AMR-TST: Abstract Meaning Representation-based Text Style Transfer

Kaize Shi[1], **Xueyao Sun**[1,2], **Li He**[1], **Dingxian Wang**[1,3], **Qing Li**[2], **Guandong Xu**[1]*
[1]University of Technology Sydney
[2]The Hong Kong Polytechnic University
[3]Etsy
{Kaize.Shi, Guandong.Xu}@uts.edu.au

Abstract

Abstract Meaning Representation (AMR) is a semantic representation that can enhance natural language generation (NLG) by providing a logical semantic input. In this paper, we propose the AMR-TST, an AMR-based text style transfer (TST) technique. The AMR-TST converts the source text to an AMR graph and generates the transferred text based on the AMR graph modified by a TST policy named style rewriting. Our method combines both the explainability and diversity of explicit and implicit TST methods. The experiments show that the proposed method achieves state-of-the-art results compared with other baseline models in automatic and human evaluations. The generated transferred text in qualitative evaluation proves the AMR-TST have significant advantages in keeping semantic features and reducing hallucinations. To the best of our knowledge, this work is the first to apply the AMR method focusing on node-level features to the TST task.

1 Introduction

Text style transfer (TST) is an attractive task in natural language processing, which aims to change the specific style by editing while preserving the core content of source texts. TST has been widely applied in tasks such as sentiment transfer, formality transfer, and political transfer (Jin et al., 2022; Shi et al., 2021). The lack of parallel corpus is the main challenge of the current TST tasks, making the methods based on the unsupervised generative structures that distinguish content and style features become the dominant technology. However, the entanglement of content and style features makes it difficult for these methods to balance the diversity and semantic reliability of the transferred text generation (Ramesh Kashyap et al., 2022).

Abstract Meaning Representation (AMR, Banarescu et al. 2013) is a semantic representation

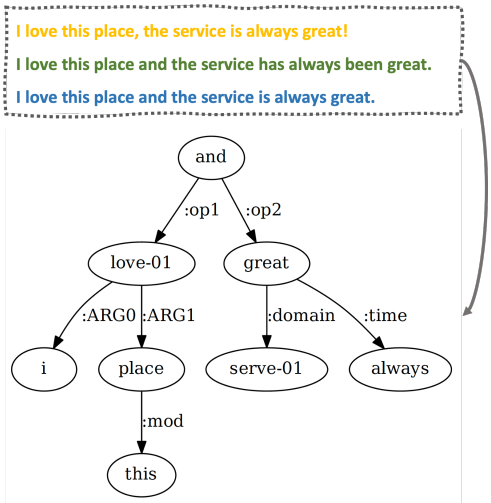

Figure 1: Sentences with the same semantics but different surface syntax can be parsed into the same AMR graph

language, which comprises the whole sentence into a rooted, labelled, directed, acyclic graph. AMR graphs can be represented by PENMAN (Goodman, 2020) symbols, and texts with the same semantic meaning can be abstracted into the same AMR graph, an example is shown in Figure 1. This characteristic allows the model to generate various texts that maintain the same semantics logic based on a constant AMR graph. Compared with other meaning representation methods, AMR allows better maintenance of sentence backbones to describe phenomena such as parameter sharing and allows adding implicit or omitted constituents to recover full sentence semantics (Socher et al., 2013). More importantly, recent research has demonstrated that robust and diverse text generation can be achieved by modifying the nodes of AMR without the complex decoder retraining process (Shou et al., 2022).

This paper proposes the AMR-TST, a novel AMR-based generative text style transfer method. AMR-TST takes the AMR graphs as the intermediate representations and generates the trans-

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2023, pages 4231–4243
July 9-14, 2023

ferred text by the style rewriting algorithm that modifies the detected AMR graphs' stylistic nodes from source to target style. This design overcomes the difficulty in previous TST methods of generating diverse transferred texts combined with target words while maintaining factual consistency of non-stylistic content. It performs well by jointly considering the sentence-level features representing the semantic logical and node-level features representing the stylistic entities, and enables AMR-TST to adaptively embed target style words into the semantic structure of the source text to generate the reasonable and readable transferred text. Meanwhile, the parsing process of the AMR graph can realize the screening of the core content entities with semantic features of the source text, avoiding hallucinations caused by semantically irrelevant content in the transferred text generation process.

The structure of the AMR-TST is shown in Figure 2, which consists of three components: (1) Text to AMR, (2) AMR Style Transfer, and (3) Transferred AMR to Text. The source text is first transduced to the AMR graph by the AMR parser; the AMR style transfer achieves graph modification by rewriting the nodes consisting of style words detected by the style detector; the diverse transferred texts with the target style are generated by the AMR decoder based on the modified graph.

To demonstrate the effectiveness of the proposed AMR-TST, we evaluate it using two public datasets, Yelp and Amazon, which are commonly employed for sentiment transfer tasks - one of the typical application scenarios in TST. All the evaluation results demonstrate that the AMR-TST achieves state-of-the-art results compared with other baseline models. To the best of our knowledge, AMR-TST is the first work to apply AMR to the TST task by rewriting node-level stylistic features.

2 Related Work

Text style transfer aims to revise the specific styles or attributes of the source texts while preserving the non-stylistic content (Hu et al., 2022). Due to the lack of a parallel corpus, implicit and explicit unsupervised methods are the mainstream techniques for this task (Jin et al., 2022). The implicit methods enable the model to map the text to the latent space through the encoder to obtain the disentangled representation, separate the content and attributes, and perform attribute transfer. Hu et al. (Hu et al., 2017) combined VAE with an attribute discriminator to control the attributes of the target sentence through structured encoding and provided feedback to optimize the generated sentence through the attribute discriminator. Luo et al. (Luo et al., 2019) regarded the mapping between the source and target text as a dual learning task and achieved style transfer by setting reward mechanisms of style accuracy and content retention in reinforcement learning.

Considering the fact that the style features of a sentence are usually reflected in unique phrases, explicit methods can achieve explainable text style transfer by only changing the stylistic words or phrases while retaining the style-independent parts. Li et al. (Li et al., 2018) first proposed the DRG framework, which achieves style transfer by deleting style words from texts, retrieving target texts similar to the source content, and generating target texts by combining target style features. Since sentiment words with higher attention weights in sentiment classification, Xu et al. (Xu et al., 2018) used an attention-based classifier to separate content and sentiment words for text style transfer.

Most research on AMR focuses on AMR parsing and generation, such as using graph neural networks (Bai et al., 2022) and pre-training language models (Xu et al., 2021a) to improve performance. It is gratifying to note that more recent research integrates AMR with downstream NLG tasks. T-STAR (Jangra et al., 2022) is a contemporaneous work with us, which transfers the text's style by training style-specific AMR encoder and decoder. In comparison, the AMR-TST achieves the text style transfer with a simple and reliable style rewriting algorithm, avoiding potential semantic bias during the complex retraining process. Kapanipathi et al. (Kapanipathi et al., 2021) introduced the AMR into knowledge base question answering (KBQA) for delegating the complexity of understanding natural language questions to AMR parsers, which relieves the pressure of labelling large amounts of data in KBQA. All this research proves the potential of AMR to power various NLP tasks.

3 Methods

3.1 Text to AMR

Text to AMR is the first component of AMR-TST. Let $x_{s^{src}} = \{x_1, ..., x_n\}$ be the source text with the style of s^{src}, and this component aims to parse the $x_{s^{src}}$ into the corresponding AMR graph $G_{s^{src}}$. Previous text-to-AMR semantic parsing methods

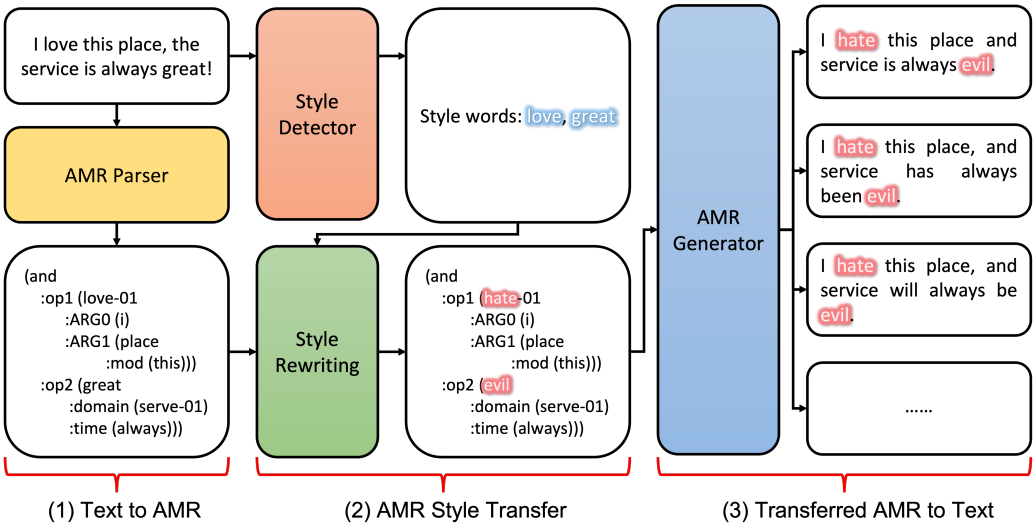

Figure 2: Overview of the proposed AMR-TST pipeline

are fine-grained, content-specific heuristics that require complex pre- and post-processing, making them difficult to apply directly to cross-domain and genre-specific tasks. The pre-trained transformer-based sequence-to-sequence (seq2seq) model powered the AMR parsing tasks by their robust performance in transfer learning (Xu et al., 2020). In this paper, we applied SPRING[1] (Bevilacqua et al., 2021) as the AMR parser, which is a BART-based (Lewis et al., 2020) model that achieves competitive performance in AMR semantic parsing.

SPRING extends its tokenization vocabulary by adding the frequently occurring relations, frames, and constituents of AMR tokens to make BART applicable for processing AMR graphs. The embeddings of the new symbols are included by a vector initialized by the average of word embeddings. Then the produced sequence can be transferred to the PENMAN notations after restoring parenthesis parity and removing the discontinuity token.

Specifically, SPRING first applies a complete graph isomorphic linearization technique to encode an AMR graph as a sequence of symbols via a DFS-based PENMAN annotation without losing adjacency information. The lack of a clear distinction between the constants and variables may confuse the seq2seq models. Since the variable names are without semantics, SPRING proposed a series of special tokens $< R0 >, < R1 >, ..., < Rn >$ to represent the variables in the linearized graph and to handle co-referring nodes. This representation also disposes of the redundant slash token "/". Through this setting, the AMR graph in Figure 1 can be represented as (<R0> and:op1 (<R1> love-01:ARG0 (<R2> i):ARG1 (<R3> place:mod (<R4> this))):op2 (<R5> great:domain (<R6> serve-01):time (<R7> always))). The SPRING here is pre-trained on AMR 3.0 (LDC2020T02) [2].

3.2 AMR Style Transfer

3.2.1 Style Detector

Let $a_{s^{src}} = \{a_1, ..., a_m\} \in x_{s^{src}}$ be the stylistic words in $x_{s^{src}}$, and the style detector aims to detect $a_{s^{src}}$ significantly contributes to s^{src}. Specifically, we applied the RoBERTa (Ott et al., 2019), a Transformer-based model that achieves state-of-the-art results in several text classification tasks as our style classifier. The RoBERTa-based style classification process can be expressed as Eq. 1.

$$p(s|x) = g(v, \alpha) \tag{1}$$

where v is a tensor such that v_i is encoded x_i; α_i is the corresponding attention weight in determining probabilities of each style label s over the whole style label set $S = \{s^{src}, s^{tgt}\}$, which can be understood as an importance score to detect style words. However, the Style Detector has multiple attention heads and layers that encode different semantic and linguistic structures. Inspired by Sudhakar et al. (Sudhakar et al., 2019), we calculate and extract the specific attention head and layer that can significantly encode the style features representing the importance score contributing to the style classification results.

[1] https://github.com/SapienzaNLP/spring

[2] https://catalog.ldc.upenn.edu/LDC2020T02

Let $< h, l >$ be the potential head-layer pair to be iterated to extract the attention score of the i^{th} word $x_i \in x$. The calculation is as Eq. 2.

$$\alpha_{h,l}(x_i) = softmax_{x_i \in x}(Q_{h,l,[CLS]} K^T_{h,l,x_i}) \quad (2)$$

where "$[CLS]$" is a special token added to each sentence's beginning. This symbol, without obvious semantic information, will more fairly integrate the semantic information of each word, thus better representing the semantics of the whole sentence. "Q" and "K" are the query and key vectors defined in Vaswani et al. 2017. Then we define a proportion parameter γ to select the top $\gamma \cdot n$ words representing the style words $a_{s^{src}}$ from x based on the importance score calculated in Eq. 2, where n represents the number of words in x. After that, we select the potential head-layer pair that can better fit the style word features, and the score $z(a_{h,l})$ can be calculated in Eq. 3:

$$z(a_{h,l}) = \frac{p(s|z(a_{h,l})) + \lambda}{\sum_{s'} p(s^{'}|z(a_{h,l})) + \lambda} \quad (3)$$

where s is the style label with the maximum probability assigned by the softmax distribution over S, and $s^{'} = S - \{s\}$; λ represents a smoothing parameter. The final head-layer pair $< h_s, l_s >$ can be selected as Eq. 4:

$$< h_s, l_s >= \arg\max_{h \in H, l \in L} \frac{\sum_{a \in D} z(a_{h,l})}{|D|} \quad (4)$$

where H and L represents the whole head set and layer set separately; D is the validation set.

3.2.2 Style Rewriting

Style rewriting aims to transfer the AMR graph with the source style ($G_{s^{src}}$) to the AMR with the target style ($G_{s^{tgt}}$), which lays the intermediate representation for the decoder to generate sentences in the target style. Shou et al. (Shou et al., 2022) proposed AMR-DA, a novel AMR-based method that shows remarkable performance in the NLP data argumentation task. Inspired by the synonym replacement operation in their research, we transfer the style of $G_{s^{src}}$ by modifying its nodes of style words with antonyms. WordNet (Miller, 1998) is currently the mainstream antonym recognition tool in the English vocabulary database. However, WordNet can only identify antonyms corresponding to a limited number of words. In order to improve the coverage of style words, we propose a style rewriting algorithm based on the idea of query rewriting in information retrieval, as shown in Algorithm 1:

Algorithm 1: Style Rewriting Algorithm

```
def Style_Rewriting(a_{s^src}, c_{s^src}):
for a_{s_i^src} ∈ a_{s^src} do
    a_{s^tgt} ← a_{s^tgt} + WordNet(a_{s_i^src})
if a_{s^tgt} is not None then
    for a_{s_i^tgt} ∈ a_{s^tgt} do
        x_tmp = c_{s^src} + a_{s_i^tgt}
        if RoBERTa(x_tmp) ≠ s^src then
            return a_{s_i^tgt}
else
    for a_{s_i^src} ∈ a_{s^src} do
        a_{s^src} ← a_{s^src} + Fasttext(a_{s_i^src})
    return Style_Rewriting(a_{s^src}, c_{s^src})
```

For the style word $a_{s_i^{src}} \in a_{s^{src}}$, if it is consistent with the node in $G_{s^{src}}$, we transfer the style of the AMR graph through the style rewriting algorithm. In this algorithm, we introduce the Fasttext (Bojanowski et al., 2017) as a style words expander and the previous pre-trained RoBERTa (Ott et al., 2019) as a style gating unit, focusing on the stylistic features of words and sentences separately. The Fasttext is a word vector-based model that solves the out-of-vocabulary (OOV) problem by mining character-level n-gram features. We train the Fasttext model using each dataset's training set and extend the $a_{s^{src}}$ by calculating the corresponding top ten synonyms through the pre-trained Fasttext model. However, sometimes the $a_{s^{tgt}}$ generated based on the expanded $a_{s^{src}}$ will have a "style backtracking" problem, that is, the $a_{s^{tgt}}$ and $a_{s^{src}}$ represent the same style feature. Therefore we introduce RoBERTa to filter expanded style words with the same style features as $a_{s^{src}}$. If the style feature of $a_{s^{tgt}}$ is opposite to that of $a_{s^{src}}$, it will pass through the gate; otherwise, it will be blocked. To adapt to the features of RoBERTa, we embed each $a_{s_i^{tgt}} \in a_{s^{tgt}}$ to a sentence x_{tmp}, which is "$c_{s^{src}}$ + $a_{s_i^{tgt}}$", where $c_{s^{src}}$ represents the non-stylistic content, the words after removing $a_{s^{src}}$ from $x_{s^{src}}$. If x_{tmp} can pass through the gating unit, the algorithm returns the corresponding $a_{s_i^{tgt}}$; otherwise, it executes recursively. This algorithm transfers the style of the AMR graph by rewriting the stylistic nodes with $a_{s^{src}}$ to $a_{s^{tgt}}$ that is maximally opposed to the source style. The style rewriting algorithm constrains the non-stylistic nodes to be consistent

in order to maintain factual consistency. Moreover, this algorithm overcomes the dependence on parallel corpora, enabling the model to use the ubiquitous style opposites in natural language introduced by general or fine-tuned language models to rewrite the stylistic nodes in the AMR graphs. The computational complexity of the proposed style rewriting algorithm is reported in Appendix A.

3.3 Transferred AMR to Text

This component aims to generate the text in the target style from the modified AMR graph. The pre-trained transformer-based models have become the mainstream for this task (Mager et al., 2020; Ribeiro et al., 2021). These transfer learning-based models can adapt to generation tasks without the complex retraining processes. We use the SPRING (Bevilacqua et al., 2021) as the generator, which is the inverse task of AMR parsing. Compared with other generative methods in TST, the AMR-based method allows the model to generate diverse but semantically reasonable texts following the same semantic logic structure in simple ways (Figure 2). More importantly, for text style transfer tasks like sentiment transfer based on the review data with colloquial and non-normalized features, the AMR parsing and generation process can automatically correct the semantic normality of the source text, making the generated content more understandable. We discuss this advantage in Appendix B.

4 Experiments

4.1 Datasets

We conduct the experiments on two datasets that provide human gold standard references, Yelp and Amazon, commonly used in text style transfer tasks. The Yelp dataset includes users' positive and negative reviews of specific businesses, while the Amazon dataset contains reviews with sentiment polarity of the products sold on Amazon. We use the same train-dev-test split as Li et al. 2018, and the statistics of the datasets are shown in Table 1.

Table 1: Dataset statistics

Dataset	Style	Train	Dev	Test
Yelp	Positive	270K	2000	500
	Negative	180K	2000	500
Amazon	Positive	277K	985	500
	Negative	279K	1015	500

4.2 Evaluation Metrics

The widely-agreed goals of the text style transfer tasks are that the transferred text conforms to the target style intensity, preserves the content consistent with the non-stylistic part of the source text, and with natural human writing characteristics (Mir et al., 2019). In accordance with these goals, we applied automatic evaluation metrics commonly used in text style transfer tasks for evaluating the methods from the following aspects:

• **Style Transfer Intensity (Sty.)**: We train the FastText[3] (Joulin et al., 2017) as a style classifier on the training set following the train-dev-test split shown in Table 1. In addition, we use the previously fine-tuned RoBERTa model as an additional style evaluation classifier, which is more sensitive to style features based on the detected words. We use these classifiers to measure the accuracy (AC_f and AC_b) with which the style of generated texts are successfully transferred to the target style.

• **Content Preservation (Cont.)**: We use BLEU (Papineni et al., 2002) score to measure the overlap between the transferred text and the source text or the human-written reference, represented as $BLEU_s$ and $BLEU_r$. Narasimhan et al. 2022 mentioned that the BLEU scores alone are insufficient to measure relevance to the target content. Following their conclusions, we borrow the metric widely used in machine translation and text summarization tasks, ROUGE-L (Lin, 2004), for content preservation evaluation. This metric shows more correlations with human judgment, and RL_s and RL_r represent the ROUGE-L calculated with source and reference text separately.

• **Naturalness (Nat.)**: We fine-tune the OpenAI GPT-2 (Radford et al., 2019), a large pre-trained language model, using the training set following the same train-dev-test split in Table 1. We calculate the perplexity (PPL) of the transferred texts by this fine-tuned language model for evaluating the model in generating natural and fluent text.

• **Geometric Mean (GM)**: Following Yi et al. 2020, we report the geometric mean of AC_f, AC_b, $BLEU_s$, $BLEU_r$, RL_s, RL_r, and $\frac{1}{\ln PPL}$ as an overall evaluation metric.

4.3 Baseline Methods

We compare the proposed AMR-TST with novel state-of-the-art TST methods based on various mainstream techniques: **B-GST** (Sudhakar et al.,

[3] https://fasttext.cc/

2019): explicitly separates content and style features to generate transferred text by inputting non-stylistic content and target style; **G-GST** (Sudhakar et al., 2019): retrieves style words from the target corpus and generates the transferred text based on the retrieved target style words and non-stylistic content; **DAAE** (Shen et al., 2020): augments adversarial auto-encoders with denoising objectives to enable zero-shot text style transfer; **VT-STOWER** (Xu et al., 2021b): based on the VAE structure with pivot words enhancement learning that learns decisive words for a specific style; **EPAAE** (Narasimhan et al., 2022): controls the strength of style transfer by clustering stylistically similar sentences based on latent space produced by a finely adjustable noise component; **RLPrompt** (Deng et al., 2022): a discrete prompt optimization method with reinforcement learning that generates the desired discrete prompts formulated by the parameter-efficient policy network.

5 Results and Analysis

5.1 Automatic Evaluation

The automatic evaluation results of the proposed AMR-TST and other baselines are shown in Table 2. It can be observed that most baselines have difficulty balancing the strength between style transfer and content preservation, because rewriting style words necessarily affects the word overlap between texts, thus creating the contradiction between these two goals. Compared to other baselines, DAAE (Shen et al., 2020) has achieved better results in balancing these two goals. VT-STOWER (Xu et al., 2021b) shows good results in target style accuracy because it is good at learning the keywords that determine the target style. RLPrompt (Deng et al., 2022) performs better in perplexity since the introduction of prompt constrains the randomness of the generation process, thereby enhancing the readability of the generated text.

In compression, the AMR-TST achieves state-of-the-art results in the GM metric that comprehensively evaluates the model from the three aforementioned goals. These results also prove that the text generated by the AMR-TST model can maximize the transfer of the source text to the target style while retaining the core content and constraining the semantic logic to best conforms to the natural language specification. In particular, the AMR-TST model has a significant advantage in the perplexity metric, which is closest to the perplexity calculated from the source text (SRC). This result proves that AMR-TST can constrain the text generation process by relying on the global logical information represented by the graph structure even after the local information represented by the nodes has been modified. This advantage allows the AMR-TST to adapt the transferred local information to the remaining non-stylistic nodes with content information, thus making the generated text comprehensible by maintaining the semantic and factual features of the source text.

5.2 Human Evaluation

We invited ten volunteer annotators with extensive experience in English natural language understanding for human evaluation to evaluate AMR-TST and DAAE (Shen et al., 2020), which shows competitive results in the automated evaluation and the followed qualitative evaluation. Each annotator was asked to anonymously rate the ten randomly selected texts generated by these models from perspectives including style transfer intensity, content preservation, and naturalness. For each item, the annotators need to choose which of the generated texts is better, or neither one can decide.

Table 3 shows human evaluation results, which are the percentage representing which model generates the texts preferred by the annotators. It is evident that the AMR-TST-generated texts attract more preference, proving the AMR-TST transferred text is more in line with the language features that better fit human understanding habits.

5.3 Qualitative Evaluation

The quantitative evaluation results of the transferred text generated by the AMR-TST and baselines are shown in Table 4. The words with style features are marked as different colours, which are the target words of the style detector in our method. The AMR-TST can successfully generate the transferred text that conforms to the target style based on the rewritten graph nodes with the target style and the graph structure representing the semantic logic of the source text. In comparison, due to the lack of semantic control, some baselines are confused in keeping the natural semantic and factual consistency while hitting the target words in the transferred text. More importantly, the transferred texts generated by AMR-TST are most in line with human expression style and semantic norms since they are constrained by the semantic structure. Although we found the transferred target words some-

Table 2: Automatic evaluation results, where SRC and H represent the source text and human reference

Dataset	Model	Sty. ↑		Cont. ↑				Nat. ↓	GM ↑
		AC_f	AC_b	$BLEU_s$	$BLEU_r$	RL_s	RL_r	PPL	
Yelp	SRC	0.16	0.03	-	-	-	-	32.99	-
	H	0.73	0.65	-	-	-	-	92.57	-
	B-GST	0.33	0.26	0.36	0.16	0.53	0.37	537.38	0.29
	G-GST	0.39	0.39	0.35	0.15	0.63	0.32	625.57	0.31
	DAAE	0.56	0.63	**0.41**	0.18	0.58	0.32	208.85	0.37
	VT-STOWER	**0.82**	0.91	0.01	0.01	0.08	0.08	151.69	0.10
	EPAAE	0.59	0.68	0.20	0.09	0.32	0.28	416.39	0.27
	RLPrompt	0.63	0.73	0.10	0.08	0.18	0.22	51.71	0.23
	AMR-TST	0.76	**0.92**	0.40	**0.19**	**0.68**	**0.40**	**43.83**	**0.45**
Amazon	SRC	0.32	0.14	-	-	-	-	39.39	-
	H	0.44	0.33	-	-	-	-	92.58	-
	B-GST	0.45	0.44	0.29	0.16	0.72	0.50	324.61	0.34
	G-GST	0.41	0.36	0.28	0.17	**0.75**	0.54	417.60	0.33
	DAAE	0.57	0.66	0.19	0.11	0.63	**0.58**	59.75	0.35
	VT-STOWER	0.58	0.69	0.14	0.07	0.48	0.45	160.66	0.29
	EPAAE	0.55	0.61	0.16	0.09	0.59	0.54	92.61	0.32
	RLPrompt	0.45	0.43	0.31	0.17	0.55	0.55	58.01	0.36
	AMR-TST	**0.64**	**0.82**	**0.33**	**0.18**	0.41	0.41	**43.71**	**0.39**

Table 3: Human evaluation results

	Model	Sty.	Cont.	Nat.	All
Yelp	DAAE	25.7	37.1	31.4	20.0
	AMR-TST	**62.9**	**62.9**	**68.6**	**57.1**
	None	11.4	0.06	0.03	22.9
Amazon	DAAE	17.2	45.4	31.7	14.3
	AMR-TST	**65.7**	**48.6**	**62.3**	**68.6**
	None	17.1	0.06	0.06	17.1

times slightly blunt when observing other generated samples, the text transferred by the AMR-TST did not have obvious grammatical errors compared with other baselines, which is reliable for applying the TST models in real-world scenarios. Moreover, the text generated by AMR-TST does not require complex post-processing, it can directly process abbreviations or punctuation marks for easy reading.

6 Ablation Study

In this section, we conduct the ablation study to verify the positive impact of the proposed style rewriting algorithm. Specifically, we evaluate the performance of models without the style words expander ("w/o se") and style gating unit ("w/o sg") components, as well as the model that has neither of these components ("w/o sr"). The results of this study are shown in Table 5.

The results show that the proposed AMR-TST achieves the best results in GM compared to the models without the style rewriting components, demonstrating the positive impact of the style rewriting components on the model performance. Specifically, for the AC_f and AC_b metrics, there is a gradual decrease from w/o se, w/o sg, to w/o sr. This phenomenon demonstrates that the style rewriting algorithm and its style words expander and style gating unit components actively promote the style transfer from the source text to the target text. In contrast, among the metrics $BLEU_s$, $BLEU_r$, RL_s, and RL_r that reflect content preservation, the w/o sr model has better results, proving the style rewriting algorithm can promote the model to rewrite the source text to the greatest extent according to the target style, resulting in reduced content consistency between the transferred text and the source text. In addition, the results of w/o sr model on the $BLEU_s$ and RL_s metrics also prove that AMR can better reconstruct source text while standardizing semantic representation. AMR-TST achieves competitive results in the PPL metric, which proves that the target style words generated by the style rewriting algorithm are in line with the natural expression habits of humans.

7 Conclusion

This paper proposes the AMR-TST, an abstract meaning representation-based text style transfer

Table 4: Examples of transferred text generated by the AMR-TST and baselines

Example #1	Yelp (Positive to Negative)	Yelp (Negative to Positive)
SRC	the calzones are awesome and the lunch special they have is perfect .	bad service in these areas and really ruined our visit .
B-GST	all calzones , they have is .	unfortunately service in these areas and really ruined our visit .
G-GST	the calzones amazing are have .	service in these areas and really ruined our visit .
DAAE	the kabobs are awesome but the lunch special they would have it .	great service in the area and enjoy the evening .
VT-STOWER	their lunch special is a terrible value !	great service in these areas and really appreciate our visit .
EPAAE	their lunch special could be a great value about value .	great service in tap and terrific food .
RLPrompt	it's not that we don't want to have a lunch special or a lunch special.	and that's a good thing.
AMR-TST	**The calzone is evil and they have an imperfect lunch special.**	**The good service in this area really blesses our visit.**
Example #2	Amazon (Positive to Negative)	Amazon (Negative to Positive)
SRC	nothing bad to say about this wonderful innovation at all .	i ve come to the conclusion that i ve wasted my money .
B-GST	really bad than nothing to say case about this at all .	sometimes i also come to the conclusion that i changed my .
G-GST	absolutely terrible bad to say about vation this at all	unfortunately i been to the conclusion that i ve ruined my money .
DAAE	nothing bad bad why i heard about the taste .	i ve come on the manual that i ve wasted my money .
VT-STOWER	nothing bad to say about this product of problems .	i have used any coffee and i have this for my money .
EPAAE	nothing bad to say about this wonderful game all there .	i ve come to the conclusion that i ve ever wasted my money .
RLPrompt	it's not a new innovation.	i ve got a chance to make a living on a small scale and i am now a millionaire.
AMR-TST	**There is nothing good to be said at all for this unattractive innovation.**	**I've concluded that I'm conserving my money.**

Table 5: Automatic evaluation results of ablation study for verifying the effectiveness of the components in the style rewriting algorithm

Dataset	Model	Sty. ↑		Cont. ↑				Nat. ↓	GM ↑
		AC_f	AC_b	$BLEU_s$	$BLEU_r$	RL_s	RL_r	PPL	
Yelp	w/o se	0.52	0.88	0.49	0.21	0.68	**0.41**	48.76	0.44
	w/o sg	0.53	0.85	0.44	0.20	0.61	0.40	57.98	0.42
	w/o sr	0.31	0.80	**0.57**	**0.22**	**0.77**	0.39	**42.61**	0.43
	AMR-TST	**0.76**	**0.92**	0.40	0.19	0.68	0.40	43.83	**0.45**
Amazon	w/o se	0.47	0.72	0.38	0.20	0.47	**0.42**	59.87	0.38
	w/o sg	0.45	0.71	0.37	0.18	0.47	0.41	54.92	0.37
	w/o sr	0.38	0.68	**0.43**	**0.22**	**0.48**	**0.42**	52.47	0.38
	AMR-TST	**0.64**	**0.82**	0.33	0.18	0.41	0.41	**43.71**	**0.39**

method. AMR-TST transduces source text to AMR graphs, detects and rewrites the stylistic nodes by the style rewriting algorithm, and generates texts in the target style with transferred AMR graphs.

This design overcomes the difficulty of maintaining factual consistency in non-stylistic content when generating texts with target words. Compared with other baselines, the AMR-TST achieves state-of-the-art results in TST benchmarks.

Future research can expand the scope of our method to other text style transfer tasks by designing more flexible style rewriting components. For instance, adding or removing specific stylistic nodes or structures in the AMR graph can result in a more diverse and controllable text style transfer.

Limitations

The current AMR-TST is based on the style rewriting algorithm to rewrite the stylistic nodes of AMR graphs from source style to target style. However, this method relies on style opposites features contained in the general natural language corpus. The advantage of such a method is that it does not need complex decoder retraining processes for different datasets, which maximizes the use of generic natural language knowledge and reduces training costs. However, this also leads to a limitation that the current AMR-TST is applicable to text style transfer tasks with significant style polarity, such as sentiment features. For other text style transfer tasks like political and gender transfer, our current style rewriting algorithm cannot precisely rewrite the implicit style words in these tasks. To address this limitation, our future work will improve the style rewriting algorithm by finely identifying implicit style words and exploring their correlations, enabling the revised algorithm can be embedded in the current AMR-TST framework that focuses on the node-level stylistic features.

Ethical Statement

This paper honors the ethical code set out in the ACL Code of Ethics.

Acknowledgment

This work is partially supported by Australian Research Council under Grant No. DP220103717, DP200101374, and LE220100078 and National Science Foundation of China under Grant No. 62072257.

References

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12564–12573.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning.

Michael Wayne Goodman. 2020. Penman: An open-source library and tool for AMR graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 312–319, Online. Association for Computational Linguistics.

Zhiqiang Hu, Roy Ka-Wei Lee, Charu C. Aggarwal, and Aston Zhang. 2022. Text style transfer: A review and experimental evaluation. *SIGKDD Explor. Newsl.*, 24(1):14–45.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1587–1596. JMLR.org.

Anubhav Jangra, Preksha Nema, and Aravindan Raghuveer. 2022. T-star: Truthful style transfer using amr graph as intermediate representation. *arXiv preprint arXiv:2212.01667*.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep Learning for Text Style Transfer: A Survey. *Computational Linguistics*, 48(1):155–205.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Gangi Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G P Shrivatsa Bhargav, and Mo Yu. 2021. Leveraging Abstract Meaning Representation for knowledge base question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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*, pages 7871–7880, Online. Association for Computational Linguistics.

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

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Xu Sun, and Zhifang Sui. 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*, pages 5116–5122. International Joint Conferences on Artificial Intelligence Organization.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online. Association for Computational Linguistics.

George A Miller. 1998. *WordNet: An electronic lexical database*, volume 38. Communications of the ACM.

Remi Mir, Bjarke Felbo, Nick Obradovich, and Iyad Rahwan. 2019. Evaluating style transfer for text. 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)*, pages 495–504, Minneapolis, Minnesota. Association for Computational Linguistics.

Sharan Narasimhan, Suvodip Dey, and Maunendra Desarkar. 2022. Towards robust and semantically organised latent representations for unsupervised text style transfer. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 456–474, Seattle, United States. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Abhinav Ramesh Kashyap, Devamanyu Hazarika, Min-Yen Kan, Roger Zimmermann, and Soujanya Poria. 2022. So different yet so alike! constrained unsupervised text style transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 416–431, Dublin, Ireland. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Tianxiao Shen, Jonas Mueller, Regina Barzilay, and Tommi Jaakkola. 2020. Educating text autoencoders: Latent representation guidance via denoising. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Kaize Shi, Yusen Wang, Hao Lu, Yifan Zhu, and Zhendong Niu. 2021. Ekgtf: A knowledge-enhanced model for optimizing social network-based meteorological briefings. *Information Processing & Management*, 58(4):102564.

Ziyi Shou, Yuxin Jiang, and Fangzhen Lin. 2022. AMR-DA: Data augmentation by Abstract Meaning Representation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3082–3098,

Dublin, Ireland. Association for Computational Linguistics.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 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)*, pages 3269–3279.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2020. Improving AMR parsing with sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2501–2511, Online. Association for Computational Linguistics.

Dongqin Xu, Junhui Li, Muhua Zhu, Min Zhang, and Guodong Zhou. 2021a. XLPT-AMR: Cross-lingual pre-training via multi-task learning for zero-shot AMR parsing and text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 896–907, Online. Association for Computational Linguistics.

Haoran Xu, Sixing Lu, Zhongkai Sun, Chengyuan Ma, and Chenlei Guo. 2021b. VAE based text style transfer with pivot words enhancement learning. In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pages 162–172, National Institute of Technology Silchar, Silchar, India. NLP Association of India (NLPAI).

Jingjing Xu, Xu Sun, Qi Zeng, Xiaodong Zhang, Xuancheng Ren, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 979–988, Melbourne, Australia. Association for Computational Linguistics.

Xiaoyuan Yi, Zhenghao Liu, Wenhao Li, and Maosong Sun. 2020. Text style transfer via learning style instance supported latent space. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3801–3807. International Joint Conferences on Artificial Intelligence Organization. Main track.

A Computational Complexity

The proposed style rewriting algorithm utilizes a recursive structure, and on average, the number of recursions observed in the experiments was 2.78 times. The algorithm was implemented on a server equipped with an Intel Xeon E-2288G CPU and NVIDIA RTX 6000, and the average time required to transfer one source text was 0.92 seconds.

B Semantic Normality Correction

This appendix demonstrates the advantages of AMR-TST in checking and correcting the semantic normality of source text. The examples are in Table 6. It is shown that when the source text contains irregular symbol representations such as "...", the transferred text generated by AMR-TST can automatically remove these symbols (#1 and #2). Moreover, the AMR-TST can also understand and convert some symbols of the source text into words (#3: e.g. "&" → "and"). When the source texts contain some colloquial superlatives, AMR-TST can understand their semantics and paraphrase them in a normalized form (#4 and #5). All these advantages improve the transferred text's readability, making it more intuitive and easier to understand.

Table 6: Examples of the AMR-TST for normalizing semantic representation

ID	SRC	AMR-TST
#1	i love the food... however service here is horrible.	I hate food, but the service here has been nice.
#2	the food is good with very generous portions of everything.....	Evil food and a very stingy portion of everything.
#3	egg drop soup & spring rolls are excellent.	The egg drop soup and spring rolls are terrible.
#4	what a mess! i m so, so, so upset with this cleaner.	How tidiness! How lovable is this cleanser?
#5	so so good... the food is of great quality and prices are reasonable.	How awful, the food is bad quality and the price is unreasonable..

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section "Limitations"

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 3 and 4

☑ B1. Did you cite the creators of artifacts you used?
Section 3 and 4

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Not applicable. Left blank.

C ☑ Did you run computational experiments?

Appendix A

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Not applicable. Left blank.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not applicable. Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4.2

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Section 5.2

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Section 5.2

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Section 5.2

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. The annotators are only there for human evaluation, not to annotate the dataset.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. The annotators are only there for human evaluation, not to annotate the dataset.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. The annotators are only there for human evaluation, not to annotate the dataset.