


ARTICLE

# Masked transformer through knowledge distillation for unsupervised text style transfer

Arthur Scalercio and Aline Paes 

Institute of Computing, Universidade Federal Fluminense, Niterói, RJ, Brazil

**Corresponding author:** Aline Paes; Email: [alinepaes@ic.uff.br](mailto:alinepaes@ic.uff.br)

(Received 17 December 2021; revised 11 May 2023; accepted 13 June 2023; first published online 25 July 2023)

## Abstract

Text style transfer (TST) aims at automatically changing a text's stylistic features, such as formality, sentiment, authorial style, humor, and complexity, while still trying to preserve its content. Although the scientific community has investigated TST since the 1980s, it has recently regained attention by adopting deep unsupervised strategies to address the challenge of training without parallel data. In this manuscript, we investigate how relying on sequence-to-sequence pretraining models affects the performance of TST when the pretraining step leverages pairs of paraphrase data. Furthermore, we propose a new technique to enhance the sequence-to-sequence model by distilling knowledge from masked language models. We evaluate our proposals on three unsupervised style transfer tasks with widely used benchmarks: author imitation, formality transfer, and polarity swap. The evaluation relies on quantitative and qualitative analyses and comparisons with the results of state-of-the-art models. For the author imitation and the formality transfer task, we show that using the proposed techniques improves all measured metrics and leads to state-of-the-art (SOTA) results in content preservation and an overall score in the author imitation domain. In the formality transfer domain, we paired with the SOTA method in the style control metric. Regarding the polarity swap domain, we show that the knowledge distillation component improves all measured metrics. The paraphrase pretraining increases content preservation at the expense of harming style control. Based on the results reached in these domains, we also discuss in the manuscript if the tasks we address have the same nature and should be equally treated as TST tasks.

**Keywords:** Evaluation; Machine learning; Natural language generation; Style transfer

## 1. Introduction

Natural language generation (NLG) (Gatt and Krahmer 2018) is a subfield of natural language processing aiming to enable computers the ability to write correct, coherent, and appealing texts. NLG includes popular tasks like machine translation (Bahdanau, Cho, and Bengio 2015), summarization (Rush, Chopra, and Weston 2015), and dialogue response generation (Yarats and Lewis 2018).

An increasingly prominent task in NLG is *text style transfer* (TST). TST aims at transferring a sentence from one style to another without appreciably changing the content (Hu *et al.* 2022; Jin *et al.* 2022). TST encompasses several sub-tasks, including sentiment transfer, news rewriting, storytelling, text simplification, and writing assistants, among others. For example, author imitation (Xu *et al.* 2012) is the task of paraphrasing a sentence to fit another author's style. Automatic poetry generation (Ghazvininejad *et al.* 2016) applies style transfer to create poetry in different fashions.



TST inherits the challenges of NLG, namely the lack of parallel training corpora and reliable evaluation metrics (Reiter and Belz 2009; Gatt and Krahrmer 2018). Since parallel examples from each domain style are usually unavailable, most style transfer works focus on the unsupervised configuration (Han, Wu, and Niu 2017; He *et al.* 2020; Malmi, Severyn, and Rothe 2020a). Our work also follows the *unsupervised style transfer setting*, learning from non-parallel data only.

Most previous models that address the style transfer problem adopt the sequence-to-sequence encoder-decoder framework (Shen *et al.* 2017; Hu *et al.* 2017; Fu *et al.* 2018). The encoder aims at extracting a style-independent latent representation while the decoder generates the text conditioned on the disentangled latent representation plus a style attribute. This family of methods learns how to disentangle the content and style in the latent space. Disengaging the content from the style means that it is impossible to recover the style from the content. Nevertheless, Lample *et al.* (2019) show that disentanglement is hard to do, difficult to judge the quality, and particularly unnecessary. Thus, following previous work (Lample *et al.* 2019; Dai *et al.* 2019), we make no assumptions about the disentangled latent representations of the input sentences.

We rely on transformer (Vaswani *et al.* 2017) as our base neural sequence-to-sequence (Seq2Seq) architecture. The transformer is a deep neural network that has succeeded in many NLP tasks, particularly when it allies with pretrained masked language models (MLM), such as BERT (Devlin *et al.* 2019). However, using MLM in text generation tasks is less prevalent. This is because models like BERT focus on encoding bidirectional representations through the masked language modeling task, while text generation better fits an auto-regressive decoding process.

We explore training the Seq2Seq model with two procedures. One approach trains it from scratch with the dataset itself, and the other pretrains the model using available paraphrased data before presenting it to the dataset examples. We show that pretraining a Seq2Seq model on a massive paraphrase data benefits TST tasks, mainly the ones that can be considered rewriting tasks. Furthermore, we investigate if we can leverage *masked language models* to benefit the style transfer task, even though their use is less widespread than auto-regressive models for language generation-based tasks. Notably, we want to investigate if TST with a masked language model can output texts with the desired style while still preserving the input text main topic or theme. The investigation considers the following research questions:

RQ1: *Does extracting knowledge from a Masked Language Model improve the performance of Seq2Seq models in the style transfer task and consequently generate high-quality texts?*

RQ2: *What is the impact of pretraining the Seq2Seq model on paraphrase data to the style transfer task?*

To answer the research questions, we build our model upon the neural architecture block proposed in Dai *et al.* (2019). We leverage their transformer neural network and training strategies. Our adopted transformer network is similar to the original one (Vaswani *et al.* 2017), except for an additional style embedding inserted into the encoder as the first embedding component. Regarding the training techniques, we inherited the adversarial training and the back-translation techniques of Dai *et al.* (2019). Nevertheless, we formulate the main cost function to extract knowledge from a MLM.

To show we can take advantage of an MLM to improve the performance of the style transfer task, we try to *distill the knowledge* of a pretrained MLM to leverage its learned bidirectional representations. We modify the training objective by transferring the knowledge it contains to the Seq2Seq model. We hypothesize that the predictive power of an MLM improves the performance of TST. As our model uses both Transformers and a MLM for training, we call it MATTES (MAsked Transformer for TExt Style transfer).<sup>3</sup> Furthermore, to evaluate if pretraining the Seq2Seq model benefits the TST task, we select a large-scale collection of diverse paraphrase

<sup>3</sup>The source code is available at <https://github.com/MeLLL-UFF/mattes>.

data, namely, the PARABANK 2 dataset (Hu *et al.* 2019). For comparison, we experimented training from scratch with the dataset examples and starting from the pretrained model.

For evaluating the proposed model and learning strategies, we select the author imitation (Xu *et al.* 2012), formality transfer (Rao and Tetreault 2018), and the polarity swap (Li *et al.* 2018) tasks, all in English, given the availability of benchmark datasets. The first task aims at paraphrasing a sentence in an author's style. The second task consists of rewriting a formal sentence into its informal counterpart and vice-versa. The remaining task, also known as sentiment transfer, aims at reversing the sentiment of a text from positive to negative and vice-versa, preserving the overall theme. These tasks are usually gathered in the literature under the general style transfer label and addressed with the same methods. However, they have critical differences: author imitation and formality transfer imply rephrasing a sentence to match the desired target style without changing its meaning. On the other hand, polarity swap aims to change a text with positive polarity into a text with negative polarity (or vice-versa). In this case, while the general topic must be preserved, the meaning is not maintained (e.g., turning "I had a bad experience" into "I had a great experience"). In this sense, author imitation and formality transfer can be seen much more as rewriting than polarity swap. This way, it is possible to consider them similar to the broader objective of paraphrasing. In this manuscript, we also address these tasks similarly, but this is to investigate how their different nature affects the models, task modeling, and evaluation.

Regarding task evaluation, the literature commonly assesses the performance of TST on style strength and content preservation. To verify that the generated text agrees with the desired style, we train a classifier using texts of both styles, with style as the class, and measure its accuracy. To measure content preservation, the most important feature a style transfer model should possess, three metrics were used: BLEU (Papineni *et al.* 2002), semantic similarity (SIM) (Wieting *et al.* 2019), and BARTScore (Yuan, Neubig, and Liu 2021). The results pointed out that using a pretrained Seq2Seq as starting point and using a pretrained MLM throughout the main training of the model improves the quality of the generated texts. We show that the former is extremely helpful for rewriting tasks while the latter is more task agnostic oriented but slightly impacts the performance.

The contributions of this paper are:

1. A novel unsupervised training method that distills knowledge from a pretrained MLM. To our knowledge, this is the first study that uses an MLM within the training objective on the style transfer task. We show that extracting the rich bidirectional representations of an MLM benefits the TST task.
2. An evaluation of two training strategies of the Seq2Seq model. We show that rewriting tasks, such as author imitation and formality transfer, benefit from a Seq2seq model pretrained with paraphrase data.
3. In the experiments with author imitation and formality transfer, we achieve state-of-the-art results when pretraining the Seq2Seq model with paraphrase data and using our distillation training technique.

The rest of the manuscript is organized as follows. Section 2 briefly describes basic concepts related to our proposal. Section 3 reviews related work to highlight how MATTES is placed within the literature on the theme. Next, we state the problem tackled here and present our approach in Section 4. Section 5 devises the experiments, and Section 6 concludes the manuscript and points out future directions.

## 2. Key concepts

This section reviews essential concepts that form the architectural backbone of our proposal. We describe the general paradigm widely used to handle NLP tasks that demand transforming a sequence of tokens into another sequence. Next, we review the Transformer architecture, which

relies on the attention mechanism to enhance the sequence generation. Finally, we discuss the autoregressive and MLMs, as we rely on the latter to build MATTES.

### 2.1 Sequence-to-sequence models with the transformer architecture

Sequence-to-Sequence (Seq2Seq) models are frameworks based on deep learning to address tasks that require obtaining a sequence of values as output from a sequence of input values (Sutskever, Vinyals, and Le 2014). At a high level, a Seq2Seq model comprises two neural networks, one encoding the input and the other decoding the encoded representation to produce the output.

Early models included recurrent neural networks as the encoder and decoder components. In this case, the encoder's role is to read the input sequence  $X = \langle x_1, \dots, x_n \rangle$ , where  $x_i$  is a token, and generate a fixed-dimension vector  $C$  representing the context. Then, the decoder generates the output sequence  $Y = \langle y_1, \dots, y_m \rangle$ , starting from the context vector  $C$ . However, compressing a variable-length sequence into a single context vector is challenging, especially when the input sequence is long. Thus, Seq2Seq models with vanilla recurrent networks fail to capture long textual dependencies due to the information bottleneck when relying on a single context vector.

The attention mechanism emerged to address the information bottleneck problem of seq2seq recurrent neural networks (Bahdanau *et al.* 2015). Instead of using only one context vector at the end of the encoding process, the attention vector provides the decoding network with a full view of the input sequence at every step of the decoding process. Thus, in the output generation process, the decoder can decide which tokens are important at any given time.

Using only attention mechanisms and dismissing recurrent and convolutional components, Vaswani *et al.* (2017) created an architecture named *Transformer*, which was successful in numerous NLP tasks (Radford *et al.* 2018; Devlin *et al.* 2019). The Transformer architecture follows the Seq2Seq paradigm comprising an encoder–decoder architecture. The encoder consists of six identical encoding layers. Each comprises a multihead self-attention mechanism, residual connections, an add-norm mechanism, and a feed-forward network. The decoder also consists of six identical layers similar to the encoding layers. However, the decoder gets two inputs and applies the attention twice, where, in one of them, the input is masked. This prevents the token in a certain position from having access to tokens after it during the generation process. Also, the final decoder layer has a size equal to the number of words in the vocabulary.

### 2.2 Autoregressive and masked language models

Language models compute the probability of the occurrence of a token given a context. Autoregressive language models compute the probability of occurrence of a token, given the previous tokens in the sequence. Thus, given a sequence  $\mathbf{x} = (x_1, x_2, \dots, x_m)$ , an autoregressive language model calculates the probability  $p(x_t | x_{<t})$ . The probability of a sequence of  $m$  tokens  $x_1, x_2, \dots, x_m$  is given by  $P(x_1, x_2, \dots, x_m)$ . Since it is computationally expensive to enumerate all possible combinations of tokens that come before a token,  $P(x_1, x_2, \dots, x_m)$  is usually conditioned to a window of  $n$  previous tokens instead of all the previous ones. In these cases,

$$P(x_1, x_2, \dots, x_m) = \prod_{i=1}^m P(x_i | x_1, \dots, x_{i-1}) \approx \prod_{i=1}^m P(x_i | x_{i-n}, \dots, x_{i-1}) \quad (1)$$

Pretraining neural language models using self-supervision from a large volume of texts has been highly effective in improving the performance of various NLP tasks (Peters *et al.* 2018; Radford *et al.* 2018; Howard and Ruder 2018; Devlin *et al.* 2019). The pretrained language models can be later adjusted with fine-tuning to handle specific downstream tasks or domains.

Different self-supervised goals were explored in the literature to pretrain a language model, including autoregressive and MLM strategies (Yang *et al.* 2019, Devlin *et al.* 2019; Yang *et al.*

2019; Lan *et al.* 2020; Lan *et al.* 2020). An autoregressive language model, given a sequence  $\mathbf{x} = (x_1, x_2, \dots, x_m)$ , factors the probability into a left-to-right product  $p(\mathbf{x}) = \prod_{t=1}^T p(x_t | x_{<t})$  or from right-to-left  $p(\mathbf{x}) = \prod_{t=T}^1 p(x_t | x_{>t})$ . The problem with these models is unidirectionality since, during training, a token can only be aware of the tokens to its left or right. Many works point out that it is fundamental for several tasks to obtain bidirectional representations incorporating the context of both the left and the right (Devlin *et al.* 2019).

Devlin *et al.* (2019) introduced the task of masked language modeling (MLM) with BERT, a transformer-encoder. MLM consists of replacing a percentage (in the original work, 15%) of the sequence tokens with a token [MASK] and then predicting those masked tokens. In BERT (Devlin *et al.* 2019), the probability distribution of the masked tokens regards both the left and right contexts of the sentence.

### 2.2.1 ALBERT: A lite version of BERT

Overall, increasing the size of a pretrained neural language model positively impacts subsequent tasks. However, growing the model becomes unfeasible at some point due to GPU/TPU memory limitations and huge training time. ALBERT (Lan *et al.* 2020) addressed these problems with two techniques for reducing parameters. This manuscript relies on an ALBERT model to extract knowledge from its rich contextualized representations. Although any other MLM model could have been selected, we adopt ALBERT because it uses fewer computational resources when compared to a BERT of the same size.

The architectural skeleton of ALBERT is similar to BERT, which means that it uses a transformer encoder with GELU activation function (Hendrycks and Gimpel 2016). ALBERT makes three main contributions to BERT design choices, as follows. Following BERT notation,  $E$  is the size of the vocabulary vector representations,  $L$  is the number of encoder layers, and  $H$  is the hidden layers' representation size.

*Parameter factorization.* BERT ties  $E$  to the size of the hidden states representation  $H$ , that is,  $E = H$ , which is not efficient, both for modeling and practical reasons. From a modeling point of view, the learned representations for vocabulary tokens are context-independent, while the hidden layers of learned representations are context-dependent. As the representational power of BERT lies in the possibility of obtaining rich contextualized representations from non-contextualized representations, untying  $E$  from  $H$  is a more efficient use of model parameters. From a practical point of view, as in most NLP tasks, the vocabulary size  $V$  is usually large; if  $E = H$ , increasing  $H$  increases the representation matrix of vocabulary tokens, which is of length  $V \times E$ . This can result in a model with billions of parameters, many of which are sparsely updated during training. Thus, ALBERT decomposes the matrix of vocabulary representations into two smaller matrices, reducing the parameters from  $O(V \times H)$  to  $O(V \times E + E \times H)$ . This is a significant reduction when  $H \gg E$ .

*Shared parameters between layers.* ALBERT has a scheme for sharing parameters between layers to improve handling parameters efficiently. Although it is possible to share the parameters between the layers partially, the default configuration shares all the parameters, both neural network and attention parameters. With this, it is possible to increase the depth without increasing the number of parameters.

*Coherence component between sentences in the loss function.* BERT has an extra component in the loss function besides predicting the masked tokens, called next-sentence prediction (NSP). During training, NSP learns whether two sentences appear consecutively in the original text. Arguing that this component is not practical as a task when compared to the MLM task, Lan *et al.* (2020) proposed a component called *sentence-order prediction* (SOP). This component uses as a positive example two consecutive text segments from the same document (such as BERT).

As a negative example, the same two segments, but with the order changed. The results indicated that ALBERT improved performance in subsequent tasks that involved coding more than one sentence.

Although masked pretrained models, such as BERT and ALBERT, benefit several NLP tasks, they fit less text generation tasks since they are transformer encoders, and decoders are more suitable to generate texts. However, it is common when writing a text that words that appeared before can be changed after writing a later sequence, giving an intuitive idea of context and bidirectionality. In order to make use of rich bidirectional representations from a MLM, MATTES distills knowledge from ALBERT to benefit the TST task.

### 3. Literature review and related work

TST aims to automatically change stylistic features, such as formality, sentiment, author style, humor, and complexity, while trying to preserve the content. This manuscript focuses on unsupervised TST, given that creating a parallel corpus of texts with different styles is challenging and requires much human effort. In this scenario, the approaches differ in investigating how to disentangle style and content or not disentangling them at all (Hu *et al.* 2022). Lample *et al.* (2019) argued that it is difficult to judge whether content and style representations obtained are disentangled and that disentanglement is not necessary for the TST task either. Recent studies, including this manuscript, explore the TST task without disentangling content and style. Next, we elicit works that explicitly decouple the content from the style, try to detach them implicitly using latent variables, or do not rely on disentanglement strategies to position our work within the current TST literature.

#### 3.1 Explicit disentanglement

Models following this strategy generate texts through **direct replacement** of keywords associated with the style (Li *et al.* 2018; Xu *et al.* 2018; Zhang *et al.* 2018a; Sudhakar, Upadhyay, and Maheswaran 2019; Wu *et al.* 2019a, 2019b; Malmi, Severyn, and Rothe 2020b).

The method *Delete, Retrieve, Generate* (Li *et al.* 2018) explicitly replaces keywords in a text with words of the target style. First, it removes the words that best represent the original style. Then, it fetches the text most similar to the input from the target corpus. Next, it extracts the words most closely associated with the target style from the returned text and combines them with the sentence acquired in the first step to generate the output text using a sequence-to-sequence neural network model. Sudhakar *et al.* (2019) extended the model *Delete, Retrieve, Generate* to improve the *Delete* step using a transformer (Vaswani *et al.* 2017). The method *POINT-THEN-OPERATE* (Wu *et al.* 2019) also changes the input sentence but relies on hierarchical reinforcement learning. One RL agent points out the positions that should be edited in the sentence, and another RL agent changes it.

Zhang *et al.* (2018a) adopted a keyword substitution technique similar to *Delete, Retrieve, Generate* to transfer sentiment in texts. Also focusing on sentiment transfer, Xu *et al.* (2018) developed a model with a *neutralization* and an *emotionalization* components. The former extracts semantic information without emotional content, while the latter adds sentiment content to the neutralized positions.

The *Mask and Infill* method (Wu *et al.* 2019b) works in two stages. First, it masks words associated with the style using frequency rates. Next, it fills the masked positions with the target style using a pretrained MLM. Malmi *et al.* (2020b) also used a pretrained MLM to remove snippets and generate the replacement snippets. Although these works rely on MLMs, they differ from our proposal as they use language models only to predict tokens to replace previously removed ones.



### 3.2 Implicit disentanglement

Models focusing on implicit disentanglement detach content and style from the original sentence but do not explicitly alter the original sentence. They learn latent content and style representations for a given text to separate content from style. Then, the content representation is combined with the target style representation to generate the text in the target style.

Most recent solutions leverage adversarial learning (Hu *et al.* 2017; Shen *et al.* 2017; Fu *et al.* 2018; Zhao *et al.* 2018b; Chen *et al.* 2018; Logeswaran, Lee, and Bengio 2018; Lai *et al.* 2019; John *et al.* 2019; Yin *et al.* 2019) to obtain style-agnostic representations of the sentence content. After learning the latent content representation, the decoder receives as input that representation along with the label of the desired style to generate a variation of the input text with the desired style. Yang *et al.* (2018) followed that strategy and used two language models, one for each stylistic domain. The model minimizes the perplexity of sentences generated according to these pretrained language models.

Other techniques separate content from the style by artificially generating parallel data via back-translation, then converting them back to the original domain, forcing them to be the same as the input. Besides addressing the lack of parallel data, such a strategy can normalize the input sentence by stripping away information that is predictive of its original style. With back-translation, one-direction outputs and inputs can be used as pairs to train the model of the opposite transfer direction. Prabhumoye *et al.* (2018) used an English-French neural translation model to rephrase the sentence and remove the stylistic properties of the text. The English sentence is initially translated into French. The French text is then translated back to English using a French-English neural model. Finally, this style-independent latent representation learned is used to generate texts in a different style using a multiple-decoder approach. Zhang *et al.* (2018b) followed back-translation first to create pseudo-parallel data. Then, those data initialize an iterative back-translation pipeline to train two style transfer systems based on neural translation models. Krishna *et al.* (2020) used a paraphrasing model to create pseudo-parallel data and then trained style-specific inverse paraphrase models that convert these paraphrased sentences back into the original stylized sentence. Despite adopting back-translation, our proposal does not expect that the model output is deprived of style information. On the contrary, during training, we try to control the style of the generated output.

Some disentanglement strategies explore learning a style attribute to control the generation of texts in different styles. The method presented in (Hu *et al.* 2017) induces a model that uses a variational autoencoder (VAE) to learn latent representations of sentences. These representations are composed of unstructured variables (content)  $z$  and structured variables (style)  $c$  that aim to represent salient and independent features of the sentence semantics. Finally,  $z$  and  $c$  are inserted into a decoder to generate text in the desired style. Tian *et al.* (2018) extended this approach, adding constraints to preserve style-independent content, using Part-of-speech categories and a content-conditioned language model. Zhao, Kim, Zhang, Rush and LeCun (2018) proposed a regularized adversarial autoencoder that expands the use of adversarial autoencoders to discrete sequences. Park *et al.* (2019) relied on adversarial training and VAE to expand previous methods to generate paraphrases guided by a target style.

### 3.3 Non-disentanglement approaches

Although the models included in this category do not assume the need to disentangle content and style from input sentences, they also rely on controlled generation, adversarial learning, reinforcement learning, back-translation, and probabilistic models, similar to some models in the previous categories.

Jain *et al.* (2019) proposed a framework for controlled natural language transformation that consists of an encoder–decoder neural network reinforced by transformations carried out through auxiliary modules. Zhang, Ding, and Soricut (2018) devised the SHAPED method with an architecture that has shared parameters updated from all training examples and not-shared parameters updated with only examples from their respective distributions. Zhou *et al.* (2020) proposed a Seq2Seq model that dynamically evaluates the relevance of each output word for the target style. Lample *et al.* (2019) also used the strategy of learning attribute representations to control text generation. They demonstrated that it is difficult to prove that the style is separated from the disentangled content representation and that performing this disentanglement is unnecessary for the TST task to succeed. Dai *et al.* (2019) proposed a transformer-based architecture with trainable style vectors. Our proposal follows this architecture, but it differs in the training strategy since it alters the loss function of the generating network to extract knowledge from a MLM.

The methods presented in Mueller, Gifford, and Jaakkola (2017); Wang, Hua, and Wan (2019); Liu *et al.* (2020); Xu, Cheung, and Cao (2020) have in common the fact that they manipulate the hidden representations obtained from the input sentence to generate texts in the desired style. The method proposed in Mueller *et al.* (2017) comprises a recurrent VAE and an output predictor neural network. By imposing boundary conditions during optimization and using the VAE decoder to generate the revised sentences, the method ensures that the transformation is similar to the original sentence, is associated with better outputs, and looks natural. The method devised in Liu *et al.* (2020) has three components: (1) a VAE, which has an encoder that maps the sentence to a smooth continuous vector space and a decoder that maps back the continuous representation to a sentence; (2) attribute predictors, which use the continuous representation obtained by the VAE as input and predict the attributes of the output sentence; and (3) content predictors of a Bag-of-words (BoW) variable for the output sentence. The method proposed in Wang *et al.* (2019) comprises an autoencoder based on Transformers to learn a hidden input representation. Next, the task becomes an optimization problem that edits the obtained hidden representation according to the target attribute. VAE is also the core of the method proposed in Xu *et al.* (2020) to control text generation. The method proposed in He *et al.* (2020) addresses the TST task with unsupervised learning, formulating it as a probabilistic deep generative model, where the optimization objective arises naturally, without the need to create artificial custom objectives.

Gong *et al.* (2019) leveraged reinforcement learning with a generator and an evaluator network. The evaluator is an adversarially trained style discriminator with semantic and syntactic constraints punctuating the sentence generated by style, content preservation, and fluency. The method proposed in Luo *et al.* (2019) also uses reinforcement learning and considers the problem of transferring style from one domain to another as a dual task. To this end, two rewards are modeled based on this framework to reflect style control and content preservation.

Lai *et al.* (2021) created a three-step procedure on top of the large pretrained seq2seq model BART (Lewis *et al.* 2020). First, they further pretrain the BART model on an existing collection of generic paraphrases and synthetic pairs created using a general-purpose lexical resource. Second, they use iterative back-translation with several reward strategies to train two models simultaneously, each in a transfer direction. Third, using their best systems from the previous step, they create a static resource of parallel data. These pairs are used to fine-tune the original BART with all reward strategies in a supervised way. As this model, ours also has a preliminary pretraining phase on the same paraphrase data. Nevertheless, our main training procedure differs, relying mostly on adversarial and distillation techniques for style control.

Table 1 exhibits different features of some works from the last 5 years that, like MATTES, do not follow any disentanglement strategy to decouple content from style in the unsupervised TST task.



**Table 1.** Non-disentaglemet publications according to model characteristics

Model	Model features							
	Multiple styles	Back-translation	Adversarial training	Pre-trained LM	Pre-trained MLM	Transformers-based	VAE	Human evaluation
Mueller <i>et al.</i> (2017)							X	
Lample <i>et al.</i> (2019)	X	X						X
Dai <i>et al.</i> (2019)		X	X			X		X
Jain <i>et al.</i> (2019)				X				X
Wang <i>et al.</i> (2019)						X		X
Gong <i>et al.</i> (2019)			X	X				X
Luo <i>et al.</i> (2019)								X
Zhou <i>et al.</i> (2020)				X				X
Liu <i>et al.</i> (2020)	X						X	X
Xu <i>et al.</i> (2020)							X	
He <i>et al.</i> (2020)		X		X			X	
Krishna <i>et al.</i> (2020)		X		X		X		X
Kim and Sohn (2020)						X		
Yi <i>et al.</i> (2020)			X					X
Chen <i>et al.</i> (2021)	X		X			X		X
Goyal <i>et al.</i> (2021)	X			X		X		X
Liu, Neubig, and Wieting (2021)			X	X		X		X
<b>MATTES</b>		X	X		X	X		X

#### 4. MATTES: Masked transformer for text style transfer

This Section presents MATTES, our proposed approach that distills knowledge from a MLM aiming at improving the quality of the text generated by a Seq2Seq model for the TST task. The MLM adopted here is ALBERT (Lan *et al.* 2020), a lighter version of the popular BERT that still gets better or competitive results but relies on parameter reduction techniques to improve training efficiency and reduce memory costs.

MATTES follows the architecture established in Dai *et al.* (2019), which in turn uses an adversarial training framework (Radford, Metz, and Chintala 2016). Training adopts two neural networks. One is a discriminator network operating only during training. The discriminator is a style classifier aiming at making the model learn to differentiate the original and the reconstructed sentences from the translated sentences. The other neural network is the text generator. It comprises an encoder and a decoder, based on a transformer architecture. The generator network receives as input a sentence  $X$  and a target output style  $s^{\text{tgt}}$  and produces a sentence  $Y$  in the target style, making the proposed model a mapping function  $Y = f_{\theta}(X, s^{\text{tgt}})$ . The final model learned by MATTES and used for inference is the generator network, while the discriminator network is used only during training.

Before going into further details about the specific components of MATTES, Section 4.1 formalizes how we investigate the textual style transfer task in this manuscript. Afterward, the main paradigms that constitute the proposed method are presented: Section 4.2 presents the Seq2Seq learning component, and Section 4.3 devises the MLM and the knowledge distillation method that extracts knowledge from the MLM. Finally, Section 4.4 describes the adversarial learning algorithm proposed here and gives additional details about the model architecture and design choices.

##### 4.1 Problem formulation

This manuscript assumes the styles as elements of a set  $S$ . For example,  $S = \{\text{positive, negative}\}$  for the polarity swap task, where the text style can be positive or negative. For training the model, there is a set of sentences  $D = \{(X_1, s_1), \dots, (X_k, s_k)\}$  labeled with their style, *that is*,  $X_i$  is a sentence and  $s_i \in S$  is the style attribute of the sentence. From  $D$ , we extract a set of style sentences  $D_s = \{X: (X, s) \in D\}$ , which represent all the sentences of  $D$  with attribute  $s$ . In the polarity swap task, it would be all sentences with attribute positive, for example. All sequences in the same dataset  $D_i$  share specific characteristics related to the style of the sequences.

The goal of the textual style transfer learning task tackled in this manuscript is to build a model that receives as input a sentence  $X$  and a target style  $s^{\text{tgt}}$ , where  $X$  is a sentence with style  $s^{\text{src}} \neq s^{\text{tgt}}$ , and produces a sentence  $Y$  that preserves as much as possible the content of  $X$  while incorporating the style  $s^{\text{tgt}}$ . MATTES addresses the problem of style transfer with unsupervised machine learning. Thus, the only data available for training are the sequences  $X$  and their style source  $s^{\text{src}}$ . MATTES do not have access to a template sentence  $X^*$ , which would be the conversion of  $X$  to the target style  $s^{\text{tgt}}$ .

When we adopt the strategy of pretraining the Seq2Seq model, there is another style element in the set  $S$  that we call the paraphrase style. For the polarity swap task,  $S$  would be  $S = \{\text{positive, negative, paraphrase}\}$ , for example. This new style is not part of the domain dataset, and it only exists to make it possible to start from the pretrained model in the overall training.

##### 4.2 Sequence-to-sequence model

This section describes the sequence-to-sequence architecture and how it is pretrained on a large amount of generic data to improve the model.

4.2.1 Sequence-to-sequence learning architecture

As usual, we adopt the Seq2Seq encoder–decoder paradigm to solve the TST task. Formally, during learning, the model trains to generate an output sequence  $Y = (y_1, \dots, y_N)$  of length  $N$ , conditioned on the input sequence  $X = (x_1, \dots, x_M)$  of length  $M$ , where  $x_i \in X$  and  $y_i \in Y$  are tokens. The encoder–decoder neural network achieves the goal of generating the output sequence by learning a conditional probability distribution  $P_\theta(Y|X)$ , by minimizing the cross entropy loss function  $\mathcal{L}(\theta)$

$$\mathcal{L}(\theta) = -\log P_\theta(Y|X) = -\sum_{t=1}^N \log P_\theta(y_t|y_{1:t-1}, X) \tag{2}$$

where  $\theta$  are the model parameters.

Following (Dai *et al.* 2019), in this manuscript, the *Transformer* (Vaswani *et al.* 2017) architecture is adopted as the Seq2Seq model. During the training and inference process, in addition to the embeddings of the input sentence  $X$ , the model also receives as input an embedding of the target style. Hence, the input embeddings passed to the encoder are the target style embedding followed by the embeddings of the input sentence, which is the sum of the token embeddings and the positional embeddings, following the Transformer architecture. Thus, the method aims to learn a model that represents a probability distribution conditioned not only on  $X$  but also on the desired target style  $s^{\text{tgt}}$ . Thus, Equation (2) is modified to meet this characteristic, giving rise to the loss function

$$\mathcal{L}(\theta) = -\log P_\theta(Y|X, s^{\text{tgt}}) = -\sum_{t=1}^N \log P_\theta(y_t|y_{1:t-1}, X, s^{\text{tgt}}) \tag{3}$$

4.2.2 Sequence-to-sequence pre-training

Pretraining and fine-tuning are usually adopted when target tasks have few examples available (Radford *et al.* 2018; Devlin *et al.* 2019). Several pretraining approaches adopt MLM, a kind of denoising auto-encoder trained to reconstruct the text where some random words have been masked. This kind of pretraining has mainly improved the performance of natural language understanding tasks, which can be justified by the fact that MLMs are composed only of a bidirectional encoder, while generation tasks adopt a left-to-right decoder. In this sense, the model BART (Lewis *et al.* 2020) combined bidirectional and auto-regressive transformers to show that sequence-to-sequence pretraining can benefit downstream language generation tasks.

Following this rationale, we pretrain our Seq2Seq model in a large collection of paraphrase pairs (Hu *et al.* 2019). With that, we expect the model to learn the primary task of rewriting, allowing the generation of pseudo-parallel data and, consequently, handling the style transfer in a supervised fashion. We added another style embedding inside our style embedding layer to adapt this technique to our training framework. This way, besides the actual styles of the sentences, there is an additional one that we refer to as  $s^{\text{para}}$ . It is the only style inserted into the model during the pretraining phase. During this phase, we minimize the following loss function

$$\mathcal{L}(\theta) = -\log P_\theta(Y|X, s^{\text{para}}) = -\sum_{t=1}^N \log P_\theta(y_t|y_{1:t-1}, X, s^{\text{para}}) \tag{4}$$

4.3 Knowledge distillation

This section starts by describing the MLM adopted in this manuscript. Next, it devised the knowledge distillation strategy proposed here to leverage the MLM.

### 4.3.1 Masked language model

One of the main contributions of this manuscript is to introduce the ability to transfer knowledge contained in the rich bidirectional contextualized representations provided by a MLM to a Seq2Seq model. MATTES adopts ALBERT (Lan *et al.* 2020) as the MLM, whose architecture is similar to the popular BERT model (Devlin *et al.* 2019) but has fewer parameters.

From the language model learned by ALBERT, one can obtain the probability distribution of the masked tokens according to

$$P_\phi(X^m|X^u) = P_\phi(x_1^m, \dots, x_l^m|X^u) \quad (5)$$

where  $x_*^m \in X^m$  are the masked tokens,  $l$  is the number of masked tokens in  $X$ ,  $X^u$  are the unmasked tokens, and  $\phi$  are ALBERT parameters.

Before training the main model, we fine-tuned ALBERT with the available training dataset. Although ALBERT is prepared to handle a couple of sentences as inputs, given the problem definition, only one sentence is required. This is true either when fine-tuning or when training the main Seq2Seq model. Because of that, MATTES does not adopt ALBERT Sentence-order prediction loss component during the fine-tuning. When training the main TST model, ALBERT parameters do not change. As the next section explains, MATTES uses the probability distribution provided by ALBERT for each token of the input sentence as a label for training the model in one of the components of the loss function. Thus, instead of forcing the model to generate a probability distribution with the entire probability mass in a single token, the model is forced to have a smoother probability distribution, injecting probability mass into several tokens.

### 4.3.2 Knowledge distillation from the masked language model

This section details how we introduce the knowledge distillation strategy into the model loss function. To preserve the content of the original message, several previous TST methods adopt a *back-translation* (BT) strategy (Lample *et al.* 2019; Dai *et al.* 2019; He *et al.* 2020; Lai *et al.* 2021), proposed in Sennrich, Haddow, and Birch (2016) for the machine translation task. BT generates pseudo-parallel sequences for training when no parallel sentences are available in the examples, thus generating latent parallel data. Thus, in the context of the TST task, pairs of sentences are created to train the model by automatically converting sentences from the training set to another style.

During training, the BT component takes a sentence  $X$  and its style  $s$  as input and converts it to a target style target  $\hat{s} \neq s$ , generating the sentence  $\hat{Y} = f_\theta(X, \hat{s})$ . After that, the generated sentence  $\hat{Y}$  is passed as input to the model along with the original style  $s$ , and the network is trained to learn to predict the original input sentence  $X$ . The input sentence is first converted to the target style and then converted back to its original style.

The method proposed in Dai *et al.* (2019) adopts the *back-translation* strategy to learn a TST model, minimizing the negative value of the logarithm of the probability that the generated sentence is equal to the original sentence with

$$\mathcal{L}_{\text{BT}}(\theta) = -\log P_\theta(Y = X|f_\theta(X, \hat{s}), s) \quad (6)$$

where  $f_\theta(X, \hat{s})$  indicates the converted sentence and  $s$  is the original input style.

Sequence-to-sequence models are normally trained from left to right. Thus, when generating each token to compose the sentence, the vocabulary probability distribution is conditioned only to the previous tokens in the sentence. This is to avoid each token seeing itself and the others after it. However, such an approach has the disadvantage of estimating the probability distribution only with the left context.

MATTES uses a configuration called masked transformer during training to overcome this limitation and generate a distribution that includes bidirectional information. The adoption of this

architecture in the style transfer task comes from the observation that the probability distribution of a masked token  $x_t^m$  given by an MLM contains both past and future context information. Thus, as no sentence pairs are available to perform supervised training, the central idea is to force MATTES to generate a distribution provided by ALBERT for each token. By doing that, MATTES smooths the probability curve and spreads probability mass over more tokens. Such additional information can improve the quality of the generated texts and, in particular, the style control of the model. Thus, during the optimization of the loss function component related to the *back-translation* (Equation 6), the target is no longer the distribution in which the entire probability mass is in a single token. Instead, it becomes the probability distribution given by the MLM  $P_\phi(x_t^m|X^u)$  (Equation 5), for each token  $x_t$  of the sentence input. The distribution provided by the MLM becomes a softer target for text generation during training, moving the model away from learning a more abrupt and unreal distribution, where the entire probability mass is in a single token.

Another point that MATTES leverages with MLM is the use of a knowledge distillation scheme (Hinton, Vinyals, and Dean 2015). Distilling consists of extracting the knowledge contained in one model through a specific training technique. This technique is commonly used to transfer information from a large, already trained model, a teacher, to a smaller model, a student, better suited for production. In such a scheme, the student uses the teacher’s output values as the goal, instead of relying on the training set labels. The distillation process controls better the optimization and regularization of the training process (Phuong and Lampert 2019).

Several previous distillation methods train both the teacher and the student in the same task, aiming at model compression (Hinton *et al.* 2015; Sun *et al.* 2019). Here, we have a different goal, as we use distillation to take advantage of pretrained bidirectional representations generated through a MLM. Thus, ALBERT provides smoother labels to be used as targets during training in the knowledge distillation component of the Seq2Seq loss function, inspired by the method proposed in Chen *et al.* (2020), to improve the quality of the generated text.

MATTES benefits from the distillation scheme by making the MLM assume the teacher’s role, while the unsupervised Seq2Seq model behaves like the student. Equation (7) shows how we adapt the *back-translation* component to be a bidirectional knowledge distillation component:

$$\mathcal{L}_{\text{bidi}}(\theta) = - \sum_{t=1}^N \sum_{w \in V} P_\phi(x_t = w|X^u) \cdot \log P_\theta(y_t = w|y_{1:t-1}, f_\theta(X, \hat{s}, s)) \tag{7}$$

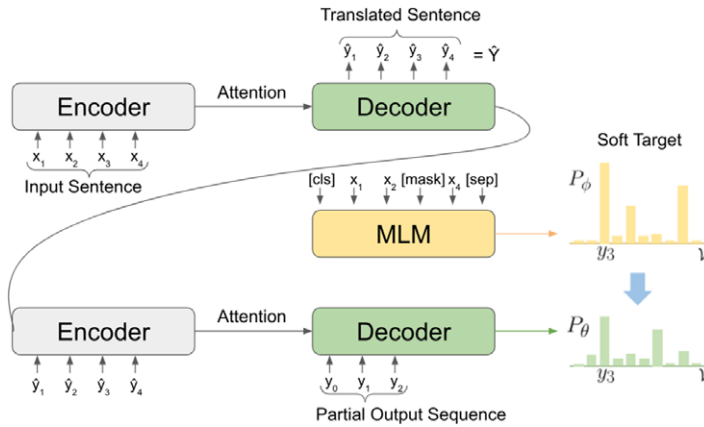
where  $P_\phi(x_t)$  is the soft target provided by the MLM with learned parameters  $\phi$ ,  $N$  is the sentence length, and  $V$  denotes the vocabulary. Note that the parameters of the MLM are fixed during the training process. Figure 1 illustrates the learning process, where the objective is to make the probability distribution of the word  $P_\theta(y_t)$ , provided by the student, be closer to the distribution provided by the teacher,  $P_\phi(x_t)$ .

MATTES is trained with an adapted *back-translation* method that implements a *knowledge distillation procedure*:

$$\mathcal{L}_{\text{KD}}(\theta) = \alpha \mathcal{L}_{\text{bidi}}(\theta) + (1 - \alpha) \mathcal{L}_{\text{BT}}(\theta) \tag{8}$$

where  $\alpha$  is a hyperparameter to adjust the relative importance of the soft targets provided by the MLM and the original targets.

With the introduction of the new loss function term, the distribution is forced to become smoother during training. When the traditional *one-hot* representations are used as a target during training, the model is forced to generate the correct token. All other tokens do not matter to the model, treating equally tokens that would be more likely to occur and tokens with almost zero chance of occurring. In this way, by relying on a smoother distribution, the model increases its potential to generate more fluent sentences as the probability spreads over more tokens. Also,



**Figure 1.** Training illustration when the model is predicting the token  $y_3$  using an MLM.  $P_\theta$  is the student distribution, while  $P_\phi$  is the teacher soft distribution provided by the MLM.

a smoother distribution avoids token generation entirely out of context and better controls the desired style.

**4.4 Learning algorithm**

Training the model proposed here follows the algorithm defined in Dai *et al.* (2019). Both the discriminator and the generator networks are trained adversarially. First, we describe how to train the discriminator network and then the generator, where the masked transformer proposed in this manuscript is inserted.

**4.4.1 Learning the discriminator network**

The discriminator neural network is a multiclass classifier with  $K + 1$  classes.  $K$  classes are associated with  $K$  different styles, and the remaining class refers to the converted sentences generated by the masked Transformer. The discriminator network is trained to distinguish the original and the reconstructed sentences from the translated sentences. It means the classifier is trained to classify a sentence  $X$  with style  $s$  and its reconstructed sentence  $Y$ , as belonging to the class  $s$ , and the translated  $\hat{Y}$  as belonging to the class of translated sentences. Accordingly, its loss function is defined as

$$\mathcal{L}_{\text{discriminator}}(\rho) = -\log P_\rho(c|X) \tag{9}$$

where  $\rho$  are the parameters of the discriminator network,  $c$  is the stylistic domain of the sample that can assume  $K + 1$  categories. The parameters  $\theta$  of the generator network are not updated when training the discriminator.

**4.4.2 Learning the generator network**

A reconstruction component, a knowledge distillation component, and an adversarial component compose the final loss function of the masked transformer, as follows.

*Input sentence reconstruction component.* When the model receives as input a sentence  $X$  along with its style  $s$ , the model must be able to reconstruct the original sentence. The following reconstruction component is added to the loss function to make the model achieve this ability:

$$\mathcal{L}_{\text{self}}(\theta) = -\log P_\theta(Y = X|X, s) \tag{10}$$



During training,  $\mathcal{L}_{\text{self}}$  is optimized in the traditional way, that is, from left to right, masking the future context, according to Equation (3). Despite being possible, this component does not adopt the technique of knowledge distillation as we would like to isolate the distillation to a single component and verify its benefit.

*Knowledge distillation component.* To extract knowledge from a pretrained MLM to improve the transductive process of converting a sentence from one style to another, we introduce the following knowledge distillation component

$$\mathcal{L}_{\text{KD}}(\theta) = \alpha \mathcal{L}_{\text{bidi}}(\theta) + (1 - \alpha) \mathcal{L}_{\text{BT}}(\theta) \tag{11}$$

where  $\mathcal{L}_{\text{BT}}$  is as Equation (6) and  $\mathcal{L}_{\text{bidi}}$  is defined as Equation (7). With this, we expect to smooth the probability distribution of the style converter model, producing more fluent sentences resembling the target domain texts.

During training, to generate the knowledge distillation component (Equation 11) of our cost function, the translated sentence style  $\hat{s}$  inserted into the generator network to obtain  $\hat{Y}$  depends on whether we are training from scratch or using the generator network already pretrained on paraphrase data. In the former case, as we do not have the style  $s^{\text{para}}$ , we convert it to the other existing style. On the latter,  $\hat{Y}$  is created according to the paraphrase style  $s^{\text{para}}$ . In both approaches,  $\hat{Y}$  is inserted back into the network along with the original style  $s$  to generate our final probability distribution  $P_\theta$ . This architectural modification unlocks our model to handle multiple styles at once. This way, regardless of the number of style domains, during training, we translate to  $s^{\text{para}}$  and then back to  $s^{\text{src}}$ . In inference time, we first translate to  $s^{\text{para}}$  and then to  $s^{\text{tgt}}$ . Translating to the paraphrase style can be thought of as normalizing the input sentence, stripping out stylistic information regarding the source style.

*Adversarial component.* When adopting the training from scratch approach, if the model is only trained with the sentence reconstruction and knowledge distillation components, it could quickly converge to learn to copy the input sentence, that is, stick to learning the identity function. Thus, to avoid this problematic and unwanted behavior, an adversarial component is added to the cost function to encourage texts converted to style  $\hat{s}$ , different from the input sentence style  $s$ , to get closer to texts from the style  $\hat{s}$ .

In the scenario with the presence of paraphrase style, the adversarial component tries to modify the generator such that the generated paraphrase sentence  $\hat{Y}$  is pushed to become similar to other existing styles of the training set, as long as they are different from  $s^{\text{src}}$ .

Generalizing for both approaches, the converted sentence  $\hat{Y}$  is inserted into the discriminator neural network and, during training, the probability of the generated sentence being of the style  $s^{\text{tgt}}$ , such that  $s^{\text{tgt}} \neq s^{\text{src}}$  and  $s^{\text{tgt}} \neq s^{\text{para}}$ , is maximized through optimization of the loss function defined in Equation 12. The  $\rho$  parameters of the discriminator network are not updated when training the generator network.

$$\mathcal{L}_{\text{adversarial}}(\rho) = -\log P_\rho(c = s^{\text{tgt}} | f_\theta(X, \hat{s})) \tag{12}$$

These three loss functions are merged, and the overall objective for the generator network becomes:

$$a_1 \mathcal{L}_{\text{self}}(\theta) + a_2 \mathcal{L}_{\text{KD}}(\theta) + \mathcal{L}_{\text{adversarial}}(\rho) \tag{13}$$

where  $a_1$  and  $a_2$  are hyperparameters to adjust each component’s importance in the loss function of the generator network.

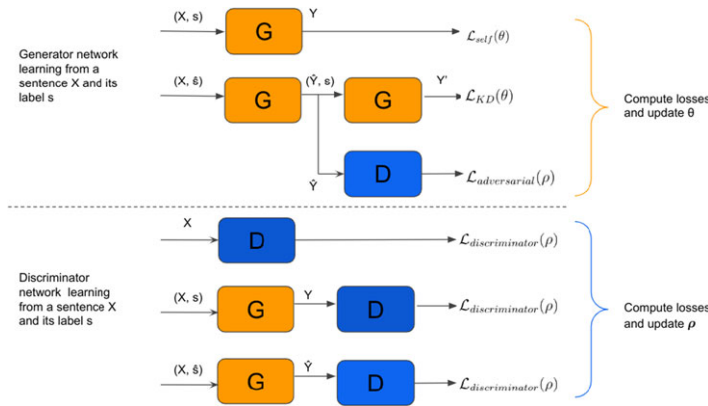


Figure 2. Adversarial training illustration. *G* denotes the generator network, while *D* denotes the discriminator network.

#### 4.4.3 Adversarial training

Generative adversarial networks (GANs) (Goodfellow *et al.* 2014) are differentiable generator neural networks based on a game theory scenario where a generator network must compete against an opponent. The generator network produces samples  $x = g(z; \theta^g)$ . The discriminator network’s adversary aims to distinguish between samples from the training set and the generator network. Thus, during training, the discriminator learns to classify samples as real or artificially generated by the generator network. Simultaneously, the generator network tries to trick the discriminator and produces samples that look like they are coming from the probability distribution of the dataset. At convergence, the generator samples will be indistinguishable from the actual training set samples, and the discriminator network can be dropped.

In the context of this manuscript, the adversarial strategy can be better understood according to the approach adopted. When training from scratch, it aims at converting texts to the desired style without incurring the failure to copy the input text. When training from a pretrained paraphrase model, the adversarial technique tries to push the generated paraphrases toward different styles  $s^{tgt}$ , such that  $s^{tgt} \neq s^{src}$  and  $s^{tgt} \neq s^{para}$ . The general training procedure consists of, repeatedly, performing  $n_d$  discriminator training steps, minimizing  $\mathcal{L}_{discriminator}(\rho)$ , followed by  $n_f$  generator network training steps, minimizing  $a_1 \mathcal{L}_{self}(\theta) + a_2 \mathcal{L}_{KD}(\theta) + \mathcal{L}_{adversarial}(\rho)$ , until convergence. During the discriminator training steps, only the  $\rho$  parameters of the discriminator network are updated. Analogously, during the generator network training steps, only the parameters  $\theta$  of the masked transformer are updated. Figure 2 illustrates the adversarial training adopted by MATTES.

When back-translating during training (Figure 1), MATTES do not generate the transformed sentence  $\hat{Y}$  through sampling or greedy decoding from the transductive probability distribution provided by the generator network. This decision is because propagating the gradients backward through discrete stochastic operations is not differentiable, typically requiring techniques such as REINFORCE (Williams 1992) or Gumbel-Softmax distribution approximation (Jang, Gu, and Poole 2017), which suffer from high variance (He *et al.* 2020). To overcome this difficulty, the softmax distribution generated by the MATTES decoder is inserted again into the MATTES encoder, along with the original sentence style, as in Dai *et al.* (2019). This configuration makes it possible to directly propagate the gradients from the discriminator and generator networks to the reverse model that generated the target style translated sentences.

Overall, the MATTES architectural scheme includes a four-layer transformer architecture with four attention heads in each layer, for both the encoder and the decoder, and the discriminator network, as in Dai *et al.* (2019). The vector representations of tokens, hidden states, and token position in the sentence have 256 dimensions. When training from scratch, we have a style embedding layer with two 256-dimensional vectors representing the stylistic domains. When starting

from the pretrained Seq2seq, we have three vectors as we append the paraphrase style to the set of styles. The target style vector is inserted into the encoder along with the embeddings of each token of the sentence.

In the experiments, we do not include the start sentence token, only the end sequence token. During training, we replaced this token embedding with the style embedding as the first input embedding of the sequence. Thus, the vectors representing the styles are also trained and are part of the model.

This section described a machine learning method that addresses the unsupervised TST task using a Seq2Seq model pretrained in a large amount of paraphrase data and an MLM inserted into the training process with a knowledge distillation configuration. The Seq2Seq pretraining phase allows the model to create synthetically generated paraphrase pairs on-the-fly during training to help the transduction process. The distillation schema aims to extract knowledge from the MLM and enrich the text-generated quality. To verify if the proposed method improves the performance of TST, we conduct an experimental evaluation in the next section.

## 5. Experiments

We evaluated MATTES on three different style transfer tasks in the English language: author imitation, formality transfer, and polarity swap. This section presents the experimental methodology adopted to carry out the experiments, the quantitative results from five automated metrics, qualitative analyses, human evaluations, comparisons with related work, and ablation studies.

### 5.1 Experimental methodology

Here, we describe the datasets of the style transfer tasks, the models whose results we compare, the assessment metrics, and model training details.

#### 5.1.1 Datasets and tasks

MATTES was evaluated in three transfer style tasks in the English language: author imitation, formality transfer, and polarity swap. The **author imitation** task consists of converting the style of a sentence to the style of a particular author. The aim is to generate a paraphrase of the original sentence but with a different textual style. To verify the ability of MATTES to perform this task, we rely on a set of 21,000 (twenty-one thousand) sentences from William Shakespeare's plays, transcribed into modern English. As they are translations from English into English, this dataset is also an intra-linguistic translation. The author imitation task can be considered an adaptation, where a text is modified within the language. In this case, it is a sum of two processes: a temporal and a linguistic adaptation. This dataset was curated in Xu *et al.* (2012) and previously used in unsupervised textual style transfer jobs (He *et al.* 2020; Krishna *et al.* 2020). The dataset is divided into training, validation, and testing sets. We adopt  $s_1$  to denote modern English and  $\mathcal{D}_1$  for the dataset referring to this style. In contrast,  $s_2$  denotes Shakesperian English, and  $\mathcal{D}_2$  denotes the domain of the sentences in Shakesperian English. We evaluate MATTES both to transfer sentences from  $s_1$  to  $s_2$  and from  $s_2$  to  $s_1$ . Although sentence pairs exist, they are not used in pairs during training but only to evaluate results.

In the **formality transfer** task, our goal is to change the formality of the sentence while preserving its meaning. For this task, we adopted the GY AFC corpus (Rao and Tetreault 2018), which contains formal and informal sentences from two domains: Entertainment & Music (E&M) and Family & Relationships (F&R). For every sentence from validation and test sets, there are four human-written references. In the experiments, we use the most commonly used F&R domain. The dataset contains 51,967 training examples in each formal and informal class. The test set has 2351 examples. We denote the formal style as  $s_1$  and its domain as  $\mathcal{D}_1$ , while the informal style is

denoted by  $s_2$  and its domain is  $\mathcal{D}_2$ . MATTES is evaluated both to transfer sentences from  $s_1$  to  $s_2$  and from  $s_2$  to  $s_1$ .

The **polarity swap** task consists of converting the sentence to a different sentiment, preserving its general theme. In this task, we rely on the YELP dataset, collected in Shen *et al.* (2017), which includes establishment evaluations performed within the YELP application. The dataset contains 250,000 negative sentences and 380,000 positive sentences. To quantitatively assess the generalization skills of the transfer models, we rely on a test set with 1000 parallel sentences annotated by humans, introduced in Li *et al.* (2018). We denote the positive sentiment style as  $s_1$  and its domain as  $\mathcal{D}_1$ , while the negative sentiment style is denoted by  $s_2$  and its domain is  $\mathcal{D}_2$ . Again, MATTES is evaluated both to transfer sentences from  $s_1$  to  $s_2$  and from  $s_2$  to  $s_1$ .

Although these tasks have been handled indiscriminately as TST tasks, they have crucial differences that might indicate they should be treated differently during modeling and evaluation. More specifically, in polarity swap, the content is not strictly preserved (the message is actually the opposite). The important is to preserve the overall topic. In author imitation and formality transfer, instead, the “translation” happens more at the style level, and content must remain the same. In YELP, we can see that words related to the theme are intended to be retained, while polarity words are expected to be modified or replaced. On the contrary, in author imitation and formality transfer, the modifications happen at the stylistic level, affecting even noun phrases, but the essential message should be conserved. Since these two tasks can be seen much more as a rewriting task than the polarity swap, we expect the paraphrase pretraining strategy to benefit them more.

### 5.1.2 Baselines

We compare MATTES to recent models that made available the transferred test suite sentences. In the author imitation task, the results were compared with the *Deep Latent Sequence* (DLSM) (He *et al.* 2020) and STRAP (Krishna *et al.* 2020) models, which showed relevant results in the metrics of content preservation. As the *Style Transformer* (Dai *et al.* 2019) model did not perform this task, we built an in-house implementation of this model so that its output samples could be compared with MATTES. Regarding the polarity swap task, we compare MATTES with the results of other models that obtained state-of-the-art (SOA) results for this task, including again the DLSM (He *et al.* 2020) and *Style Transformer* (Dai *et al.* 2019), in addition to the model proposed in Lai *et al.* (2021) and also the RETRIEVEONLY, RULE-BASED, and DELETEANDRETRIVE models, which obtained the best results in the experiments published in Li *et al.* (2018). For the formality transfer task, we also compared with the SOTA models that made their outputs available (Luo *et al.* 2019; Yi *et al.* 2020; Lai *et al.* 2021).

### 5.1.3 Metrics for quantitative evaluation

Developing reliable automatic evaluation metrics for NLG tasks that reflect human judgment accordingly is still an open research field. However, automatic evaluation is cheaper and faster to run than human evaluation. Thus, first, we gather the most used automatic evaluation metrics from the literature (Yang *et al.* 2018; Lample *et al.* 2019; He *et al.* 2020) focusing on two dimensions that effective textual transfer style systems must preserve.

Considering that **content preservation** is the most desired feature to style transfer, three distinct metrics were used to evaluate the models according to this dimension: BLEU (Papineni *et al.* 2002), BartScore (Yuan *et al.* 2021) and semantic similarity (SIM) (Wieting *et al.* 2019). The values of the three metrics are calculated using the sentence generated by the model and a previously annotated reference sentence. For the polarity swap task, as we only have annotated sentences for the test set, during the evaluation part of the training, we use the input as the reference for calculating the metrics.

BLEU (Bilingual Evaluation Understudy Score) is a fast and inexpensive metric widely used in NLP tasks, initially proposed for machine translation. BLEU is widely used in many languages and correlates well with human judgments. Although BLEU is traditionally used for inter-linguistic translation, it was adopted in this study to evaluate the adaptations of Shakespeare’s texts, which consist of an intra-linguistic translation task. In the experiments, BLEU was calculated using the NLTK (Bird 2006) package. BLEU is calculated as

$$\begin{aligned}
 p_n &= \frac{\#n\text{-grams in the candidate found in the reference}}{\#n\text{-grams in the candidate}} \\
 BP &= e^{\min(0, 1 - \frac{c}{r})} \\
 BLEU &= BP \times \prod_{i=1}^k p_n^{w_n} \tag{14}
 \end{aligned}$$

where  $c$  is the length of the translated candidate sentence,  $r$  is the length of the reference sentence,  $BP$  is a term that penalizes the difference between the length of the candidate and reference sentences,  $k$  is the maximum  $n$ -gram one wants to evaluate, and  $p_n$  is the precision score for the grams of length  $n$ . According to the values proposed in Papineni *et al.* (2002), we adopt  $k = 4$  and uniform weights  $w_n = 1/N$ .

Wieting *et al.* (2019) introduced a metric called SIM to measure semantic similarity between sentences. SIM aims to overcome the limitations of the BLEU metric and avoid giving partial credit and penalizing semantically correct candidates when they differ lexically from the reference sentence. For that, a coding model  $g$  that tries to maximize the similarity of pairs of sentences present in a dataset of paraphrases (Wieting and Gimpel 2018) was trained. The  $g$  encoder averages the vector representations of each sentence token to create a vector representation of the sentence. The similarity between a pair of sentences  $\langle X, \hat{X} \rangle$  is obtained by encoding the two sentences with  $g$  and then calculating the cosine similarity of the two representations:

$$SIM = \cos(g(X), g(\hat{X})) \tag{15}$$

BARTScore (Yuan *et al.* 2021) is a recently proposed metric tailored for generation tasks. It evaluates generated text from different perspectives, for example, informativeness and fluency. Although simple, BARTScore has been shown to correlate better with human judgments in different generation tasks and achieved the best performance on 16 of 22 settings against existing top-scoring metrics. Mathematically, BARTScore is the log probability of one text  $y$  given another text  $x$ .

$$BARTScore = \frac{1}{N} \sum_{t=1}^N \log p(y_t | y_{1:t-1}, x, \theta) \tag{16}$$

Regarding **attribute control**, following previous work (Shen *et al.* 2017; Yang *et al.* 2018; Luo *et al.* 2019; He *et al.* 2020; Lai *et al.* 2021), we train a neural convolutional classifier (Kim 2014) to measure the extent to which the style is controlled. We train one classifier for each task to determine the stylistic domain to which a sentence belongs. The style control metric for each task is the accuracy this classifier gives to the generated sentences. The training set used to train the classifiers is the same one used during our main training. The classifiers have an accuracy of 82.7%, 85.1%, and 97.1% on the validation sets of Shakespeare, GYAFC and YELP, respectively. On the test sets, the values are 81.4%, 86.2%, and 97.0%, respectively.

Finally, as the overall score for model selection and for comparison to previous work (Luo *et al.* 2019; Lai *et al.* 2021), we compute the harmonic mean (HM) of style accuracy and BLEU.

To select our best model, we settled thresholds for the style accuracy, and the model with the highest harmonic mean (HM) was selected to run on the test set. Since it is necessary to provide a greater weight to the content preservation metric, the HM suits well for our case. The threshold

is necessary to avoid selecting models with a high degree of input copying, achieving a high harmonic mean even with a shallow style accuracy, and failing to control the desired style. For author imitation, we empirically established 66.6% accuracy, which is a bottom limit for style control, considering each transfer direction. For formality transfer and polarity swap, this limit was set to 70%.

#### 5.1.4 Human evaluation

As the automatic metrics only give us shallow perceptions of the quality of the translated sentences, we compare the transfer quality across state-of-the-art models and ours by a small-scale human study. We conduct a human evaluation on the test set of both the YELP dataset and the GYAF corpus. We left the Shakespeare domain out since it is more difficult for non-native speakers and non-experts to evaluate. For each dataset, we randomly select 48 samples for the human evaluation (24 for each style attribute). Each sample contains the transformed sentences generated by different models given the same source sentence. Annotators were asked to rate each output on three criteria on a Likert scale from 1 to 5: style control, content preservation, and an overall quality score. The annotators also rate human references for each dataset.

#### 5.1.5 Hyperparameters and training details

During the experiments, some combinations of hyperparameters were evaluated according to the performance of MATTES in the validation set. The term  $\alpha$  in the loss function of the knowledge distillation component (Equation 11) varied within the values {0.1, 0.5, 0.65}. The knowledge distillation temperature  $T_{\text{KD}}$  also varied by {1, 5, 10}. Values for the number of steps of the discriminator  $n_d$  and the number of steps of the generator  $n_f$  were also experimented. The tuple  $(n_d, n_f)$  varied with the values {(7.5), (9.5), (10.5)}. The contribution of each component to the MATTES loss function (Equations 10, 11, and 12) also varied ( $a_1 \mathcal{L}_{\text{self}}(\theta) + a_2 \mathcal{L}_{\text{KD}}(\theta) + \mathcal{L}_{\text{adversarial}}(\rho)$ ). As Dai *et al.* (2019), it was concluded that executing random dropout on the input sentence tokens during the sentence reconstruction step positively impacts the model results. The dropout rate varied in {0.2, 0.3, 0.4}.

We perform all the experiments using the Python programming language with the PyTorch (Paszke *et al.* 2017) framework. The ALBERT implementation uses the *HuggingFace transformers* (Wolf *et al.* 2020) framework. Before the main MATTES training, for each task performed, we selected the available model `albert-large-v2`, and fine-tuned two ALBERT models, one for each stylistic domain, using only the traditional MLM task. In this adaptive domain pretraining step, as in Gururangan *et al.* (2020), we train each ALBERT model for 100 epochs, using the AdamW with  $\epsilon$  equal to  $1e-6$ , linear learning rate with a warm-up, and maximum learning rate of  $1e-5$ . ALBERT does not use dropout or regularization during training. Training took place on four NVIDIA P100 GPUs with a training batch of eight examples per GPU. After fine-tuning ALBERT in each stylistic domain, the logits<sup>b</sup> were extracted for each token of each sentence in the training set. Following Chen *et al.* (2020), for saving computational resources, only the top-8 logits were considered to be used as labels during the main MATTES training.

To take advantage of available generic data, we pre-trained our Seq2seq model on pairs of paraphrase (Hu *et al.* 2019) and used the trained model as starting point for the main adversarial training. The training details followed Vaswani *et al.* (2017). We trained the model for 300,000 steps, using the Adam optimizer (Kingma and Ba 2015) with  $\beta_1 = 0.9$ ,  $\beta_2 = 0.98$  and  $\epsilon = 1e-9$ . Each training step took about 0.12 seconds, and the whole pre-training was about 20 hours on a single P100 GPU. We apply dropout with a rate of  $P_{\text{dropout}} = 0.1$ . As in Vaswani *et al.* (2017), we varied the learning rate throughout training, increasing the learning rate linearly for the first 4000 steps and decreasing it proportionally to the inverse square root of the step number. From

<sup>b</sup> Logits are the non-normalized representations provided by ALBERT before going through the Softmax layer.



the complete PARABANK 2 (Hu *et al.* 2019) containing 19.7M pairs of paraphrases, we filtered out pairs in which at least one of the sentences had less than six words or more than 50 words. We split the remaining pairs into training and validation sets. Our final paraphrase training set reached 11.4M pairs, and the validation set 2.8M.

All primary training experiments were run on Tesla P100-SXM2 GPUs using the Ubuntu operating system on a machine with an Intel(R) Xeon(R) CPU E5-2698 v4. On average, the Yelp dataset takes 30 hours with a training batch of 192, the GYAFC corpus takes 15 hours with a training batch of 128, while the Shakespeare dataset takes 15 hours with a training batch of 64. This time drops by approximately half when starting from the pretrained Seq2Seq model.

## 5.2 Results

The results include the values obtained using quantitative metrics, a statistical test, a human evaluation, and ablation studies. We also perform a qualitative analysis.

### 5.2.1 Quantitative results

Table 2 shows the results using the automatic metrics. In the YELP dataset, the best models displayed in Li *et al.* (2018) were compared and complemented with Yi *et al.* (2020), He *et al.* (2020), Dai *et al.* (2019), and Lai *et al.* (2021). In the Shakespeare dataset, MATTES results were compared to the results of He *et al.* (2020) and Krishna *et al.* (2020). In addition, the results were also compared with our implementation of the style transformer (Dai *et al.* 2019) model in the author imitation task, as the original work did not include this task. This implementation is similar to MATTES but uses the traditional back-translation component in the loss function rather than the knowledge distillation component. In the GYAFC corpus, MATTES results were compared to the results of Luo *et al.* (2019), Yi *et al.* (2020), and Lai *et al.* (2021), the state-of-the-art model. We also compared it with our implementation of the style transformer (Dai *et al.* 2019) model.

In Shakespeare's dataset, MATTES surpassed all previous approaches regarding all content preservation metrics. It also achieved the best overall performance (HM) on this task. The Deep Latent Sequence Model (DLSM) (He *et al.* 2020) has the best style control metric, however, at the expense of low values in the content preservation metrics. On the other hand, we can increase the style accuracy threshold if we desire more control over the style. In Appendix D, we show models selected using a higher threshold.

In the YELP dataset, MATTES achieved the second-highest results for all the content preservation metrics and the overall score, staying very close to the best model. Nevertheless, the style control metric was the worst among all the compared models.

In the GYAFC corpus, MATTES achieves the highest value for style control. As in YELP, it achieves the second-highest results for all the content preservation metrics and the overall score. As we can see in Section 5.2.3, the style control result is mainly due to the knowledge distillation technique.

By examining the results, we can see that our full model is quite effective in preserving the content of the input while keeping a reasonable style accuracy in all tasks. In the author imitation task and in the formality transfer task, our approach achieved new state-of-the-art results in terms of content preservation and style control, respectively. In Section 5.2.3, we evaluate the impact of our proposed training strategy; specifically, we want to quantify the impact of the Seq2Seq pretraining phase and the knowledge distillation training technique.

*Statistical significance analysis.* We conducted statistical tests to compare the BARTScore and SIM metrics achieved by MATTES and the other two models that obtained the best content preservation metrics. We left out BLEU since it is a more limited metric that measures lexical matching only. We denote  $H_0$  as the null hypothesis and  $H_1$  as the alternative hypothesis to be tested. The

**Table 2.** Results with automatic evaluation metrics. BScore stands for BARTScore, and HM is the harmonic mean of BLEU and accuracy. The best results are in bold.

Task	Model	Acc.↑	BLEU↑	SIM↑	BScore↑	HM↑
Polarity swap	Rule-based (Li <i>et al.</i> 2018)	85.6	15.57	53.4	-3.407	0.263
	DeleteAndRetrieve (Li <i>et al.</i> 2018)	89.1	12.05	48.5	-3.690	0.212
	Deep Latent Sequence (He <i>et al.</i> 2020)	84.4	16.59	48.8	-3.515	0.277
	Style Transformer (Dai <i>et al.</i> 2019)	84.6	21.86	60.9	-2.975	0.348
	StylIns (Yi <i>et al.</i> 2020)	<b>93.4</b>	18.7	56.6	-3.159	0.312
	Generic (Lai <i>et al.</i> 2021)	89.5	<b>24.19</b>	<b>65.4</b>	<b>-2.824</b>	<b>0.381</b>
	MATTES (ours)	75.5	23.65	64.6	-2.838	0.360
Author imitation	DLSM (He <i>et al.</i> 2020)	<b>80.3</b>	9.05	43.4	-4.432	0.163
	STRAP (Krishna <i>et al.</i> 2020)	70.0	8.27	56.4	-4.063	0.148
	Style transformer	61.4	11.29	52.8	-3.688	0.191
	MATTES (ours)	69.5	<b>14.49</b>	<b>58.9</b>	<b>-3.373</b>	<b>0.240</b>
Formality transfer	DualRL (Luo <i>et al.</i> 2019)	64.7	24.0	53.9	-2.639	0.350
	StylIns (Yi <i>et al.</i> 2020)	74.6	29.2	63.8	-2.216	0.419
	Style Transformer	78.0	33.0	60.0	-2.352	0.464
	Generic (Lai <i>et al.</i> 2021)	87.0	<b>51.1</b>	<b>71.6</b>	<b>-1.931</b>	<b>0.644</b>
	MATTES (ours)	<b>87.8</b>	39.8	64.0	-2.130	0.547

significance level  $\alpha = 0.05$  was adopted, and we assume that the random variables  $\bar{B}$  and  $\bar{S}$ , which correspond, respectively, to the population mean of BARTScore and SIM, have a normal distribution for each model. The tests were based on the sample means of the two metrics in the test set. Under the null hypothesis, the p-value is the probability of obtaining a statistic equal to or more extreme than that observed in a sample. Thus, when a p-value smaller than the significance level is obtained, this indicates an unlikely event, indicating rejection of  $H_0$ .

The results of the statistical tests in Table 3 reinforced the results of Table 2. In the author imitation task, MATTES outperformed the other baseline models on both metrics. In the polarity swap, MATTES outperforms the Style Transformer Model on both metrics. When comparing with the SOTA model (Lai *et al.* 2021), for a confidence interval of 95%, it is not possible to affirm the superiority of any model regarding both metrics. In the formality transfer task, Lai *et al.* (2021) is statistically superior in both metrics, and MATTES confirms the superiority over the style instance model only for the BARTScore metric.

### 5.2.2 Human evaluation results

Due to the limitations of automatic metrics, we also conduct human evaluations on YELP and GYAFD datasets. Table 4 shows the evaluation of the style transfer accuracy, content preservation,

**Table 3.** T-test over the mean population for BARTScore and SIM metric on the test set

Task	t-test	H_0	H_1	Result	p-value
Polarity swap	MATTES-GEN	$\bar{B}_{GEN} = \bar{B}_{MATTES}$	$\bar{B}_{GEN} > \bar{B}_{MATTES}$	Accept $H_0$	$p = 0.38$
	MATTES-STYLE	$\bar{B}_{MATTES} = \bar{B}_{STYLE}$	$\bar{B}_{MATTES} > \bar{B}_{STYLE}$	Reject $H_0$	$p = 0.002$
Author imitation	MATTES-STRAP	$\bar{B}_{MATTES} = \bar{B}_{STRAP}$	$\bar{B}_{MATTES} > \bar{B}_{STRAP}$	Reject $H_0$	$p < 10^{-3}$
	MATTES-STYLE	$\bar{B}_{MATTES} = \bar{B}_{STYLE}$	$\bar{B}_{MATTES} > \bar{B}_{STYLE}$	Reject $H_0$	$p < 10^{-3}$
Formality transfer	MATTES-GEN	$\bar{B}_{GEN} = \bar{B}_{MATTES}$	$\bar{B}_{GEN} > \bar{B}_{MATTES}$	Reject $H_0$	$p < 10^{-3}$
	MATTES-STYINS	$\bar{B}_{MATTES} = \bar{B}_{STYIN}$	$\bar{B}_{MATTES} > \bar{B}_{STYIN}$	Reject $H_0$	$p = 0.002$
Polarity swap	MATTES-GEN	$\bar{S}_{GEN} = \bar{S}_{MATTES}$	$\bar{S}_{GEN} > \bar{S}_{MATTES}$	Accept $H_0$	$p = 0.22$
	MATTES-STYLE	$\bar{S}_{MATTES} = \bar{S}_{STYLE}$	$\bar{S}_{MATTES} > \bar{S}_{STYLE}$	Reject $H_0$	$p < 10^{-3}$
Author imitation	MATTES-STRAP	$\bar{S}_{MATTES} = \bar{S}_{STRAP}$	$\bar{S}_{MATTES} > \bar{S}_{STRAP}$	Reject $H_0$	$p < 10^{-3}$
	MATTES-STYLE	$\bar{S}_{MATTES} = \bar{S}_{STYLE}$	$\bar{S}_{MATTES} > \bar{S}_{STYLE}$	Reject $H_0$	$p < 10^{-3}$
Formality transfer	MATTES-GEN	$\bar{S}_{GEN} = \bar{S}_{MATTES}$	$\bar{S}_{GEN} > \bar{S}_{MATTES}$	Reject $H_0$	$p < 10^{-3}$
	MATTES-STYINS	$\bar{S}_{MATTES} = \bar{S}_{STYIN}$	$\bar{S}_{MATTES} > \bar{S}_{STYIN}$	Accept $H_0$	$p = 0.52$

**Table 4.** Results from the human evaluation on Yelp and GYAFc datasets. The best results are in bold.

Task	Model	Style↑	Content↑	Quality score↑
Polarity swap	Style Transformer (Dai <i>et al.</i> 2019)	3.6	3.2	3.3
	Generic (Lai <i>et al.</i> 2021)	4.1	<b>3.6</b>	3.9
	MATTES (ours)	3.4	3.5	3.5
	Human	<b>4.2</b>	3.4	<b>4.0</b>
Formality transfer	DualRL (Luo <i>et al.</i> 2019)	2.6	2.4	2.3
	Generic (Lai <i>et al.</i> 2021)	3.7	<b>4.3</b>	<b>3.9</b>
	MATTES (ours)	3.5	3.6	3.6
	Human	<b>4.0</b>	3.8	3.8

and an overall quality score obtained by MATTES and the compared models. All three aspects are rated with a range of 1–5. The results are in line with our automatic evaluations and add confidence to the efficacy of our proposed techniques in achieving style transfer across multiple dimensions. Our scores are around the state-of-the-art values regarding both style control and content preservation in the formality transfer and polarity swap tasks, respectively. Additionally, it is worth mentioning the high scores achieved by the baseline Lai *et al.* (2021), mainly for the content preservation metric, in some cases even surpassing the scores reached by the references which are sentences created by humans.

### 5.2.3 Ablation studies

In this section, we measure the individual impact of the pretraining phase and the model training components. We also show how the content preservation metrics correlate between them.

**Table 5.** Evaluation results for the ablation study. Acc. stands for accuracy, BScore for BARTScore, and HM for the harmonic mean between BLEU and accuracy

Task	Model	Acc.↑	BLEU↑	SIM↑	BScore↑	HM↑
Polarity swap	FULL MODEL	75.5	23.65	64.6	-2.838	0.360
	NO KD	71.8	23.03	64.4	-2.878	0.349
	NO PARA	83.0	22.06	61.3	-2.964	0.349
	NO PARA KD	80.2	20.85	59.4	-3.024	0.331
	NO ADV	8.1	24.11	65.0	-2.933	0.121
	NO ADV KD	3.9	24.34	65.5	-2.919	0.067
Author imitation	FULL MODEL	69.5	14.49	58.9	-3.373	0.240
	NO KD	68.7	14.30	58.4	-3.411	0.237
	NO PARA	68.9	11.72	54.0	-3.591	0.200
	NO PARA KD	61.5	11.29	52.8	-3.688	0.191
	NO ADV	26.6	13.87	60.8	-3.270	0.182
	NO ADV KD	19.5	14.56	64.7	-3.139	0.167
Formality transfer	FULL MODEL	87.8	39.80	64.0	-2.130	0.547
	NO KD	74.5	37.86	64.0	-2.170	0.502
	NO PARA	81.5	33.89	54.5	-2.326	0.479
	NO PARA KD	78.0	33.05	60.0	-2.352	0.464

*Paraphrase pre-training and model components.* To confirm that TST models can benefit from our proposals, we conducted an ablation study on all datasets by eliminating or modifying a certain component (e.g., objective functions or pre-training phase). We tested the following variations: (1) FULL MODEL: our proposed model with all the components described in Section 4; (2) NO KD: the model without the knowledge distillation component ( $\alpha = 0$ ); (3) NO PARA: the model trained from scratch, without pre-training the Seq2Seq model on paraphrase data; (4) NO PARA KD: the model trained from scratch and without knowledge distillation; (5) NO ADV: the model trained without the adversarial loss component; and (6) NO ADV KD: the model trained without the adversarial loss component and without the knowledge distillation ( $\alpha = 0$ ). Table 5 exhibits the results.

The results indicate that the pretraining strategy significantly impacts content preservation. On all tasks, pretraining considerably increased the content preservation metric. In the Shakespeare dataset and in the GYAFC corpus, the style accuracy also improved with pre-training, which makes sense since the diversity of the paraphrase model should lead to a lower degree of input copying. On the other hand, in the YELP dataset, the pretraining approach harmed the model style control. This also makes sense since the polarity swap task is not a rewriting task, so having a paraphrase of a sentence does not help. For that, something that changes the meaning is necessary. This observation indicates that these tasks should not be indiscriminately tackled as TST tasks since there is a clear need to change the meaning of the polarity swap task.

Regarding the isolated impact of our proposed knowledge distillation technique, all the model variations from Table 5 trained with the distillation achieved higher accuracy and HM when compared with the variations trained without. Although adversarial training is the most crucial component for style control, the distillation also positively affects style accuracy. Comparing

**Table 6.** Pearson correlation between content preservation metrics over  $N$  systems

Task	$N$	BS	SIM	BLUE
		SIM	BLUE	BS
Polarity swap	17	0.787	0.656	0.612
		$p = 0$	$p = 0$	$p = 0$
Author imitation	12	0.682	0.555	0.488
		$p = 0$	$p = 0$	$p = 0$

the two model variations without adversarial training (NOADV and NOADVKD), we realize distillation increases the accuracy, yet not so impressively as the adversarial component.

Since we decided to use the harmonic mean (HM) of the BLUE with the style accuracy for model selection, to interpret the impact of the pre-training phase better, in Figure 3 located in Appendix C, we plot these three metrics achieved by our FULL MODEL and its variation NO PARA along the training process on the Yelp validation set. From the graphics, we can realize that the training dynamics are different. When training the model from scratch, the content preservation metric starts almost from zero until reaching its limit. In contrast, the pretrained model starts at its maximum but with low accuracy. Once its highest value is reached, the HM tends downward after many training steps. At last, we notice the FULL MODEL reaches its highest HM faster than the model trained from scratch, showing another benefit from pretraining.

The hyperparameters of the models in Table 5 are listed in Appendix A.

*Metric analysis.* Since evaluating generated texts is an open field and a challenging task, we adopted three metrics for content preservation. To glean further insights about these choices, we calculate the Pearson Correlation between evaluation metrics for content preservation over  $N$  systems (Table 6).

The results show that while SIM and BARTScore highly correlated with each other, BLEU does so to a lesser extent, suggesting this might be a less robust measure to assess the goodness of TST tasks. Since transfer style often involves lexical changes, n-gram-based matching metrics such as BLEU usually fail to recognize information beyond the lexical level. This result strengthens the necessity for adopting metrics concerned with capturing the overall semantics of a sentence, like SIM and BARTScore.

#### 5.2.4 Qualitative analyses

At last, we carry out some qualitative analyses: (1) compare samples generated by MATTES and DLSSM and (2) compare our model and training approach with the state-of-art method in the Yelp dataset (Lai *et al.* 2021).

*MATTES versus deep latent sequence model.* We examine the output from our model and the Deep Latent Sequence model for the author imitation task. We picked this task because the reference outputs for the test set are provided. From the qualitative analysis, we observed that both models can generate good translations, but MATTES tends to preserve the content better. On the other hand, Deep Latent Sequence tends to create too short sentences when the input sentence is long, losing content. This observation concurs with the fact that MATTES's content preservation metric is the highest. The attention mechanisms might have helped in handling long sentences. Besides, the sentences generated by the Deep Latent Sequence model have lower

**Table 7.** Transferred sentences for the author imitation task

Model	Shakespeare to modern
Source	Lo, here upon thy cheek the stain doth sit Of an old tear that is not washed off yet.
Ref.	There's still a stain on your cheek from an old tear that hasn't been washed off yet.
Deep Lat.	Right on, now.
Ours	Okay, here on your cheeks the stigma do the sitting of an old tears that's not washed out yet.
Source	These happy masks that kiss fair ladies'brows, Being black, puts us in mind they hide the fair.
Ref.	Look at other beautiful girls.
Deep Lat.	These!
Ours.	These happy masks that kiss beautiful ladies'leggeds, being black, puts us in mind they hide the beautiful.

perplexity than the test set, indicating short or trivial sentences. Table 7 exhibits examples of the generated sentences for long inputs.

Appendix B shows other examples of transferred sentences by MATTES, DLSSM, and Style Transformer in the sentiment transfer task.

*MATTES versus generic resources model.* The state-of-the-art (SOTA) model in the polarity swap (Lai *et al.* 2021), which also leverages generic resources for pretraining, adopts the same Transformer architecture as MATTES. The primary difference lies in MATTES additional style embedding layer and the depth of both networks. While MATTES has 23 M parameters, the SOTA model starts its training from an already pretrained BART model (Lewis *et al.* 2020) with 139 M parameters. Its impressive results were thanks to further pretraining using synthetic pairs of sentences with opposite polarities.

Analyzing MATTES metrics achieved in the YELP dataset and comparing them with Lai *et al.* (2021), we realize that adopting a pretraining phase using generic paraphrase data made all the content preservation metrics of both models very similar. Nevertheless, in the case of MATTES, this benefit came at the expense of style accuracy reduction, damage that did not occur in the author imitation task. These facts strengthen pretraining as a great skill to increase content preservation and that both tasks should not be treated and approached equally as style transfer tasks. As polarity swap is not a rewriting task, the pretrained paraphrase system gets in the way of the style control, putting all the effort to change the style into the adversarial mechanisms.

### 5.3 Limitations and final remarks

Although theoretically attractive, adopting the proposed components and the training schema has some drawbacks. Adversarial training is challenging due to its instability and likely divergence after some epochs. The knowledge distillation strategy requires using a pretrained MLM to generate the soft targets and adds two more parameters for tuning during training. Also, although beneficial, pretraining the Seq2seq model increases the overall running time. At last, when adopting the pretrained model strategy, during inference, the model first converts the input to the paraphrase style and then to the target style, finishing the translation process. The models trained from scratch, as they do not consider the paraphrase style, convert directly to the target style.

To answer the research questions enunciated in Section 1, in this section, we presented an experimental evaluation of the proposed model and compared it to other state-of-the-art models. We also conducted an ablation study with two main contributions, namely, the pretrained Seq2Seq and the knowledge distillation component, to observe their individual impact



on the results. Following the experimental results, this section showed qualitative analyses and MATTES's comparisons with other works related to the unsupervised TST task.

## 6. Conclusions

This manuscript proposed a new method based on machine learning to perform the unsupervised textual style transfer task, where only the sentence and its respective textual style are available for training. Interpreting a text and converting it to the desired style is a fundamental skill for communication between people. Equipping machines with these skills is necessary to make them part of the communication process.

Given how difficult it is to manually create annotated TST datasets with parallel texts transformed from one style to another, we adopted an unsupervised approach in this manuscript. The proposed method, MATTES, as far as we know, is the first to use the bidirectional representations produced by a MLM as labels to support the training of a generator neural network in the TST task.

The overall architecture of MATTES was inspired by a previous method, the *Style Transformer* but significantly changing the training procedure. In addition to the pretraining phase of the Seq2Seq model, during the main training, MATTES adopts as labels the probability distribution provided by a MLM for each token of the input sentence as part of its loss function. The experimental results indicated that MATTES reaches the state of the art in the author imitation task, considering both content preservation and an overall score for style transfer. In the formality transfer task, although we do not reach the best model regarding content preservation, our proposed techniques lead to significant improvements in both content preservation and style control. In the polarity swap task, MATTES content preservation metrics were close to the SOTA model, but the style measure was not impressive. We believe the polarity swap did not benefit as much as the other tasks from the pretraining because the former is a task that needs to change the meaning to accomplish its goal and cannot be considered a rewriting task. On the other hand, the knowledge distillation technique is task agnostic and can be suitable in many situations. The less information you have to build a proper probability distribution to generate texts, the more helpful the distillation might be.

Future work includes adopting MATTES in the challenging task of text simplification Al-Thanyan and Azmi (2021). In addition, we intend to add knowledge distillation to other existing models where the distillation may fit to provide more evidence that it improves text generation in the unsupervised textual style transfer. Knowledge distillation can also be added to the sentence reconstruction component of the generator network. Another promising idea, inspired by machine translation, is to optimize directly a differentiable metric during training that takes into account both style control and content preservation. The adoption of a paraphrase style enables the model to handle the multiattribute transfer, which is an exciting direction to follow. Finally, we want to train MATTES in other languages, such as Brazilian Portuguese.

To sum up, we reinforce the need for a better evaluation of text generation methods in natural language. With the increasing presence of machines as parts of the communication process, it is expected that this subarea of NLP continues to gain prominence in the academic and industrial world. The advance of machine learning algorithms that work or are compatible with the TST task enables the creation of models capable of smoothly controlling the text attributes.

**Acknowledgments.** This research was partially financed by CNPq (National Council for Scientific and Technological Development) under grant 311275/2020-6 and FAPERJ—*Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro*, process SEI-260003/000614/2023 and E-26/202.914/2019.

**Competing interests.** The authors declare that they have no competing or conflict of interest.

## References

- Al-Thanyyan S. S. and Azmi A. M. (2021). Automated text simplification: A survey. *ACM Computing Surveys (CSUR)* 54(2), 1–36.
- Bahdanau D., Cho K. and Bengio Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio Y. and LeCun Y. (eds), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. <http://arxiv.org/abs/1409.0473>
- Bird S. (2006). NLTK: The natural language toolkit. In Calzolari C., Cardie C. and Isabelle P., (eds), *ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006*. The Association for Computer Linguistics. <https://aclanthology.org/P06-4018/>
- Chen L., Dai S., Tao C., Shen D., Gan Z., Zhang H., Zhang Y., Zhang R., Wang G. and Carin L. (2018). Adversarial text generation via feature-mover’s distance. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS’18*. Red Hook, NY: Curran Associates Inc., pp. 4671–4682.
- Chen X., Zhang S., Shen G., Deng Z.-H. and Yun U. (2021). Towards unsupervised text multi-style transfer with parameter-sharing scheme. *Neurocomputing* 426, 227–234. <https://www.sciencedirect.com/science/article/pii/S0925231220315915>
- Chen Y., Gan Z., Cheng Y., Liu J. and Liu J. (2020). Distilling knowledge learned in BERT for text generation. In Jurafsky D., Chai J., Schluter N. and Tetreault J. R. (eds), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 7893–7905. <https://doi.org/10.18653/v1/2020.acl-main.705>
- Dai N., Liang J., Qiu X. and Huang X. (2019). Style transformer: Unpaired text style transfer without disentangled latent representation. In Korhonen S., Korhonen D. R. and Márquez L. (eds), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 5997–6007. <https://doi.org/10.18653/v1/p19-1601>
- Devlin J., Chang M.-W., Lee K. and Toutanova K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. pp. 4171–4186.
- Fu Z., Tan X., Peng N., Zhao D. and Yan R. (2018). Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*. AAAI Press, pp. 663–670.
- Gatt A. and Krahmer E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research* 61, 65–170. <https://doi.org/10.1613/jair.5477>
- Ghazvininejad M., Shi X., Choi Y. and Knight K. (2016). Generating topical poetry. In Su J., Carreras X. and Duh K. (eds), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. The Association for Computational Linguistics, pp. 1183–1191. <https://doi.org/10.18653/v1/d16-1126>
- Gong H., Bhat S., Wu L., Xiong J. and Hwu W.-M. (2019). Reinforcement learning based text style transfer without parallel training corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, MN: Association for Computational Linguistics, pp. 3168–3180. <https://aclanthology.org/N19-1320>
- Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A. and Bengio Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, Vol. 27.
- Goyal N., Srinivasan B. V., A. N. and Sancheti A. (2021). Multi-style transfer with discriminative feedback on disjoint corpus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online*, pp. 3500–3510. <https://aclanthology.org/2021.naacl-main.275>
- Gururangan S., Marasovic A., Swayamdipta S., Lo K., Beltagy I., Downey D. and Smith N. A. (2020). Don’t stop pre-training: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, pp. 8342–8360.
- Han M., Wu O. and Niu Z. (2017). Unsupervised automatic text style transfer using LSTM. In Huang X., Jiang J., Zhao D., Feng Y. and Hong Y. (eds), *Natural Language Processing and Chinese Computing - 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8-12, 2017, Proceedings*, Lecture Notes in Computer Science, Vol. 10619. Cham: Springer, pp. 281–292. [https://doi.org/10.1007/978-3-319-73618-1\\_24](https://doi.org/10.1007/978-3-319-73618-1_24)
- He J., Wang X., Neubig G. and Berg-Kirkpatrick T. (2020). A probabilistic formulation of unsupervised text style transfer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=HJLA0C4tPS>
- Hendrycks D. and Gimpel K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.
- Hinton G., Vinyals O. and Dean J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv: 1503.02531*.
- Howard J. and Ruder S. (2018). Universal language model fine-tuning for text classification. In Gurevych I. and Miyao Y. (eds), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 328–339.

- Hu J. E., Singh A., Holzenberger N., Post M. and Van Durme B. (2019). Large-scale, diverse, paraphrastic bitexts via sampling and clustering. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Hong Kong: Association for Computational Linguistics, pp. 44–54. <https://aclanthology.org/K19-1005>
- Hu Z., Lee R. K.-W., Aggarwal C. C. and Zhang A. (2022). Text style transfer: A review and experimental evaluation. *ACM SIGKDD Explorations Newsletter* 24(1), 14–45.
- Hu Z., Yang Z., Liang X., Salakhutdinov R. and Xing E. P. (2017). Toward controlled generation of text. In Precup D. and Teh Y. W. (eds), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, Proceedings of Machine Learning Research*, PMLR, Vol. 70, pp. 1587–1596. <http://proceedings.mlr.press/v70/hu17e.html>
- Jain P., Mishra A., Azad A. P. and Sankaranarayanan K. (2019). Unsupervised controllable text formalization. *Proceedings of the AAAI Conference on Artificial Intelligence* 33(01), 6554–6561. <https://ojs.aaai.org/index.php/AAAI/article/view/4623>
- Jang E., Gu S. and Poole B. (2017). Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=rkE3y85ee>
- Jin D., Jin Z., Hu Z., Vechtomova O. and Mihalcea R. (2022). Deep learning for text style transfer: A survey. *Computational Linguistics* 48(1), 155–205.
- John V., Mou L., Bahuleyan H. and Vechtomova O. (2019). Disentangled representation learning for non-parallel text style transfer. In Korhonen A., Traum D. R. and Márquez L. (eds), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 424–434.
- Kim H. and Sohn K.-A. (2020). How positive are you: Text style transfer using adaptive style embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona: International Committee on Computational Linguistics (Online), pp. 2115–2125. <https://aclanthology.org/2020.coling-main.191>
- Kim Y. (2014). Convolutional neural networks for sentence classification. In Moschitti A., Pang B. and Daelemans V. (eds), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, pp. 1746–1751. <https://doi.org/10.3115/v1/d14-1181>
- Kingma D. P. and Ba J. (2015). Adam: A method for stochastic optimization. In Bengio Y. and LeCun Y. (eds), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. <http://arxiv.org/abs/1412.6980>
- Krishna K., Wieting J. and Iyyer M. (2020). Reformulating unsupervised style transfer as paraphrase generation. In Webber B., Cohn T., He Y. and Liu Y. (eds), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, pp. 737–762. <https://doi.org/10.18653/v1/2020.emnlp-main.55>
- Lai C.-T., Hong Y.-T., Chen H.-Y., Lu C.-J. and Lin S.-D. (2019). Multiple text style transfer by using word-level conditional generative adversarial network with two-phase training. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 3579–3584. <https://aclanthology.org/D19-1366>
- Lai H., Toral A. and Nissim M. (2021). Generic resources are what you need: Style transfer tasks without task-specific parallel training data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, pp. 4241–4254. <https://aclanthology.org/2021.emnlp-main.349>
- Lample G., Subramanian S., Smith E. M., Denoyer L., Ranzato M. and Boureau Y. (2019). Multiple-attribute text rewriting. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. <https://openreview.net/forum?id=H1g2NhC5KQ>
- Lan Z., Chen M., Goodman S., Gimpel K., Sharma P. and Soricut R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. <https://openreview.net/forum?id=H1eA7AEtV5>
- Lewis M., Liu Y., Goyal N., Ghazvininejad M., Mohamed A., Levy O., Stoyanov V. and Zettlemoyer L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, pp. 7871–7880. <https://aclanthology.org/2020.acl-main.703>
- Li J., Jia R., He H. and Liang P. (2018). Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, pp. 1865–1874.
- Liu D., Fu J., Zhang Y., Pal C. and Lv J. (2020). Revision in continuous space: Unsupervised text style transfer without adversarial learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(05), 8376–8383. <https://ojs.aaai.org/index.php/AAAI/article/view/6355>
- Liu Y., Neubig G. and Wieting J. (2021). On learning text style transfer with direct rewards. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, pp. 4262–4273. <https://aclanthology.org/2021.naacl-main.337>

- Logeswaran L., Lee H. and Bengio S.** (2018). Content preserving text generation with attribute controls. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18*. Red Hook, NY: Curran Associates Inc., pp. 5108–5118.
- Luo F., Li P., Zhou J., Yang P., Chang B., Sun X. and Sui Z.** (2019). A dual reinforcement learning framework for unsupervised text style transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, pp. 5116–5122. <https://doi.org/10.24963/ijcai.2019/711>
- Malmi E., Severyn A. and Rothe S.** (2020a). Unsupervised text style transfer with padded masked language models. In Webber B., Cohn T., He Y. and Liu Y. (eds), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, pp. 8671–8680.
- Malmi E., Severyn A. and Rothe S.** (2020b). Unsupervised text style transfer with padded masked language models. In Webber B., Cohn T., He Y. and Liu Y. (eds), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, pp. 8671–8680.
- Mueller J., Gifford D. and Jaakkola T.** (2017). Sequence to better sequence: Continuous revision of combinatorial structures. In Precup D. and Teh Y. W. (eds), *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70. PMLR, pp. 2536–2544. <http://proceedings.mlr.press/v70/mueller17a.html>
- Papineni K., Roukos S., Ward T. and Zhu W.** (2002). Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*. ACL, pp. 311–318. <https://aclanthology.org/P02-1040/>
- Park S., Hwang S.-w., Chen F., Choo J., Ha J.-W., Kim S. and Yim J.** (2019). Paraphrase diversification using counterfactual debiasing. *Proceedings of the AAAI Conference on Artificial Intelligence* 33(01), 6883–6891.
- Paszke A., Gross S., Chintala S., Chanan G., Yang E., DeVito Z., Lin Z., Desmaison A., Antiga L. and Lerer A.** (2017). Automatic differentiation in pytorch. In *NIPS-W*.
- Peters M., Neumann M., Iyyer M., Gardner M., Clark C., Lee K. and Zettlemoyer L.** (2018). Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 2227–2237.
- Phuong M. and Lampert C.** (2019). Towards understanding knowledge distillation. In *International Conference on Machine Learning*. PMLR, pp. 5142–5151.
- Prabhumoye S., Tsvetkov Y., Salakhutdinov R. and Black A. W.** (2018). Style transfer through back-translation. In Gurevych I. and Miyao Y. (eds), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 866–876. <https://aclanthology.org/P18-1080/>
- Radford A., Metz L. and Chintala S.** (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In Bengio Y. and LeCun Y. (eds), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. <http://arxiv.org/abs/1511.06434>
- Radford A., Narasimhan K., Salimans T. and Sutskever I.** (2018). Improving language understanding by generative pre-training. <https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf>
- Rao S. and Tetreault J. R.** (2018). Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer. In Walker M. A., Ji H. and Stent A. (eds), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, pp. 129–140. <https://doi.org/10.18653/v1/n18-1012>
- Reiter E. and Belz A.** (2009). An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics* 35(4), 529–558.
- Rush A. M., Chopra S. and Weston J.** (2015). A neural attention model for abstractive sentence summarization. In Márquez L., Callison-Burch C., Su J., Pighin D. and Marton Y. (eds), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*. The Association for Computational Linguistics, pp. 379–389.
- Sennrich R., Haddow B. and Birch A.** (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics. <https://doi.org/10.18653/v1/p16-1009>
- Shen T., Lei T., Barzilay R. and Jaakkola T. S.** (2017). Style transfer from non-parallel text by cross-alignment. In Guyon I., von Luxburg U., Bengio S., Wallach H. M., Fergus R., Vishwanathan S. V. N. and Garnett R. (eds), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6830–6841. <https://proceedings.neurips.cc/paper/2017/hash/2d2c8394e31101a261abf1784302bf75-Abstract.html>
- Sudhakar A., Upadhyay B. and Maheswaran A.** (2019). “transforming” delete, retrieve, generate approach for controlled text style transfer. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*. Association for Computational Linguistics, pp. 3267–3277.



- Sun S., Cheng Y., Gan Z. and Liu J.** (2019). Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 4323–4332. <https://aclanthology.org/D19-1441>
- Sutskever I., Vinyals O. and Le Q. V.** (2014). Sequence to sequence learning with neural networks. In Ghahramani Z., Welling M., Cortes C., Lawrence N. D. and Weinberger K. Q. (eds), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3104–3112. <https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html>
- Tian Y., Hu Z. and Yu Z.** (2018). Structured content preservation for unsupervised text style transfer. *arXiv preprint arXiv:1810.06526*.
- Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L. and Polosukhin I.** (2017). Attention is all you need. In Guyon I., von Luxburg U., Bengio S., Wallach H. M., Fergus R., Vishwanathan S. V. N. and Garnett R. (eds), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008. <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>
- Wang K., Hua H. and Wan X.** (2019). Controllable unsupervised text attribute transfer via editing entangled latent representation. *Advances in Neural Information Processing Systems* **32**, 11036–11046.
- Wieting J., Berg-Kirkpatrick T., Gimpel K. and Neubig G.** (2019). Beyond BLEU: Training neural machine translation with semantic similarity. In Korhonen A., Traum D. R. and Márquez L. (eds), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 4344–4355. <https://doi.org/10.18653/v1/p19-1427>
- Wieting J. and Gimpel K.** (2018). ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne: Association for Computational Linguistics, pp. 451–462. <https://aclanthology.org/P18-1042>
- Williams R. J.** (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* **8**(3-4), 229–256.
- Wolf T., Debut L., Sanh V., Chaumond J., Delangue C., Moi A., Cistac P., Rault T., Louf R., Funtowicz M., Davison J., Shleifer S., von Platen P., Ma C., Jernite Y., Plu J., Xu C., Scao T. L., Gugger S., Drame M., Lhoest Q. and Rush A. M.** (2020). Transformers: State-of-the-art natural language processing. In Liu Q. and Schlangen D. (eds), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP. 2020 - Demos, Online, November 16-20, 2020*. Association for Computational Linguistics, pp. 38–45. <https://doi.org/10.18653/v1/2020.emnlp-demos.6>
- Wu C., Ren X., Luo F. and Sun X.** (2019a). A hierarchical reinforced sequence operation method for unsupervised text style transfer. In Korhonen A., Traum D. R. and Márquez L. (eds), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 4873–4883. <https://doi.org/10.18653/v1/p19-1482>
- Wu X., Zhang T., Zang L., Han J. and Hu S.** (2019b). Mask and infill: Applying masked language model for sentiment transfer. In Kraus S. (ed), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, pp. 5271–5277. <https://doi.org/10.24963/ijcai.2019/732>
- Xu J., Sun X., Zeng Q., Zhang X., Ren X., Wang H. and Li W.** (2018). Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In Gurevych I. and Miyao Y. (eds), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Volume 1: Long Papers*. Association for Computational Linguistics, pp. 979–988.
- Xu P., Cheung J. C. K. and Cao Y.** (2020). On variational learning of controllable representations for text without supervision. In *International Conference on Machine Learning*. PMLR, pp. 10534–10543.
- Xu W., Ritter A., Dolan B., Grishman R. and Cherry C.** (2012). Paraphrasing for style. In Kay M. and Boitet C. (eds), *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*. Indian Institute of Technology Bombay, pp. 2899–2914. <https://aclanthology.org/C12-1177/>
- Yang Z., Dai Z., Yang Y., Carbonell J. G., Salakhutdinov R. and Le Q. V.** (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Wallach H. M., Larochelle H., Beygelzimer A., d'Alché-Buc F., Fox E. B. and Garnett R. (eds), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 5754–5764. <https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html>
- Yang Z., Hu Z., Dyer C., Xing E. P. and Berg-Kirkpatrick T.** (2018). Unsupervised text style transfer using language models as discriminators. In Bengio S., Wallach H. M., Larochelle H., Grauman K., Cesa-Bianchi N. and Garnett R. (eds), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 7298–7309. <https://proceedings.neurips.cc/paper/2018/hash/398475c83b47075e8897a083e97eb9f0-Abstract.html>

- Yarats D. and Lewis M.** (2018). Hierarchical text generation and planning for strategic dialogue. In Dy J. G. and Krause A. (eds), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, Vol. 80. PMLR, pp. 5587–5595. <http://proceedings.mlr.press/v80/yarats18a.html>
- Yi X., Liu Z., Li W. and Sun M.** (2020). Text style transfer via learning style instance supported latent space. In Bessiere C. (ed), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*. International Joint Conferences on Artificial Intelligence Organization, pp. 3801–3807. <https://doi.org/10.24963/ijcai.2020/526>
- Yin D., Huang S., Dai X.-Y. and Chen J.** (2019). Utilizing non-parallel text for style transfer by making partial comparisons. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, pp. 5379–5386. <https://doi.org/10.24963/ijcai.2019/747>
- Yuan W., Neubig G. and Liu P.** (2021). Bartscore: Evaluating generated text as text generation. In Ranzato M., Beygelzimer A., Dauphin Y., Liang P. and Vaughan J. W. (eds), *Advances in Neural Information Processing Systems*, Vol. 34. Curran Associates, Inc., pp. 27263–27277. <https://proceedings.neurips.cc/paper/2021/file/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Paper.pdf>
- Zhang Y., Ding N. and Soricut R.** (2018). SHAPED: Shared-private encoder-decoder for text style adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, LA: Association for Computational Linguistics, pp. 1528–1538. <https://aclanthology.org/N18-1138>
- Zhang Y., Xu J., Yang P. and Sun X.** (2018a). Learning sentiment memories for sentiment modification without parallel data. In Riloff E., Chiang D., Hockenmaier J. and Tsujii J. (eds), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, pp. 1103–1108. <https://doi.org/10.18653/v1/d18-1138>
- Zhang Z., Ren S., Liu S., Wang J., Chen P., Li M., Zhou M. and Chen E.** (2018b). Style transfer as unsupervised machine translation. *arXiv preprint arXiv: 1808.07894*.
- Zhao J., Kim Y., Zhang K., Rush A. and LeCun Y.** (2018a). Adversarially regularized autoencoders. In Dy J. and Krause A. (eds), *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 80. PMLR, pp. 5902–5911. <http://proceedings.mlr.press/v80/zhao18b.html>
- Zhao Y., Bi W., Cai D., Liu X., Tu K. and Shi S.** (2018b). Language style transfer from sentences with arbitrary unknown styles. *arXiv preprint arXiv: 1808.04071*.
- Zhou C., Chen L., Liu J., Xiao X., Su J., Guo S. and Wu H.** (2020). Exploring contextual word-level style relevance for unsupervised style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, pp. 7135–7144. <https://aclanthology.org/2020.acl-main.639>

## Appendix A. Best performance models

In order to make the experiments conducted in this manuscript reproducible, in the Tables 8 and 9, we list the hyperparameters used in the training of the models of Table 5. The learning rate, the

**Table 8.** Hyperparameters used in training. For sentiment transfer 0 means negative and 1 positive. For author imitation 0 means shakespearean and 1 modern English

Model	Sentiment transfer				Author imitation			
	$T_{KD}$	$\alpha$	$(n_d, n_f)$	$(a_1, a_2)$	$T_{KD}$	$\alpha$	$(n_d, n_f)$	$(a_1, a_2)$
FULL 0→1	10	0.1	(10,5)	(0.25,0.5)	1	0.5	(7,5)	(0.15,0.3)
FULL 1→0	10	0.1	(10,5)	(0.25,0.5)	5	0.1	(7,5)	(0.15,0.3)
NO KD 0→1	–	0	(7,5)	(0.15,0.3)	–	0	(7,5)	(0.15,0.3)
NO KD 1→0	–	0	(10,5)	(0.25,0.5)	–	0	(7,5)	(0.15,0.3)
NO PARA	10	0.1	(10,5)	(0.25,0.5)	5	0.5	(7,5)	(0.2,0.5)
NO PARA KD	–	0	(10,5)	(0.25,0.5)	–	0	(10,5)	(0.25,0.5)
NO ADV	10	0.5	–	(0.25,0.5)	5	0.5	–	(0.15,0.3)
NO ADV KD	–	0	–	(0.25,0.5)	–	0	–	(0.15,0.3)



**Table 9.** Hyperparameters for training. For formality transfer 0 means informal and 1 formal

Model	Sentiment transfer			
	$T_{KD}$	$\alpha$	$(n_d, n_f)$	$(a_1, a_2)$
FULL 0→1	10	0.65	(7,5)	(0.15,0.3)
FULL 1→0	5	0.65	(7,5)	(0.15,0.3)
NO KD	-	0	(7,5)	(0.15,0.3)
NO PARA 0→1	1	0.5	(7,5)	(0.15,0.3)
NO PARA 1→0	5	0.5	(7,5)	(0.15,0.3)
NO PARA KD	-	0	(10,5)	(0.25,0.5)

maximum sentence size, and the word dropout were kept untouched throughout the experiments. For the sentiment transfer task, these values are 0.0001, 32, and 0.2, respectively. For the author imitation task, the values are 0.0001, 64, and 0.25, respectively. And for the formality transfer task, the values are 0.0001, 32, and 0.25, respectively.

**Appendix B. Examples of sentences collected from the experiments conducted on the sentiment transfer task**

Table 10 brings some transferred sentences collected from MATTES (FULL MODEL), DLSM, and Style Transformer.

**Table 10.** Examples of transferred sentences in the sentiment transfer task

Model	Negative to positive
Source	I guess she wasn't happy that we were asking the prices.
Deep Lat	I love it and i love the happy hour we were expecting the prices.
Style Tranf.	I recommend she was always happy that we were asking the prices.
MATTES	I recommend she wasn't happy that we were asking the prices!
Source	Went to the sunday brunch to celebrate our daughter's college graduation.
Deep Lat	Went to the sunday brunch to celebrate our daughter's college favorite.
Style Tranf.	Easy to the sunday brunch to celebrate our daughter's college graduation.
MATTES	Came to the sunday brunch to celebrate our daughter's college graduation!
Source	You can not judge people based on appearance.
Deep Lat	You can definitely count based on time on appearance.
Style Tranf.	You can definitely judge people based on appearance.
MATTES	You can a judge people based on appearance!
Source	Did they not have a fountain machine on site?
Deep Lat	Lots of fun, outdoor seating on the site!
Style Tranf.	Did they well have a fountain machine on site!
MATTES	Did they also have a great jazz machine on site?

Table 10. Continued

Model	Positive to negative
Source	Rick is a seriously cool guy!
Deep Lat	Ugh.
Style Tranf.	Rick was a seriously over guy?
MATTES	Rick is a seriously cold guy!
Source	I highly recommend e & m painting.
Deep Lat	I wouldn't recommend & m m.
Style Tranf.	I won't be e & m painting.
MATTES	I do not recommend e & m painting.
Source	We recommend imports & american auto service to everyone we know.
Deep Lat	We would rather buy an american auto service to everyone we know.
Style Tranf.	We wouldn't billed & american auto service to everyone we know.
MATTES	We wouldn't imports or american auto service to everyone we know.
Source	Loved the burgers, i had the jalapeo ranch burger it was really tasty.
Deep Lat	However, the burgers i had the jalapeo ranch burger it was really disappointing.
Style Tranf.	Ordered the burgers, i had the jalapeo ranch burger it was really tasty.
MATTES	Used to the burgers, i had the jalapeo ranch burger it was really tasty.

### Appendix C. Evolution of metrics

Figure 3 shows the style accuracy, self-bleu and harmonic mean (HM) achieved throughout training by our FULL MODEL and its variation without the pre-training strategy (NO PARA), on the Yelp validation set.

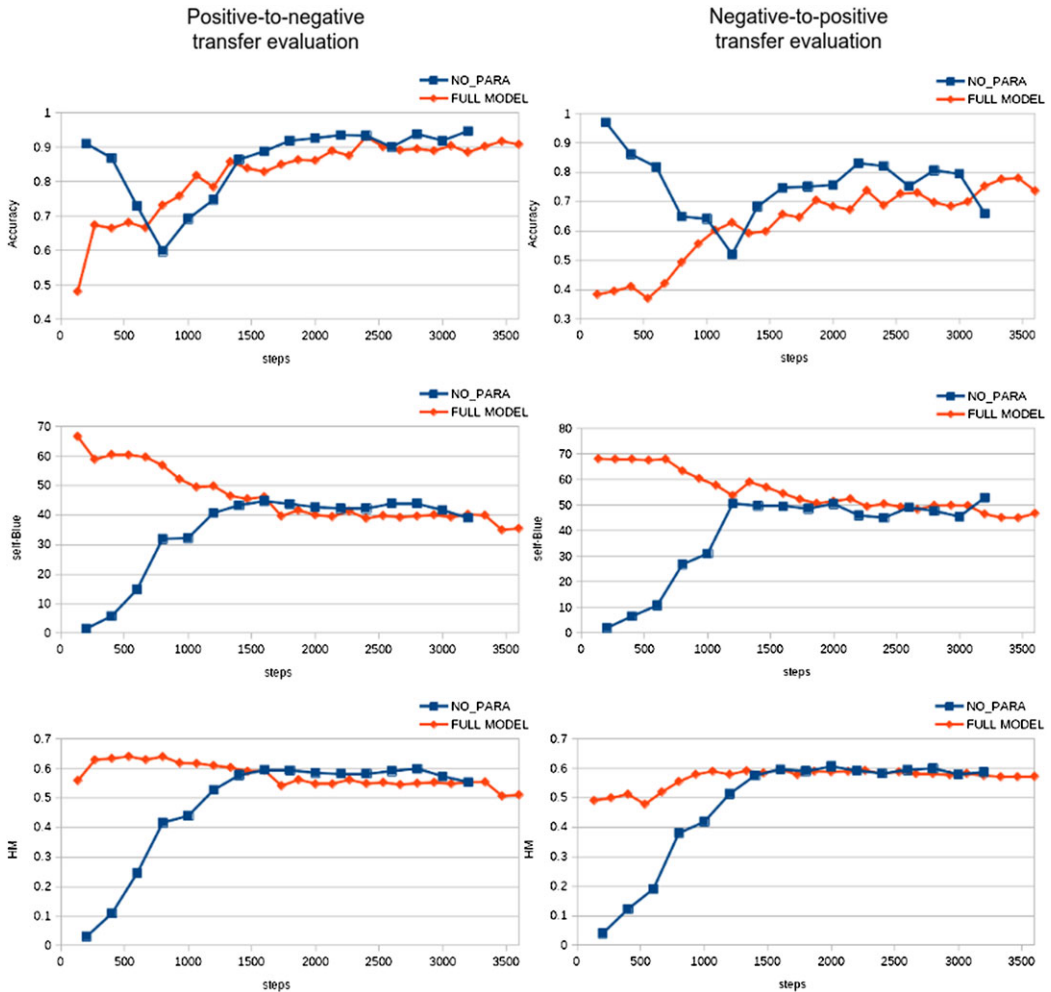


Figure 3. Evaluation metrics, for both transfer directions, during training for our FULL MODEL and its variation NO PARA on the Yelp dataset

### Appendix D. Trade-off: Bleu versus Accuracy

Table 11 shows models selected using a higher threshold for the style accuracy. As shown, the improvement comes at the expense of reducing the content preservation.

**Table 11.** Metrics form model variations trained from scratch. The best results are in bold.

Task	Model	Acc↑	BLEU↑	HM↑
Sentiment	$T = 5; \alpha = 0.5$	<b>89.9</b>	19.16	0.316
transfer	$T = 10; \alpha = 0.5$	88.3	<b>20.52</b>	<b>0.333</b>
Author	$T = 5; \alpha = 0.5$	<b>81.3</b>	9.13	0.164
imitation	$T = 1; \alpha = 0.5$	72.0	<b>11.24</b>	<b>0.194</b>