NeurIPS 2019 Reproducibility Challenge: Controllable Unsupervised Text Attribute Transfer via Editing Entangled Latent Representation

Zhi Wen **Shih-Chieh Fuh** **Andrei Romascanu**
McGill University
School of Computer Science
{zhi.wen, shih-chieh.fuh, andrei.romascanu}@mail.mcgill.ca

Abstract

Adversarial editing is a common technique used for attribute transfer. In the reviewed paper, the authors applied the technique on entangled latent representations to build a controllable and flexible model for text attribute transfer. In our ablation study, we studied the effect of latent space dimension and number of Transformer layers on the performance of the original model. We found that the pre-trained model provided by the authors had a lower performance than the reported performance. In addition, we have reported several issues regarding the model implementation, but we believe that the overall structure of the model design remains correct and valid.

1 Introduction

The goal of unsupervised text attribute transfer is to modify text by changing specific attributes such as sentiment, while preserving its attribute-independent content and the integrity of its original linguistic characteristics. The task is often performed without additional information beyond the reference text to be modified. One of the challenges is that there are few datasets that contain well-defined texts with different attributes along with the counterpart references, hence most studies of text attribute transfer are unsupervised.

Several models have been proposed for text attribute transfer. Common ways to perform text attribute transfer include either separating attribute and content representations into different models or integrating the content representation with attribute representation to perform adversarial sample generation. However, different attributes require different models. Moreover, the separation of attribute from other contents may lead to models that successfully modify the attribute at the expense of other contents, for example decreasing the readability and integrity of the output text. In the proposed model, the authors alleviate these problems by using an entangled latent representation for both the attributes and the contents, which they claim preserves the integrity of the text. Furthermore, the proposed Fast Gradient Iterative Modification (FGIM) algorithm can iteratively modify the latent representation through a set of weights. Compared to other models for text attribute transfer, the proposed model can perform in a more controllable and flexible style by adjusting the weights, and the same model may be applied to different attributes without remodeling.

2 Model Description

The authors introduce a model with three subcomponents: encoder, decoder, and classifier. The encoder E takes the input text x and encodes it to a latent representation z, which is used by the decoder D to produce the output text $\hat{x}$. The same latent representation is also used by the classifier C to classify the attributes of the latent representation and return the attribute value y. For each subcomponent:

$$z = E(x); y = C(z); \hat{x} = D(z) \qquad (1)$$

The task of text attribute transfer can be considered as an optimization task. The goal is to find the optimal representation that changes the text attribute while preserving

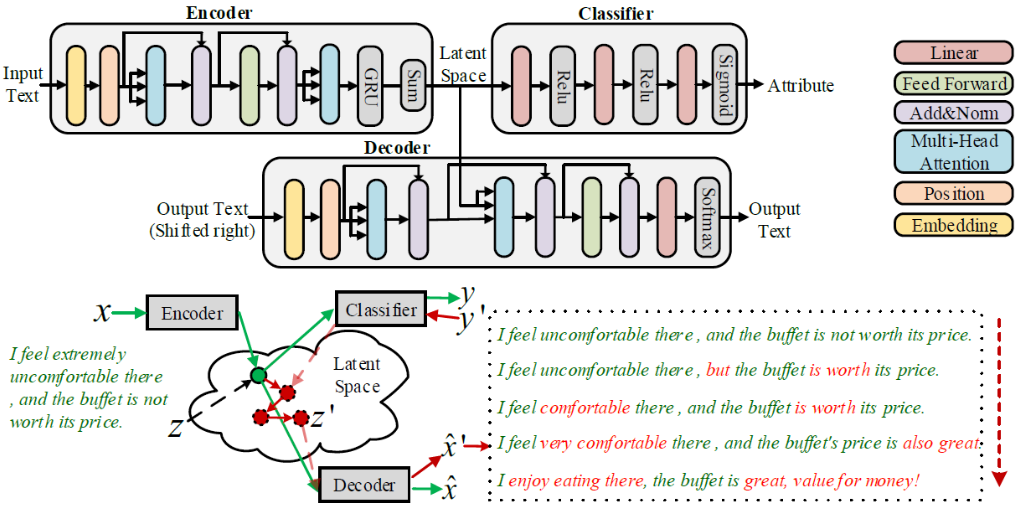
Encoder
Input Text
GRU
Sum
Latent Space
Classifier
Relu
Relu
Sigmoid
Attribute
Decoder
Output Text (Shifted right)
Softmax
Output Text
Linear
Feed Forward
Add&Norm
Multi-Head Attention
Position
Embedding
x
Encoder
Classifier
y
y'
I feel extremely uncomfortable there , and the buffet is not worth its price.
Latent Space
z
z'
$\hat{x}'$
Decoder
$\hat{x}$
I feel uncomfortable there , and the buffet is not worth its price.
I feel uncomfortable there , but the buffet is worth its price.
I feel comfortable there , and the buffet is worth its price.
I feel very comfortable there , and the buffet's price is also great.
I enjoy eating there, the buffet is great, value for money!

Figure 1: The model design (courtesy of Wang et al. (2019))

the attribute-independent content and linguistic fluency. The authors use a Transformer-based autoencoder as described in the work of Vaswani et al. (2017). In addition to the two-layer multi-head-attention feed-forward structure of the original Transformer encoder, the authors also apply a GRU layer and a sigmoid layer along with an extra positional embedding layer to get the final latent representation. The decoder is also implemented in a similar fashion to the Transformer decoder, which is extended from the two-layer Transformer encoder, with one extra multi-head attention layer inserted, followed by linear and softmax layers. The attribute classifier was implemented with two linear layers followed by sigmoid activation, with cross-entropy as the attribute classification loss. The authors have reported that the model performed better by optimizing the two loss functions from the encoder-decoder and the classifier separately compared to a joint optimization.

In order to find the desired latent space representation of the text with the desired attribute, gradient back-propagation is used to compute the attribute classification loss for the desired attribute. Based on the algorithm proposed by Goodfellow et al. (2014), the authors propose the FGIM algorithm. In the original fast gradient sign method by Goodfellow et al., the adversarial example is created by adding the perturbation term:

$$\eta = \epsilon sign(\nabla_x J(\theta, x, y)) \qquad (2)$$

which is based on the magnitude ϵ, the gradient ∇_x, and the cost function used to train the neural network J that takes the input text x and the target attribute y. This is based on the modification on the input text, while the authors proposed the update on the latent space z:

$$z^* = z - w_i \nabla_z \mathcal{L}_c(C_{\theta_c}(z), y') \qquad (3)$$

where z^* is the modified latent space, and the update term is based on the weight w, gradient with respect to the latent space $_z$, and the attribute classification loss function $\mathcal{L}_c$. The model implemented by the authors applies KL divergence to the loss function while the model described in the paper applies cross-entropy (see detail in the loss function discrepancy section).

Moreover, FGIM applies a Dynamic-weight-initialization mechanism that initializes over a set of weights to avoid the algorithm stopping at the local optimum. The FGIM algorithm iterates through a set of weights dynamically to find the optimized latent representation. The advantage is that the degree of attribute transfer can in principle be altered simply by modifying this weight. In addition, as seen in the update algorithm, given the latent representation z, the update is dependent on only the output of classifier

$C(z)$ and the desired target output y'. Different from other models that require additional information such as the attribute embedding, the model can be applied to multiple aspects with flexibility (Subramanian et al., 2018).

3 Dataset

The authors used three datasets: Yelp review for flipping sentiment (Yelp), Amazon review for flipping sentiment (Amazon), and image captions for flipping types between romantic and humorous (Caption). We focused our study on two smaller datasets Yelp and Caption due to the computational resource constraints.

4 Ablation Study

4.1 Experiments

We built our ablation study[1] upon the code base provided by the authors[2]. On each of the dataset provided by the authors, we modified the authors' code to fix bugs and ensure compatibility across modules, and evaluated the trained models on their respective test sets. We then trained the same models from scratch using the exact structure and configurations provided by the authors. As part of our ablation study, we then modified the models in ways not reported by the authors to investigate the impact of different hyper-parameters on model performance as well as the robustness of the proposed approach. Specifically, we studied the impact of latent space dimension and encoder depth on model performance. We also explored LSTM and self-attention as encoder and decoder blocks, however their generated sentences were of significantly lower quality compared to other models and thus we do not include them in our comparison. We adopted the authors' original evaluation methodology to compare the effects of these hyper-parameters on the test set of the respective datasets. We provide an analysis of our evaluation results, along with our main findings in terms of reproducibility throughout our experiments.

4.2 Reproducibility

Throughout our study, several features of the original paper and repository helped us reproduce the authors' methodology:

- The proposed model uses standard building blocks such as Transformer (Vaswani et al., 2017), and are implemented in popular open-source libraries such as pytorch.

- The paper includes a diagram of the model's overall structure.

- The authors provide a documented, generally well-organized code base.

We also noticed several issues during our experiments that hampered reproducibility of the paper:

- **Model structure the authors actually used in their code is different from the one described in their paper**

  This was the main issue troubling us in the beginning stage. For instance, as described in the paper, the model has a GRU block at the top of their encoder, which is described both in the model diagram (Figure 1) and in Section 3.2. However, the authors codebase includes two models for each dataset, one of which was noticeably incomplete for each dataset. Unfortunately, certain architecture components such as the GRU reported in the paper seemed to appear only in the incomplete implementation. Upon reaching out to the authors for clarification, they confirmed that they actually used the implementation without GRU. They also informed us they will update their implementation with GRU, although it performs similarly to the one without GRU. Their original reply can be found here:

[1] Code available here https://github.com/BruceWen120/neurips-reproducibility-challenge-2019

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

https://github.com/Nrgeup/controllable-text-attribute-transfer/issues/12issuecomment-565020857.

- **Loss function actually used in the code is different from the one described in the paper**

  We noticed this in a GitHub issue (in Mandarin)[3]. The loss function in the paper is cross-entropy loss as described in Equation 4 of original function, whereas in the code authors replaced this with KL-divergence.

- **Testing files of 2 classes in the Caption dataset are identical**

  The caption dataset provided by the authors have identical positive samples and negative samples in its test set. We suspect they are either all positive samples or negative samples, and since we followed the original code by evaluating on sentences modified to positive class from negative class, this issue might have two possible consequences. One is, if negative samples are actually positive samples, the sentence modification process would barely modify the input sentence before outputting it. This is because when the classifier in the FGIM algorithm decides the latent vector belongs to positive class it would stop the modification process. The other possible scenario is, if positive samples are actually negative, the reported multi-BLEU score would be meaningless, since it would be basically comparing the modified sentence with the unmodified one.

- **Despite the code base being generally well-organized, there are minor bugs and inconsistency of notations that need to be resolved first.**

  There is no major error in code that could either challenge the claims made by the authors or prohibit the reproduction of their results, however there are still minor bugs that needed to be resolved before being able to reproduce experiments. There is also minor inconsistency in using notations, for example the initial weight $\mathbf{w}$ in the paper is actually denoted as `epsilon` in code, without any clarifying comments.

- **Our experiments suggest the model developed by the authors is susceptible to variations in certain parameters or structural characteristics, and evaluation metrics used in this paper may not perfectly reflect the models' performance.**

  We found that the model proposed in the paper performed similarly when some parameters are changed, while performing better or worse in certain aspects when other changes are made. Furthermore, we find that the evaluation methodology and reported metrics of perplexity, BLEU and accuracy might be misleading. We elaborate on these findings in Section 4.3.

4.3 Evaluation

We replicated the same evaluation as the original authors of (Wang et al., 2019), specifically measuring the perplexity of transformed text to evaluate fluency, the multi-BLEU score relative to human-transformed text to evaluate similarity, and the attribute classifier accuracy to evaluate success of attribute transfer. While the authors provided code to measure multi-BLEU scores, we had to independently reimplement the evaluation of perplexity and classifier accuracy based on the methodology described by the authors.

For perplexity, we use the SRILM toolkit (Stolcke, 2002) to first train a language model on each dataset corpus (which we interpreted as the concatenation of the train, dev and test sets for a given dataset) using *ngram-count* with order 2 and no smoothing; and then measure perplexity on a given text using *ngram*

[3] https://github.com/Nrgeup/controllable-text-attribute-transfer/issues/5

with the trained language model as described in (Levy, 2015). Due to potential limitations of a language model trained on the original corpus with no transformed text, we also investigated using pre-trained GPT-2 as a language model to measure perplexity. However, we found that pre-trained GPT-2 has a significant bias towards longer sentences, leading to very large (> 1000) perplexities for short yet grammatical sentences; as a result we did not include it in our analysis.

For classifier accuracy, we trained a fastText (Joulin et al., 2017) attribute classifier on the training set of each dataset as described by the authors. We use the preprocessing and methodology recommended in the official fastText repository, obtaining test set accuracies of 75.84%, 63.47% and 50.00% for the amazon, yelp and imagecaption datasets respectively. The accuracy on the imagecaption dataset is due to the fact that every example in the dataset appears twice, once with each label. We then use the trained classifiers on the transformed text to predict the attribute. The original authors do not describe how they compute accuracy and our numbers do not match, so we generate two accuracy variants: (1) relative to the target attribute and (2) relative to the classifier prediction on the human-transformed text. As a benchmark, we also compute the accuracy of the classifier on the human-transformed examples relative to the target labels.

4.4 Results

Evaluation results of different models on the Yelp dataset and the Caption dataset are summarized in Table 1 and Table 2 respectively. Specifically, we reduced latent space dimension from 256 to 128, increased it to 512, and reduced the number of Transformer layers from 2 to 1.

We made the following observations from the results:

- **Overall, the pre-trained model provided by the authors had a lower performance compared to the reported performance.** On the Yelp dataset, we also observed that compared to pre-trained model the replicated model trained from scratch had lower BLEU score and higher perplexity, but with high accuracy. We have yet to identify the source of this discrepancy, however it is likely due to the slight differences in evaluation methodology resulting from us having to reimplement it.

- **Compared to human references, models tend to have lower perplexity and higher accuracy.** We hypothesize this is because both language model and classifier were trained on the same datasets as the main model was. This might lead to the language model and classifier ”overfitting” to the datasets, i.e. not generalizing to accommodate sentences written by human.

- **BLEU score and accuracy have inconsistent trends, despite the fact that they should both measure the success of attribute transfer.** We believe accuracy is more meaningful here, because BLEU compares against one gold standard, whereas for style transfer task there could be more than one candidate that is reasonably good. In contrast, using classifier is not restricted to the gold standard. Furthermore, we found that the BLEU score for untransformed text was much higher than that of the transformed text, showing that a model which did not modify the text at all would achieve a better score. This is likely due to the fact that BLEU is agnostic to pivot words, and thus poorly suited as a measure of text attribute transfer.

- **We also observed reducing the number of layers had minimal impact on model's performance based on all three metrics.** This suggests it might not be necessary to use 2 Transformer blocks in the encoder or decoder as proposed in the paper. The model with reduced latent space dimension had a slight decrease in

Table 1: Evaluation of models and human references on Yelp dataset

	BLEU	**Perplexity**	**Accuracy**
Reported performance	24.6%	46.2	95.4%
Trained model by authors	19.8%	59.6	96.1%
Replicated model trained from scratch	10.9%	78.4	97.1%
Model with reduced latent space dimension	17.0%	61.6	98.0%
Model with increased latent space dimension	13.1%	42.5	44.9%
Model with reduced number of layers	22.3%	46.4	91.4%
Human references	NA	51.5	64.3%

Table 2: Evaluation of models and human references on Caption dataset

	BLEU	**Perplexity**	**Accuracy**
Reported performance	23.7%	17.6	92.3%
Trained model by authors	9.0%	18.9	73.8%
Replicated model trained from scratch	17.3%	18.9	74.0%
Model with reduced latent space dimension	15.3%	18.7	76.0%
Model with increased latent space dimension	19.4%	18.9	71.5%
Model with reduced number of layers	20.7%	17.6	76.3%
Human references	NA	21.2	68.2%

performance in terms of the BLEU score and perplexity. **Meanwhile, we noticed with the increase of latent space dimension both perplexity and accuracy dropped significantly.** We suspect with a larger latent space the information bottleneck for reconstructing sentences was significantly mitigated, therefore lowering perplexity, while the increased number of parameters prohibited the model from converging, and thus the drop in accuracy.

- **Negative samples in Caption test set might actually be positive.** As mentioned in Section 4.2, there are two possible consequences of the Caption dataset having identical positive and negative samples. From the evaluation results on Caption dataset, we noticed the model was not as susceptible to variations in parameters or structures as on the Yelp dataset, even in terms of perplexity and accuracy which do not reply on gold standard. This indicates that perhaps all negative samples are actually positive, since in this case generated sentences were barely modified and thus close to input sentences, which are the same for all models.

5 Discussion and Conclusion

The paper we studied proposed a framework for controllable attribute transfer based on an auto-encoder structure. The novel design of the proposed model with its FGIM algorithm on entangled latent space editing allows flexible attribute transfer. In this ablation study, we set out to examine this work by reproducing their model and exploring its variants. Through experiments, we noted this work's features that helped our reproduction as well as issues we encountered. We further compared and analyzed our experiment results from pre-trained and re-trained models against results reported in the paper to examine reproducibility, as well as against models with modified structures to evaluate robustness. In conclusion, we identified certain aspects of the original work that can be improved for better reproducibility, while there are also issues that we were unable to discover the causes of, as well as ones that require further investigation to confirm or reject our hypotheses.

References

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Roger Levy. 2015. Working withn-grams in srilm.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*.

Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552*.

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.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. *arXiv preprint arXiv:1905.12926*.

A Exemplary sentences

Table 3 show an example of sentences generated by different models and by human.

Table 3: Examples from different models and human

Source	**Example sentence**
Human reference	it is n't perfect , but it is very good .
Example in paper	it is n't terrible , but it is very good and delicious .
Pre-trained model	it is n't terrible , but it is n't very good either .
Retrained model	it is good delicious , but it is is very good great either !
Reduced latent space dimension model	it is great variety , but it is definitely a great good too !
Increased latent space dimension model	it is n't terrible , but it is n't very good either .
1-layer model	it is n't terrible , but it is n't very good either .