Style versus Content: A distinction without a (learnable) difference?

Somayeh Jafaritazehjani, Gwénolé Lecorvé, Damien Lolive, John D Kelleher

Style versus Content: A distinction without a (learnable) difference?

Somayeh Jafaritazehjani
Univ Rennes, CNRS, IRISA
TU Dublin, ADAPT Centre
somayeh.jafaritazehjani@irisa.fr

Gwénolé Lecorvé
Univ Rennes, CNRS, IRISA
gwenole.lecorve@irisa.fr

Damien Lolive
Univ Rennes, CNRS, IRISA
damien.lolive@irisa.fr

John D. Kelleher
TU Dublin, ADAPT Centre
john.d.kelleher@tudublin.ie

Abstract

Textual style transfer involves modifying the style of a text while preserving its content. This assumes that it is possible to separate style from content. This paper investigates whether this separation is possible. We use sentiment transfer as our case study for style transfer analysis. Our experimental methodology frames style transfer as a multi-objective problem, balancing style shift with content preservation and fluency. Due to the lack of parallel data for style transfer we employ a variety of adversarial encoder-decoder networks in our experiments. Also, we use a probing methodology to analyse how these models encode style-related features in their latent spaces. The results of our experiments which are further confirmed by a human evaluation reveal an inherent trade-off between the multiple style transfer objectives and indicate that style cannot be usefully separated from content within these style-transfer systems.

1 Introduction

Style transfer can be defined as a Natural Language Generation (NLG) task where an input word sequence (a text, a sentence, an utterance...) from an original style is rephrased in order to fit a target style while keeping its semantic information, leading to a *style-transferred* sequence. This task can be applied to many facets of the natural language: sentiment, complexity, authorship, formality, and so on. The primary use-case of textual style transfer is to improve language understanding between humans or between humans and machines by adapting messages to the social context or abilities of an interlocutor. However, work on style transfer can also provide insight into language more generally, for example by providing insight into the relationship between style and content. Indeed, a pre-requisite of style transfer is the presumed ability to separate style from content, and so research on style transfer can inform whether computational models can usefully learn this distinction.

Research on style-transfer often focuses on one specific aspect of textual productions, (e.g., preserving the meaning of the original sequence) and disregards the others (e.g., how present is the target style in the generated output, and the fluency of the generated sentences). However, this paper highlights the importance of evaluating style transfer systems in terms of multiple objectives where fitting the target style is counter-balanced by the needs to generate a linguistically fluent output sequence and at the same time preserving the original meaning. Furthermore, based on an analysis of a number of evaluation experiments of style-transfer systems, including a probing experiment that analyses the latent space of these systems, we identify an inherent trade-off in style-transfer between transfer strength, fluency and content preservation. We conclude that style and content cannot be usefully disentangled by these systems (at least not for the task of sentiment transfer), and so a holistic perspective on style and content that focuses on finding a suitable trade-off for the target application is likely the best approach to developing a useful style-transfer system.

The paper structure is as follows. Section 2 reviews related work on style-transfer and distinguishes between: (i) systems that conceive of style as being encoded in a set of explicitly identifiable features that can be removed and replaced, and (ii) systems that do not explicitly identify stylistic features. Section 3

introduces the style transfer evaluation objectives we use; Section 4 describes the models we evaluate in our experiments, and Section 5 describes the dataset and our experimental methodology. In Section 6 we evaluate the style-transfer models, and in Section 7 probe their representations of the input to examine what stylistic information they encode. In Section 8 we set out our conclusions and future work.

2 Literature review

To deal with textual style transfer, a first group of approaches consider a two-step process where markers of the source style are removed before generating the target style output sequence. Marking and removing the original style features can be achieved through frequency-based methods (Li et al., 2018) or neural networks that integrate attention modules (Leeftink and Spanakis, 2019). The generation step relies on techniques such as the retrieval of textual segments from a corpus of the target style (Ramos, 1999; Leeftink and Spanakis, 2019), or employing neural network generation techniques (Li et al., 2018).

The second group of approaches which have been mostly implemented in the literature focus on end-to-end learning strategies to frame the task of style transfer. End-to-end learning approaches enable learning of a latent representations of an input (Kelleher, 2019), encoding potentially both the input content and style. Historically, such style transfer approaches have been inspired by the Neural Machine Translation techniques, basically a sequence-to-sequence encoder-decoder architecture based on recurrent neural networks (Sutskever et al., 2014; Bahdanau et al., 2015). Ma and Sun (2017) applied this approach to text simplification and summarization (considering the style here as "verbosity"). The key idea is that the text embedding vector generated by the encoder is a style-free representation of the input text (Rabinovich et al., 2017). A major limitation of this approach is the need for parallel data. In some scenarios this data challenge can be addressed by either relying on intermediate resources–e.g., using "zero-shot translation" (Johnson et al., 2017; Carlson et al., 2017)–or monolingual data–e.g. using "back translation" (Sennrich et al., 2016; Prabhumoye et al., 2018).

Alternatively, Generative Adversarial Networks (Goodfellow et al., 2014) have been implemented for style transfer, removing the need for parallel data. These models include a generator block which generates two sequences for each input sequence (a style transferred sequence and a reconstructed sequence), and a discriminator block (a classifier) which tries to guess whether a given sequence is style transferred or reconstructed. Generators are usually standard encoder-decoders (Shen et al., 2017; Singh and Palod, 2018; Fu et al., 2018; Romanov et al., 2019), but various refinements have been proposed. For instance, several generators (decoders) can be used, one for each style (Fu et al., 2018), or a variational encoder can be used for encoding (Hu et al., 2017; John et al., 2018). In most cases, the decoders are conditioned on a latent representation (embedding) of the input sequence concatenated with an embedding of the target output style. The concept of discriminator can also be extended, for instance in order to force encoded representations of the input to either focus on its style or meaning or both (Romanov et al., 2019; John et al., 2018).

3 Evaluation aspects

This section introduces the evaluation methodologies we use in our experiments. Each of these methods is taken from the literature and each focuses on a different aspect of textual style transfer: content preservation, style transfer strength and fluency.

Style shift power Style shift power investigates how well the output of a model fits with the target style. One approach is to employ a pre-trained classifier to measure the percentage of the style-shifted sequences which are labeled with the desired style (Fu et al., 2018; Li et al., 2018; Leeftink and Spanakis, 2019; Singh and Palod, 2018; Prabhumoye et al., 2018; Shen et al., 2017; John et al., 2018; Hu et al., 2017).

Here, we employ the same classifier type as (Shen et al., 2017), the TextCNN model (Kim, 2014) and train it on the Yelp dataset. It achieves an accuracy of $\%97$ on predicting sentiment (positive and negative classes) on the test set of the Yelp dataset.

Content preservation Many approaches, usually inherited from historical NLG tasks, have been employed to compare how well the generated output preserves the meaning of the original sequence.

We compute a content preservation rate between an input sequence and a generated sequence as the cosine similarity between their embedded representations. The process we use to generated the embedding of a sequence follows the method proposed by Fu et al. (2018). Given a sequence of words $\mathbf{x} = [x_1 \dots w_N]$, each token x_i is represented by an embedding $e_i \in \mathbb{R}^{100}$. These embeddings are generated using a pre-trained 100-dimensional GloVe model (Pennington et al., 2014). Then, we calculate the min, mean and max pooling of these word embeddings, namely m, μ, M as follows:

$$m = \left(\min_{1 \leq i \leq N} e_{i,j} \right)_{1 \leq j \leq 100} \quad (1)$$

$$\mu = \left(\frac{\sum_{1 \leq i \leq N} e_{i,j}}{N} \right)_{1 \leq j \leq 100} \quad (2)$$

$$M = \left(\max_{1 \leq j \leq 100} e_{i,j} \right)_{1 \leq i \leq N} \quad (3)$$

The embedding of the sequence $\mathbf{x}$ is then created by the concatenation of m, μ and M. In practice, sentiment markers[1] are removed from the sequence $\mathbf{x}$ to make sure the content preservation metric indeed measures the content similarity. Finally, when processing a dataset, content preservation is averaged over all its sentences.

Fluency Following Zhao et al. (2018) and John et al.(2018), we measure the fluency of an output sequences using language model perplexity. For our experiments, we trained an RNN-based language model, consisting of single-layer RNN with GRU cells (Chung et al., 2014), on the Yelp dataset. Again, tokens are embedded using a 100-dimensional layer which was initialized at training time with a pre-trained GloVe model (Pennington et al., 2014).

4 Models

This section introduces three models employed in this work to deal with textual style transfer: a state-of-the-art adversarial network, which we refer to as the *base model* (Section 4.1); an encoder extension of the base model employing a novel type of encoder (in Section 4.2); a generator extension of the base model employing multi-generators (Section 4.3).

4.1 Base model

The base model is the adversarial style transfer model proposed by Shen et al. (2017). As depicted in Figure 1, this model is composed of the following components:

Encoder An encoder $\mathbf{E}$ which reads a sequence $\mathbf{x}$ of style $s \in \{1, 2\}$, denoted as $\mathbf{x}^{(s)}$ and outputs an embedded representation denoted as $\mathbf{z}$.

Generator A generator $\mathbf{G}$ which is initialized with $\mathbf{z}$ and the desired output style s. The output is a sequence of words where the content is supposed to be the same as in $\mathbf{x}$ and style should be s.

$\mathbf{E}$ and $\mathbf{G}$ are based on a standard sequence-to-sequence architecture (Sutskever et al., 2014). In the case where s is the original style, the model forms an auto-encoder. During training, $\mathbf{G}$ is used twice for each input: once to generate a reconstructed sequence (i.e., in the same style as the input), and once to generate a style-transferred sequence (i.e., in the opposite style to the input).

Discriminators Two discriminators, $\mathbf{D_1}$ and $\mathbf{D_2}$, one for each style s_1, and s_2. The style-specific discriminator, $\mathbf{D_s}$ takes a generated sequence and predicts the probability that this sequence has the style s. In other words, depending on whether s is the same as the original style of the input sequence or different from it, it predicts whether the style of the generated sequence has been "preserved" or "transferred".

[1] The sentiment dictionary is the one from Fu et al. (2018)

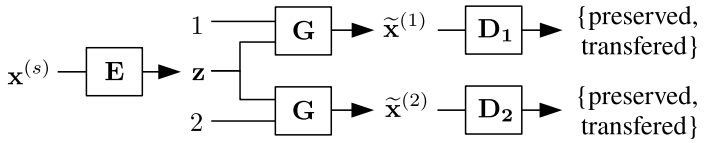

Figure 1: Given input sequences $\mathbf{x}$ with an original style s, $\mathbf{E}$ encodes $\mathbf{x^{(s)}}$ in a latent representation $\mathbf{z}$. $\mathbf{G}$ is initialized with $\mathbf{z}$ and the desired output style, 1 or 2, leading to $\tilde{\mathbf{x}}^{(1)}$ or $\tilde{\mathbf{x}}^{(2)}$, respectively. Discriminators $\mathbf{D_1}$ and $\mathbf{D_2}$ aim at detecting whether the styles of their inputs (i.e., the sequences generated by $\mathbf{G}$) are "preserved" or "transferred" relative to s.

This architecture is designed to be trained in an adversarial manner: the generator, $\mathbf{G}$ when conditioned with style s, aims at generating a sequence that should convince the discriminator, $\mathbf{D_s}$, that the generated sequence has the style s, whereas $\mathbf{D_s}$ aims at detecting style-shifted sequences.

The training process involves processing two differently styled input sequences in parallel $\mathbf{x}_1^{(s_1)}$ and $\mathbf{x}_2^{(s_2)}$, where $s_2 = \overline{s_1}$. This results in four generated output sequences, one output sequence per style for each input sequence, leading to two reconstructed sequences of $\widetilde{\mathbf{x}}_1^{(s_1)}$, $\widetilde{\mathbf{x}}_2^{(s_2)}$, and two style-transferred sequences $\widetilde{\mathbf{x}}_2^{(s_1)}$, and $\widetilde{\mathbf{x}}_1^{(s_2)}$.

The adversarial training relies on 3 elementary losses. For simplification, we assume that training data has the two sequences $\mathbf{x}_1^{(s_1)}$ and $\mathbf{x}_2^{(s_2)}$.

- The reconstruction loss, $\mathcal{L}_{rec}$, is the negative log probability of the training data, and is computed as the cross-entropy between an original sentence, and the reconstructed sentence (equation 4). Backpropagation is carried out for the couple ($\mathbf{E}$, $\mathbf{G}$) using this loss.

$$\mathcal{L}_{rec} \quad = \quad -\log \Pr(\widetilde{\mathbf{x}}_1^{(s_1)}|\mathbf{x}_1^{(s_1)}) - \log \Pr(\widetilde{\mathbf{x}}_2^{(s_2)}|\mathbf{x_2}^{(s_2)}) \tag{4}$$

- The adversarial loss is defined such that minimising this loss requires minimising the precision of the discriminators in detecting that the input sentence has been transferred. Equation 5 lists this loss, note that it is computed solely on the transferred sequences. This loss is the log of the inverse probability predicted by the discriminator and so minimising this measures causes the generation block to generate style-shifted sequences with lower probability of being detected as transferred. L_{adv,s_2} is defined symmetrically to Equation 5.

$$\mathcal{L}_{adv,s_1} \quad = \quad \log(1 - D_{s_1}(\widetilde{\mathbf{x}}_2^{(s_1)})) \tag{5}$$

- Equation 6 lists the discrimination loss, $\mathcal{L}_{Ds}$, for a given style s. This loss is defined as the binary cross-entropy over "transferred" and "preserved" classes where the true labels of style-shifted and reconstructed outputs are considered as "transferred" and "preserved" respectively. For each style s, we train $\mathbf{D}_s$ to maximize the probability of assigning these true labels to the output sequences by minimizing this loss.

$$\mathcal{L}_{D_s} \quad = \quad -\log D_s(\widetilde{\mathbf{x}}_1^{(s)}) - \log(1 - D_s(\widetilde{\mathbf{x}}_2^{(s)})) \tag{6}$$

The backpropagation for the encoder-generator couple ($\mathbf{E}$, $\mathbf{G}$) is carried out using the following:

$$\mathcal{L}_{total} = \quad \mathcal{L}_{rec} + \mathcal{L}_{adv,s_1} + \mathcal{L}_{adv,s_2} \tag{7}$$

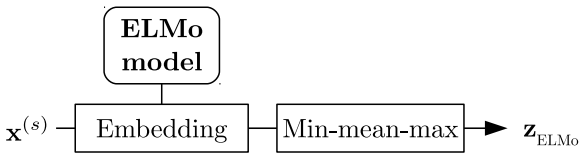

Figure 2: Detail of the ELMo-based encoding.

4.2 Encoder variant: ELMo-based encoder

ELMo is a state-of-the-art NLP framework which provides deep contextualized word representations (Peters et al., 2018) and has proved efficient in many NLP-related tasks recently. Here, we focus on the encoder architecture of the baseline model and replace it with a pre-trained ELMo model. As shown by Figure 2, in this model embeddings for the tokens in the input sequence are generated using a pre-trained ELMo model (Peters et al., 2018) and the embedding for the sequence is the generated from these token embeddings using the min-mean-max representation described in Section 3. The training of this model is the same as for the base model, except that the encoder (the pre-trained ELMo model) is not modified during training. Hence, reconstruction and total losses are only used to update $\mathbf{G}$.

4.3 Generator variant: style-specific generators

We propose a multi-generator extension of the base model which employs style-specific generators $\mathbf{G_1}$, $\mathbf{G_2}$ (one for each style). The generators share the encoder component and given a $\mathbf{z}$ generated by this shared encoder, each generator generates a sequence in its corresponding style. Therefore, unlike the base model, in this framework, there is no need to condition the generation on the desired style vector. The role and functioning of the discriminators in this model are the same as those of the base model.

In this model, there are two reconstruction losses, one for each generator (equations 8 and 9). Adversarial loss, discriminator losses, and total loss are the same as those of the base model (equations 5, 6 and 7) and L_{rec} in equation 7 is the summation of the L_{rec1} and L_{rec2}. The training process of this model is also the same as the base model apart from the fact that the backpropagations following the reconstruction losses and total loss are carried out for the the couples ($\mathbf{E}$, $\mathbf{G_1}$) and ($\mathbf{E}$, $\mathbf{G_2}$).

$$\mathcal{L}_{rec1} = \quad -\log \Pr_{G_1}(\widetilde{\mathbf{x}}_1^{(s_1)} | x_1^{(s_1)}) \tag{8}$$

$$\mathcal{L}_{rec2} = \quad -\log \Pr_{G_{s_2}}(\widetilde{\mathbf{x}}_2^{(s_2)} | x_2^{(s_2)}) \tag{9}$$

5 Datasets and experimental setup

Datasets For this work we limit the style transfer task to *sentiment and opinion polarity transfer in text* and evaluate the proposed frameworks using the Yelp dataset of restaurant reviews. It is a large-scale review dataset (4.7 million reviews) where text reviews are annotated by the stars (on a scale of 5) given by users. The reviews with rating above three and below three are considered as positive and negative corpora respectively, and three-starred reviews are discarded.

The dataset is annotated at a review level. However, since the unit of analysis in our experiments is sentence, we must project the review level annotations to the sentence level. This involves a one to many mapping of labels: one review label to multiple sentences. A question that arises here is whether it is appropriate to label every sentence in a review with the overall sentiment label of the review. For example, a review may have an overall sentiment rating that is very positive but certain sentences in the review may be quite neutral. To address this problem, and following the process of Shen et al. (2017), we first filtered all reviews that had more than 10 sentences, and then we removed all sentences that had more than 15 tokens. The intuition behind these filtering processes is that longer reviews are more likely to have neutral sentences, and that longer sentences are more likely to be neutral. We split the resulting dataset, including 250K negative, and 350K positive sentences, into training (70% of the sentences), development (10%) and test (20%) sets while maintaining the relative distributions of positive

and negative instances within each split. Finally, we created a balanced training dataset by upsampling the negative sentences in the training split, by randomly selecting negative sentences without replacement for repetition within the training split. Upsampling was not applied in the development or test set. The vocabulary size of the final dataset is 10K after replacing words occurring less than 5 times with the <unk> token.

Model settings The encoder and generator cells of the models in this work are single-layer RNNs with GRU and the size of hidden cells is set to 700 in *base model* and *multi-generator model*, whereas this is 3072 when using ELMo in the encoder part. Token vectors are initialized by pre-trained GloVe (Pennington et al., 2014) and their size is set to 100. Discriminators are TextCNN classifiers from Kim(2014).

6 Multi-Dimensional System Evaluation

This section reports on four multi-dimensional experiments on sentiment transfer. Firstly, a comparison between the standard and ELMo-based encoders is presented (Section 6.1). Then, attention shifts to the generator component (Sections 6.2 and 6.3). Section 6.2 evaluates the benefits of using multiple style specific generators, and Section 6.3 evaluates the effects on content preservation of reinforcing the latent representation of the input throughout the output generation process. Finally, Section 6.4 validates our multi-dimensional experimental framework for style-transfer against human evaluations. In the following sections, we will discuss the results of each of these experiments which are listed in Table 1.

6.1 Experiment 1: ELMo-based encoder

Given the many recent examples of ELMo embeddings being used in state-of-the-art NLP systems, our first experiment evaluated whether using an ELMo-based encoder would improve the performance of the baseline style-transfer model. The ELMo-based variant of the base model was introduced in Section 4.2. The performance of the base model and its ELMo-based variant are reported in rows a and b of Table 1.

The results of these two models reveals that using ELMo-based representations of the input sequences reduces the performance of the model in all three aspects of evaluation. To better understand these results, we designed a probing experiment to see how much the latent representations of the input sequences generated by each system encoder (i.e., the $\mathbf{z}$-vectors) encodes source stylistic information. We will introduce the probing experiments in detail in section 7; however, for now, the results of this probing experiment for the base model and ELMo variant are listed in rows a and b of Table 2.

Comparing rows a and b of Table 2 reveals that ELMo does appreciably worse in encoding stylistic features compared to a style transfer-specific encoder. We hypothesise that the role of encoders in style transfer models is not only encoding the information, but also, distinguishing between stylistic and content-related features which helps the style-shifting generation process. In more detail, we hypothesise that the performance of the ELMo-based model drops because relative to the style-transfer specific encoders the ELMo-base $\mathbf{z}$-vector representations have encoded less information relating to the input style and this makes it more difficult for the generator to decide whether it is reconstructing the input or style-shifting the input. When a $\mathbf{z}$-vector strongly encodes the input style the model has information relating to both the input style and the target style (it receives this through the style vector fed directly into the generator) and so has a comparative signal as to whether the generation task involves reconstructing or style-shifting the input. However, as stylistic information is stripped from the $\mathbf{z}$-vector this signal is weakened and the task of the generator becomes less well defined and so more difficult.

The main finding of this experiment is that there is a need for style transfer-specific encoders. Given this result, in the next experiment we will test whether it is possible to improve style transfer by adjusting the generator component of the pipeline.

6.2 Experiment 2: Style-specific generators

We aim at modifying the generator component of the architecture by implementing style-specific generators and investigate their effect on the the style-shift strength. The style-specific generator variant of the baseline architecture evaluated in this experiment was described in Section 4.3.

	Model	Style-shift power	Fluency	Content preservation
(a)	Base model	78.8%	43.5	0.925
(b)	ELMo-based encoder	39.3%	134.5	0.895
(c)	Multi-generator	**94.4%**	**39.1**	0.910
(d)	Base model (reinforce)	66.0%	49.6	**0.930**
(e)	Multi-generator (reinforce)	90.7%	83.0	0.860

Table 1: Fluency, style shift strength and content preservation of transferred sequences. The higher the style-shift power and content preservation power are the better it is, but for the fluency (perplexity) the lower results show the better performance.

The performance of the model with multiple style-specific generators is listed in row c of Table 1. Comparing these results with the performance of the baseline system reveals an improvement in the style-shift power and also fluency. The increase in style-shift power is due to the fact that in the multi-generator structure, each generator can learn a specific style. Also, the increase in fluency is due to the fact that each generator learns the distribution of one corpus (i.e., the distribution over the samples in the relevant style). This can provide further explanation for the increase of style-shift power in multi-generator frameworks (an increase from 78.76% to 94.41% in rows a and c of Table 1).

Furthermore, results of probing classification task from Table 2 indicates that multi-generator architectures (rows c and e) encode less source stylistic features compared to single-generator systems (rows a and d). As noted above, in a single generator model encoding input style in $\mathbf{z}$-space provides a signal to the generator relating to whether it is reconstructing or style-shifting the input. However, the fact that multi-generator systems have style-specific generators obviates the need for this signal and this is reflected in the multi-generator system encoding less source stylistic features in $\mathbf{z}$-vectors compared to single-generator models while avoiding the deterioration in style-shift and fluency performance observed with the ELMo encoder system. However, in spite of an increase in style-shift power and fluency, we lose content preservation power in this framework compared to base model. To deal with this, in section 6.3 we implement techniques to reinforce content throughout the generation process.

6.3 Experiment 3: Reinforcing the content

In this section, we investigate whether reinforcing the latent representation of the input sequence ($\mathbf{z}$-vectors) during generation improves the content preservation power of a model. We do this experiment for both the base model and the multi-generator model.

In the standard setup of the base model and also the multi-generator model $\mathbf{z}$ is provided once to the generator at the start of of the generation process, i.e. $\widetilde{x}_1^{(s)} = G(\mathbf{z}, \widetilde{\mathbf{x}}_\mathbf{0}^{(\mathbf{s})}, \mathbf{s})$ but $\widetilde{x}_{i>1}^{(s)} = G(\mathbf{h}_{i-1}, \widetilde{x}_{i-1}^{(s)}, s)$ with $\mathbf{h}_{i-1}$ being the hidden state of the generator, $\widetilde{x}_0$ a special `<start>` token, and $\widetilde{x}_i$ the token generated at step i. The strategy we test for reinforcing content involves $\mathbf{z}$ being provided to the generator at the start of the generation process and also being reinforced at each generation step by concatenating $\mathbf{z}$ to the GRU logits output vector prior to passing it to the softmax layer (this concatenation method was proposed by Tanti et al. (2018)).

Table 1 row d shows that reinforcing the content throughout generation slightly improves content preservation in the base model but also results in a drop in style-shift power and fluency. For the multi-generator model (Table 1 row e), reinforcing content negatively affects all three aspects of evaluation. To provide an overall baseline for the content preservation of these systems for each system we calculated the cosine distance between the style shifted sentences generated by each system when applied to the test set and randomly selected sentences from the training set with the same style. The average of these scores across different frameworks can be considered as lower bound of content preservation rate and is 0.817. This shows that all the models perform better in content preservation than the lower bound.

Intuitively, the effect of reinforcing $\mathbf{z}$ throughout the generation process is dependent on what information is encoded in $\mathbf{z}$. For example, if $\mathbf{z}$ only encodes content information then reinforcing $\mathbf{z}$ should not

diminish style shift power; however, if $\mathbf{z}$ encodes both style and content information then reinforcing $\mathbf{z}$ may help with content preservation but this will likely be at the cost of a loss in style shift power. Also, if $\mathbf{z}$ encodes information from a distinct distribution of language (e.g., positive versus negative sentiment) reinforcing $\mathbf{z}$ may inhibit the fluency of the language model. The results of our probing experiment in Section 7 provide some insight into these dynamics.

The results of the probing task in rows d and e of Table 2 illustrates that $\mathbf{z}$-reinforcement leads the models to embed less style-specific information in $\mathbf{z}$. Our hypothesis for what causes this is that the generator component is trying to output a sequence in a specific style; however, if a lot of stylistic information is encoded in $\mathbf{z}$ and this stylistic information is repeatedly reinforced during generation then $\mathbf{z}$ interferes with the ability of generators to shift to a new style. In order to overcome this dynamic the systems learn to encode less stylistic information in $\mathbf{z}$, thereby reducing the interference in the generator between the source style and desired style. This leads the single-generator model to lose more power to transfer the style compared to the multi-generator model (a drop of 12.8% in style-shift strength in single-generator models versus 3.7% in multi-generator frameworks in rows d and e of Table 1). The reason is the former was relying more on style-specific information in $\mathbf{z}$ than the latter, as we noted earlier the single-generator model may use the style information in $\mathbf{z}$ to determine whether to reconstruct or style shift the input. Overall, there is an inherent trade-off for single-generator models with respect to encoding stylistic information in $\mathbf{z}$: reducing stylistic information in $\mathbf{z}$ makes the initial decision between reconstruction and style-shift more difficult, but increasing the stylistic information in $\mathbf{z}$ makes style-shifted generation more difficult. Multi-generator models do not need source style signals for their style-specific generators; therefore, they drop more source style in their z-space compared to single-generator models (7% versus 2%). However, this negatively affects the content preservation power of these models, since, seemingly, dropping more source style in z-space leads to losing some features which are either content or the type of features which can be considered as the intersection between content and style.

6.4 Experiment 4: Validating the Methodology with Human Evaluations

To investigate whether the results of our automatic evaluation correlate with the human judgment, we did a manual test considering style-shift, content preservation and fluency. The tests were conducted in a strictly blind fashion by shuffling samples and models to avoid biases while answering the questions. To do the test, 450 samples were considered: 150 samples randomly selected from the Yelp test set and their corresponding style-transferred sequences of the base and multi-generator models. Each sample was evaluated at least 3 times and at most 8 times. The final scores are reported as the proportions for each answer, averaged over all testers and all samples. Moreover, Krippendorff's inter-rater agreements (Krippendorff, 1980) is reported. Below are the details for each aspect.

Style shift power In this test the testers considered one style-shifted sample at a time, and labelled it as "positive", "negative", or "neutral". There were 29 participants with an inter-rater agreement of 0.752. The results indicate that base model is successfully transferring the style of the sequences in 58.3% of the cases while this is 67.6% for the multi-generator model.

Content preservation This test was conducted in a comparative manner, i.e. evaluators were shown a source sentence and two style-shifted sentences from the two models and were asked which sequence most closely resembles the source sentence in terms of content (disregarding the sentiment). Possible options were "equally good", "equally bad", "first sample is better", or "second sample is better". There were 22 participants in this study with an the inter-rater agreement of 0.772. In 52.2% of the cases, the base model appears to preserve the content, while this is 41.6% for the multi-generator model.

Fluency This test was evaluated in a similar way as style transfer. The evaluators judged the structural correctness of generated samples as "incorrect", "partly correct" and "correct". As a post-processing step, the "partly correct" answers were discarded since the task appeared as ambiguous for the annotators. The were 25 participants with inter-rater agreement of 0.568. The results show that the multi-generator performs slightly better than the base model (67.6% versus 64.2% of "correct" answers).

	Model	Accuracy
(a)	Base model	**99.97**%
(b)	ELMo-based encoder	93.84%
(c)	Multi-generator	97.56%
(d)	Base model (reinforce)	98.39%
(e)	Multi-generator (reinforce)	91.85%

Table 2: The accuracy of the classifiers trained for each of the sentiment-transfer frameworks.

In conclusion, these results are in line with the rows a and c of the Table 1 and so we interpret them as confirming the validity of the automatic metrics selected in our work.

7 Probing the Relationship between Style and Content

In Section 6.3 we observed that reinforcing the latent representation of the input ($\mathbf{z}$) during generation could result in improved content preservation but also consistently resulted in a deterioration in style-shift power. In this section we examine whether this trade-off is a result of stylistic information being encoded within the latent representation of the input. Inspired by the probing experiments described by Conneau et al. (2018), we designed a probing experiment to examine what style information each of our models encoded in their latent representations.

To do the probing classification experiment, for each style-transfer framework we trained a classifier that took the $\mathbf{z}$ representation generated by the encoder of the framework and predicted the style of the input sequence that had been feed into the encoder. The idea here is that if we can train a classifier to accurately predict the style of an input from the $\mathbf{z}$ representation of an input sequence generated by the encoder, then this $\mathbf{z}$ vector must have information related to the style of the input sequence. Therefore, by examining the accuracy of the style classifier for each style-transfer framework we can get an insight into the amount of stylistic information the framework encodes in its $\mathbf{z}$ space. For each style-transfer framework, we trained a separate feed-forward network with a single hidden layer and a sigmoid output layer as the classifier using pairs of $\mathbf{z}$ vectors (embedded representations) of the input sequences and their original style. We employed the Yelp review dataset for training where the distribution of the "positive" and "negative" labels are balanced. Table 2 shows the performance of each classifier. For all style-transfer frameworks we were able to train accurate input style classifiers. This indicates that all the encoders include style information in the $\mathbf{z}$ state. There is, however, variations across the style-transfer frameworks in terms of the amount of style information they encode in their $\mathbf{z}$ state. In particular, reinforcing the input during generation leads to a reduction in stylistic information being encoded in $\mathbf{z}$. As discussed earlier we believe this is caused by the systems attempting to ameliorate the interference caused by reinforcing the input style when attempting to generate in another style.

8 Further discussion and future steps

Our results indicate that style cannot be totally separated from content. This is because encoders mistakenly strip out content when attempting to remove source stylistic features. This is most clearly seen in the interaction between the results of the probing experiment and the content preservation results for the different systems. The probing experiment revealed that reduction in encoding the stylistic features in the latent representation of the inputs as a result of reinforcing the content (table 2) can lead to a reduction in content preservation of the model. This indicates a direct relationship between the content preservation power of a model and how much of source style the model encodes in its latent space. Therefore, it can be inferred that style and content are entangled elements, which suggests that style should be considered as an integral component of a text which cannot totally be separated from the content of the text. This raises questions about the conceptual basis of the computational modelling of style which considers style as independent from content and fragmentizes style into a set of explicit linguistic elements such as specific words, markers, or syntactic structures (Li et al., 2018; Leeftink and Spanakis, 2019).

To summarize, the results of this study show the necessity of employing style transfer-specific encoders as opposed to pre-trained embeddings. Furthermore, the results indicate that each of the tested frameworks has its own strength and weaknesses. Indeed, our result indicate the benefits of framing style-transfer as a multi-dimensional task: given the trade-off we observed in terms of style-shift and fluency versus content preservation it is important to consider all of these aspects when evaluating a system as understanding the trade-off systems make can inform the design of improved style-transfer systems. Finally, our results suggest that stylistic and contextual features cannot really be separated or at least not in the case context of framing sentiment as a style.

Overall, we consider the multi-generator framework as a good basis for future research, and we plan to implement alternative techniques to improve content preservation power in this architecture. Furthermore, considering the significance of doing the evaluation in a comprehensive manner, we would like to focus on introducing a single evaluation metric for this task which takes the three evaluation aspects of style-shift, content maintenance and fluency into consideration.

9 Acknowledgements

This study has been realized under the ANR (French National Research Agency) projects TREMoLo (ANR-16-CE23-0019) and TextToKids (AAPG 2019).

This project has also been supported by the ADAPT Centre for Digital Content Technology which is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Keith Carlson, Allen Riddell, and Daniel N. Rockmore. 2017. Zero-shot style transfer in text using recurrent neural networks. *CoRR*, abs/1711.04731.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of the 28th Neural Information Processing Systems (NIPS), Workshop on Deep Learning*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 2126–2136.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the 32nd AAAI conference on artificial intelligence*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Proceedings of the 28th conference in Neural Information Processing Systems (NIPS)*, pages 2672–2680. Curran Associates, Inc.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Controllable text generation. *CoRR*, abs/1703.00955.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2018. Disentangled representation learning for text style transfer. *CoRR*, abs/1808.04339.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics (TACL)*, 5:339–351.

John D. Kelleher. 2019. *Deep Learning*. MIT Press.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

K. Krippendorff. 1980. Content analysis: An introduction to its methodology. Beverly Hills: Sage Publications.

Wouter Leeftink and Gerasimos Spanakis. 2019. Towards controlled transformation of sentiment in sentences. *CoRR*, abs/1808.04365.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 16th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long Papers)*, pages 1865–1874.

Shuming Ma and Xu Sun. 2017. A semantic relevance based neural network for text summarization and text simplification. *Computational Linguistics*, Volume: 1(1).

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 866–876. Association for Computational Linguistics.

Ella Rabinovich, Raj Nath Patel, Shachar Mirkin, Lucia Specia, and Shuly Wintner. 2017. Personalized machine translation: Preserving original author traits. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL): Volume 1, Long Papers*, pages 1074–1084. Association for Computational Linguistics.

Juan Ramos. 1999. Using tf-idf to determine word relevance in document queries.

Alexey Romanov, Anna Rumshisky, Anna Rogers, and David Donahue. 2019. Adversarial decomposition of text representation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long and Short Papers)*, pages 815–825.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 86–96. Association for Computational Linguistics.

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

Ayush Singh and Ritu Palod. 2018. Sentiment transfer using seq2seq adversarial autoencoders. *CoRR*, abs/1804.04003.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the Conference in Neural Information Processing Systems (NIPS)*, pages 3104–3112.

Marc Tanti, Albert Gatt, and Kenneth P Camilleri. 2018. Where to put the image in an image caption generator. *Natural Language Engineering*, 24(3):467.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5902–5911, Stockholmsmässan, Stockholm Sweden, 10–15 Jul. PMLR.

Appendix A. Samples of the style-shifted sequences

From negative to positive
1. so nasty .
2. so awesome .
3. ugh delicious
4. so delicious .
5. so fun .
6. so awesome ! .
1. my goodness it was so gross .
2. my husband was so delicious .
3. my experience was great .
4. my goodness it was amazing .
5. my goodness was amazing .
6. my , it was also
1. the cake portion was extremely light and a bit dry .
2. the prime rib was very smooth and very reasonable .
3. the pizza is fresh and a very nice .
4. the tuna was a bit dry and satisfying .
5. the portion was extremely tasty and reasonably priced .
6. the crust was very dry and dry a bit .
From positive to negative
1. i highly recommend this place !
2. i wo n't be it .
3. i do n't even go back .
4. i do not recommend this place !
5. i highly recommend this place place !
6. i loved not recommend this place .
1. my appetizer was also very good and unique .
2. my chicken was just a little hot and texture .
3. my pie is just thin and just like pie .
4. my boyfriend was n't and had a very dry .
5. my entree was very unique and also very good .
6. my appetizer was also very good and lacked lacked lacked lacked .
1. the food is fresh and the environment is good .
2. the food was tasty and the quality is pretty expensive .
3.the food was rude and do n't waste the time .
4. the food is fresh and the sandwich was too salty .
5. the food is good and the food is not good .
6. the food is the food and the food is the food comes .

Table 3: Considering the input sentences in lines 1, lines 2-6 represent the style transferred sequences as the output of the following frameworks: 2: Multi-generator model, 3: Reinforced multi-generator model 4: Base model 5: Reinforced base model and 6: ELMo-based encoder model.