Local or Global: The Variation in the Encoding of Style Across Sentiment and Formality

Somayeh Jafaritazehjani[1,3], Gwénolé Lecorvé[2], Damien Lolive[3], and John D. Kelleher[4]

[1] Technological University Dublin, ADAPT Centre, Ireland
[2] Orange, Lannion, France
[3] Univ Rennes, CNRS, IRISA, France
[4] Maynooth University, ADAPT Research Centre, Ireland
somayeh.x.jafaritazehjani@mytudublin.ie, John.Kelleher@mu.ie

Abstract. Research on textual style transfer has observed that the concept of style can vary across domains. This research examines the encoding of style across the sentiment and formality domains and observes that formality appears to be more globally encoded, and sentiment more locally encoded. The work also shows how the encoding of a style can inform the appropriate choice of method to compute content preservation during textual style transfer.

Keywords: Sentiment · Formality · Style · Local · Global.

1 Introduction

Textual Style Transfer (TST) attempts to generate text in a given style from a input text in a different style while preserving as much information from the input as possible. Even though style is a key element in this field, there is yet no consensus and clear definition for it. Previous work has identified that the concept of style varies across different domains [10]. However, this prior research does not explain how the encoding of style differs across these domains. This is the question that this paper addresses. By the encoding of style we mean the distribution of the components of a text that carry stylistic information. If a style is locally encoded this implies that frequently only a relatively small number of local changes are necessary to transfer the style (such as changing a few keywords), whereas if a style is globally encoded then typically a general reworking of a text is required in order to transfer the style.

We focus on sentiment and formality (two commonly used styles in *TST* research) and examine the differences in the style encoding when it is equated with sentiment polarity versus formal register of a text. Based on observations from a number of experiments, we argue that style varies in terms of its local versus global encoding. In particular, we find that transforming formality requires a more global adaptation of a text as compared to sentiment-transfer.

The paper begins by proposing a refinement to training regime of a state-of-the-art transformer-based (T-based) TST model [4] which reduces the computational cost[5]. The paper then reports a set of probing experiments that examine the entanglement of style across the layers of a transformer architecture in both the sentiment and formality domains. Building on these results it presents a unigram based analysis of the overlap between original and human style adapted paraphrases of a text in both domains. The probing experiments and unigram overlap results indicate that the encoding of formality is more global (requiring extra layers of processing to encode, and resulting in lower unigram overlap) as compared to the encoding of sentiment. Finally, informed by the distinctions between the encoding of these style domains we reflect on the standard metrics used to assess content preservation (CPP) within TST research, and show that contextual embeddings [26] are more accurate for computing semantic similarity between source and paraphrased texts in the formality domain, whereas, GloVe embeddings [22] result in more accurate CPP scores in sentiment domain.

2 Literature review

Most TST approaches use end2end strategies to learn a latent representation of the input [13] and condition the generation of style-shifted text on this representation. In the case of using parallel data to train models, TST is similar to a supervised NMT problem [19]. However, the majority of previous TST research has addressed the task in an unsupervised framing and compensate for the lack of parallel labelled data using adversarial techniques to guide the training towards generating text in a desired style.

The majority of TST networks have employed standard seq2seq RNN-based models [30, 1] with some integrating variational encoders, multi-encoder, or multi-decoder [4, 27, 29, 28, 7, 8, 12, 9, 10]. Some previous TST work has analysed the role of different subnetworks of end2end RNN-based models as well as investigating the intermediate representations created by them [15, 6, 9, 10].

Recently, [4] proposed a transformer-based (T-based) TST model. In a pilot experiment on our datasets we found that this model outperformed state-of-the-art RNN-based models (table 1, see rows 1, 2, 3). Consequently, we the decided to use this model for our experimental work. However, this T-based model took a significant amount of time to converge, and so we adapted the training regime in order to reduce the computational cost.

3 Transformer-based (T-based) TST model

In this section, we describe the architecture and the adapted training regime of our T-based TST model consisting of: (i) a generator and (ii) a discriminator.

[5] The code is released: https://github.com/somayeJ/Transformer-based-style-transfer

Dataset	Yelp			GYAFC		
Model/ Evaluation metrcis	**CPP**	**PPLX**	**SSP**	**CPP**	**PPLX**	**SSP**
RNN-based model [28]	0.9261	**37.98**	81.8%	0.9088	26.81	**65.11%**
Multi-E RNN-based model [10]	0.9289	41.37	79.8%	0.911	**28.84**	58.82%
T-based model [4]	0.9717	106.07	78.50%	0.9516	289.20	28.99%
Our T-based model	**0.9718**	126.12	**83.00 %**	**0.9741**	141.44	47.19%

Table 1. Higher CPP and SSP show better performance, but lower values of $PPLX$ reflect better fluency. α and β (equation3) of T-based models are 0.25 & 0.5.

Generator (***Gen***) is a seq2seq pipeline where the encoder (***E***) and decoder (***D***) are transformers [31]. ***E*** consists of a sequence of 4 stacks, each including a fully connected self-attention, fully connected point-wise feed-forward, and normalization layers. ***E*** takes a text x (of length T and original style s_1) and a desired style s_2 as the input.The processing within ***E*** projects the input tokens through layers where the model learns the contextual and positional information of the tokens.

The final layer of ***E*** creates a sequence of latent token representations: $z = (z_0, z_1, ..., z_T)$ where z_0 is the dense vector of the desired style. ***D*** is similar to ***E***, but also contains an attention layer where z is fed from ***E***. ***D*** also has a projection layer which takes the output of ***D*** and generates tokens.

Reconstruction Loss The reconstruction loss is designed to encourage the model to retain relevant content. We use two types of reconstruction loss during training, the self-reconstruction and cycle loss. During training (figure 1), ***Gen*** generates two text for each input $\mathbf{x}$ it receives, i.e. given $\mathbf{x}$, ***Gen*** creates its reconstructed version $\widetilde{\mathbf{x}}^{rec}$ as well as its style-shifted version $\widetilde{\mathbf{x}}^{trf}$. $\widetilde{\mathbf{x}}^{rec}$ is used to compute the self-reconstruction loss as follows. During training the negative log probability of x and $\widetilde{x}^{rec}$ is minimized.

$$\mathcal{L}_{self_{rec}} = -\log \Pr(\widetilde{x}^{rec} = x | x, s) \tag{1}$$

Using the self-reconstruction loss, ***Gen*** is trained to reconstruct text where the style of the input and the desired target style of the output text are the same. Here, the model functions as an auto-encoder (AE).

The style-shifted text, on the other hand, is used as an intermediate representation for the cycle loss which is calculated in equation 2.

$$\mathcal{L}_{cycle_{rec}} = -\log \Pr(\widetilde{\widetilde{x}}^{rec} = x | \widetilde{x}^{trf}, s) \tag{2}$$

$L_{cycle_{rec}}$ is designed to encourage the model to preserve the non-stylistic information of the input. To do so, the model is trained to generate $\widetilde{\widetilde{x}}^{rec}$ (a reconstructed version the input x) given $\widetilde{x}^{trf}$ (style-shifted version of x) by following these cycle of generation steps where $s_1 \neq s_2$:

- Generating $\widetilde{x}^{trf}$ by feeding ***Gen*** with x and the desired style s_2.
- Generating $\widetilde{\widetilde{x}}^{rec}$ by feeding ***Gen*** with $\widetilde{x}^{trf}$ and the desired style s_1.

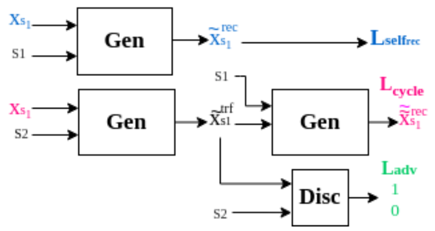
x_{s_1}
S1
Gen
$\widetilde{x}_{s_1}^{rec}$
$L_{selfrec}$
S1
L_{cycle}
$\widetilde{\widetilde{x}}_{s_1}^{rec}$
x_{s_1}
S2
Gen
$\widetilde{x}_{s1}^{trf}$
Gen
L_{adv}
Disc
1
0
S2

Fig. 1. For input x_{s_1}, ***Gen*** creates $\widetilde{\mathbf{x}}_{s_1}^{rec}$ & $\widetilde{\mathbf{x}}_{s_2}^{trf}$, also re-creates $\widetilde{\widetilde{\mathbf{x}}}_{s_1}^{rec}$. ***Disc*** gets $\widetilde{\mathbf{x}}_{s_1}^{trf}$ & its desired style s_2, and labels it as style-shifted (0) or reconstructed (1).

While training, equation 2 is used to minimize the negative log probability of each input x and its reconstructed text $\widetilde{\widetilde{x}}^{rec}$.

The reconstruction loss is then computed using the equation 3 as the weighted summation of a self-reconstruction and a cycle-reconstruction loss.

$$\mathcal{L}_{rec} = \alpha \mathcal{L}_{self_{rec}} + \beta \mathcal{L}_{cycle_{rec}} \tag{3}$$

Discriminator (***Disc***) consists of a sequence of a transformer (with the same architecture as the transformer E), and a classifier (a feed-forward network with a single hidden layer and a softmax output layer), similar to the discriminator used in [24] and [5]. It takes as input a text and a style and attempts to learn whether or not the style matches the original style of the given text. Specifically, it is trained to label pairs where the style matches the original style of the text as positive and pairs where the style is not the original style of the text as negative, i.e. it learns to return true for ($\mathbf{x}_{s_1}$, s_1) and ($\widetilde{\mathbf{x}}_{s_1}^{rec}$, s_1) and false for ($\widetilde{\mathbf{x}}_{s_1}^{trf}$, s_2) and ($\mathbf{x}_{s_1}$, s_2). This is done by minimizing the equation 4, the binary cross-entropy over the two classes where $s_1 \neq s_2$. ***Disc*** is trained in parallel with ***Gen***.

$$\mathcal{L}_{Disc} = -\log(Disc(\widetilde{\mathbf{x}}_{s1}{}^{rec}, s_1)) - \log(1 - Disc(\widetilde{\mathbf{x}}_{s^{trf}, s_2))}) \tag{4}$$

Adversarial Loss A key component of the adversarial training is ***Disc*** which is in competition with ***Gen***: ***Gen*** attempts to generate style-shifted text that ***Disc*** will categorize as original text, and ***Disc*** is attempting to detect style-shifted sequences. To encourage ***Gen*** to generate style-shifted text that can convince ***Disc*** that the style-shifted text is actually original text, an adversarial loss is defined in equation 5 ($\overline{s}$ is the source and s is the desired style of text) and is minimized during the training together with the total loss ($\mathcal{L}_{rec} + \mathcal{L}_{adv}$).

$$\mathcal{L}_{adv} = -\log(Disc(\widetilde{\mathbf{x}}_{\overline{s}}^{trf}, s)) \tag{5}$$

Training regime Although the architecture and loss functions we use (sections 3, 3, and 3) are the same as [4], the training regime we propose is different. First, we do not use a pre-training phase for ***Gen***, instead ***Gen*** and ***Disc*** are trained in parallel throughout. However, following [28], we do not use the adversarial loss

in the training of ***Gen*** until the loss of ***Disc*** falls below a pre-set threshold (a hyper-parameter set to 1.2 based on [28]). This is to ensure that the adversarial loss is not used in training ***Gen*** until ***Disc*** is of sufficient quality that the adversarial loss it returns is informative. Also, when updating ***Gen*** weights, we only do a single backpropogation+weight update pass, using the summation of the losses, as compared to [4] that uses two backpropogation+weight update passes, first, the self reconstruction, the summation of the adversarial and cycle loss. Our training regime is as follows where given a corpus, X_{s_1} and X_{s_2} are portions of this corpus containing styles s_1 and s_2.
Step 1: Sample two different-styled mini-batches of size k (k indicates batch-size and is set to 1 here for simplicity): $\{x_{s_1}\} \in X_{s_1}$, and $\{x_{s_2}\} \in X_{s_2}$.
Step 2: Generates a reconstructed and a style-shifted text for each sequence:
$\widetilde{\mathbf{x}}^{rec}_{s_1} = Gen(x_{s_1}, s_1)$, $\widetilde{\mathbf{x}}^{trf}_{s_2} = Gen(x_{s_1}, s_2)$
$\widetilde{\mathbf{x}}^{rec}_{s_2} = Gen(x_{s_2}, s_2)$, $\widetilde{\mathbf{x}}^{trf}_{s_1} = Gen(x_{s_2}, s_1)$
Step 3: Compute $L_{self_{rec}}$ using the equation 1.
Step 4: Compute L_{Disc} using the equation 4 and update θ_{Disc}.
Step 5: If $L_{Disc} < 1.2$, compute $L_{cycle_{rec}}$ and L_{adv} and update θ_{Gen} using the total loss. Otherwise, perform update θ_{Gen} using the equation 1.
Step 6: For batches of one epoch, repeat steps 1-5.
Step 7: Use the model with lowest total loss (best model) as the initial model in the next epoch. Stop after 20 epochs.

4 Data, experimental set up and evaluation methodology

Data Yelp and GYAFC datasets are used throughout the experiments of this paper. Yelp is a restaurant review dataset where the positive or negative label of each review is considered as its style. For our experiments, we use the dataset provided and preprocessed by [17].

GYAFC (Grammarly's Yahoo Answers Formality Corpus) [25] is a parallel dataset which is used in a non-parallel mode in our experiments. It contains text from the domains of Entertainment & Music and Family & Relationships. We compose and shuffle the data from these domains and do the following preprocessing steps (around 2% of the data is removed in the resulting dataset).

First, we make the tokens more consistent by: 1. Lower casing the tokens. 2. Replacing the numbers, website addresses, email addresses and emojis with special tokens. 3. Inserting space between token and punctuation as well as punctuation and punctuation. 4. In informal data, for the tokens with high frequency such as *oh* converting all non-standard forms, such as *ohhhh* or *oooohhhhh* into one non-standard form *ohh* to reduce the size of vocabulary and also the number of unknown tokens (`<unk>`). 5. Filtering the non-English sequences. To do so, we manually filter the list of sequences marked as non-English by the python language detector library. This is to reduce the possibility of removing the English sequences which are detected as non-English specifically due to the characteristics of informal data which contains many non-standard variations of the tokens. 6. Considering a box plot of the sentence length distribution and filtering all sequences whose lengths were outside the whiskers of the plot.

Experimental set up While building the vocab, the tokens with frequency lower than 5 were considered as `<unk>`. Each stack of $\boldsymbol{E}$ and $\boldsymbol{D}$ of out T-based model has 4 attention heads. The size of token embeddings, positional embeddings,

Data	Yelp		GYAFC	
Style	Positive	Negative	Formal	Informal
Train	267314	176787	102502	104044
Dev	2000	2000	5064	5111
Test	500	500	2076	2739
Avg-l	8.45	9.66	12.4	12
Vocab-size	9352		11409	

Table 2. Data distribution of the datasets, (l:length).

style vectors and the hidden size of the model are 256[6].

Evaluation methodology We use three evaluation metrics designed to cover the multiple objectives of TST: style-shift, content maintenance and fluency [11].

Style-shif power (SSP) determines the power of a TST model in shifting the style, prior work has trained separate classifiers for each domain to measure the presence of a desired style in the style-shifted text [7, 18, 16, 29, 23, 28, 12, 8]. Accordingly, we train two separate binary classifier using GYAFC and Yelp data using the TextCNN model proposed by [14]. Style shift power SSP metric is the score returned by this classifier for the target style of style-shifted text.

To compute Content Preservation Power (CPP), we use an embedding-based approach to measure the similarity between an input x and style-shifted text $\widetilde{x}^{trf}$. First, the tokens of x and $\widetilde{x}^{trf}$ are mapped into an embedding space using a pretrained model. Vector representations of x and $\widetilde{x}^{trf}$ are then created by taking the average of their token embeddings. Finally, these vector pairs are compared with cosine similarity [7]. In our experiments, we use both a 100-dimensional GloVe model [22] and a 768-dimensional SBERT model[7][26] as pre-trained embeddings models. To improve the interpretability of CPP scores, we compute the semantic similarity between randomly selected sequences as the CPP lower bound (LB) scores. The respective GloVe- and SBERT-based LB scores for the sentiment domain are 0.86 and 0.09387 and for the formality domain are 0.87 and 0.0672.

To evaluate the fluency of the generated texts we follow previous research [32, 12] and train separate language models for each domain. These LMs are single-layer RNN with GRU cells [2]. We then use the average perplexity scores

[6] Other hyperparameters adapted from (http:/github.com/fastnlp/style-transformer).
[7] http:/huggingface.co/cross-encoder/stsb-TinyBERT-L-4

of these models on the style-shifted texts of a TST model as its PPLX. Lower PPLX indicates the TST model generates more fluent text.

5 Experiments

To investigate the variations in sentiment and formality encoding, an interesting experiment is re-weighting the TST model so that it performs more similar to an AE (section 5). The idea being that the more a TST model is weighted towards acting as an AE the less likely it is to perform global rewrites and so this change is likely to be reflected in changes of the SSP of the system in domains where style is globally encoded. Also, we study how sentiment and formality are encoded by different layers of the T-based encoder of our TST model (section 5) and compare it with the variations observed by studying the human-generated data across these style domains (section 5).

Reconstruction versus adversarial Balance Table 1 lists the results generated using the training regime from [4] (row 3) and our adapted training regime (row 4). The results reveal a slight improvement with the new training regime with improvements in CPP and SSP on Yelp and a larger improvement in CPP for GYAFC with a drop in SSP. However, these results are recorded from single runs of the model and so we do not claim a statistical difference here. More importantly, however, the adapted training regime required much less training time to achieve these results[8].

Neither the total loss nor the reconstruction loss (equation 3) normalize the contribution of the losses. Therefore, α and β summing to less and more than 1 indirectly puts greater emphasis during training on the adversarial loss and reconstruction loss, respectively. T-models listed in table 1, use a total weight of 0.75 (α=0.25 and β=0.5) to reconstruction and 1 to adversarial loss.

We investigate the effect of increasing the weight of reconstruction loss by doubling the summation of α and β and training two new models: T_1; α=1, β= 0.5, and T_2; α=0.5, β=1. Comparing the results of T_1 and T_2 in table 3 with the scores of our T-based model (table 1) shows that in Yelp, T_1 performs better than the T-based model in every evaluation aspect and T_2 also has a better CPP and fluency. In GYAFC, however, this weight modification does not appear to be as beneficial overall, although, in T_1, it results in an improvement in CPP and fluency, the SSP drops by a large amount. The fact that increasing the weighting of the reconstruction loss relative to the adversarial loss encourages a model to act more like an AE, and that this is beneficial for both CPP and SSP in the sentiment domain but results in much lower SSP in the formality domain, suggests that shifting sentiment requires fewer text changes compared to the formality (i.e., sentiment is more locally encoded compared to formality).

[8] Training of our T-based model on Yelp took around 36 hours using single Quadro RTX 8000s GPU, compared to 75 hours while applying the training regime from [4].

	Datasets		Yelp		GYAFC	
	Models		**T1**	**T2**	**T1**	**T2**
Automatic Evaluation	**SSP**		83.8%	70.9%	32.71%	41.52%
	PPLX		107.07	99.88	101.57	154.35
	CPP	GloVe	0.9732	0.9767	0.9743	0.9714
		SBERT	0.5869	0.6177	0.8595	0.8108
Layer-wise probing of Transformer encoder	**GloVe-baseline**		85.80%		71.01%	
	Embedding layer		89.9%	87.4%	75.74%	78.69%
	Stack1		100%	90.5%	74.18%	80.48%
	Stack2		100%	100%	99.63%	88.2%
	Stack3		100%	100%	100%	100%
	Stack4		100%	100%	100%	100%

Table 3. α and β (equation3) of T_1: α=1, β= 0.5, and T_2: α=0.5, β=1.

Layer-wise probing of transformer $\boldsymbol{E}$ An interesting aspect of transformer models is that they include multiple self-attention layers. Indeed, researchers interested in understanding how transformers encode linguistic information have probed how the encoding of this information varies across the layers of transformers trained for different text NLP-related problems [20, 21]. However, to the best of our knowledge the encoding of style across the layers of a TST transformer has not yet been examined. Inspired by previous work on probing [3, 9, 10] we designed a classification experiment to examine the extent to which style is encoded at each layer of the transformer $\boldsymbol{E}$ for formality and sentiment domains.

We train 5 separate probes, one for each of the layers of $\boldsymbol{E}$: the embedding layer, and each of the 4 stacks of $\boldsymbol{E}$. Each probe, a feed-forward network with a single hidden layer and a sigmoid output layer, is trained to detect the source style of an input text from the embedding of the text generated by that probe's corresponding layer in the transformer.The higher the accuracy of a probe is, the more source-stylistic features are present in the text embedding. The text embedding for a layer is the average of the embeddings of the text tokens generated by that layer.

As a baseline for this task, for each dataset we train a probe on GloVe-based embeddings to identify the style of a text sequence. The GloVe-based embedding for a given sequence is computed by mapping its tokens to their pre-trained GloVe embeddings and then taking the average of these token embeddings. The accuracy of the source style identification probes trained on GloVe-based embeddings of Yelp and GYAFC test sequences (i.e., original text sequences that have not been style-shifted) are 85.80% and 71.01%, respectively. GloVe token embeddings are trained on the nonzero elements in a word-word co-occurrence matrix [22,], and we generated the GloVe-based sequence embeddings by averaging GloVe token embeddings. Consequently, it is likely that our GloVe-based sequence embeddings primarily encode word co-occurrence information and neglect word order, essentially functioning as a bag-of-words. Given this, the higher score on identifying source style in the sentiment domain using GloVe embeddings compared to the formality domain suggests that a bag-of-word representation is better at identifying sentiment compared to formality. This also suggests that sentiment is more locally encoded as compared to formality (sentiment being more readily identifiable based on the presence/absence of particular words, whereas the identification of formality may require more structural information).

Table 3 lists the results from the layer-wise probing of transformer embeddings. The attention mechanism within each layer of $\boldsymbol{E}$ allows the embeddings for a word to be fine-tuned to its context of use by integrating information from across the input sequence. Consequently, as we move up through the layers of the network it is to be expected that the embeddings at each subsequent layer encode a more global perspective on the meaning of a sequence, as more and more information from across the sequence is integrated into each of the token embeddings. As a result, comparing the performance of a probe within a style domain across the layers of a transformer architecture can provide insight into sensitivity of that style to global structure of the sequence. Given this, it is interesting that in the sentiment domain the probe achieves 100% performance at an earlier layer as compared with the formality domain, suggesting that in general the encoding of formality requires more information from across a sequence to be integrated into the embeddings of each of the tokens in the sequence.

Taken together, these probing results suggest that sentiment is encoded in a more local (e.g., keywords are very informative) manner as compared with formality. Based on this observation, we hypothesise that in general formality transfer requires more global changes to a text as compared with sentiment shift, that may be achieved in some instances by just swapping a single word.

Unigram based analysis To further test our hypothesis, sentiment is encoded more locally compared to formality, we run a word overlap (WO) analysis. The intuition being that if sentiment is more local in its encoding relative to formality, a higher WO between style-shifted texts and inputs are expected in this domain compared to the formality. To do this analysis, we use the gold style-shifted text for the test sets of Yelp and GYAFC. In GYAFC, there are 4 human gold sets for each domain and style. We use all these files for analysis and the results reported are the average of the scores computed for each files.

To compute WO, following [12] given x and $\widetilde{x}^{trf}$, we first filter stop words,then compute the score as $\frac{count(x \cap \widetilde{x}^{trf})}{count(x \cup \widetilde{x}^{trf})}$. We augment our analysis of the WO of manually style-shifted texts with an analysis of how successfully the human 'style translators' were at the task. To do this, we use accuracy of the SSP classifiers (introduced in section 4) in detecting the desired style of human-generated files across the two domains. The intuition being that if a human style translator has successfully shifted the style of the text into the desired style the SSP classifier should recognise this desired style with high confidence.

The results of our WO analysis show slightly higher WO between source and gold style-shifted text in sentiment 0.4253 as compared to the formality 0.4057 (LB of WO is 0.0035). This together with higher accuracy of classifiers in labeling sentiment 77.2% compared to 70.45% in formality illustrates that even though more unigrams are swapped in formality transfer, we still observe lower SSP in the style-shifted files which supports our hypothesis.

6 The interaction of style characteristics and CPP

The experiment presented here investigates the performance of GloVe- and SBERT-based CPP metrics for the sentiment- and formality-transfer tasks. For each style domain, we randomly selected 200 samples from the test set of that domain as the source texts. For each of these source sequences we composed a target set containing its corresponding gold style-shifted text and 499 other randomly selected texts. Then, we computed the CPP scores between each source sequence and each of the sequences in its target set using both GloVe- and SBERT-based embeddings. We expect that given a source text a good CPP metric assigns a higher value to the pair <source text, gold style-shifted text> rather than to the pairs of <source text, random text 1>, ..., <source text, random text 499>.

The SBERT-based CPP metric assigns the highest value to the <source text, gold style-shifted text> pair in 95.5% of cases in the formality domain and 75.5% in sentiment domain. The GloVe-based CPP metric assigns the highest value to <source text, gold style-shifted text> pair in 71% cases in the formality domain and in 84% in sentiment domain. These results indicate that for formality the SBERT-based CPP metric works better than the GloVe-based metric (95.5% > 71%), whereas the GloVe-based CPP outperforms SBERT-based CPP for sentiment (84% > 75.5%). The variation in the relative performance of SBERT and GloVe across the two domains is inline with the hypothesis that formality is relatively globally encoded (SBERT is better) whereas sentiment is locally encoded. Indeed, texts having different sentiments seem to be very close in the GloVe embedding space as compared to the SBERT embedding space. Computing the similarity of the text while ignoring their sentiment variations makes GloVe-based CPP metrics more suitable for the sentiment domain.

7 Conclusion

Throughout our experiments we observed that sentiment is more locally encoded whereas formality is more globally encoded. In brief, this observation indicates that sentiment TST can often be achieved by changing a small number of keywords in a text, whereas formality TST frequently required more global reworking of a text. This clarification can improve TST research in a number of ways. First, we observed that SBERT-based CPP metric works better for formality, whereas, GloVe-based metric computes more accurate scores in sentiment domain. This is inline with the insight that formality is encoded as a global property of a text (beyond the representational capacity of a bag-or-words) compared to the encoding of the sentiment which is more token-based. Clarifying the encoding of style in different domains can also inform the appropriate use of TST modelling approaches. Some approaches that attempt to directly filter markers of style in the input, assume that stylistic features are detectable and separable from the content [18, 16]. However, other approaches consider each style to be a separate language and adapt methods inspired by Neural Machine Translation to the TST problem [19, 4, 27, 29, 28, 7, 8, 12]. This distinction between approaches may be inline with the observed global versus local style encoding distinction.

References

1. Bahdanau, D., Cho, K.H., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, ICLR 2015 (2015)
2. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. In: Proceedings of the 28th Neural Information Processing Systems (NIPS), Workshop on Deep Learning (2014)
3. Conneau, A., Kruszewski, G., Lample, G., Barrault, L., Baroni, M.: 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. pp. 2126–2136 (2018)
4. Dai, N., Liang, J., Qiu, X., Huang, X.: Style transformer: Unpaired text style transfer without disentangled latent representation. CoRR **abs/1905.05621** (2019), http://arxiv.org/abs/1905.05621
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Elazar, Y., Goldberg, Y.: Adversarial removal of demographic attributes from text data. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 11–21. Association for Computational Linguistics, Brussels, Belgium (Oct-Nov 2018). https://doi.org/10.18653/v1/D18-1002, https://aclanthology.org/D18-1002
7. Fu, Z., Tan, X., Peng, N., Zhao, D., Yan, R.: Style transfer in text: Exploration and evaluation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
8. Hu, Z., Yang, Z., Liang, X., Salakhutdinov, R., Xing, E.P.: Controllable text generation. CoRR **abs/1703.00955** (2017), http://arxiv.org/abs/1703.00955
9. Jafaritazehjani, S., Lecorvé, G., Lolive, D., Kelleher, J.: Style versus content: A distinction without a (learnable) difference? In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 2169–2180. International Committee on Computational Linguistics, Barcelona, Spain (Online) (Dec 2020). https://doi.org/10.18653/v1/2020.coling-main.197, https://aclanthology.org/2020.coling-main.197
10. Jafaritazehjani, S., Lecorvé, G., Lolive, D., Kelleher, J.D.: Style as sentiment versus style as formality: The same or different? In: ICANN (2021)
11. Jin, D., Jin, Z., Hu, Z., Vechtomova, O., Mihalcea, R.: Deep Learning for Text Style Transfer: A Survey. Computational Linguistics **48**(1), 155–205 (04 2022). https://doi.org/10.1162/coli$_{a0}$0426, $https://doi.org/10.1162/coli\_a\_00426$
12. John, V., Mou, L., Bahuleyan, H., Vechtomova, O.: Disentangled representation learning for non-parallel text style transfer. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 424–434 (2019)
13. Kelleher, J.D.: Deep Learning. MIT Press (2019)

14. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751 (2014)
15. Lample, G., Subramanian, S., Smith, E., Denoyer, L., Ranzato, M., Boureau, Y.L.: Multiple-attribute text rewriting. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=H1g2NhC5KQ
16. Leeftink, W., Spanakis, G.: Towards controlled transformation of sentiment in sentences. In: Proceedings of the 11th International Conference on Agents and Artificial Intelligence-Volume 2: ICAART. pp. 809–816. SCITEPRESS (2019)
17. Li, J., Jia, R., He, H., Liang, P.: 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). pp. 1865–1874 (2018)
18. Li, J., Jia, R., He, H., Liang, P.: 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). pp. 1865–1874 (2018)
19. Ma, S., Sun, X.: A semantic relevance based neural network for text summarization and text simplification. Computational Linguistics **Volume: 1**(1) (2017)
20. Nedumpozhimana, V., Kelleher, J.: Finding BERT's idiomatic key. In: Proceedings of the 17th Workshop on Multiword Expressions (MWE 2021). pp. 57–62. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.mwe-1.7, https://aclanthology.org/2021.mwe-1.7
21. Nedumpozhimana, V., Klubička, F., Kelleher, J.D.: Shapley idioms: Analysing bert sentence embeddings for general idiom token identification. Frontiers in Artificial Intelligence **5** (2022). https://doi.org/10.3389/frai.2022.813967, https://www.frontiersin.org/article/10.3389/frai.2022.813967
22. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
23. Prabhumoye, S., Tsvetkov, Y., Salakhutdinov, R., Black, A.W.: Style transfer through back-translation. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers. pp. 866–876. Association for Computational Linguistics (2018), http://aclweb.org/anthology/P18-1080
24. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
25. Rao, S., Tetreault, J.R.: Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In: NAACL-HLT (2018)
26. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (11 2019), https://arxiv.org/abs/1908.10084
27. Romanov, A., Rumshisky, A., Rogers, A., Donahue, D.: 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). pp. 815–825 (2019)
28. Shen, T., Lei, T., Barzilay, R., Jaakkola, T.: Style transfer from non-parallel text by cross-alignment. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Proceedings of the Conference in

Neural Information Processing Systems 30 (NIPS), pp. 6830–6841. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7259-style-transfer-from-non-parallel-text-by-cross-alignment.pdf

29. Singh, A., Palod, R.: Sentiment transfer using seq2seq adversarial autoencoders. CoRR **abs/1804.04003** (2018), http://arxiv.org/abs/1804.04003
30. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of the Conference in Neural Information Processing Systems (NIPS). pp. 3104–3112 (2014)
31. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
32. Zhao, J., Kim, Y., Zhang, K., Rush, A., LeCun, Y.: Adversarially regularized autoencoders. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 5902–5911. PMLR, Stockholmsmässan, Stockholm Sweden (10–15 Jul 2018), http://proceedings.mlr.press/v80/zhao18b.html