Unsupervised Lyrics Style Transfer

Hammaad Adam **Sirui Hu** **Eunseo Choi** **Chenwei Wu**

Abstract

While interest in text style transfer has grown in recent years, little work has addressed its potential applications to music. In this paper, we investigate an intriguing question: what would a lyric from a Taylor Swift song look like if it were written by Drake? And what would a Drake verse look like if it were penned by Taylor Swift? This problem is challenging, as both artist have distinct styles and there is no set of parallel corpora to translate between the two. Our work uses unsupervised text style transfer to address this problem and translates each artist's lyrics into the other's style. We consider a variety of approaches from recent literature that combine a sequence-to-sequence autoencoder (with or without attention) with either a classifier or discriminator. We evaluate these models on fluency, content preservation, and style, and find that a simple model that uses a non-attention autoencoder and an adversarial discriminator achieves the best results.

1 Introduction

Text style transfer is a fast growing field that develops methods to transform sentences written in a source style to a specified target style. Many current applications of style transfer have focused on general tasks: changing from negative sentiment to positive, from formal English to informal, from impolite to polite (Jin et al., 2021). In this project, we aim to apply unsupervised text style transfer to a task not previously discussed in the literature: changing song lyrics from one artist's style to another's. As a motivating example, consider the tweet in Figure 1.

While this line was mostly tweeted as a joke, it does capture some important facets. Drake and Taylor Swift are two of the most successful artists of the 2000s; they both write about love, heartbreak, and fame, and appeal to wide, wide audiences. However, their songwriting styles are completely different, in ways that range from genre to vocabulary to sentence structure. Given that the two artists talk about similar topics in very different styles, is it possible to train an algorithm to rewrite Taylor Swift lines in Drake's lyrical flow? And can we transform Drake's raps to match Taylor's pop country vibe? In short, can we go from Champagne Poetry to champagne problems, and vice versa?

We adopt a sequence-to-sequence approach to this problem, and use an Encoder-Decoder architecture to achieve style transfer. Our baseline model–adapted from Santos et al. (2018)–combines an autoencoder with a style classifier. The autoencoder ensures content preservation, while the classifier enforces style transfer. We also consider models that (1) add attention to the decoder, and (2) replace the classifier with a pair of discriminators that distinguish between real and generated examples in each style. We consider all combinations of these methods, and evaluate the resulting models on fluency, content preservation, and style. Overall, we find that a simple model with a non-attention autoencoder and a discriminator yields the best results, producing transferred sentences that are fluent, true to the original content, and in the required lyrical style.

Other than being a fun exploration of an exciting NLP topic, we hope that our work can be used to inspire creativity in songwriting. For songwriters who write lyrics first, the style of lyrics they write can greatly influence the type of melody they write. A system that assists in style transfer of lyrics could inspire these songwriters to explore different genres and expand their creative boundaries. While a semester is not enough to come up with a functional product, we hope to lay the initial foundations of a system that can help artists in this way.

Figure 1: A motivating tweet

2 Related Work

Interest in text style transfer has exploded in recent years; Jin et al. (2021) provide an excellent summary of the field and popular approaches. In our project, we don't have access to parallel corpora (i.e. the same sentence written in both source and target styles); we will thus rely on methods of unsupervised text style transfer. In particular, we will explore the "disentanglement" approach to this problem (Jin et al., 2021), which uses an encoding decoding architecture to transform text styles. Shen et al. (2017) details a particularly compelling approach; the authors use an RNN encoder to transform the source text to a latent representation, an RNN decoder to translate the latent text to sentences in both the source and target styles, then use a binary discriminator to enforce that the decoded outputs are in the requisite styles. Santos et al. (2018) add to this method by using a single classifier instead of two discriminators, and a cycle reconstruction of the original text to ensure content preservation. Many other papers have considered similar approaches, using different encoders (Hu et al., 2018), back-translation (Prabhumoye et al., 2018), and other forms of adversarial training (Romanov et al., 2019). While existing work has considered some atypical, challenging tasks–detoxifying hate speech (Santos et al., 2018), changing political slant (Prabhumoye et al., 2018) and simplifying medical texts (van den Bercken et al., 2019)–no existing papers (to our knowledge) have considered music lyrics. In our work, we will start with a simplified version of the approach from Santos et al. (2018), and layer on additional complexity as necessary.

3 Methods

We adopt an Encoder-Decoder approach to unsupervised text style transfer. Our model takes as input a song line x_i, which is either a Taylor Swift or Drake line, and transforms it into the other artist's style. Of course, in the absence of parallel corpora, this translation is challenging, as it can't rely on simple sequence to sequence models. In this section, we describe our baseline model in detail, then discuss the key changes we intend to make to improve its performance.

3.1 Baseline Model

Given an input line x_i^j in style j–where $j = 0$ for Drake and $j = 1$ for Taylor Swift–we first extract its style-independent context z_i. We do so by passing x_i^j and its style label j through an recurrent neural network (RNN) encoder E. The output of this encoder is its final hidden state, which we denote as z_i. Now, we need this latent representation to summarize the content but not the style of the original line. To ensure that z_i captures content, we pass it through a decoder D along with its original style label j to reconstruct the line in its original form, $x_i^{j \to j}$. This first part of the model is thus an autoencoder, as it simply attempts to reconstruct the original input. We measure the error of the reconstruction using the negative log likelihood of the observed tokens in x_i^j with the predicted probabilities from the autoencoder, and denote this loss as $\mathcal{L}_{rec}(x_i^{j \to j}, x_i^j)$.

Next, we need to ensure that the information contained in z_i is sufficient to generate a line in either artist's style. We thus pass the latent representation z_i through the same decoder as before to reconstruct the line in both its original style $x_i^{j \to j}$ and

transferred style $x_i^{j \to 1-j}$. As this step involves text generation, we use self-feeding greedy decoding from the same RNN as above, using the Gumbel softmax as a differentiable approximation to the argmax operator. Now, we need to ensure that $x_i^{j \to j}$ and $x_i^{j \to 1-j}$ are in the correct styles; to do so, we pass each of these through a classifier C that predicts the style of a song line. This classifier uses the binary cross entropy loss, and is trained along with the encoder and decoder. To ensure that the classifier learns to differentiate between the actual Taylor Swift and Drake lines, we also pass it the original line x_i^j and its original label. This gives us the classification loss

$$\begin{aligned}\mathcal{L}_{class}(x_i^j) =& \mathcal{L}_{class}(C(x_i^j), j) + \\ & \mathcal{L}_{class}(C(x_i^{j \to j}), j) + \\ & \mathcal{L}_{class}(C(x_i^{j \to 1-j}), 1-j)\end{aligned}$$

Finally, we put these two losses together to obtain our final loss

$$\mathcal{L}_{total}(x_i^j) = \mathcal{L}_{class}(x_i^j) + \mathcal{L}_{rec}(x_i^{j \to j}, x_i^j)$$

We train the whole system–encoder, decoder, and classifier–together to minimize this loss using stochastic gradient descent (SGD). We now briefly discuss each of the three components, then describe updates we make to this baseline to improve performance.

Encoder Our model's encoder is an RNN with GRU cells, $E(x_i^j, j) = z_i$, which takes as input a sentence x_i^j together with its style label j and outputs z_i, a sequence of hidden states. We have included dropout in the RNN encoder for regularization. Our variable-length inputs are also padded to the same length for batching and we pack for computational efficiency.

Decoder The model's decoder is also an RNN with GRU cells. We applied two techniques to the basic decoder to reduce the instability of the generated sentences caused by training over the discrete generated words (Lamb et al., 2016; Hu et al., 2018; Shen et al., 2017; Yang et al., 2019). First, when using the decoder to reconstruct lines in their original style, we use teacher forcing to speed up convergence. Second, we apply Gumbel-Softmax distribution as continuous approximation to the output (π) of the previous layer which is in categorical distribution when training the generator and get new sample vectors (y) as the input for the next layer (Yang et al., 2019; Jang et al., 2017). The Gumbel-Softmax trick has γ, the temperature, to tune the distribution. When γ approaches infinity, the transformed output will be uniformly distributed. When γ approaches 0, the output is the one-hot representation.

$$u \sim Uniform(0, 1)$$

$$g = -log(-log(u))$$

$$y_i = \frac{exp(\frac{\pi_i + g}{\gamma})}{\sum_{j=1}^{V} exp(\frac{\pi_j + g}{\gamma})}$$

for i =1, ..., V where V is the vocabulary size.

Classifier We use a bidirectional LSTM to classify lyrics as either being from a Taylor Swift or Drake song. The classifier takes as input either a raw lyric or a sequence of predicted word probabilities from the decoder, and outputs the probability that the line was written by Taylor Swift (vs. Drake). This architecture is used both as a module within the larger baseline model, as well as to evaluate the success of our style transfer (as discussed in section 4).

3.2 Changes to Baseline Model

Now that we have described our baseline model, we describe two additional strategies that we investigate to improve performance: (1) adding attention, and (2) using a discriminator to distinguish between real examples and generated examples in a given style. Note that the discriminator can be used instead of or in addition to the classifier, while attention can be added to the decoder of any of these approaches. In this section, we discuss each of these strategies in further detail.

3.2.1 Attention Decoder

A common issue in sequence to sequence modeling is that the decoder relies entirely on the final hidden state to generate text; this dependence creates a bottleneck, and can hinder accurate reconstruction. We can directly address this issue using the attention mechanism. Specifically, we add Bahdanau Attention, also known as Additive Attention, on top of the decoder used in the baseline model. (Bahdanau et al., 2016). This addition is in line with Nogueira dos Santos et al. (2020), who found that adding an attention module dramatically improved the quality of their sentiment style transfer.

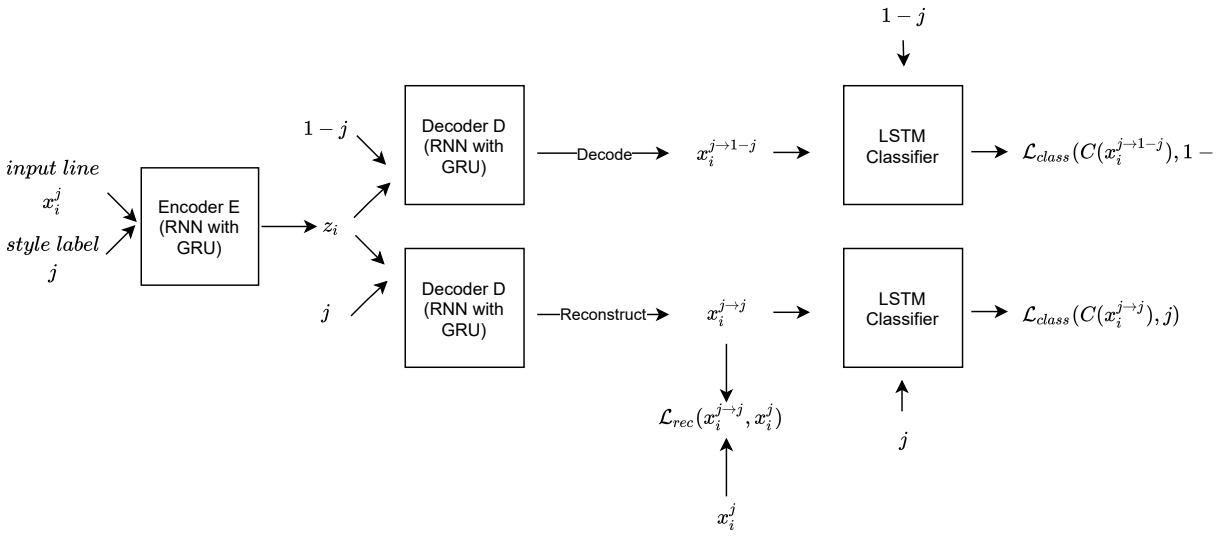

Figure 2: Model Architecture

3.2.2 Discriminator

In our baseline model, the classification loss enforces the style transfer, as the decoder learns to generate text in a way that resembles the specified style. However, this approach may produce poor results if isn't enough semantic overlap between the two styles considered. For example, Nogueira dos Santos et al. (2020) found that the classifier approach works best when changing one or two words is sufficient to change the style (e.g. changing "good" to "bad" for sentiment transfer). However, such small changes are unlikely to be meaningful in our lyrical context, as Drake and Taylor Swift may express the same thought using extremely different vocabulary.

To overcome this issue, we use the adversarial discriminator approach from Shen et al. (2017). Specifically, we train two discriminators D_0 and D_1, where D_0 distinguishes between real and fake (i.e. generated) Drake lines and D_1 between real and fake Taylor Swift lines. To make this more concrete, consider a Drake line x_i^0. As in our baseline model, we both reconstruct this line as a Drake lyric ($x_i^{0\to 0}$) and transfer it into a Taylor Swift lyric ($x_i^{0\to 1}$). We can then define the adversarial loss for this line as $\mathcal{L}_d(D_0(x_i^{0\to 0}), real) + \mathcal{L}_d(D_1(x_i^{0\to 1}), fake)$, where $\mathcal{L}_d$ is a binary cross entropy loss between the predicted discriminator label (i.e. real or fake) and the actual label. More generally, we obtain the adversarial loss for an input line x_i^j as

$$\mathcal{L}_{adv}(x_i^j) = \mathcal{L}_d(D_j(x_i^{j\to j}), real) + \mathcal{L}_d(D_{1-j}(x_i^{j\to 1-j}), fake)$$

$\mathcal{L}_{adv}$ is then subtracted from the total loss in training, that is, the style transfer model is trained to maximize the error of the discriminators. Of course, the discriminators are also simultaneously trained to minimize $\mathcal{L}_{adv}$, yielding an adversarial game between the style transfer model and the discriminators.

We make two important notes about this approach. First, though we express the discriminator input as $x_i^{j\to j}$ and $x_i^{j\to 1-j}$ above for notational convenience, we actually pass in the decoded hidden states–not the generated text–to the discriminator. This choice allows us to work with smooth latent variables instead of peaked softmax distributions, and is in line with the Professor-Forcing approach adopted in Shen et al. (2017). Second, as Shen et al. (2017) explain, this approach effectively aligns the transferred lines from one style with the true lines of the other. As the latent space for both styles are forced to be aligned, this strategy should be less prone to the semantic overlap issue the classifier approach may suffer from.

4 Evaluation

To measure the success of style transfer for each experiment, we will consider the following automatic evaluation criteria: fluency (how natural the

transferred sentence sounds), semantic (how similar input and output sentence is in its meaning), and style (whether the output sentence can cheat our pre-trained classifier to identify it same as the transferred styles). While automatic evaluation metrics are not perfect given their undesirable properties identified in previous studies (Jin et al., 2021), we checked whether any of our models are robust to all metrics, implying stability. We did not consider human evaluation given our lack of time and financial resources to run human experiments. Nonetheless, if a model does not receive good automatic metric scores, it is likely that it will not receive good evaluation scores from humans as well.

4.1 Fluency

To measure how fluency of the sentences being decoded by our model, we will score the general perplexity (PPLg) for both the decoded sentences back to its own style and to a different style. We used the pre-trained GPT-2 model to compute the perplexity score in order to reflect the fluency level in a general setting(Radford et al., 2019; Lee, 2020). The perplexity score measures how well the GPT-2 model can predict the sentences generated by our encoder.

We considered using the fixed-length perplexity score. Howeever, we ended up with the fully-factorized perplexity, after considering a number of factors: the compute time, the size of the input tokenized with GPT-2 tokenizer, and the maximum numebr of tokens GPT-2 model can take (1024 tokens).

4.2 Content Preservation

Unsupervised style transfer requires the content of the original sentence to be preserved even after going through the decoder. There are a number of metrics to measure semantic similarity. These metrics either use n-grams or word embeddings.

BLEU, for example, uses n-grams. Previous studies have reported varying correlations between BLEU and human evaluation scores (Jin et al., 2021; Liu et al., 2016; Papineni et al., 2002; Niu and Bansal, 2018). Nonetheless, BLEU is the most commonly metric used in automatic evaluation of model translation to measure how good the model is at content preservation by comparing source and target sentence pairs (Papineni et al., 2002). BERTscore, which uses contextualized word embeddings, is another popular measure to evaluate model translation (Shimanaka et al., 2019). Pre-trained Universal Sentence Encoder, SentenceTransformers with all-MiniLM-L6-v2 model and all-mpnet-base-v2 model, and DistilRoBERTa model can be used to measure the cosine similarity between the input sentence and the decoded sentence as well (CSg and CSuse)(Fu et al., 2017; Lee, 2020). Universal Sentence Encoder is developed by Google with a pre-trained model available in TF-hub (Cer et al., 2018). The pre-trained SetenceTransformers are downloaded from HuggingFace according to which the pre-trained all-mpnet-base-v2 model generates better quality than all-MiniLM-L6-v2 which is relatively faster. DistilRoBERTa model is basically a smaller RoBERTAa model which is faster but still retain most of the language understanding capabilities. These pre-trained models can capture a more comprehensive context representation of lyrics than models trained only with data made up of lyrics from two artists like TF-IDF and BoW.

We considered BLEU, Bert, and cosine similarity between the input sentence and the output sentence to quantify this semantic preservation. For cosine similarity, we decided to incorporate all four encoders mentioned above in case there is any obvious difference in any one of the models because the context to train these models are different from lyrics.

4.3 Style

We evaluated style accuracy with a pre-trained classifier independent from the one used in training. We used a BiLSTM classifier to measure the percentage of correctly labeled lines where labels are binary (either the style of Taylor Swift or Drake). This accuracy is the key metric by which we judged the efficacy of the transfer. If our style transfer is successful, then the classifier should label the style transferred lines with the target style, while still labelling the reconstructed lines with the original style. We thus measured the AUC of our classifier on the reconstructed and transferred lines separately to evaluate the style change.

5 Experiments

5.1 Data

Our dataset contains $17,997$ lines of lyrics from 492 Drake's songs and $17,997$ lines from 479 Taylor Swift's songs. To construct this dataset, we used the Genius API to obtain lyrics for every song by these two artists. We then split songs up into lines

by splitting on the newline character. We tokenized these lines using a simple word tokenizer, keeping punctuation and transforming all words to lower case. The vocabulary extracted from this corpus yields $V = 12,850$ unique tokens, including the start of line, end of line, padding, and unknown word tokens. We excluded any lines fewer than 5 tokens or longer than 15 tokens; including the appended start and end of line tokens, this implies a maximum line length L of 17. In our experiments, we split this data set into training, validation, and test sets using a 7:1:2 split.

5.2 Models

There are several components to our style transfer model which we evaluate. The first component is a simple autoencoder to make sure our sequence to sequence approach is feasible in this setting. We do so both with and without attention, and present results in section 6.1.1. Next is the BiLSTM classifier to distinguish between raw Drake and Taylor Swift lyrics. Since classifier is used in both our transfer model and in evaluation, it is important to ensure that this task is actually possible. We describe the classifier's performance in section 6.1.2. The final step is training and evaluating the actual style transfer models. Overall, we consider six models with different combinations of attention, classifier, and discriminator modules. First, we consider our baseline model from section , which uses a non-attention autoencoder with a classifier. We then consider a model that replaces the classifier with a discriminator, as well as one that uses both a classifier and a discriminator. Last but not least, we consider each of these three approaches adding an attention module to the decoder.

6 Results and discussion

6.1 Quantitative Evaluation

6.1.1 Autoencoder

We first compared the BLEU and BERTscore of an autoencoder and an autoencoder with attention (Table 1). The autoencoder with attention had the highest BERTscore. Both the autoencoder model and the autoencoder with attention had very low classification accuracy (about 0.1, which is worse than random). Such low scores are unsurprising given that the autoencoder by itself is reconstructing sentences and attention is just copying and retaining local content from the source sentence. Even though both models do not transfer style, it is essential to review their performances to make sure our models were preserving content. Results of these two models also illustrate why we must look at all three metrics (content preservation, fluency, and style) when evaluating the quality of transferred sentences.

6.1.2 Classifier

Our approach uses bi-directional LSTM classifiers to both train and evaluate the style transfer models. It is thus vital to confirm that it is possible to build such a classifier to distinguish between Drake and Taylor Swift lyrics. To see if our classifiers' performance could potentially be on par with the human evaluation and also to see if one style is harder to distinguish, we tested respectively on mixed styles real sentences test set, All Drake lyrics test set and All Taylor test set.

Our classifier has a ROC AUC score for real sentences (both Drake and Taylor) of 0.923. For classifying Drake individually it has an ROC AUC score of 0.808, and for Taylor Swift it has an ROC AUC score of 0.900. As for accuracy it has 0.807 on Drake and 0.900 for Taylor. Thus the Drake lines are indeed harder to classify compared to Taylor Swift lines, an interesting fact we could also tell from the quality of sentences generated when doing the qualitative evaluation.

6.1.3 Style Transfer

Fluency Pre-trained GPT2 model is used to calculate the perplexity score of the decoded sentences. We have the perplexity scores of the reconstructed sentences generated from the autoencoder model as our baseline of the perplexity score. The baseline is already high with 163.645 for autoencoder only and 176.212 for autoencoder with attention layer. This can be caused by the shortness of the lyrics in comparison and how lyrics are written is different from the data source that are used to trained the GPT2 model.

The perplexity scores range from 160 to 900 which are mostly higher than the baseline score we set and show the potential problem of the generated sentences' fluency and grammatically correctness. From Table 2, we can see the perplexity score is generally better without attention than with attention. This pattern suggests that some grammatical error might be inevitable in order to preserve the score of style classification.

Content preservation As mentioned in section 4.2, the content of the lyrics should be preserved

in the transferred sentence. We use cosine similarity, BLEU and BERTscore to measure the content preservation.

It is worth noting that different semantic similarity measures rank each model differently in Table 2. For example, the autoencoder model with attention and classifier does worst in preserving content while the autoencoder with classifier and discriminator does best when the criteria is measured with cosine similarity using DistilRoBERTa model. When semantic is measured using BLEU, the autoencoder with classifier and discriminator does best and the autoencoder with a combination of attention, classifier, and discriminator does worst. Last but not least, the model using autoencoder with attention and discriminator does best in terms of BERTscore. For selecting the "best" model out of six candidate models, we used BERTscore as our main measure since it uses contextualized word embeddings and is not as susceptible to synonyms like BLEU. Previous studies show that BLEU is also bad at encouraging output diversity and identifying semantically more words (Krishna et al., 2020). Nonetheless, industry practitioners testing and selecting from a set of models should be wary of these score differences across metrics that claim to measure semantic similarity between sentences.

Style The AUC scores for the reconstructed sentences are much higher than the transferred sentences which is expected because the transferred sentence should have some preserved contexts that are not present in the training set of lyrics. Comparing classification accuracy of the transferred sentences in Table 3, the model using autoencoder with a combination of attention, classifier, and discriminator gives the highest AUC for the transferred sentences. Therefore, in terms of the fooling the classifer, the last model outperforms any other models. However, this might not be the case if we have a human evaluator to evaluate the style of the transferred sentences.

Balancing three metrics Overall, Tables 2 and 3 show that the autoencoder with a discriminator does the best job at balancing the three criteria. All other models fail significantly at one of the three tasks. For example, the attention autoencoder with discriminator and classifier does exceedingly well on the style metric (AUC), but produces low quality sentences in terms of fluency. The autoencoder with discriminator does sufficiently well in all three metric categories; this conclusion is further supported by the qualitative evaluation in section 6.2.

Before proceeding, we make two important notes. First, we think that discriminator's better performance compared to that of our classifier is partly due to the nature our style transfer task. As Santos et al. (2018) point out, the classifier approach works better in easier tasks that requires small change to a sentence instead of transferring the whole sentence. For example, the autoencoder with a classifier would have work better for sentiment style transfer from "I am so stressed this week" to "I am so happy this week," where a change in one word would do the job. Lyrics style transfer requires a more subtle transfer of style. Second, from our initial analysis of these models, we learn that adding more complexity does not always give the best result. Adding extra modules often just convoluted the final result, and made the models more likely to fail in one of the three key evaluation categories. This finding is in line with previous studies that have also reported simpler models outperforming more complex ones (Krishna et al., 2020).

6.2 Qualitative Evaluation

While automatic metrics have provided a more objective understanding of how good our model performs in terms of content and fluency, they may have their limitations. For example, style classifiers can not possibly outperform human evaluation. Thus, to better understand how well the sentences were reconstructed and transferred, we conducted extensive qualitative evaluations for lyrics from both Drake and Taylor Swift, with respect to the grammatical correctness, style transfer strength and similarity of reconstruction.

Table 4 and 5 showcase two examples from each styles for each of our model architectures. Table 4 is an example of Taylor to Drake style transfer with the original sentence being "Cause you were Romeo, I was a scarlet letter." The first autoencoder and second autoencoder with attention simply reconstruct the original sentence. This is unsurprising since no style transferring loss were adding to the model. As we start to add classifier and discriminator to the model, the styles are starting to transfer to Drakes'. Our best example here is the autoencoder with discriminator that transfers the original sentences into "Silk, tasteful man, then I was a prescription you", which notably matches Drake's rap

		Content		Fluency
Model	**CS_distRob**	**BLEU**	**BERTscore**	**PPL_gpt**
Autoencr	0.619	72.613	0.918	163.645
Autoencr_Atten	0.906	71.817	0.951	176.212

Table 1: Sanity check of our auto-encoders

		Content		Fluency
Model	**CS_distRob**	**BLEU**	**BERTscore**	**PPL_gpt**
Autoencr_Classif	0.476	71.329	0.918	163.645
Autoencr_Discrim	0.424	70.62	0.862	239.02
Autoencr_ClassifDiscrim	0.484	71.775	0.883	178.259
Autoencr_AttenClass	0.1673	63.132	0.797	901.088
Autoencr_AttenDiscrim	0.822	68.658	0.933	253.74
Autoencr_AttenClassDiscrim	0.202	61.316	0.800	821.625

Table 2: Overall Evaluation

	Style		
Model	**AUC_real**	**AUC_recons**	**AUC_tsf**
Autoencr_Class	0.923	0.908	0.629
Autoencr_Discrim	0.923	0.901	0.695
Autoencr_ClassDiscrim	0.928	0.896	0.59
Autoencr_AttenClass	0.926	0.779	0.793
Autoencr_AttenDiscrim	0.923	0.919	0.105
Autoencr_AttenClassDiscrim	0.918	0.855	0.809

Table 3: Overall Evaluation cont.

style in both punctuation and lyrics. We also observed that the addition of the attention mechanism improves content preservation of the original sentences. The addition of classifier and discriminator, especially discriminator, significantly strengthens the style transfer. While we expected the autoencoder with attention, classifier and discriminator to be perform the best, our qualitative example shows that the model does well in style transfer but does quite poor in generating grammatically correct sentences.

Table 5 is also a good example of Drake lyrics undergoing style transfer to Taylor Swift. The vanilla autoencoder does not do well in reconstructing the raw sentences. Adding attention improves the reconstruction component. The performance of our candidate models remains comparable to our quantitative analysis. The autoencoder with discriminator still performs the best in style transfer while outputing a readable sentence. The autoencoder with attention, classifier and discriminator, albeit having high score in style transfer, fails to deliver grammatically sensible lyrics.

7 Conclusion and Future Work

In this work, we model style transfer between Taylor Swift's and Drake's lyrics. We employ the encoder-decoder approach to do unsupervised style transfer, and we present results from our experiments testing different combinations of autoencoders. We show that the autoencoder with discriminator shows the best performance for the purpose of our work, though these are only initial results.

In future work, we would like to consider the following:

7.0.1 Improvement and aggregation of metrics

We chose the "best" model from the set of our candidate models by comparing performance across three independent metrics, yet our selection method could be more rigorous. Our results also showed that an improvement in one style transfer evaluation criterion can lead to decrease in another. Thus, we see value in aggregating our three style transfer metrics to see how combination of three scores would affect the performance ranking of our candidate models. This would be important to avoid selecting a model that only optimizes for a subset of metrics among all that we are interested. For example, as we described in section 6.1.1, a system optimizing for content preservation and perplexity would wrongly assume that an autoencoder with attention is the best model for style transfer when the model is only just copying sentences. More work is needed to identify the best way to jointly optimize all metrics.(Krishna et al., 2020)

7.0.2 Human Evaluation

Human evaluation is the gold standard for evaluating the quality of transferred sentences. If we have more time and resources in the future, we would like to recruit and assign a number of human annotators (preferably fans of Taylor Swift and/or Drake). These annotators would evaluate 100 outputs from each of our candidate models for both style transfer directions (Swift to Drake and Drake to Swift).

7.0.3 Transformer Encoder-Decoder

Transformers allow a shorter path between any combination of sequence positions and make it easier to learn long-range dependencies within the sequence. Unlike earlier self-attention models that still rely on RNNs for input representations, the transformer model is solely based on attention mechanisms without any convolutional or recurrent layer (Vaswani et al., 2017). We hope to improve our model performance in the future by introducing transformer architectures.

7.0.4 Style transfer between artists and genres

Notably, the autoencoder with attention and classifier for both style transfers generates a repetition of words like "mischief" and "forever" in Table 5. This could be due to the limitation of the size of our data so that model is dominated by several signature words of each artist. In the future, we could try style transfer on more artists and even different music genres with a larger corpus to train on.

Model	Reconstruct	Transferred
Autoencr	cause you were romeo, i was a scarlet letter	Cause you were romeo, i, i was a way
Autoencr_Atten	cause you were romeo, i was a scarlet letter	cause you were romeo, i was a scarlet letter
Autoencr_Class	cause you were romeo, i was a scarlet times	Cause you you were romeo damn a race i i
Autoencr_AttenClass	lead daddy you lead lead stand june lead lead lead 27th lead daddy daddy stand you	lead lead lead lead east lead lead east baby
Autoencr_ClassDiscrim	cause you were romeo, i was a scarlet letter	cause you were romeo, i was a scarlet letter
Autoencr_AttenClassDiscrim	august slipped a little a a crown a a crown a boy	drizzy forseen was a clown
Autoencr_Discrim	rough you i was a, i was lost a	Silk, tasteful man, then I was a prescription you
Autoencr_AttenDiscrim	cause you were romeo, i was a scarlet letter	cause you were romeo, i was a scarlet letter

Table 4: Qualitative Evaluation: Taylor to Drake
(**Source**: Cause you were Romeo, I was a scarlet letter)

Model	Reconstruct	Transferred
Autoencr	i taste and yall stare in your black	i taste when and and in your eyes
Autoencr_Atten	i taste pain and regret in your sweat	i taste pain and regret in your sweat
Autoencr_Class	i purchase and and in in is	i watched looked and all your your songs
Autoencr_AttenClass	lead of east treatin of east of doing	lead daddy of lead lead of
Autoencr_ClassDiscrim	i wine-and-dine here covered in your your reflection	i here in here and your highest
Autoencr_AttenClassDiscrim	i ever your friends back forever back your friends back back helplessness your helplessness your forever	mischief mischief boy forever forever forever boy forever forever forever boy forever boy forever giftwrapped forever
Autoencr_Discrim	i run your expectations and in in your head	i m down in your car and smile
Autoencr_AttenDiscrim	i taste pain and regret in your sweat	i taste pain and regret in your sweat

Table 5: Qualitative Evaluation: Drake to Taylor
(**Source**: I taste pain and regret in your sweat)

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate.

Laurens van den Bercken, Robert-Jan Sips, and Christoph Lofi. 2019. Evaluating neural text simplification in the medical domain. pages 3286–3292.

Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2017. Style transfer in text: Exploration and evaluation.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2018. Toward controlled generation of text.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2021. Deep learning for text style transfer: A survey.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating Unsupervised Style Transfer as Paraphrase Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–762, Online. Association for Computational Linguistics.

Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks.

Joosung Lee. 2020. Stable style transformer: Delete and generate approach with encoder-decoder for text style transfer.

Chiao-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*.

Tong Niu and Mohit Bansal. 2018. Polite dialogue generation without parallel data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

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 of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Alexey Romanov, Anna Rumshisky, Anna Rogers, and David Donahue. 2019. Adversarial decomposition of text representation.

Cicero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through ranking by generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1722–1727, Online. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. *arXiv preprint arXiv:1805.07685*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. *arXiv preprint arXiv:1705.09655*.

Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. 2019. Machine translation evaluation with bert regressor.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2019. Unsupervised text style transfer using language models as discriminators.

Model	CS_univ	CS_tsfmr	CS_mpnet	CS_distRob
Autoencr	0.637	0.648	0.599	0.619
Autoencr_Classif	0.501	0.5131	0.450	0.476
Autoencr_Discrim	0.438	0.468	0.396	0.424
Autoencr_ClassifDiscrim	0.508	0.523	0.459	0.484
Autoencr_Atten	0.879	0.918	0.901	0.906
Autoencr_AttenClass	0.083	0.152	0.153	0.167
Autoencr_AttenDiscrim	0.7747	0.847	0.809	0.822
Autoencr_AttenClassDiscrim	0.135	0.185	0.195	0.204

Table 6: Appendix 1: Difference in cosine similarity scores

		Content		Fluency
Model	CS_distRob	BLEU	BERTscore	PPL_gpt
Autoencr	0.648	72.8	0.924	159.559
Autoencr_Class	0.499	71.45	0.884	250.026
Autoencr_Discrim	0.448	70.816	0.867	237.23
Autoencr_ClassDiscrim	0.507	71.902	0.887	181.374
Autoencr_AttenC	0.908	71.825	0.952	176.701
Autoencr_AttenDiscrim	0.1705	63.426	0.798	903.917
Autoencr_AttenDiscrim	0.8251	68.68	0.9338	249.96
Autoencr_AttenClassDiscrim	0.2351	63.134	0.805	597.803

Table 7: Appendix 2: Evaluation of reconstructed sentences

		Content		Fluency
Model	CS_distRob	BLEU	BERTscore	PPL_gpt
Autoencr	0.590	0.911	72.42	167.731
Autoencr_Class	0.4518	71.208	0.875	255.4219
Autoencr_Discrim	0.400	70.431	0.8578	240.8043
Autoencr_ClassDiscrim	0.46	71.648	0.879	175.145
Autoencr_Atten	0.906	71.825	0.9511	175.723
Autoencr_AttenClass	0.164	62.837	0.796	898.260
Autoencr_AttenDiscrim	0.8197	68.637	0.9331	257.53
Autoencr_AttenClassDiscrim	0.170	59.499	0.796	1045.448

Table 8: Appendix 3: Evaluation of transferred sentences