Transformer-based Disentanglement of Natural Language for Counterfactual Explainability

Cooper Stevens, Wei Jie Lee, Tianliang Xu
{coopstev, lweijie, tianlix}@umich.edu

Abstract

Style transfer is an emerging topic in the field of Natural Language Processing (NLP). Inspired by the style transfer literature for images, we experiment with various structures of natural language generative models that incorporate transformers and arbitrary classifiers to produce counterfactual examples. The goal of the counterfactual examples is to enhance the explainability of any arbitrary classifier model. Compared with the classical transformer structured model, we focus on training style embeddings between encoders and decoders. We hypothesize that the new architecture could improve the state-of-the-art in terms of the stylistic coherence and fluency of the generated text. Additionally, our objective is to generate examples that exhibit stylistic differences across multiple dimensions (beyond simple sentiment) and considering subtle stylistic nuances.

1 Problem Statement

Many of these style transfer techniques utilize Generative Adversarial Networks (GANs), and we will refer to such style-transfer-GANs as StyleGANs. Modern StyleGAN techniques have mostly been applied to images, but have been applied in a variety of fashions. To name a few:

- Karras et al. (2019) used it to allow a user to make a requested edit to the image that is completed through some GAN's use of the styles.
- Karras et al. (2018) used it to "mash" images together to create hybrid-image results using GANs.
- Li et al. (2018) used it for prototype extraction of groups (good for unsupervised clustering methods) using GANs.
- Lang et al. (2021) used it to improve model explainability in post-processing (also using GANs).

We aim to extend and combine these methods to build a StyleGAN model for NLP tasks. In particular, we would build such a StyleGAN with the objective of modifying emotionally weighted sentences to invoke different emotions. Our goal here is to improve the explainability of NLP models. Specifically, we would like to generate counterfactual examples for a given text input to aid us in understanding the behavior of the model. In the context of a classifier C and input sentence m, our goal here is to generate m' such that the output of C on input m' would be $y' \neq y$, where y is the output of C on m (which is comparable to m'). Providing such examples would improve explainability for a user by allowing them to see (by inspection) what characteristics of the input are leading a certain classification, thus informing the user on how the model makes its decisions.

2 Related Works

Lang et al. (2021) introduce StylEx, a method to generate counterfactual examples given an image classifier. This paper also attempts to improve explainability of models—specifically classifiers—by

elucidating how the classifiers come to a decision. Our work here is inspired by this paper, but our aim is to work on natural language instead of images.

Dai et al. (2019) introduce Style Transformer, which, given a source sentence, disentangles the content and style information in the latent space. The authors use the Style Transformer to transfer the style of a sentence while preserving the content. Compared to our work, this paper mainly focuses on changing the sentiment of the text (positive or negative), and the model architecture is not designed to take in an arbitrary classifier. As such, the style transformer here will not be useful for arbitrary model explainability. Moreover, our work will include more than a transformation of sentiment—we intend to work on higher-dimensional transformations and consider other style-related properties that are found to result in a different classifier output.

2.1 Obtaining a Performance Baseline: Investigating Style Transformer's Performance

Our Multi-class Experiment in Section 5.4 is performed by modifying the Style Transformer code to use the GoEmotions dataset as described in Section 3.2. Hence, in order to properly assess any results from the experiment, we first looked to obtain a baseline of performance that the original code from Style Transformer can provide. These baselines of performance will be essential in evaluating the performance of the resulting multi-class model that is obtained from our Multi-class Experiment. As a reminder, Dai et al. (2019) trained and tested on the Yelp Open Dataset where a review has an associated sentiment that is positive or negative. After training for 19,175 iterations, the following plots display the model performance over time for accuracy, BLEU score, and Perplexity. These metrics are explained further in Section 4.

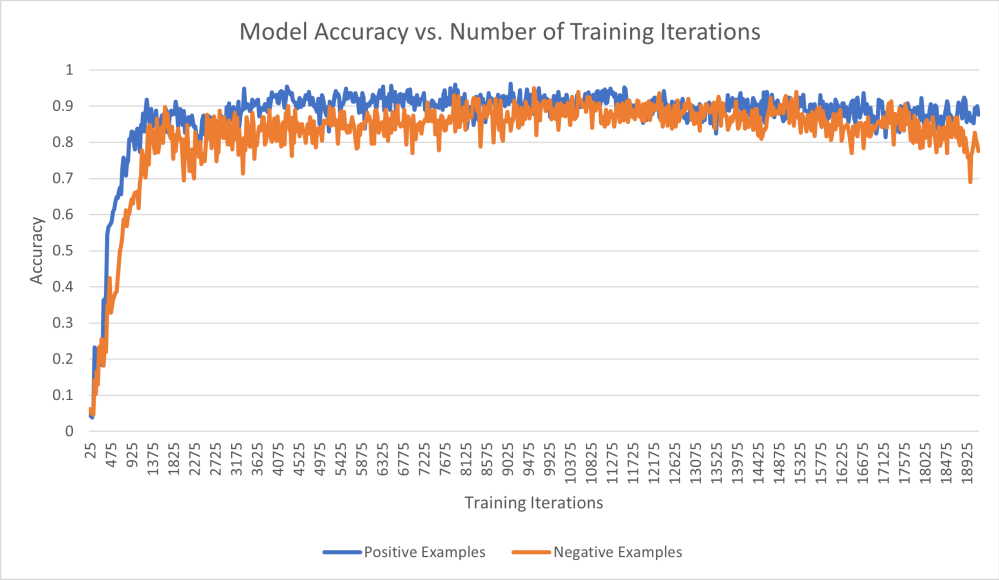


Figure 1: Plot of model accuracy as a function of number of training iterations for Style Transformer

As can be seen in Figure 1, accuracy for both positive and negative examples converged very quickly (after about 6,000 iterations). Additionally, it appears that the model begins to overfit past 11,000 iterations. Finally, the accuracy on negative examples seems to be a few percentage points higher than that for positive examples. In future work, it might be worth investigating the disparity in performance on the two types of examples.

As can be seen in Figure 2, the model converges to a BLEU score that is lower than the initial, indicating that the loss functions for training are not encouraging good translations, as measured by BLEU. This is indicative of an inherent disconnect between the training process objectives and the performance/testing objectives. This is certainly something to keep in mind as we

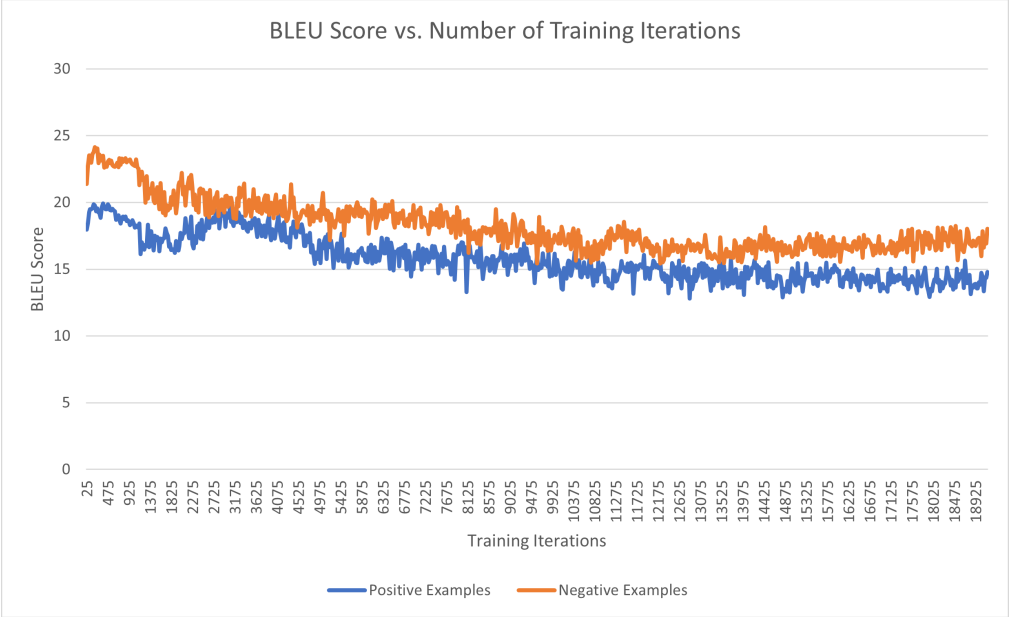


Figure 2: Plot of model's BLEU score as a function of number of training iterations for Style Transformer

assess our multi-class model in Section 5.4.

As can been seen in Figure 3, the perplexity also converges to a value that is higher than

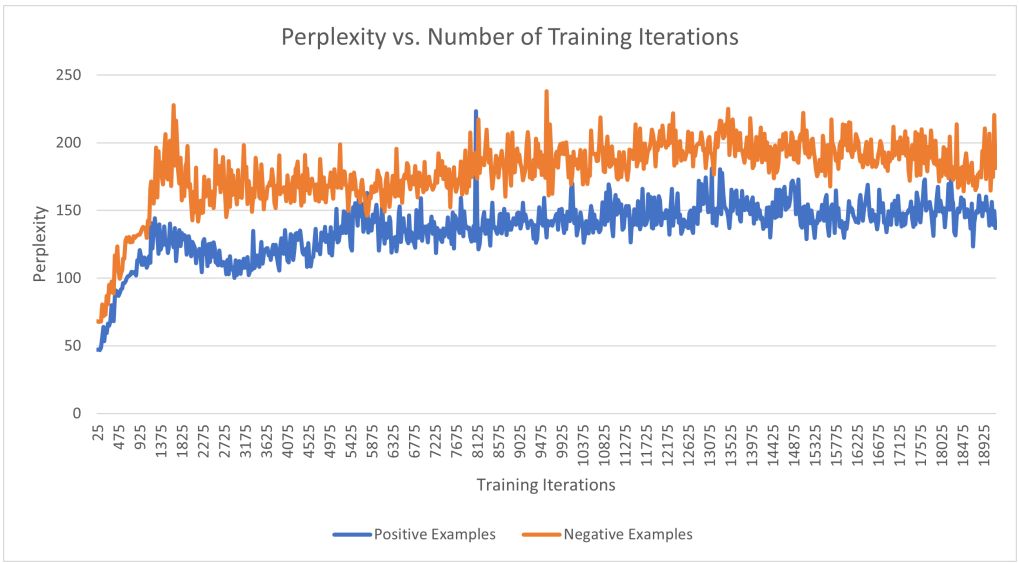


Figure 3: Plot of model perplexity as a function of number of training iterations for Style Transformer

the initial, indicating that the loss functions for training are causing it to learn a probability model that is not representative. This is comparable to the trend of the model's BLEU score in that the performance converges to state that is worse than the initial state. Again, this is something to keep in mind as we assess our multi-class model in Section 5.4.

It's worth noting that we were not able to reproduce the quality of translations that the Style Transformer paper produced. For example, when transferring the review "fantastic place to see a show as every seat is a great seat !" from a positive to a negative sentiment, our generator outputs

"bad service to see a show as every seat is a better seat service ." whereas Dai et al. (2019) reports having its generator output "disgusting place to see a show as every seat is a terrible seat !". This is most likely due to a lack of computing resources in training our generator for this task when compared to the computing resources used by Dai et al. (2019).

3 Datasets

3.1 Yelp Open Dataset

The Yelp dataset, provided by the Yelp Dataset Challenge, consists of restaurant and business reviews with binary sentiment labels. Following previous work, we use the preprocessed dataset provided by Dai et al. (2019). Specifically, it contains $443,259$ examples in the training set and 500 positive examples as well as 500 negative examples in the test set. This dataset also includes human reference sentences in the test set for BLEU score computation.

3.2 GoEmotions Dataset

Our next dataset comes from (Demszky et al., 2020) and contains messages posted on Reddit that have been human-annotated to provide labels to more than the basic human emotions. Instead, the taxonomy include 12 positive, 11 negative, 4 ambiguous, and 1 neutral emotion category. The expanded taxonomy of the dataset is also helpful to us as it could be used to generate counterfactual examples that differ in more subtle nuances. We do not make use of the sentiment (positive/negative/ambiguous) membership for each of the 28 styles—we only use the styles themselves. Figure 4 displays the frequencies of occurrence of each of the 28 styles in the training set.

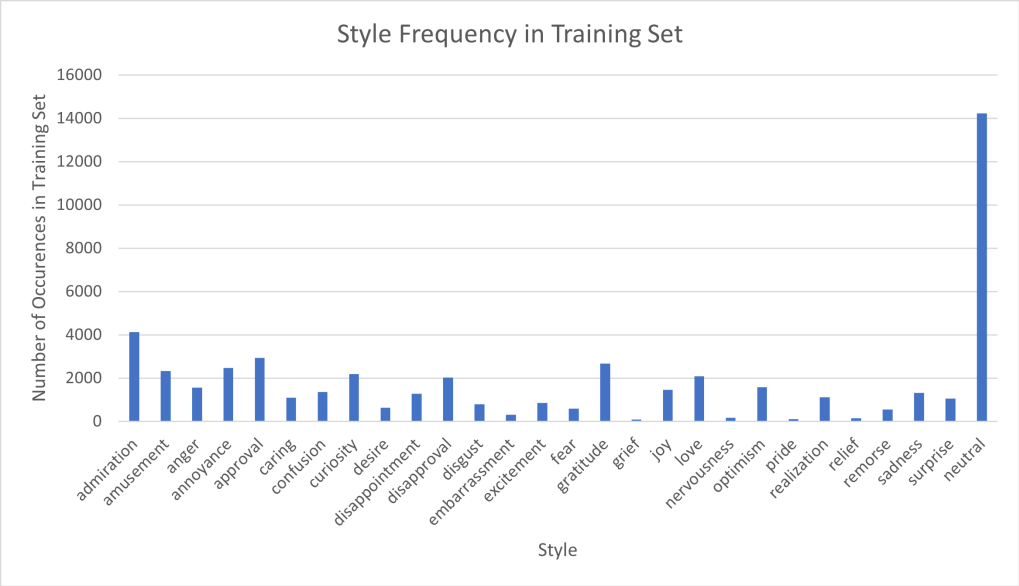


Figure 4: Number of occurrences in the GoEmotions training set for each of the 28 styles

It is clear to see in Figure 4 that the neutral style is, by far, the most common style in the training set. In the context of our goals, it is reasonable to hypothesize that this may make style transfer an easier task on average. This hypothesis is reasonable because a neutral sentence is, reasonably, close to all other styles via small modifications (e.g. adding an exclamation point to change the style to surprise, or adding a sad face :(at the end to change the style to sadness, etc.), and so transferring the style may require less modification to the original sentence than, say, transferring a sentence with an "amusement" style to one with a "remorse" style. The effect of this abundance of neutral styles may be worth investigating in future work by, for example, removing all sentences with a neutral style from the dataset and proceeding as usual with training.

The dataset comes pre-partitioned into a training set of 43,410 posts, and a testing set of 5,427 posts. To lessen the computational load in evaluation (since we will be evaluating every 10 training iterations in order to identify trends in performance), we use a subset of the first 200 posts from the testing set for all of our evaluations. We also truncate all posts to 32 tokens (the longest post is 486 tokens, and the shortest is 3) so that the batch size can be reasonably large to encourage faster convergence.

Some of the posts in the dataset are members of multiple classes. For example, the post from the training set "We need more boards and to create a bit more space for [NAME]. Then we'll be good." emotes both desire and optimism. However, in order to more easily use the Style Transformer work as a base, all posts are designated as a style chosen uniformly at random from those classes that the dataset designates. For example, in training, the post given above has a equal chance of being trained as a "desire" example or as a "optimism" example, but not both. In future work, one may look to build a model that can handle multi-class classification so that both styles can be maintained as ground-truth values. Alternatively, one could look into the effect of including the same post twice under two different ground-truth values so that the model can train on both cases, but the single-class classification architecture is maintained.

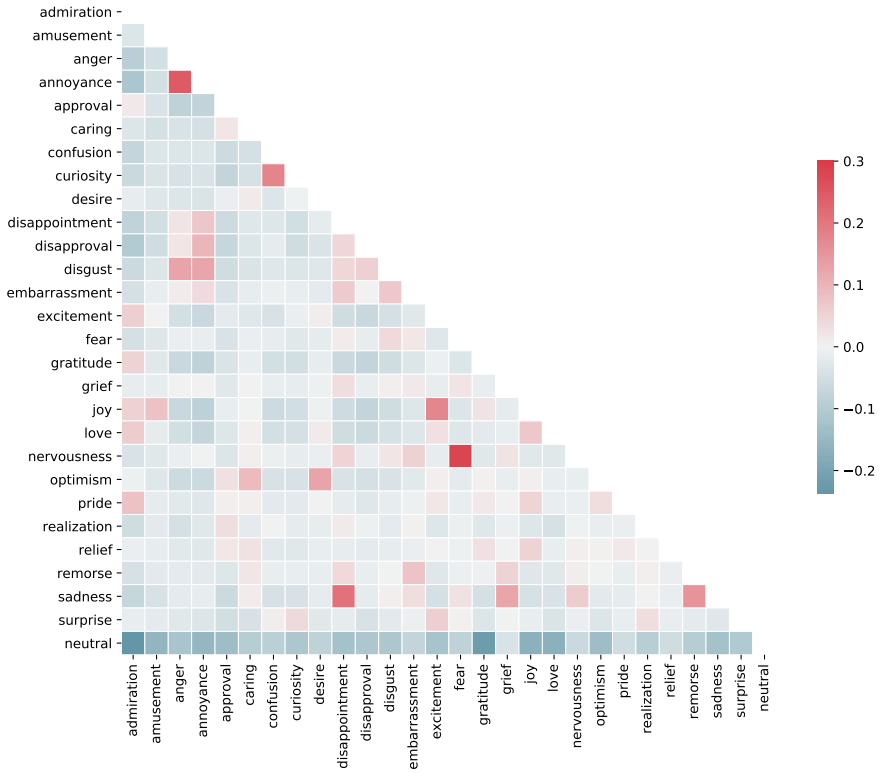


Figure 5: Correlations between different styles in the GoEmotions dataset (Demszky et al., 2020)

Figure 5 pictures the correlation matrix for different styles in the GoEmotions dataset. A more positive (red) correlation between styles x and y indicates that a post is more likely to also be style y if we already know it is style x, and vice versa. On the other hand, a more negative (blue) correlation between styles x and y indicates that a post is less likely to also be style y is we already know it is style x, and vice versa. There are a few notable insights into the dataset that can be gained upon inspection of this correlation matrix:

- The neutral style has a negative correlation with all other styles. This is intuitive because (per the meaning of a post to be neutral in style) a post cannot be both neutral and non-neutral in style simultaneously.

- There are some very strong positive correlations between some pairs of styles that we would expect. These include (annoyance and anger), (curiosity and confusion), (joy and excitement), (nervousness and fear), (sadness and disappointment), and (sadness and remorse). These positive correlations are somewhat expected due to the similar definitions of these styles, the large intersectionality of their use cases, and the interchangeability of the words in many contexts.
  - Interestingly, despite strong positive correlations between (sadness and disappointment) and (sadness and remorse), the correlation between (remorse and disappointment) is very weakly positive. This, again, is an intuitive result due to the different emotive definitions of these words, and the small intersectionality over which both of these words can be used to describe a single emotive state.

While these insights are intuitive, they are nonetheless important to verify in order to be confident that the dataset is sound in terms of the ground-truth values that it provides.

4 Evaluation Metrics

Dai et al. (2019) use accuracy, Bilingual Evaluation Understudy (BLEU), and perplexity as metrics to assess the performance of the resulting model. Hence, we will do the same.

4.1 Accuracy

Accuracy of a resulting model is measured with respect to the generator as follows: for each sentence/post in the testing corpus (suppose it has style s), use the generator to change its style to some different style $\hat{s} \neq s$. Then, using a classifier obtained through some alternate source (in particular, do **not** use the discriminator that was trained in tandem with the generator), see if the classifier can detect that the output sentences emotes the target style $\hat{s}$. If it does, this is a successful example. Otherwise, this example is considered a failure. Then, accuracy is evaluated as the proportion of successful examples in the whole testing corpus.

4.2 BLEU Score

The BLEU score of a model measures the model's translation quality in the range $[0, 100]$ where 100 indicates a perfect translation without any information loss. Note that, in any non-identical translation context (such as Style Transformer and all of our experiments), a BLEU score of 100 is considered to be unobtainable because any change in wording or style will result in a loss of information in the form implication or connotation. Obtaining the BLEU score for a sentence requires a reference sentence to compare the translated sentence to that serves as the ground truth of the encoded information from the original sentence. For our Multi-class Experiment using the GoEmotions dataset in Section 5.4, we have no reference posts for any of the posts in the corpus, so we are unable to assess translation quality using a BLEU score at this time. This is a potential shortcoming to address in future work.

4.3 Perplexity

The perplexity score of a model measures how well the generative model's probability model predicts the testing data. Dai et al. (2019) state that they obtain a probability model using 5-grams from the Yelp Open Dataset corpus for assessing the resulting models' probability models. We choose to use the same probability model (from the Yelp corpus) to assess our multi-class models due to lack of resources to obtain a probability model using 5-grams from the GoEmotions corpus.

5 Experiments

What follows is a number of experiments with the goal of providing counterfactual examples to a model user.

5.1 Experiment 1: StylText

Inspired by Lang et al. (2021), we modified the training process proposed in Style Transformer by (Dai et al., 2019).

5.1.1 Model Description

Specifically, in this experiment, our model generally consists of three parts.

For encoder, we followed the previous works, the encoder parts adopt the implementation of the classic Transformer (Vaswani et al., 2017). Specifically, it consists of a stack of identical layers, each containing two primary sub-layers: a multi-head self-attention mechanism and a position-wise feed-forward network. Additionally, residual connections and layer normalization are employed to facilitate training and improve performance.

Similarly, we adopt the manner in Transformer which is the same as Lang et al. (2021) for the decoder. The structure of the decoder is similar to the encoder, with one main primary difference that its self-attention only considers earlier positions in the output sequence. This constraint ensures that the model generates a valid output.

We train a style upsampler to classify and upsample the styles to higher dimensions for generating text. The style upsampler will be trained by the labels of the text. The original text and upsampled styles will be combined before feeding into the encoder. In this experiment, we utilized a stack of linear layers and residual blocks as a simple upsampler.

Our goal is to decompose from the binary style to multi-dimension style representations to diversify the style expression as what is mentioned in StyleGAN (Saravia et al., 2018). We employed the training approach proposed by StyleGAN. Specifically, an encoder architectures are utilized to extract style representations from the original text, thereby forming a style space. The style in this space does not correspond to the actual labels in the dataset; hence, we consider using contrastive learning to learn the style of the original and generated text.

For the contrastive loss, we used the dual contrastive loss proposed by Yu et al. (2021). Dual contrastive loss help the model converge faster by leveraging complementary information from the positive and negative contrastive losses, which has been shown outplaying other loss functions in image generating in the paper. The computation of the contrastive learning loss as following.

$$
\begin{aligned}
L_{\text{real}}^{\text{contr}}(G, D) &= \mathop{\mathbb{E}}_{\mathbf{x}\sim p(\mathbf{x})}\left[\log \frac{e^{D(\mathbf{x})}}{e^{D(\mathbf{x})}+\sum_{\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I}_d)} e^{D(G(\mathbf{z}))}}\right] \\
&= -\mathop{\mathbb{E}}_{\mathbf{x}\sim p(\mathbf{x})}\left[\log\left(1+\sum\nolimits_{\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I}_d)} e^{D(G(\mathbf{z}))-D(\mathbf{x})}\right)\right]
\end{aligned}
$$

$$
\begin{aligned}
L_{\text{fake}}^{\text{contr}}(G, D) &= \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I}_d)}\left[\log \frac{e^{-D(G(\mathbf{z}))}}{e^{-D(G(\mathbf{z}))}+\sum_{\mathbf{x}\sim p(\mathbf{x})} e^{-D(\mathbf{x})}}\right] \\
&= -\mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I}_d)}\left[\log\left(1+\sum_{\mathbf{x}\sim p(\mathbf{x})} e^{D(G(\mathbf{z}))-D(\mathbf{x})}\right)\right]
\end{aligned}
$$

$$
\min_G \max_D L_{\text{real}}^{\text{contr}}(G, D) + L_{\text{fake}}^{\text{contr}}(G, D)
$$

In pursuit of enabling the model to distinguish as many different upsampled styles as possible, it is also essential for the model to retain the original binary positive and negative labels to a certain degree. Consequently, we draw upon StylEx's training methodology (Lang et al., 2021). We initially run a specified number of pre-training steps for the model style upsampler, followed by subsequent contrastive learning.

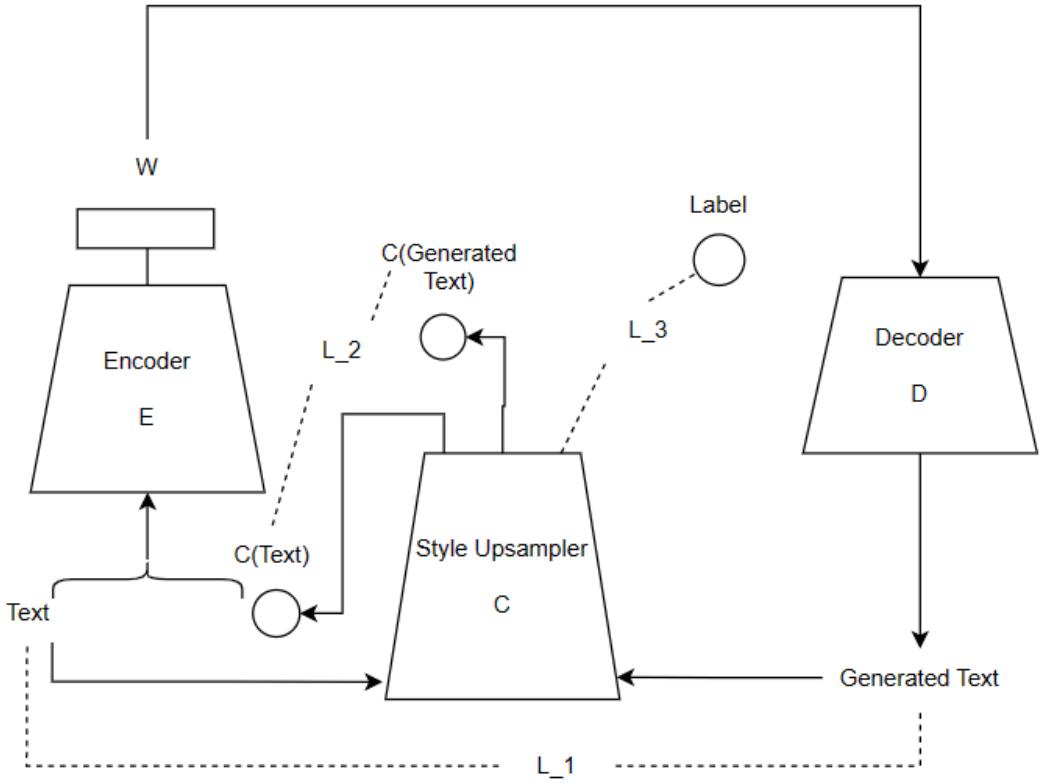


Figure 6: The proposed architecture of our model consists of the following components. First, the text is input into the style upsampler to obtain higher-dimensional style representations. The classification results are then concatenated with the tokenized text and processed through the entire transformer. During each training step, three loss components are computed: L_1 represents the sentence reconstruction loss, which is calculated using the negative log-likelihood function. L_2 denotes the style contrastive learning loss, computed via the dual contrastive loss function. L_3 corresponds to the style pre-training loss, which is determined by comparing the output logits and the true styles using the negative log-likelihood function.

5.1.2 Evaluation

The Figure 7 is the results of the StylText. The style accuracy is as low as 0.03, indicating that the first experiment fails to change the style of the input sentence almost every time. Additionally, the BLEU score and the perplexity appear unchanging, indicating that the model is not learning properly.

The potential reasons can be:

1. The loss design: In the first experiment, we do not actually design a loss for the changing styles.
2. The training process: We feed the true label into the transformer to generate text but we ignore the other styles.

5.2 Experiment 2: Experiment with different Upsamplers in Style Transformer

In light of the shortcomings of StylText, we maintained the circular training structure proposed in the style transformer (Dai et al., 2019). Additionally, we removed the contrastive learning approach employed in style classification. In this experiment, our primary objective was to examine the impact of different style upsamplers.

Figure 7: Results of the StylText indicate poor performance. The accuracy does not exceed 0.03, the BLEU score ranges between 20 and 25, and the perplexity is approximately 60. Notably, the BLEU score and the perplexity appear unchanging, indicating that the model is not learning properly.

5.2.1 Model Description

In this experiment, we feed the true style of original text into the encoder. In this training process, the loss computation of the text generator and the recognizer are alternated.

In this experiment, we explore the possibility of designing a continuous style space. However, employing a style embedding as proposed in the style transformer is not well-suited for this purpose, as it generates a discrete style space. Consequently, we devise a style upsampler that comprises a stack of linear layers and residual blocks.

For the training process, the input text and another randomly selected style S' are fed into the transformer to generate a trans-styled text G'. The discriminator D processes G' to obtain the corresponding logits. the transformer is used once more with the true upsampled style S and the trans-styled text G' to reconstruct the input text. Three losses are computed as following: L_1 is the sentence reconstruction loss, which is calculated using the negative log-likelihood function. L_2 represents the style contrastive learning loss, also computed with the negative log-likelihood function. L_3 refers to the cyclic training loss, which is the same as the sentence reconstruction loss.

Initially, to generate reasonable outputs rapidly throughout the transformer, we pre-trained the transformer with the text reconstruction training procedure. Notably, the training of the discriminator and the cyclic training are conducted alternatively. A specified number of transformer training steps are executed, followed by a certain number of discriminator training steps.

5.2.2 Evaluation

The Figure 9 refers to the results of the Experiment 2. In this experiment, the results remain unsatisfactory. Although the accuracy is higher than that in StylText, a value of 0.06 still indicates that the style is rarely changed successfully. For the BLEU score and perplexity, no substantial improvement is observed compared to StylText.

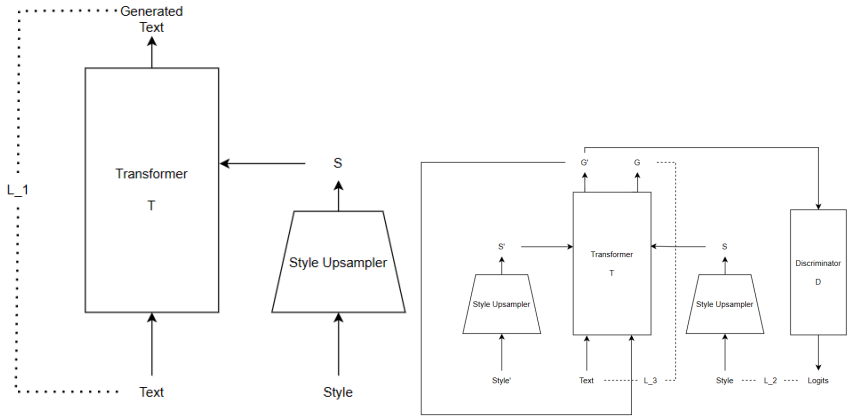
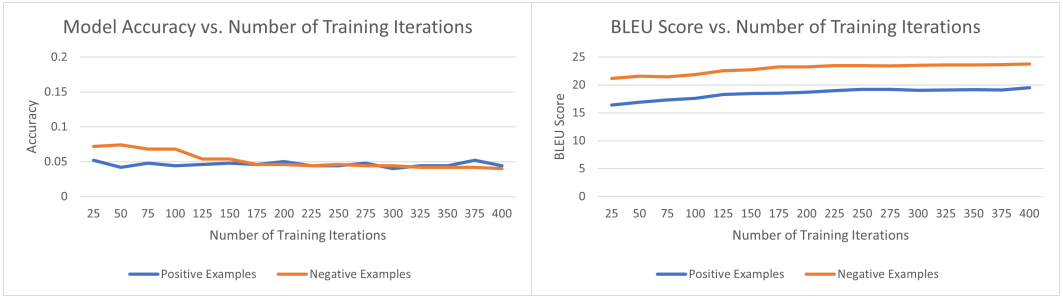


Figure 8: The proposed architecture for our second experiment comprises two parts. On the left-hand side, we have the self-construction training for the transformer. Tokens from the input text are randomly masked, and the transformer is trained to reconstruct the input text from the contaminated text and the upsampled style S of the text. On the right-hand side, we implement the entire cyclic training process proposed in Dai et al. (2019).



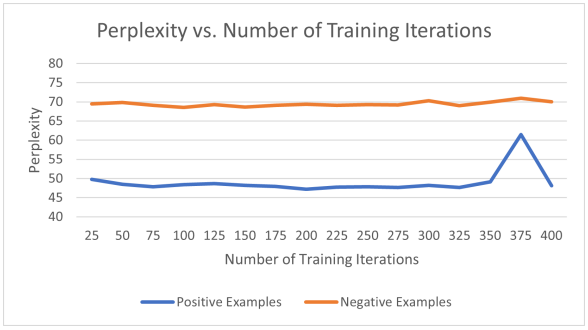


Figure 9: Results of the Experiment 2 also indicate poor performance. The accuracy is around 0.06, the BLEU score ranges between 15 and 25, and the perplexity is approximately 60. Notably, the BLEU score and the perplexity appear unchanging, indicating that the model is not learning properly.

This experiment suggests that the style upsampler, consisting of a stack of linear layers and residual blocks, does not perform well. It can be too complex to have multiple stacked layers, which increases the risk of overfitting.

5.3 Experiment 3: Expanding The Dimensionality of The Style Space

Next, we look at expanding the dimensionality of our style space. Dai et al. (2019) used discrete style spaces to represent different styles. While this may be a workable approach, we seek to use a

continuous style space instead, which should allow us to capture a wider range of styles. The intuition here is that the discrete style spaces may be good for representing discrete styles, but we wish to use the continuous space to represent more nuances or even combination of different styles. Our approach here involves taking the different discrete styles and interpolating them to obtain continuous style spaces. We can then sample from this continuous space to represent and generate text with greater granularity in style. We formulate the new transformed style S' as such:

$$S' = S + \beta T$$

where β represents the normalization constant and T represents the interpolated style space. The transformed style space can be passed into the generator such that we have $G(x, S')$. To evaluate the performance of using this interpolated style space, we perform two sub-experiments.

5.3.1 Diversity of generated text

We wish to investigate if S' would result in a generated text that is different to that from S. In other words, we want to know if $y' = y$, where $y' = G(x, S')$ and $y = G(x, S)$. We conduct this experiment and evaluate it against different values of the hyperparameter β.

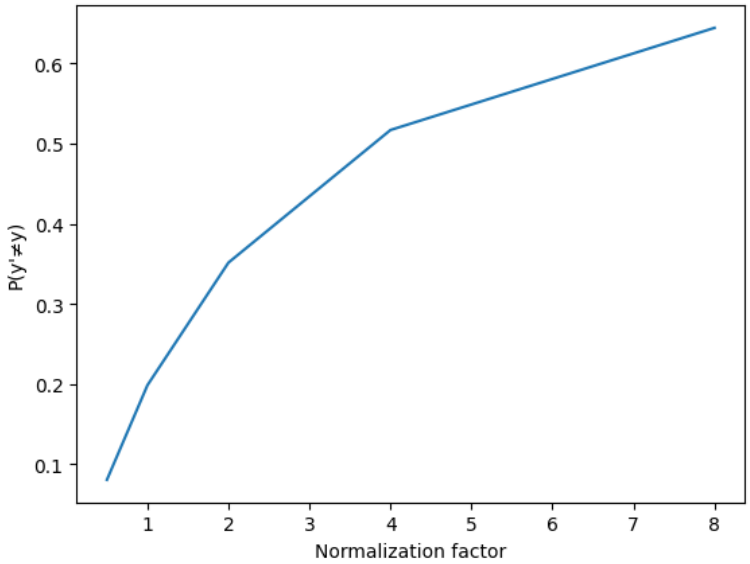


Figure 10: Graph of $Pr(y' \neq y)$ against normalization factor β

In Figure 10, we see that as we increase the normalization factor, we get more diversity in the generated text with style space S'. Intuitively this is because the style space S' is numerically more different than the original style space S. Something else to note here is also that even with $\beta = 8$, the probability that we get a generated text that differ is only approximately 0.65. This means that while the style space affects the generated text, their relationship is not completely elastic—certain style spaces can still encode similar enough style information such that the generated text are identical.

5.3.2 Styles of generated text

Next, we wish to investigate if the styles of the generated text differ. In other words, we are investigating if $style(y') = style(y)$. We do not have the ground truth for the styles of these generated text since they are generated. However, we can use the discriminator D as a proxy for this. Hence, we look at $D(y') = D(y)$ for different values of normalization constant β. In Figure 11, we see that we obtain generated text that have a different style with a normalization factor around 4. The performance here drops off for normalization factors that are too large or small. This could

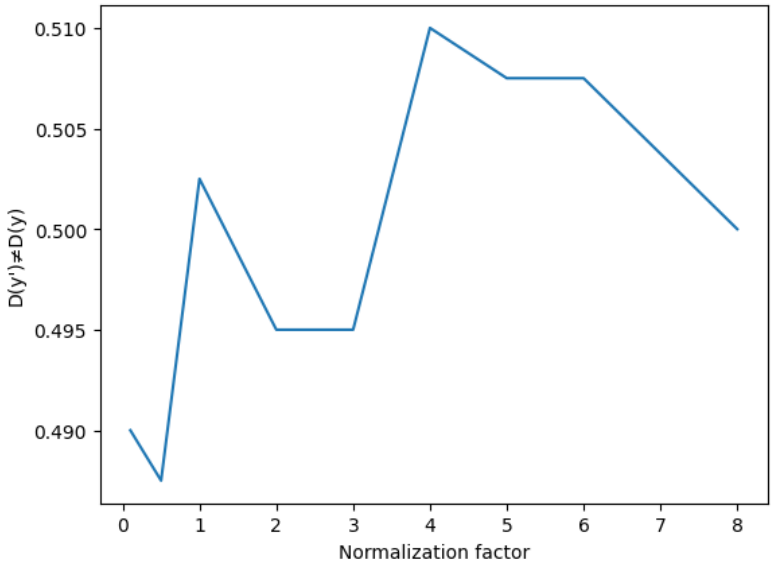


Figure 11: Graph of $D(y') \neq D(y)$ against normalization constant β

be because an excessively large or small normalization factor can lead to a style space that is either too similar to the original one or something that is not sampled well in the training. We see from the results that we obtain a different style space at a maximum probability of 0.51, which represents a fairly good performance with respect to the styles of the generated text.

5.3.3 Summary of Experiment 3

In summary, we find that transforming the style space such that we work on a continuous space can result in generated text with more granularity in styles, provided that the hyperparameter β is selected carefully. The style space transformation can provide us with generated text with styles that can be controlled more finely and with more variation.

5.4 Multi-class Experiment

In the Multi-class experiment, we look to rework the Style Transformer code that trains on binarily-classified data (positive/negative sentiment) to train on multi-class data that encompasses many complex human emotions. The dataset we we use for this experiment is the GoEmotions dataset summarize in Section 3.2. The specifications as to how the GoEmotions dataset was used for this experiment is also specified in that section.

For assessing the generator's accuracy, we are required to use an external classifier for determining the new style of the modified posts. For this classifier, we chose to use the model developed by Sanh et al. (2020) as posted on Hugging Face. We chose to use this model due to its popularity and apparent good performance in classification on the GoEmotions dataset. As can be seen in Figure 12, the accuracy did improve and seems to have mostly converged to around 0.25. Notably, this is well above a trivial performance of $\frac{1}{28} \approx 0.036$. A trivial performance of $\frac{1}{28}$ could be achieved by having the generator output a fixed sentence (ignoring the input post and style) so that the classifier always predicts the same class, which is expected to be the correct class approximately $\frac{1}{28}$ of the time since the target style is chosen uniformly at random. While the performance is well above a trivial one, the generator is still only correctly transferring the style of a post about a quarter of the time, which is quite infrequently. There is more work to be done to improve this performance, but this will have to be done in future work due to out computing and time constraints.

As can be seen in Figure 13, much like the original Style Transformer's performance illustrated in Figure 3, the perplexity is converging around a value that is higher than the initial. It is reasonable to suspect that this deterioration results from the same cause that causes the Style

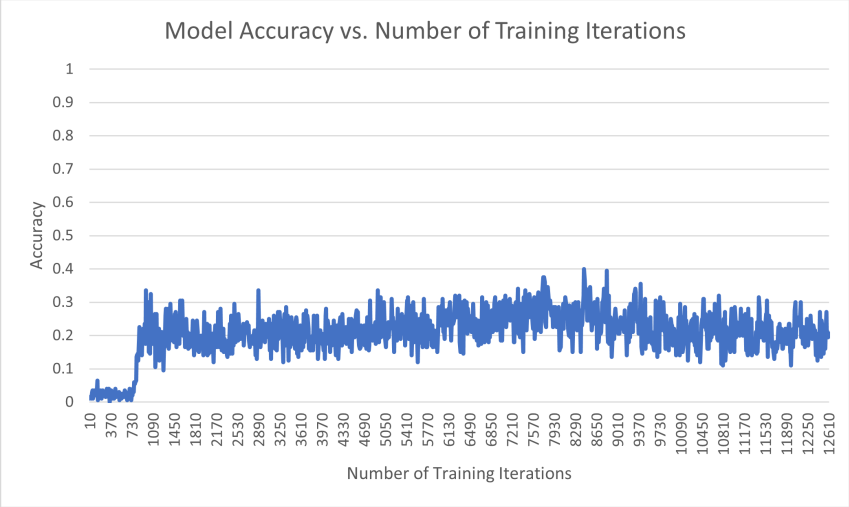


Figure 12: Plot of model accuracy as a function of number of training iterations for the Multi-class Experiment

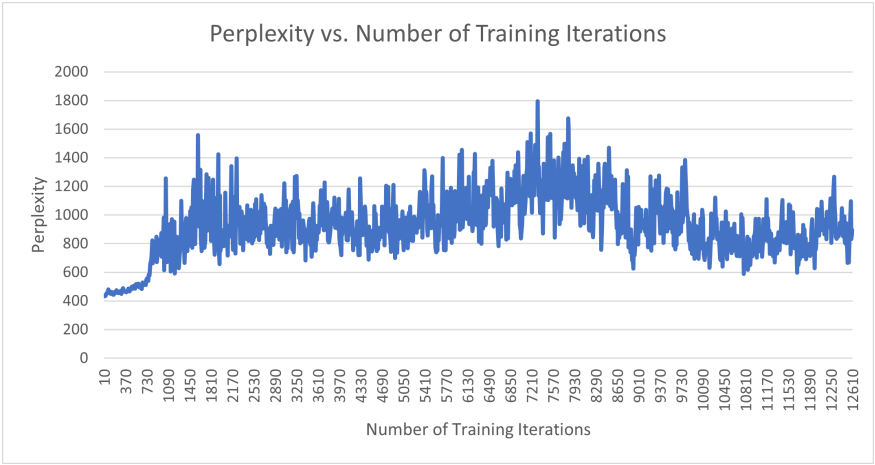


Figure 13: Plot of perplexity as a function of number of training iterations for the Multi-class Experiment

Transformer to experience this phenomenon, but the reason is still unclear. More investigation is required to uncover what might be causing this deterioration in perplexity over training.

Still, it is worth noting that both the accuracy and the perplexity experienced a steep increase after iteration 730. This is indicative of the fact that something was "learned" by the model in these iterations that was essential to the generation process regarding achieving the target style, but it is unclear why such a "learned" thing would cause the model's understanding of the probability distribution of the 5-grams in the GoEmotions corpus to deteriorate.

As mentioned in Section 3.2, we are unable to assess the BLEU score of our multi-class models due to a lack of reference sentences. This is an excellent opportunity for future work with these models. Additionally, although we had hoped to be able to acquire a human assessment of the quality of the translations, a lack of resources and time prevented us from doing so.

6 Conclusion

We have proposed a few experiments with the goal of improving explainability of arbitrary models. To do that, we first look into modifying and implementing the training process proposed by Lang et al. (2021) We also use an upsampler to upsample the styles to a higher dimension, and we implement the dual contrastive loss (Yu et al., 2021) for it. Ultimately, this experiment was not successful and the trained model did not generate text that had a different style. We postulate that this may be due to our loss design and how we fed the true label into the transformer while ignoring other styles in the training process. Next, we looked into modifying the model in a different way, where we maintained the circular training structure in Dai et al. (2019) but removed the contrastive learning approach. We also use a style upsampler comprising of a stack of linear layers and residual blocks to represent a continuous style space. After training and evaluating our model, we find that the results remain unsatisfactory. Our intuition is that the style upsampler may not perform well with respect to the rest of the training process. Third, we instead interpolate between the discrete style spaces to obtain a continuous style space. We obtain fairly good results here and were able to generate text with more nuanced styles. Finally, we modified the model to work with multi-class data, which allowed us to have multi-class representations of style, rather than the binary styles in Dai et al. (2019). This experiment was also fairly successful, with the model giving us an accuracy of about 0.25 when changing the styles of a sentence. In conclusion, we obtained significant improvements over the work in Dai et al. (2019), particularly in representing styles in a continuous space and working on multi-dimensional data.

7 GitHub Repositories

You can access the code for Experiments 1 and 2 and the Multi-class Experiment from this GitHub repository. And you can access the code for Experiment 3 from this GitHub repository.

8 Future Work

For future work, one may look to further investigate the intricacies of the Style Transformer and why our reproduced results exemplify some strange phenomena. Additional future work could involve further improving the performance of Experiment 3 and the Multi-class Experiment with the GoEmotions dataset. Currently, we are quite limited by the time and computing resources that we have access to and we believe that the results may improve with additional training. We would also like to look at how we may apply our model to aid us in identifying the bias for real-world models. Finally, we also wish to develop a more advanced UI that can enable users to generate styles with different sliders for greater granularity.

9 Work Division

We split the work for all 3 parts equally among all members.

References

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 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, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1601. URL https://aclanthology.org/P19-1601.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan S. Cowen, Gaurav Nemade, and Sujith Ravi. Goemotions: A dataset of fine-grained emotions. *CoRR*, abs/2005.00547, 2020. URL https://arxiv.org/abs/2005.00547.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018. URL https://arxiv.org/abs/1812.04948.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2019. URL https://arxiv.org/abs/1912.04958.

Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T Freeman, Phillip Isola, Amir Globerson, Michal Irani, et al. Explaining in style: Training a gan to explain a classifier in stylespace. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 693–702, 2021. URL https://arxiv.org/abs/2104.13369.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1404. URL https://www.aclweb.org/anthology/D18-1404.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Ning Yu, Guilin Liu, Aysegul Dundar, Andrew Tao, Bryan Catanzaro, Larry Davis, and Mario Fritz. Dual contrastive loss and attention for gans. *CoRR*, abs/2103.16748, 2021. URL https://arxiv.org/abs/2103.16748.