Intelligent Writing Assistant: Leveraging Text Style Transfer and Discourse Coherence Generation for Enhanced Document Editing

Student Name: C.S. Maguire
Supervisor Name: B.D. Martin
Submitted as part of the degree of BSc Computer Science to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract—Text style transfer (TST) and discourse coherence generation (DCG) are crucial for creating stylistically diverse and coherent text. However, current TST models struggle with style transfer accuracy, content preservation, and fluency. This study investigates the performance of four TST models on a more complex style transfer task and evaluates the effectiveness of incorporating a DCG module. We also compare the proposed DCG implementation with a large language model (LLM) based approach.
Our results show that the TST models generally perform poorly, with LEWIS achieving slightly better results. The DCG module improves coherence and fluency, as evidenced by increased BARTScore and decreased perplexity. However, the LLM-based DCG approach outperforms the proposed implementation.
The challenges encountered, such as the trade-off between style transfer accuracy and content preservation, highlight the need for further research to develop more effective TST and DCG techniques. Future work should explore novel architectures, investigate the impact of larger context windows, and address the limitations of the current study.
This research contributes to the growing body of work on TST and DCG, emphasizing the potential of these technologies and the need for further investigation to unlock their full potential.

Index Terms—text style transfer, Discourse Coherence Generation

1 Introduction

In the context of text, ”style” refers to the unique way a piece of writing is presented or expressed. It encompasses various elements such as tone, voice, syntax, vocabulary, and formatting choices, all of which contribute to the overall perception of the text by the reader. A subspace of style is document coherence, which refers to the quality of a written text that makes it logically organized, cohesive, and easy to follow. It involves the smooth flow of ideas and information from one sentence, paragraph, or section to the next, creating a sense of unity and clarity throughout the document.Text Style Transfer (TST) has been developed to automatically control the style attributes of text while preserving the content. The style of texts is crucial as it makes natural language tasks more user-friendly. Many applications apply TST, such as chatbots, where users prefer a distinct persona of the bot rather than an emotionless or inconsistent persona.
This paper aims to build an intelligent writing assistant application that leverages the power of Large Language Models (LLMs) such as ChatGPT for document editing and text enhancement [1]. However, a problem encountered when using LLMs is the specific style of ChatGPT outputs. We propose to tackle this issue by transferring the style of the outputs of ChatGPT, transferring the output style to the user's style. The application will have to perform complex style transfer while retaining the content of the ChatGPT message, learn the user's style from limited data quickly. Furthermore we aim to tackle the problems in document coherence that will occur due to ChatGPT not being able to see the whole document because of it's $\tilde{5}00$ word token limit.
Current TST approaches have limitations, focusing mainly on sentence-level transfer, relying on large datasets for training, and targeting easily discernible styles such as sentiment, formality, and toxicity [2], [3], [4], [5], [6]. There is minimal research into complex style transfer with the clossest task being the transfer of shakespearean english to contemporary [7]. Furthermore there is minimal research into document coherence, with most focusing on changing words to ensure information consistancy rather then sentence to sentence changes [8]. In this paper, we aim to asses TST on more fine-grained style transfer, experimenting with different frameworks to measure their success at authorship style. Additionally, we will apply Discourse Coherence Generation (DCG) to the output to ensure flow between the outputted text and the edited document. Our deliverables include analyzing large language model-based TST as well as [9], [10], [11] model specific methods for TST and DCG, analyzing style transfer-specific models on complex style tasks, and creating a novel DCG implementation to assess against large language model-based DCG output
the main contributions of this paper are:

- providing an evaluation and deeper understanding of commonly used TST approaches and their ability to transfer style and maintain meaning over a fine-

grained style transfer task;
- providing specific use cases and examples for TST in a document editing software;
- providing results on our proposed DCG method against baseline LLM approaches using human analysis and newly proposed automatic evaluation metrics.

2 Related Work

2.1 Introduction to Text Style Transfer

the main goal of text style transfer (TST) is to paraphrase a sentence by writing it from an input style to a target style while preserving the semantic meaning (content) of the text. The term style is used broadly and encompasses multiple aspects of text such as: *register (formality) [5], politeness [12], toxicity [4], political slant [13], sentiment [6] and specific author writing styles [7].* table 1 shows some examples of the common styles used in literature.

2.2 Datasets

For TST, training on parallel datasets consisting of directly translated sentence pairs in input and target styles is most efficient. However, creating large-scale, high-quality parallel datasets for TST is challenging due to the scarcity of naturally occurring parallel text with the same content but different styles, and the time-consuming, expensive, and subjective nature of manual annotation [2], [3]. In this research, we focus on non-parallel methods for TST to explore more flexible and scalable approaches that leverage readily available non-parallel corpora. By doing so, we aim to develop techniques that are more practical and applicable to real-world scenarios where parallel data is limited or unavailable.

2.2.1 Non-Parallel Datasets

Non-parallel data refers to training models on separate mono-style corpora of the src and tgt styles. due to the lack of specific parallel examples for the models new methods are devised to train the models. Three main branches have been developed to combat the lack of parallelism in the datasets [3] which include Prototype editing, Disentanglement and Pseudo-parallel corpus generation.

2.3 Current TST models

all areas of TST have seen vast improvements with most of the current literature developing from simple TST tasks such as sentiment and formality to more complex tasks such as political slant and authorship [7], [9], [10], [13], [14]. However most of the recent models and research still use formality and sentiment to baseline and compare the models against. We will now look into the subsections of TST implementations and understand how current models work as well as their limitations and advantages in reference to the projects use case.

2.3.1 Prototype editing or explicit disentanglement

Prototype editing models transfer style by masking attribute markers in the input sentence and replacing them with target style markers. The key steps in this approach are:

- Masking style-specific words using n-gram detection [15] or attention-based methods [16].
- Finding suitable substitutes for the masked words based on context, using cosine similarity of word embeddings like TF-IDF [7], [15] or averaged GloVe embedding distance [15].
- Passing the prototype sentence through a pre-trained language model to generate a fluent target sentence.

Prototype editing models are simple, explainable, and require shorter training times. However, they struggle with transferring fine-grained styles that are entangled throughout the sentence.

2.3.2 Disentanglement or Implicit Disentanglement

Disentanglement aims to separate the style attribute a from the src text into a latent representation src', which is then decoded with a learned target attribute vector a' to create the tgt style text. The main deep learning models used are:

- **Auto Encoders (AE):** Maps the input text to a lower-dimensional latent space representation to separate the content from the source attribute a. Pre-trained AEs have become prevalent for improved semantic similarity and fluency [17], [18], [19], [20].
- **Variational Auto Encoders (VAE):** Similar to AEs, but outputs a probabilistic representation in the latent space which is then sampled by the decoder to generate an output. Kullback-Leibler Divergence loss is used to regularize the function and encourage learning a desired latent space distribution [21], [22].
- **Generative Adversarial Networks (GANs):** These approximate samples from a true distribution using a generator and discriminator. The discriminator learns to spot the source attribute a and judge the fluidity of the generated text with the generator aiming to generate realistic samples that cannot be discriminated [23].

A key component of disentanglement is how the latent space is manipulated to represent style and content information. Latent space manipulation methods include:

- **Latent Representation Editing (LRE):** This ensures the latent representation src' can reconstruct the input text and incorporate a' by itteratively updating src' using an attribute classifier [24].
- **Attribute Code Control:** Ensures the latent embedding src' only contains content information through adversarial learning, and decodes the transferred output based on src' and a learned target attribute a' embedding [23], [25].
- **Latent Representation Splitting (LRS):** Disentangles the input into src' (content) and a (attribute), and replaces a with a' to generate the transferred text [24], [26], [27].

We aim to experiment with the model proposed by John et al [26] which uses a VAE with LRS. It does this by using

Task	Input sentence	Output sentence
informal to formal transfer	Gotta see both sides of the story. (informal)	You have to consider both sides of the story. (formal)
polite to impolite transfer	Could you please send me the data? (polite)	Send me the data. (non-polite)
toxic to non toxic transfer	I hope they pay out the ***, fraudulent or no. (offensive)	I hope they pay out the state, fraudulent or no. (non-offensive)
political slant transfer (left wing to right wing)	The progressive government took a bold step forward in tackling the climate crisis by announcing a comprehensive policy (left-wing)	The government announced yet another policy to address the so-called ”climate change” issue (right-wing)
sentiment transfer (negative to positive)	Great food, but horrible staff and very very rude workers! (negative)	Great food, awesome staff, very personable and very efficient atmosphere! (positive)
authorship transfer (Shakespearean to contemporary)	My lord, the queen would speak with you, and presently. (shakespearean english)	My lord, the queen wants to speak with you right away. (contemporary english)
ChatGPT to Personal writing	It is characterized by its clarity, precision, objectivity, and adherence to specific conventions and formats. (ChatGPT)	it is shown as clear and precise while sticking to specific formatting (Personal writing)

TABLE 1
A curated set of examples of common style transfer tasks with examples from [3] and our own personal writing dataset

a style classifier on the latent space which maximize the classification of the style latent space, a content classifier that classifies the content latent space by maximizing the bag of words overlap of the output and a discriminator which evaluate if the created style embedding is different to the target style embedding [26].

2.3.3 *Non-disentanglement Methods*

Recent studies have questioned the effectiveness and necessity of content disentanglement in style transfer [28]. Alternative techniques such as Reinforcement Learning (RL) [29], pseudo-parallel corpus generation [6], [30], [31], and few-shot large language model (LLM) approaches have been proposed. The task is formulated as an RL problem, where the model interacts with an environment consisting of the input text, target style, and reward function. The model learns a policy to generate text and receives rewards based on style accuracy, content preservation, and fluency. The policy is updated using techniques like policy gradient or actor-critic methods to maximize the expected cumulative reward. RL offers advantages in optimizing for specific objectives and handling discrete actions, but training can be challenging due to high reward variance and the need for careful design of the reward function and training process. [29]. Pseudo-parallel corpora provide greater training signal by mimicking parallel data. Two main types of corpora creation are:

- **Retrieval-based:** Align semantically similar sentence pairs using cosine similarity between pre-trained sentence embeddings [30].
- **Generation-based:** Employ Iterative Back-Translation (IBT) [3], [31], [32], which itteratively generates pseudo-parallel corpora using two style transfer models, $X_{a \rightarrow a'}$ and $X_{a' \rightarrow a}$. Models are trained on the corresponding generated dataset, and style classification loss is introduced to ensure the generated sentences match the desired style.

few shot LLM based approaches use minimal data (few shot) to enhance the output of an LLM to transfer style. Some examples use non-parallel data and small amounts of human annotated text to achieve competitive results for TST

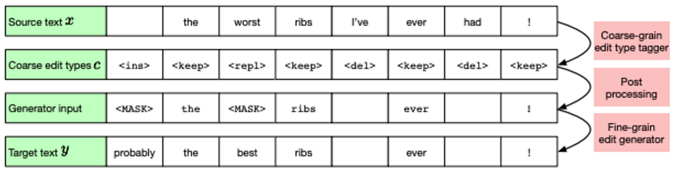


Fig. 1. inference process of lewis explaining the course to fine edits [31].

tasks [10] or use zero shot (no training data) specific prompt templates to create complex style transferred examples [9].

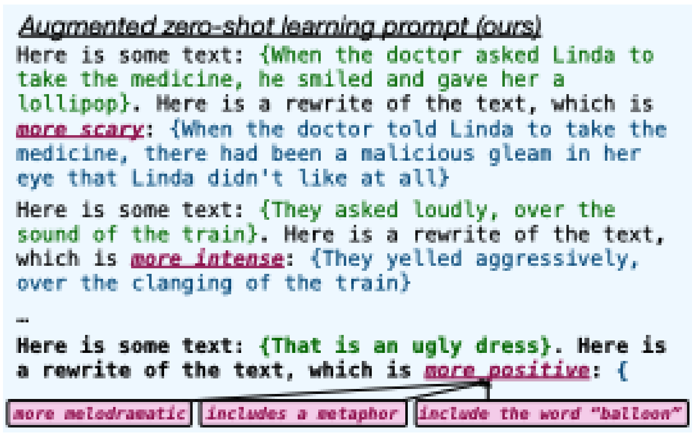
We plan on using LEWIS created by Reid and Zhong [31] which uses IBT and a two stage editor to perform TST. First the model uses IBT synthesise a pseudo-parallel corpora using BART [33] masked language models and a RoBERTa [34] style classifier. The classifiers attention weights are used to identify and remove style specific content similar to those used in prototype editing which are then infilled to generate style agnostic templates. Next, the coarse-to-fine editor is trained on the synthesized data. The editor consists of a RoBERTa-based tagger that predicts coarse-grain Levenshtein edit operations (insert, keep, replace, delete) for each token in the source text, and a fine-grain edit generator that fills in the final target text based on the edit operations. During inference, the source text is first tagged with edit operations, and then the generator produces the stylistically transferred target text. 1 shows the inference process of lewis[1].

We will also use the zero shot LLM approach created by Reif et al [9] which uses a prompt template to generate style transfered text. 2 shows the prompt template proposed which uses repeat reprompting to generate style transfered text.

2.4 Models used in DCG

Discourse Coherence Generation (DCG) plays a crucial role in Text Style Transfer (TST) tasks. While there has been extensive research into discourse coherence analysis [35], there is minimal research focusing on generating coherent text in the context of style transfer. TST tasks often focus

1. https://github.com/machelreid/lewis



Augmented zero-shot learning prompt (ours)

Here is some text: {When the doctor asked Linda to take the medicine, he smiled and gave her a lollipop}. Here is a rewrite of the text, which is more scary: {When the doctor told Linda to take the medicine, there had been a malicious gleam in her eye that Linda didn't like at all}

Here is some text: {They asked loudly, over the sound of the train}. Here is a rewrite of the text, which is more intense: {They yelled aggressively, over the clanging of the train}

…

Here is some text: {That is an ugly dress}. Here is a rewrite of the text, which is more positive: {

more melodramatic | includes a metaphor | include the word "balloon"

Fig. 2. an example of the prompt template used in zero shot transfer [9]

on short, sentence-to-sentence transfers, which can lead to issues with coherence, particularly when the transfer relies heavily on word substitution. Substituting words without considering the overall flow and structure of the text can result in a disjointed and incoherent final product. LLMs have been used as a baseline for many DCG studies [11]. However, there are no general frameworks for DCG, and most approaches focus on simplifying the task for MLMs or on Long Text Generation LTG. One recent study by liang et al [36] proposes a sliding attention mechanism to improve long text generation, which prevents the model from collapsing on longer texts by not taking in non-linked dependencies. They also propose a linear temperature decay to control the sampling process and prevent information loss over iterations. Another approach, CSP (Coherence-aware Soft Prompt) [37], focuses on generating multi-sentence text given an input. The model consists of three parts: a prompt generator, a prompt posterior generator, and a text generator. The prompt generator creates a soft prompt representing the hidden representation of the text to be generated, while the prompt posterior generator helps train the prompt generator by providing hidden representations of positive and negative samples. These prompts are then used to train an LLM to generate text with better representation of the previous text. EGRID (Entity Grid) trains a random forest classifier to classify feature vectors for coherence assessment. ParSeq (Paragraph Sequence Model) extends this idea by incorporating paragraph breaks, which are expected to be important for assessing coherence in longer documents. ParSeq uses three stacked LSTMs to produce sentence, paragraph, and document vectors, with the document vector being used to produce a distribution over coherence labels [38]. Despite these advancements, generating coherent text remains a challenge, particularly when it comes to modeling information and temporal dependencies. Most current approaches focus on long text generation which means they are not adapting the original language but focussing on generating large coherent text from an initial prompt [36], [37] There is a need for more general frameworks that can handle longer texts and maintain coherence across paragraphs and larger sections of text. As well as address the problems with temporal dependencies, and information dependencies over the span of text [35], [38], [39]. Current gaps in DCG include modelling wider dependencies of information and flow as well as creating applied generation methods rather then analysis frameworks.

2.5 Evaluation methods used in TST and DCG

in TST and DCG there are two methods of evaluation automatic and human. Human evaluation is considered the gold standard as it allows for a more fine detailed analysis of the output. Human analysis for TST is based around looking at the original and style transfered text and having the evaluator grade the fluency, style transfer intensity and content preservation of the output with DCG focusing on the content preservation and Fluency of the piece. The outputs are marked using a 1-n scale and this is then averaged over the outputs and researchers to provide and evaluation score. However, as human evalauation can be biased and a lot of the evaluation subjective most research uses human evaluation in tandem with automatic evaluation metrics. These metrics assess the same criteria as the human evalaution, fluency, style transfer strength and content preservation but use specific metrics to evaluate each. For content presrvation the most common metric is BLEU (bilingual evaluation understudy) which measures the similarity of an output with a annotated reference translation by measuring the overlap of n-grams. Although this is often used it requires human annotated references which for a lot of datasets isn't available. There are also some studies that show BLEU score does not show correlation to human evaluation [3]. To enhance the evaluation some works use METEOR [40], ROUGE [41] and Word Movers Distance (WMD) [42]. Mir et al [43] shows that METEOR and WMD correlate the best to human evaluation. For Fluency, most research uses perplexity scores which are calculated via LMs pretrained on the training data of all attributes [44]. There is still debate around how effective perplexity is as a metric as it is biased towards shorter sentences and penalises the use of less frequent words from the dataset [3]. For style evaluation common practice is to train a style classifier that then assesses the outputs of the TST model and provides an accuracy score. Currently there are no other metrics as classifier accuracy is shown to correlate with human evaluation. In DCG automatic metrics for content preservation remain the same as TST however DCG introduces greater fluency evaluation metrics that focus on large context areas then the generated sentence such as DiscoScore [39] and BARTScore [45]. These metrics evaluate the fluency and coherence of the models outputs over a larger area then the sentence level.

3 Solution

In the proposed approach we will take a users highlighted text and prompt which will be sent to ChatGPT to generate a reponse conditioned on this information. We then take the LLM generated text and apply a document transfer pipeline to the text. This will involve transferring the style of our LLM text to the users style of writing. This style transferred text will then be passed through a DCG module that will ensure the input text is coherent to the users original document before it is re-inserted. The three deliverables we aim to achieve are based around this framework and are described as follows:

Dataset	Length	Average word count
GPT-wiki-intro	150K entries	150 words
Newsroom	1.3M entries	2158 words
TinyStories	2.12M entries	192 words
Scientific-papers	203k entries	300 words

TABLE 2
table of statistics from the data we used in the study.

1) using an LLM based approach [9] and a trained style classifier we aim to transfer the text from GPT to the documents classification of style. Seeing if the categorization improves performance and the effectiveness of this method. We will also assess ChatGPTs effectiveness at document coherence using a prompt based approach we will develop.
2) applying existing TST implementations to the ChatGPT output to analyse which method of TST, disentanglement [26], prototype editing [15] and pseudo-parallel corpora generation [31] are best suited to the proposed pipeline.
3) Implementing a novel DCG framework to enhance the output of our TST models to generate a coherent output that fits seamlessly with the users original document.

3.1 Datasets

for the specific problem we will use four datasets. For the source we are going to use GPT-wiki-intro [46], for journalistic writing we will use the newsroom dataset [47], for stories we will use Tinystories [48] and for scientific writing we will use Scientific-papers dataset [49]. Statistics on the datasets can be seen in 2.

All of these datasets are available on huggingface[2] however I will be making a dataset of my own work for the final testing phase. this will be made of my own previous work as independent writing to be used for testing style transfer accuracy on writing that does not fit directly into the specific writing categories outlined as well as using it to test for discourse consistency. To pre-process the text we first apply a created function that removes latex artifacts from the scientific data and normalizes punctuation (for example making ..... into ...). **go over preprocessing in more detail** To reduce hyper-parameter optimization the sentence length is cut to 20 to be around the lengths used in the original papers and configuration files [15], [26], [31]. Due to the size of the datasets we also limited the size of the generated corpora to 500k examples to reduce training times. This is already well above the standard size of dataset in current research which is usually less then 300k examples [2].

3.2 The TST module

The objective of the TST module is to ensure the generated LLM responses style matches the style of the users document to ensure the edited document has a consistent style. A problem we face in doing this is that the users style is distinct to themselves and would involve large amounts of their own text data as well as time to train a specific TST model for their style. To adjust for this we break the problem down by addressing the users style in three distinct categories. Danielle Perry [50] proposes four distinct writing styles Persuasive which is present in editorial newspaper articles, Narrative which is present in stories, Expository which is present in scientific writing and Descriptive which is present in fiction or plays. We use this to categorize the styles our model will have to transfer too allowing us to perform three transfer tasks from ChatGPTs style to Scientific writing (Expository), journalistic writing (Persuasive) and story writing (narrative and Descriptive) which will provide our models with a solid foundation and clear training data. We can see in (figure on writing styles) the distinction between the three categories with (table of styles) providing examples from each. Each style exhibits unique characteristics in terms of sentence structure, vocabulary, and tone, allowing for a clear differentiation and targeted style transfer.

3.2.1 Data Pre-Processing

we can see in figure **??** that the scientific writing contains a lot of latex artifacts. We remove these using a created regular expression function that removes the latex artifacts and normalizes punctuation. we experimented with stopword removal however we deemed that for each style the set of most commonly used stopwords was quite different then those in another style so we did not remove stopwords. We also experimented with Lemmatization which involves splitting words into their roots such as in contractions changing "don't" to "do" and "not" however analysing the n-grams we can see that the scientific data does not contain contractions however newspapers do. This means contractions could indicate a certain style so we decide to keep contractions in and on Lemmatize the data.

3.2.2 Initial Style Classification

We aim to prepend the TST task by training a style classifier that will first categorise the users style into these three distinct groups before applying TST using the relevant model. The style classifier a RoBERTa model for text classification. We choose RoBERTa as it is pre-trained on a large amount of textual data which enables the model to understand rich contextual information from the input text. It also uses a Bidirectional encoding which allows the model to gain context from each token to the left and right of the current token giving the model a greater understanding of a words context. We train this model on the pre-processed corpora of each style using a cross-entropy loss shown in equation 1. where N is the number of samples in the batch, C is the number of classes, $y_{i,c}$ is the true label for sample i and class c (0 or 1) and $p_{i,c}$ is the predicted probability for sample i and class c.

$$\text{Cross-Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \qquad (1)$$

This model will classify the original document of the user to one of the three styles described previously allowing us to use the TST model trained on the relevant data for the specific writing style they use. In this paper we asses each type of TST method being disentanglement, prototype editing

2. https://huggingface.co/

Model	BLEU	ACC
Delete Retrieve Generate	22.8	48.0
LEWIS	24.0	93.1
John et al	*	93.0

TABLE 3
table showing the scores of the models from their papers, *is to denote john et al did not assess using bleu however achieved a 0.47 word overlap metric. all scores are obtained from the model papers themselves [15], [26], [31].

and pseudo-parallel corpora generation, as described in section 2.3. when selecting the TST methods to use we descided to select one model from each type of TST approach. we first evaluated the scores of models on common metrics such as BLEU and style classifier accuracy. We do not address fluency using PPL as it is not yet commonly used. We then chose from these results a disentaglement approach, a prototype editing approach and a pseudo-parallel corpus generation approach. figure 3 shows the results the chosen models achieved.

3.2.3 Model Evaluation and Selection

It is common when assessing TST models to evaluate their style transfer strength, content preservation and fluency. We will evaluate all of these criteria but when selecting our final implementation method we will focus predominantly on transfer strength and content preservation as fluency problems in the outputs will be addressed in the DCG module. To do this we will use the source-BLEU which measures the overlap in N-grams of the outputted transferred text with the original source, this is used as we are unable to get human annotation to perform a standard BLEU metric. For style transfer accuracy we will use a pre-trained RoBERTa classifier that will classify the text and use the accuracy score to provide a metric. We will also use perplexity generated via an LM to evaluate the fluency of the model. these metrics were chosen as they are currently standard practice for TST automatic evaluation [2], [3].

3.2.4 Disentanglement Model Explenation

For disentanglement we will use John et als DAE[3] described in section 2.3.2. We chose this model as it uses many refined techniques to ensure style transfer accuracy such as LRS which is enforced using both an attribute and style embedding classifiers as well as Discriminators. The model is also referenced many times in research [3] proving it to be a foundational model for disentanglement approaches. The model encodes the input to a latent vector using an RNN with GRUs this latent vector then applies LRS , which splits the style and content of of the latent embedding and substitues the input style for the target style. A decoder RNN reconstructs the input itself word by word from the manipulated latent space. The model uses multi-task Loss and adversarial loss for both style and content, and both the encoder and decoder are trained using a Sequence-Aggregated Cross Entropy Loss making it a Deterministic Autoencoder (DAE) [26].

3. Code available here: https://github.com/h3lio5/linguistic-style-transfer-pytorch

Sequence-Aggregated Cross Entropy Loss:

$$J_{\text{AE}}(\theta_E, \theta_D) = -\sum_{t=1}^{n} \log p_t$$

where θ_E, θ_D are the parameters of the encoder and decoder respectively and p is the probability the decoder predicts the word x_t given as $p(x_t|h, x_1 \ldots, x_{t-1})$ where h is the hidden state of the encoder [26].

Multi-Task Loss:

$$J_{\text{mul}}(\theta_E) = -\sum_{l \in \text{labels}} \log(y_c)$$

where y is the softmax of the classifiers output for style and content classifiers and θ_E is the parameters of the respective classifier, labels refers to the classification labels [26].

Adversarial Loss

$$J_{dis}(\theta_{dis}) = -\sum_{l \in \text{labels}} \log(y_d)$$

where y_d is the softmax of the discriminators guess on the style and content latent space, and $\theta_d is$ is the parameters of the respective discriminator [26].

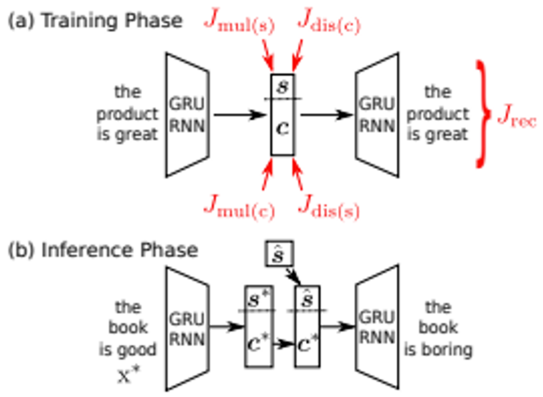


Fig. 3. The model architecture outline from [**?**]. where s and c are used to refer to the style and content space.

3.2.5 Prototype Editing Model Explenation

For the prototype editing based implementation we are using the model outline in section 2.3.1 Delete Retrieve Generate [15]. which is the first implementation of prototype editing and achieves impressive results in content preservation which is a key part of TST success. For prototype editing we will use the original Delete-Retrieve-Generate (DRG) method [15], which is the original prototype editing implementation. DRG works by deleting style specific words from the input to create a prototype sentence using n-gram analysis to find words with the most discriminative power. This is denoted formally by 3.2.5.

$$s(u, v) = \frac{count(u, D_v) + \lambda}{(\sum_{v' \in V, v' \neq v} count(u, D'_v)) + \lambda} \tag{2}$$

where D_v is the set of all words in the source style with attribute v, $count(u, D_v)$ is the number of times an n-gram

u appears in D_v, V is the set of possible attributes and λ is a smoothing parameter. u is declared an attribute marker if $s(u, v)$ is above a threshold. The model then retrieves similar sentences in the target style using TF-IDF weighted word overlap or the Euclidean distance of pre-trained content embeddings. Finally the model uses an RNN to encode the prototype sentence and uses another RNN and learned embedding to encode the retrieved target attribute markers, concatenating the two vectors and decoding these to generate Y using an RNN decoder. The models framework is shown in figure 4[4].

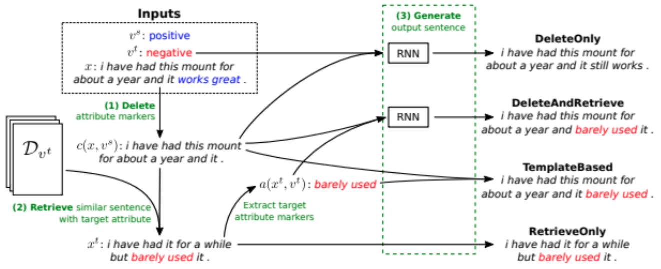

Inputs
v^s: positive
v^t: negative
x: i have had this mount for about a year and it works great .
(1) Delete attribute markers
$\mathcal{D}_{v^t}$
$c(x, v^s)$: i have had this mount for about a year and it .
(2) Retrieve similar sentence with target attribute
x^t: i have had it for a while but barely used it .
Extract target attribute markers
$a(x^t, v^t)$: barely used
(3) Generate output sentence
RNN
RNN
DeleteOnly
i have had this mount for about a year and it still works .
DeleteAndRetrieve
i have had this mount for about a year and barely used it .
TemplateBased
i have had this mount for about a year and it barely used .
RetrieveOnly
i have had it for a while but barely used it .

Fig. 4. The DRG model overview from [15]. The overview explains the separate stages of the model and the other methods proposed in the paper

For pseudo-parallel corpora generation we will use LEWIS [31]. This model advances on traditional approaches utilising IBT corpus creation. We decided to use this model as it achieves SOTA results in style transfer accuracy as well as giving a good evaluation of pseudo-parallel corpora generation based methods[5]. LEWIS is based on Levenshtein operations which are insert, replace, delete and keep. The first step of training involves fine-tuning a RoBERTa tagger [34] to tag the input with edit types for each token based on the Levenshtein edits. Next a BART [33] model which is a masked seq2seq model that is trained to take in the original source text and the source text with the RoBERTa predicted edit types applied during inference. The BART model learns to fill in the replace tags and insert tags with appropriate phrases. To assist in the training of the BART model IBT is performed to create a pseudo-parallel dataset. First a style classifier is created using RoBERTa. The attention layers from the classifier are used to identify style specific words which are indicated by their attention weights. The style specific words are then replaced with a specific SLOT token to create style agnostic templates. Two BART models are then fine tuned: one to generate source style text and the other to generate target style text from the style agnostic templates. An example of this is shown in Figure 6 The created corpus consisting of the generated source-style and target style sentences, is then used to train the final BART model that learns to infill the Levenshtein edits. An example of this process is shown in Figure 5

3.3 DCG module

for the DCG module we aim to build on existing methods by moving away from a long text generation based approach

4. https://github.com/rpryzant/delete_retrieve_generate
5. code is available here: https://github.com/machelreid/lewis

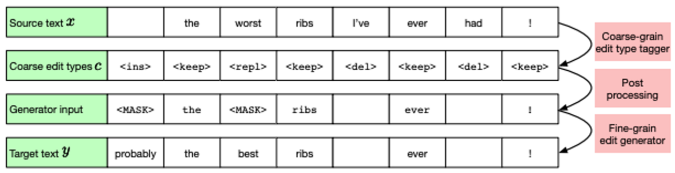


Fig. 5. an example of the inference process taken from [31]

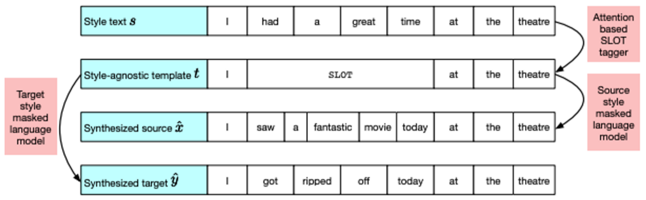

Style text s
I
had
a
great
time
at
the
theatre
Attention based SLOT tagger
Style-agnostic template t
I
SLOT
at
the
theatre
Target style masked language model
Source style masked language model
Synthesized source $\hat{x}$
I
saw
a
fantastic
movie
today
at
the
theatre
Synthesized target $\hat{y}$
I
got
ripped
off
today
at
the
theatre

Fig. 6. an example of the psuedo-parallel corpus generation process taken from [31]

used in previous works [36], [51] to a adaptation based approach that will aim to adapt the users text rather then generate. To do this we will employ the use of a classifier and a masked language model. First the classifier will be trained to identify areas of incoherence in text to allow the masked language model to use a windowed approach [51]. This method aims to narrow the problem space down to allow the model to identify key areas without having to attend to the whole output of ChatGPT. The methodology adapts a similar approach to protoype editing by using the attention weights from the classifier to then mask the input section which will then use a masked language model to fill in these masked areas to generate a coherent text similar to the course and fine grained edits used in LEWIS.

3.3.1 Dataset

as this module is an extension of the TST model we aim to train the model on the same data as the TST models. To do this we will use the same datasets as the TST methods however we will not limit the length of the input sentences and instead aim to use larger lengths of data.

3.3.2 Window Classifier

this classifier will be a RoBERTa model. We chose RoBERTa for the same reasons described in section 3.2.2. The RoBERTa model aims to classify text as coherent or incoherent. This is done by giving the model segments of the input which is k sentences long where k is a hyper-parameter. We then get the model to classify the span S_k as coherent or incoherent. To create the incoherent data for training we will aim to disorganize the data. To do this we randomly reorganize the input sequence of sentences and words. We create a dataset of randomly organized sentences and the original organized sentences with a label of 0 for incoherent and 1 for coherent assigned. To enhance training we use multi resolution training [52] which trains models on data at different levels of granularity or resolution. The idea behind multi-resolution training is to expose the model to varying levels of complexity and detail in the input data, allowing it to learn and capture information at different scales. This helps the model learn longer range dependencies which is a current problem area in DCG. To do this we will initially train the model on highly disorganized inputs, which is achieved by not only changing the order of sentences but also changing the order of words inside the sentences. As the model learns more we reduce the change in the input to just reorganizing the input sentence order. This means the model will see coherent sentences but the overall text will be incoherent allowing the model to learn longer dependencies then just word to word dependencies. We train the model on the data using a cross-entropy loss shown in equation 1 ensuring the classifier is able to classify k sentences as coherent or incoherent.

3.3.3 Masked Language Model

we will use a BART LM to fill in the masked segments of the input. BART is a pre-trained seq2seq language model [33]. BART uses a bidirectional auto encoder and a left to right transformer decoder that is pre-trained on over 100GB of data to ensure the model has a initial understanding of text data. This makes the model effective for text infilling and coherence improvements. In our implementation we will first use the attention of the classifier to identify heavily attended tokens. Tokens that have an attention above a threshold hyper-parameter λ will be masked similar to the process used in LEWIS. This λ parameter is a fixed parameter that aims to mask only words specifically causing incoherence. The heavily attended tokens indicate areas of high incoherence and by replacing these the model will improve the overall coherence of the text. The λ hyper-parameter will be important as if it is too low then the text will not retain any of it's original meaning with too much information being masked and if it is too high will not reduce the incoherence of the output enough. We need to ensure a balance of coherence and meaning is established. For training we will use a dataset similar to the one used in the classifier as w will corupt the sequences in the same way. However, instead of labels the target will be the original text. we will also premask the inputs using the classifier which will mask the altered texts for training. The objective will be to generate the original unaltered sentence by infilling the masked areas and reorganizing the sequence of sentences. For the loss function we will use a simple reconstruction loss which assesses how well the model reconstructs the original sentence.

3.3.4 Inference for the Proposed DCG Model

During inference, the proposed DCG model follows a two-step process to identify and correct incoherent spans in the input text. First, the coherence classifier is applied to the input sentences to detect the incoherent spans. Once an incoherent span is identified, it is passed to the masked language model for coherence improvement. Let's consider a set S of input sentences: $S = \{s_1, s_2, \ldots, s_n\}$ where s_i represents the i-th span of k sentences in the input text. The coherence classifier processes the input sentences sequentially. When the first example of incoherence is encountered, the model splits the text into two parts: the previous sentences that are coherent and the remaining sentences that include the incoherent span and the following sentences. The classifier continues to classify the remaining sentences until coherence is reestablished. Once coherence is detected, the classification process stops, and the identified incoherent span is passed to the masked language model for further processing. To formally define the incoherent span, let I be the set of indices of the spans classified as incoherent: $I = \{i_1, i_2, \ldots, i_m\}$ where i_j represents the index of the j-th incoherent span of length k. The incoherent paragraph S_I can be defined as: $S_I = \{s_i \mid i \in I\}$ In other words, S_I is the subset of spans from S whose indices are in the set I. Once the incoherent span S_I is identified, it is passed to the masked language model (BART) for coherence improvement. The classifiers attention of the incoherent span is used to mask the inputs to BART which takes the input and generates a coherent output by infilling the masked tokens with semantically relevant and coherent words and organizing the sentences. The generated output aims to maintain the overall meaning and context of the input text while improving its coherence. The DCG model then iteratively amends the output by passing it back to the coherence classifier. If the classifier still detects incoherence in the amended output, the process is repeated until either the classifier cannot find any incoherence or a fixed number of iterations have been achieved. This iterative refinement process ensures that the final output is coherent and free from any detected incoherence. We limit the number of itterations the model can perform for two reasons. Firstly the model inference cannot take too long as this is being used in a live application with wait times potentially being an issue. We are also concerned with information loss. If the text is repetitively passed to the model each time the model masks the input there is chance of information loss. If this is allowed to go on continually this could cause a conciderable effect on the output. By combining the coherence classifier and the masked language model in this iterative manner, the proposed DCG model effectively identifies and corrects incoherent spans in the input text, generating a more coherent and readable output.

3.3.5 Evaluation Metrics

To comprehensively evaluate the performance of the proposed DCG model, we employ a combination of automatic metrics that assess different aspects of the generated text. The primary focus is on measuring coherence and content preservation, while also considering fluency and style consistency. Coherence is evaluated using BARTScore [45], an evaluation metric that leverages the pre-trained BART model to assess the quality of generated text. BARTScore measures the semantic similarity between the generated text and human-written reference text, considering factors such as informativeness, coherence, and fluency. By comparing the generated text to the reference text, BARTScore provides an indication of how well the DCG model maintains coherence in the generated output. For our model we will measure the similarity between the original source text, and a mixed up text that has been passed through the model as we do not have access to human annotated data. Content preservation is assessed using S-BLEU (Source-Based BLEU). BLEU [3] is a widely used metric for evaluating the quality of machine-generated text by comparing it to reference text. In this case, we use S-BLEU, which calculates the BLEU score between the generated text and the input text itself. S-BLEU measures the n-gram overlap between the generated text and the input, ensuring that the DCG model preserves the key content and information from the original text. Fluency is evaluated using PPL, a metric that assesses the linguistic quality and naturalness of the generated text. PPL is calculated using a pre-trained language model and measures how well the generated text fits the language model's probability distribution. Lower perplexity scores indicate better fluency, suggesting that the generated text is more natural and easier to understand. To ensure that the DCG model maintains the style consistency of the input text, we employ a style classifier accuracy metric. The style classifier is trained to predict the writing style of the input text, such as formal, informal, or domain-specific styles. By applying the style classifier to the generated text, we can measure the accuracy of style preservation. A higher style classifier accuracy indicates that the DCG model successfully retains the style characteristics of the input text in the generated output. By utilizing these evaluation metrics—BARTScore for coherence, S-BLEU for content preservation, PPL for fluency, and style classifier accuracy for style consistency—we aim to provide a comprehensive assessment of the DCG model's performance. These metrics collectively evaluate the model's ability to generate coherent, content-preserving, fluent, and style-consistent text. Although it is understood that automatic metrics are meant to be done in tandem with human evaluation. We were unable to perform human evaluation as we were unable to get a large enough group to assess the outputs.

4 Experiments and results

4.1 Datasets

We use the scientific papers dataset [49] for training and testing for TST and DCG. We also use the GPT-wiki-intro dataset [46] to use as the source style data in the TST task. we use a pre-processing function that removes latex from the dataset and normalizes the punctuation, where the scientific and ChatGPT style data is then used in the respective TST frameworks. In the DCG framework we load the processed data into a curated dataloader that generates incoherence in the data to create a training and test set for the DCG model. This data-loader randomly shuffles the input with a probability of 0.5. The window size k was set to 2 and the length of tokens to pass to the model was set to the maximum at 514 tokens. The statistics of the dataset are provided in table 4

For DCG we split the scientific-papers dataset into an 8:2 train-test split. We reduce the size of the inputs to a length of 514 to be used in the DCG model. The information can be shown in table 5. For Both models data we use scikit-learn [53] for data splitting and analysis.

4.2 Experimental setup

We first train a Huggingface [54] RoBERTa style classifier on the dataset. This uses a cross entropy loss and classifies the styles of ChatGPT and scientific writing. The model trains until an accuracy of 99% is achieved on the test set. We obtain the Github repositories of each model and use these as the implementation. We train each TST model by creating a bash script that runs each process of the respective models training in the correct sequence. We trained the models on an A100 Nvidia Tesla GPU. After training we evaluate the models by using the models inference scripts to create a set of 100 samples. We created a dataset of 100 zero-shot LLM transfered texts by manually prompting the online interface as we could not get access to the API. We then test each set of generated texts from the test set using a curated evaluation script to generate the evaluation data. We evaluate the models using common TST evaluation methods. We use nltk[7] for Source-BLEU, the trained style classifiers accuracy, and the LMPPL[8] library for the PPL fluency score. The DCG implementation and style classifier were built using the Hugging Face RoBERTa and BART models which we manipulated the training cycles of these models using pytorch [55]. Datasets were loaded using the Hugging Face datasets module, and evaluation metrics were computed using various tools: BARTScore (Hugging Face evaluation module), Perplexity (LMPPL), BLEU scores and text processing (NLTK). We first trained the RoBERTa coherence classifier and then used the trained classifier to generate a training dataset for the BART masked language model. We conducted hyper-parameter optimization on the TST implementations using Parameter-Efficient Fine-Tuning [56] to enhance the models' performance. Due to computational constraints, we trained the models on smaller data batches to understand the effects of hyper-parameters although we did not reduce the parameters in the models. This approach allowed us to fine-tune Delete Retrieve Generate and the disentanglement DAE; however, LEWIS proved too large even with smaller data so a small grid-search of hyper-parameters was conducted. Additionally, we experimented with vocabulary sizes for the disentanglement DAE and Delete Retrieve, while LEWIS used a fixed Byte-Pair Encoding (BPE) vocabulary which did not need tuning. By doubling the hidden and embedding sizes in the disentanglement DAEs latent space, we achieved better results. For DCG, we created an LLM dataset by shuffling and prompting ChatGPT to make the information coherent. Due to the manual process and lack of API access, we generated 100 results to assess the LLM's DCG ability. We then used our curated model and inference framework to generate 100 outputs from the same inputs as ChatGPT. The original inputs were assessed for fluency and BERTScore to assess if the random resorting did generate incoherence. Additionally, we performed discourse coherence analysis on TST outputs to test if the DCG module was required.

5 Results

1) **TST Performance:** The four TST methodsxx evaluated in this study demonstrated varying levels of performance. LEWIS, which utilizes pseudoparallel corpora, achieved the best results among the TST models. In contrast, the prototype editing-based approach, DelRetRegen, performed the worst, with outputs becoming repetitive and lacking diversity in vocabulary and sentence structure (see figure

7. available here https://www.nltk.org/
8. available here https://github.com/asahi417/lmppl

	Train	Dev	test	original data
of docs	GPT:500k, SCI:500K	GPT:46K, SCI:46K	GPT:46K, SCI:46K	GPT:150K, SCI:203K
average word count	GPT:20, SCI:20	GPT:20, SCI:20	GPT:20, SCI:20	GPT:153, SCI:346

TABLE 4
Information on the selected Datasets, during the pre-processing phase the datasets are made to have an equal number of samples and length of 20 words[6].

	Train	Test
of documents	162K	41k
average word count	348	338

TABLE 5
A table showing the statistics of the scientific data used in the DCG training and testing

??). The Zero-shot LLM based approach, showed moderate performance compared to LEWIS and DelRetRegen.

2) **Discourse Coherence Generation (DCG) Effectiveness:** The incorporation of a DCG module improved the fluency and coherence of the outputs generated by all TST models, as evidenced by the increased BARTScore and decreased perplexity metrics in table 6. Notably, a simple zero-shot prompt-based approach using a LLM outperformed the proposed DCG implementation in terms of coherence and fluency. This finding aligns with recent research highlighting the effectiveness of LLM-based approaches for discourse coherence tasks [11].
3) **Quantitative Results:** Table 6 presents the quantitative results of the TST models with and without the DCG module, as well as the LLM-based DCG approach. The metrics used for evaluation include style transfer accuracy, content preservation (BLEU score), BARTScore for coherence, and perplexity for fluency.
4) **Qualitative Analysis:** Qualitative analysis of the generated outputs reveals several notable patterns. DelRetRegen, the worst-performing TST model, produces repetitive language and frequently reuses certain words and phrases (Figure **??**). This repetition highlights the limitations of the prototype editing approach in generating diverse and coherent text. LEWIS and the LLM-based DCG approach generate more coherent and fluent outputs, but they still struggle to accurately transfer the desired style while preserving content.
5) **Limitations and Sources of Error:** Several limitations and potential sources of error should be considered when interpreting the results of this study. First, the sample size used for evaluation is relatively small compared to the training data, which may limit the generalizability of the findings (100 samples in over 150k training samples). Second, due to computational constraints, it was not possible to perform extensive hyperparameter tuning for the TST models, which could have impacted their performance. Third, the data cleaning process may have introduced biases or removed important information. Finally, the use of smaller language models due to GPU limitations prevented the testing of larger context windows, which may have affected the performance of the proposed implementation due to the limited context window of the RoBERTa and BART models. Despite these limitations, the results provide valuable insights into the challenges of style transfer and the effectiveness of DCG approaches in improving the coherence and fluency of generated text. The findings suggest that while incorporating coherence modules can enhance the quality of the outputs, furhter research into the use of LLMs in DCG should be done. The findings also highlight the need for further research to develop more effective style transfer models that can accurately preserve content and transfer the desired style in a more complex task space.

6 Discussion

The results of this study provide valuable insights into the challenges and opportunities in text style transfer (TST) and discourse coherence generation (DCG). The poor overall performance of the TST models highlights the complexity of accurately transferring style while preserving the content of the original text. It also highlights how much more complex the task of authorship style transfer is compared to the formality baselines. Among the evaluated TST models, LEWIS demonstrated relatively better performance, suggesting that the use of pseudoparallel corpora may be a more effective approach compared to prototype editing or other methods. However this could be due to the complexity of the model with this being an extremely large framework compared to the others. The repetitive and less diverse outputs generated by Delete Retrieve Generate underscore the limitations of the prototype editing approach in generating high-quality stylized text. Some works argue that prototype editing simplifies the problem to such a degree that it effects the ability of the model to generate specifically styled outputs [3]. However this model is the initial prototype editing methodology and there have been major improvements in the application of prototype editing. The disentanglement based model was not as bad at transferring style as the Prototype editing method was which is to be expected as for fine grained styles simple delete and retrieve transformations would struggle to generate a sentence that is coherent but imitates the original style. The incorporation of the DCG module consistently improved the coherence and fluency of the TST model outputs, as evidenced by the increased BARTScore and decreased perplexity metrics. This finding emphasizes the importance of considering discourse coherence in text generation tasks. However, the superior performance of the LLM-based DCG approach compared to the proposed DCG implementation raises questions about the effectiveness of current DCG techniques. One limitation of the implementation is it only had a context window of 514

Model	**Style Transfer Accuracy**	**Content Preservation (S-BLEU)**	**Fluency (Perplexity)**	**Coherence (BARTScore)**
DelRetGen	0.25	**0.31**	150.5	0.32
Disentangled	0.35	0.22	98.7	0.39
LEWIS	**0.42**	0.20	85.3	0.45
LLM	0.37	0.15	68.4	0.70
DelRetGen +DCG	0.21	0.24	107.2	0.35
Disentangled + DCG	0.30	0.29	88.3	0.41
LEWIS + DCG	0.33	0.19	78.1	0.49
LLM with coherence prompt	0.34	0.13	**65.0**	**0.76**

TABLE 6
The automatic evaluation results from the testing of the models. The results demonstrate that while the DCG module improves the coherence and fluency of the TST model outputs, the overall performance remains poor. LEWIS and the LLM-based DCG approach achieve the highest scores among the evaluated models, but their performance is still limited.

tokens. We could aim to in future work implement the same architecture with larger models that can process a larger amount of text. We can also look at how the model sees long range dependencies potentially including structured graphs of the input to allow for a better understanding of the text data and it's dependencies [38]. The success of LLM-based approaches aligns with recent research findings in the field [11], suggesting that leveraging the knowledge and capabilities of large language models may be a promising direction for improving discourse coherence in generated text. Despite the improvements brought by the DCG module, the TST models still struggled to achieve a balance between style transfer accuracy and content preservation. This challenge highlights the need for further research to develop more sophisticated TST models that can effectively capture and transfer stylistic elements while minimally altering the semantic content of the input text. Potential areas for improvement include exploring novel architectures, such as adversarial networks [23] or Reinforcement Learning based approaches [29], and investigating the impact of larger context windows on style transfer and coherence. The limitations of this study should be considered when interpreting the results. The small sample size used for evaluation may limit the generalizability of the findings, and the lack of extensive hyperparameter tuning due to computational constraints could have impacted the performance of the TST models. Additionally, cleaning the data fully of latex could introduce potential biases. Future work should aim to introduce a ¡LATEX¿ token to remove the noise latex markup introduces but keep the information of the latex markup. the use of smaller language models due to GPU limitations may have affected the coherence and fluency of the generated text. Future research should address these limitations by employing larger and more diverse datasets, exploring efficient hyperparameter optimization techniques [**?**], and investigating the use of more powerful language models. Due to the gpu limits we faced we could not attempt distributed training a key part of training any large language model. This would have allowed us to assess more models and bigger models, as well as tune the hyperparameters of them. The findings of this study contribute to our understanding of the challenges in TST and DCG, highlighting the need for further research to develop more effective and robust text generation models. The potential of LLM-based approaches for improving coherence and fluency opens up new avenues for investigation, such as exploring the integration of LLMs with TST models or developing hybrid approaches that combine the strengths of different techniques. The practical applications of improved TST and DCG models are vast, ranging from content creation and text summarization to dialogue systems and creative writing assistance. In conclusion, this study sheds light on the current state of TST and DCG, revealing the challenges and opportunities in these fields. While the evaluated TST models demonstrated poor overall performance, the incorporation of the DCG module and the success of LLM-based approaches highlight the potential for improvement. Future research should focus on developing more sophisticated TST models, exploring the integration of LLMs, and addressing the limitations of current approaches to create more effective and coherent text generation systems.

7 CONCLUSION

In this study, we investigated the performance of various TST models and the effectiveness of DCG in improving the coherence and fluency of stylized text. Our results demonstrate that while TST models struggle to accurately transfer style while preserving content, the incorporation of DCG modules can significantly enhance the quality of the generated text. However, the superior performance of language-model-based DCG approaches compared to the proposed DCG implementation highlights the need for further research to develop more effective DCG techniques.

The challenges encountered in this study, such as the trade-off between style transfer accuracy and content preservation, underscore the complexity of the TST task. Our findings suggest that exploring novel architectures, leveraging larger context windows, and integrating LLMs with TST models may be promising directions for future research. Additionally, the limitations of our study, including the small evaluation size, lack of extensive hyperparameter tuning, and potential biases introduced by data cleaning and model size constraints, should be addressed in future work to improve the generalizability and robustness of the results.

Despite these challenges, our study contributes to the growing body of research on TST and DCG, highlighting the potential of these technologies for various applications, such as content creation, text summarization, and dialogue systems. As the demand for high-quality, stylistically diverse, and coherent text generation continues to grow, the development of more advanced TST and DCG models will become increasingly important.

In conclusion, our research demonstrates the need for further investigation into TST and DCG techniques to address the current limitations and unlock the full potential of these technologies. By exploring novel approaches, leveraging the power of LLMs, and addressing the challenges identified in this study, we can pave the way for more effective, coherent, and stylistically diverse text generation systems that can meet the evolving needs of users and industries alike.

References

[1] A. Haleem, M. Javaid, and R. P. Singh, "An era of chatgpt as a significant futuristic support tool: A study on features, abilities, and challenges," *BenchCouncil Transactions on Benchmarks, Standards and Evaluations*, vol. 2, no. 4, p. 100089, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772485923000066

[2] Z. Hu, R. K.-W. Lee, C. C. Aggarwal, and A. Zhang, "Text style transfer: A review and experimental evaluation," 2023.

[3] D. Jin, Z. Jin, Z. Hu, O. Vechtomova, and R. Mihalcea, "Deep Learning for Text Style Transfer: A Survey," *Computational Linguistics*, vol. 48, no. 1, pp. 155–205, 04 2022. [Online]. Available: https://doi.org/10.1162/coli_a_00426

[4] M. Tran, Y. Zhang, and M. Soleymani, "Towards a friendly online community: An unsupervised style transfer framework for profanity redaction," *arXiv preprint arXiv:2011.00403*, 2020.

[5] S. Rao and J. Tetreault, "Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer," *arXiv preprint arXiv:1803.06535*, 2018.

[6] T. Shen, T. Lei, R. Barzilay, and T. Jaakkola, "Style transfer from non-parallel text by cross-alignment," *Advances in neural information processing systems*, vol. 30, 2017.

[7] X. Zhu, J. Guan, M. Huang, and J. Liu, "Storytrans: Non-parallel story author-style transfer with discourse representations and content enhancing," *arXiv preprint arXiv:2208.13423*, 2022.

[8] L. Born, M. Mesgar, and M. Strube, "Using a graph-based coherence model in document-level machine translation," in *Proceedings of the Third Workshop on Discourse in Machine Translation*, B. Webber, A. Popescu-Belis, and J. Tiedemann, Eds. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 26–35. [Online]. Available: https://aclanthology.org/W17-4803

[9] E. Reif, D. Ippolito, A. Yuan, A. Coenen, C. Callison-Burch, and J. Wei, "A recipe for arbitrary text style transfer with large language models," 2022.

[10] S. Roy, R. Shu, N. Pappas, E. Mansimov, Y. Zhang, S. Mansour, and D. Roth, "Conversation style transfer using few-shot learning," 2023.

[11] L. Wang, C. Lyu, T. Ji, Z. Zhang, D. Yu, S. Shi, and Z. Tu, "Document-level machine translation with large language models," 2023.

[12] A. Madaan, A. Setlur, T. Parekh, B. Poczos, G. Neubig, Y. Yang, R. Salakhutdinov, A. W. Black, and S. Prabhumoye, "Politeness transfer: A tag and generate approach," *arXiv preprint arXiv:2004.14257*, 2020.

[13] S. Prabhumoye, Y. Tsvetkov, R. Salakhutdinov, and A. W. Black, "Style transfer through back-translation," *arXiv preprint arXiv:1804.09000*, 2018.

[14] E. Reif, D. Ippolito, A. Yuan, A. Coenen, C. Callison-Burch, and J. Wei, "A recipe for arbitrary text style transfer with large language models," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 837–848. [Online]. Available: https://aclanthology.org/2022.acl-short.94

[15] J. Li, R. Jia, H. He, and P. Liang, "Delete, retrieve, generate: A simple approach to sentiment and style transfer," 2018.

[16] J. Xu, X. Sun, Q. Zeng, X. Ren, X. Zhang, H. Wang, and W. Li, "Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach," *arXiv preprint arXiv:1805.05181*, 2018.

[17] D. E. Rumelhart and J. L. McClelland, *Learning Internal Representations by Error Propagation*, 1987, pp. 318–362.

[18] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014. [Online]. Available: http://arxiv.org/abs/1409.1259

[19] S. Narasimhan, S. Dey, and M. S. Desarkar, "Towards robust and semantically organised latent representations for unsupervised text style transfer," 2022.

[20] H. Lai, A. Toral, and M. Nissim, "Thank you bart! rewarding pre-trained models improves formality style transfer," 2021.

[21] J. Mueller, D. Gifford, and T. Jaakkola, "Sequence to better sequence: continuous revision of combinatorial structures," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2536–2544.

[22] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International conference on machine learning*. PMLR, 2014, pp. 1278–1286.

[23] A. Romanov, A. Rumshisky, A. Rogers, and D. Donahue, "Adversarial decomposition of text representation," *arXiv preprint arXiv:1808.09042*, 2018.

[24] D. Li, Y. Zhang, Z. Gan, Y. Cheng, C. Brockett, B. Dolan, and M.-T. Sun, "Domain adaptive 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)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3304–3313. [Online]. Available: https://aclanthology.org/D19-1325

[25] X. Yi, Z. Liu, W. Li, and M. Sun, "Text style transfer via learning style instance supported latent space," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 3801–3807.

[26] V. John, L. Mou, H. Bahuleyan, and O. Vechtomova, "Disentangled representation learning for non-parallel text style transfer," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, A. Korhonen, D. Traum, and L. Màrquez, Eds. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 424–434. [Online]. Available: https://aclanthology.org/P19-1041

[27] K. Zhu, Z. Tian, R. Luo, and X. Mao, "Styleflow: Disentangle latent representations via normalizing flow for unsupervised text style transfer," *arXiv preprint arXiv:2212.09670*, 2022.

[28] S. Subramanian, G. Lample, E. M. Smith, L. Denoyer, M. Ranzato, and Y.-L. Boureau, "Multiple-attribute text style transfer," 2019.

[29] H. Gong, S. Bhat, L. Wu, J. Xiong, and W. mei Hwu, "Reinforcement learning based text style transfer without parallel training corpus," 2019.

[30] R. Liu, C. Gao, C. Jia, G. Xu, and S. Vosoughi, "Non-parallel text style transfer with self-parallel supervision," 2022.

[31] M. Reid and V. Zhong, "LEWIS: Levenshtein editing for unsupervised text style transfer," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds. Online: Association for Computational Linguistics, Aug. 2021, pp. 3932–3944. [Online]. Available: https://aclanthology.org/2021.findings-acl.344

[32] Z. Jin, D. Jin, J. Mueller, N. Matthews, and E. Santus, "Imat: Unsupervised text attribute transfer via iterative matching and translation," 2020.

[33] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[34] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[35] M. Abdolahi and M. Zahedi, "An overview on text coherence methods," in *2016 Eighth International Conference on Information and Knowledge Technology (IKT)*, 2016, pp. 1–5.

[36] X. Liang, Z. Tang, J. Li, and M. Zhang, "Open-ended long text generation via masked language modeling," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 223–241. [Online]. Available: https://aclanthology.org/2023.acl-long.13

[37] G. Chen, J. Pu, Y. Xi, and R. Zhang, "Coherent long text generation by contrastive soft prompt," in *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, A. Bosselut, K. Chandu, K. Dhole, V. Gangal, S. Gehrmann, Y. Jernite, J. Novikova, and L. Perez-Beltrachini, Eds. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics, Dec. 2022, pp. 445–455. [Online]. Available: https://aclanthology.org/2022.gem-1.42

[38] A. Lai and J. Tetreault, "Discourse coherence in the wild: A dataset, evaluation and methods," in *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, K. Komatani, D. Litman, K. Yu, A. Papangelis, L. Cavedon, and M. Nakano, Eds. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 214–223. [Online]. Available: https://aclanthology.org/W18-5023

[39] W. Zhao, M. Strube, and S. Eger, "Discoscore: Evaluating text generation with bert and discourse coherence," 2023.

[40] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005, pp. 65–72.

[41] C.-Y. Lin and F. J. Och, "Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics," in *Proceedings of the 42nd annual meeting of the association for computational linguistics (ACL-04)*, 2004, pp. 605–612.

[42] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *International conference on machine learning*. PMLR, 2015, pp. 957–966.

[43] R. Mir, B. Felbo, N. Obradovich, and I. Rahwan, "Evaluating style transfer for text," *arXiv preprint arXiv:1904.02295*, 2019.

[44] Z. Yang, Z. Hu, C. Dyer, E. P. Xing, and T. Berg-Kirkpatrick, "Unsupervised text style transfer using language models as discriminators," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[45] W. Yuan, G. Neubig, and P. Liu, "Bartscore: Evaluating generated text as text generation," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 27 263–27 277. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Paper.pdf

[46] A. Bhat, "Gpt-wiki-intro (revision 0e458f5)," 2023. [Online]. Available: https://huggingface.co/datasets/aadityaubhat/GPT-wiki-intro

[47] M. Grusky, M. Naaman, and Y. Artzi, "Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.

[48] R. Eldan and Y. Li, "Tinystories: How small can language models be and still speak coherent english?" *arXiv preprint arXiv:2305.07759*, 2023.

[49] A. Cohan, F. Dernoncourt, D. S. Kim, T. Bui, S. Kim, W. Chang, and N. Goharian, "A discourse-aware attention model for abstractive summarization of long documents," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018. [Online]. Available: http://dx.doi.org/10.18653/v1/n18-2097

[50] D. Perry. (2020, January 27) Writing styles. Blog post. George Washington University Writing Center. Adapted from "Types of Writing Styles" by Robin Jeffrey. [Online]. Available: https://web.uri.edu/graduate-writing-center/writing-styles/#:~:text=The%20four%20main%20types%20of,of%20these%20four%20writing%20styles

[51] M. Rezapour, "Extracting emotion phrases from tweets using bart," 2024.

[52] Z. Sun, M. Wang, H. Zhou, C. Zhao, S. Huang, J. Chen, and L. Li, "Rethinking document-level neural machine translation," in *Findings of the Association for Computational Linguistics: ACL 2022*, S. Muresan, P. Nakov, and A. Villavicencio, Eds. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3537–3548. [Online]. Available: https://aclanthology.org/2022.findings-acl.279

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[54] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Huggingface's transformers: State-of-the-art natural language processing," 2020.

[55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019.

[56] L. Xu, H. Xie, S.-Z. J. Qin, X. Tao, and F. L. Wang, "Parameter-

efficient fine-tuning methods for pretrained language models: A critical review and assessment," 2023.

APPENDIX

Most common 3-grams for cleaned scientific:
(',', ',', ','): 1155 ('.', 'in', 'this'): 763 ('.', 'in', 'the'): 714 ('we', 'show', 'that'): 701 (',', 'and', 'the'): 667 ('we', 'find', 'that'): 653 ('.', 'it', 'is'): 644 ('.', 'rev', '.'): 642 ('.', 'however', ','): 630 ('phys', '.', 'rev'): 615 Most common 3-grams for scientific:
('_', '_', '_'): 10343 ('@', 'xcite', '.'): 2170 ('@', 'xcite', ','): 1030 (',', ',', ','): 891 ('.', 'in', 'this'): 747 ('we', 'show', 'that'): 701 ('.', 'in', 'the'): 698 ('fig', '.', '['): 666 ('we', 'find', 'that'): 653 ('.', 'rev', '.'): 640