate fluent texts, though the Emb2emb method performs better than the rest. Even the best model still exhibit hallucinations as it brings in meanings not present in the original sentence, implying that modifying the latent space in a fine-grained manner is still difficult. The addition model (FGIM+BART+Addition) fails to generate meaningful text. This might be caused by insufficient contexts as we only provide a 1-dimensional latent vector for its decoder to generate. This setting differs greatly from its original training objective, in which it has all the attention vectors from the encoder. For FGIM+BART+Gating, it does learn to flip the sentiment by remove the negation marker "not" (see Table 3) but still suffers from repetitions, such that the grammaticality of the output texts were compromised. During training, BFGIM+BART+Gating can reconstruct the input texts most successfully but fails to do so in inference. We suspect that this might be caused by the inappropriate modification of the hidden states. As the latent space is often sparse, how to generate meaningful latent vector is still challenging.

7.2 Evaluations on *Emb2emb*

7.2.1 Modifications on External Classifier c

Recall that we removed dropout layer and deepened the neural network of the external classifier c in *Emb2emb*. Note that this external classifier c is for copmuting loss function $\mathcal{L}_{sty}$, not the classifier for final evaluation. With these modifications, the classification accuracy of the external classifier increases from 73.0% to 75.8% on test dataset.

7.2.2 Modification on Mapping Module

With the modification of adding Batch Normalization layer before activation in the Offsetnet of mapping module, the BLEU score of output sequence against gold standard increases from 22.86 to 23.65 as shown in Table 2. However, the the classification accuracy of the generated output sequences keeps unchanged as 58.6%. We note that although the computing speed of the training decreases from 480.18 [sentences/second] to 336.93 [sentences/second], the optimization becomes faster, e.g. the loss of network with batch normalization layer drops fast with fewer epochs than that without the layer.

[4] https://arc-ts.umich.edu/greatlakes/

Model	BLEU	Accuracy
Humans	-	42.3%
Original FGIM (Wang et al, 2019)	34.1	27.2%
FGIM + BART + Addition	0	50.02%
FGIM + BART + Gating	15.1	27.2%
Emb2emb (Mai, 2020)	22.86	58.6%
Emb2emb (modified) (Mai, 2020)	23.65	58.6%

Table 2: Average BLEU and overall accuracy for model outputs

	Output
Original	so not that great for leaving on at night
Humans	perfect for night
FGIM	so not that great for watching review on not at night .
FGIM + Bart + Additon	at night night night night night night night
FGIM + Bart + Gating	so that that that great good for leaving on at at
Emb2emb	so not that great for that on at night
Emb2emb (modified)	so not that great for turning on at night

Table 3: Random Sample outputs of all models

7.2.3 Evaluation on Output Sequences

We could see the final classifier elicit a higher classification accuracy on the output sequences of *Emb2emb* (58.6%) than the result on the human written gold test set (42.3%), which means the model generated output has a stronger sentiment expression and the effective of style transfer is remarkable. The BLEU scores (22.86 or 23.65) are lower than the original FGIM due to a greater probability of using new words. Table 3 shows one random example output of all models. We could see the original *Emb2emb* removes the negative word **leaving**, and the modified *Emb2emb* further replaces it to a positive word **turning (on)**. But both of them fail to transfer the sentiment in a fluent semantic expression and are far from approaching the gold standard.

Some further sample outputs are shown in Table 4. We could see in many cases, the model tries to transfer sentiment by replacing specific word that does not exactly match the gold standard, which is acceptable since there should be many reasonable gold standards. The possible errors can be sorted in three categories, i.e. ungrammatical expression like repetition, undesired changing of unrelated words, failure of transferring sentiment words. The ungrammatical expression can be attributed to the insufficient training of the pretrain autoencoder. The latter two errors are attributed to the defect of mapping module. In general, *Emb2emb* can generate some reasonable output sequences. Due to limit of budget and time, we cannot be able to hire any annotator to do human evaluation. But by the authors' inspection, 10% outputs are reasonable, 40% outputs try to transfer sentiment words, and 55% outputs have ungrammatical expression.

References

University of Michigan ARC-TS, Advanced Research Computing-Technology Services. Great lakes slurm cluster server.

Allan Bell. 1984. Language style as audience design. *Language in society*, 13(2):145–204.

NJ Coupland. 2011. The sociolinguistics of style. *The Cambridge handbook of sociolinguistics*.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuan-Jing Huang. 2019. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5997–6007.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

	Output
Original	1. after a few days i noticed i was unusually fatigued . 2. went to the second one using the same method , no go . 3. ridiculous ! i had trouble getting it on with zero bubbles . 4. if your bike had a **kickstand** on the **plate** it won t lock down .
Humans	1. after a few days i noticed i was unusually energetic. 2. went to the second one using the same method, it worked. 3. great ! i had no trouble getting it on with zero bubbles 4. if your bike had a **kickstand** on the **plate** it would lock down.
Emb2emb	1. after a few days i noticed i was satisfied . 2. went to the second one using the same method , go . 3. great ! ! i had trouble with it it with zero bubbles . 4. if your head had **a a on on** the **rack** it won t lock down .
Emb2emb (modified)	1. after a few days i noticed i was satisfied . 2. went to the second one using the same method , go . 3. great ! i had trouble getting it with with zero bubbles . 4. if your head had **a a** on the **bottom rack** it won t **down** down .

Table 4: Further good and bad sample outputs of *Emb2emb*. Blue means the sentiment word is exactly matching gold standard. Red means there is undesired repetition. Yellow denotes the sentiment words fail to be transferred. Green denotes the sentiment words are transferred but do not match the gold standard. **Underline Bold** denotes some non-sentiment words are changed mistakenly.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *AAAI*.

Cristina Garbacea and Qiaozhu Mei. 2020. Neural language generation: Formulation, methods, and evaluation. *arXiv preprint arXiv:2007.15780*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

S. Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*.

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*, pages 424–434.

N. Keskar, B. McCann, L. R. Varshney, Caiming Xiong, and R. Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*, abs/1909.05858.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. Gedi: Generative discriminator guided sequence generation.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Fuli Luo, Peng Li, Pengcheng Yang, Jie Zhou, Yutong Tan, Baobao Chang, Zhifang Sui, and Xu Sun. 2019a. Towards fine-grained text sentiment transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2013–2022.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Xu Sun, and Zhifang Sui. 2019b. A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 5116–5122. AAAI Press.

Florian Mai, Nikolaos Pappas, Ivan Montero, Noah A Smith, and James Henderson. 2020. Plug and play autoencoders for conditional text generation. *arXiv preprint arXiv:2010.02983*.

Remi Mir, Bjarke Felbo, Nick Obradovich, and Iyad Rahwan. 2019. Evaluating style transfer for text. In

Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 495–504, Minneapolis, Minnesota. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.

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*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6830–6841. Curran Associates, Inc.

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.

Chen Wu, Xuancheng Ren, Fuli Luo, and Xu Sun. 2019. A hierarchical reinforced sequence operation method for unsupervised text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4873–4883.

Yi Zhang, Tao Ge, and Xu Sun. 2020. Parallel data augmentation for formality style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3221–3228, Online. Association for Computational Linguistics.

Chulun Zhou, Liangyu Chen, Jiachen Liu, Xinyan Xiao, Jinsong Su, Sheng Guo, and Hua Wu. 2020. Exploring contextual word-level style relevance for unsupervised style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7135–7144, Online. Association for Computational Linguistics.