

# Stylized Data-to-Text Generation: A Case Study in the E-Commerce Domain

LIQIANG JING, Shandong University, China  
 XUEMENG SONG\*, Shandong University, China  
 XUMING LIN, Alibaba Group, China  
 ZHONGZHOU ZHAO, Alibaba Group, China  
 WEI ZHOU, Alibaba Group, China  
 LIQIANG NIE, Harbin Institute of Technology (Shenzhen), China

Existing data-to-text generation efforts mainly focus on generating a coherent text from non-linguistic input data, such as tables and attribute-value pairs, but overlook that different application scenarios may require texts of different styles. Inspired by this, we define a new task, namely stylized data-to-text generation, whose aim is to generate coherent text for the given non-linguistic data according to a specific style. This task is non-trivial, due to three challenges: the logic of the generated text, unstructured style reference, and biased training samples. To address these challenges, we propose a novel stylized data-to-text generation model, named StyleD2T, comprising three components: logic planning-enhanced data embedding, mask-based style embedding, and unbiased stylized text generation. In the first component, we introduce a graph-guided logic planner for attribute organization to ensure the logic of generated text. In the second component, we devise feature-level mask-based style embedding to extract the essential style signal from the given unstructured style reference. In the last one, pseudo triplet augmentation is utilized to achieve unbiased text generation, and a multi-condition based confidence assignment function is designed to ensure the quality of pseudo samples. Extensive experiments on a newly collected dataset from Taobao<sup>1</sup> have been conducted, and the results show the superiority of our model over existing methods.

CCS Concepts: • **Information systems** → **Business intelligence**; **Online advertising**; • **Computing methodologies** → **Natural language generation**.

Additional Key Words and Phrases: Stylized Data-to-text Generation, Logical Text Generation, E-commerce, Advertising

\*Corresponding author.

<sup>1</sup><https://www.taobao.com>.

Authors' addresses: Liqiang Jing, [jingliqiang6@gmail.com](mailto:jingliqiang6@gmail.com), Shandong University, No. 72 Binhai Road, Jimo, Qingdao, Shandong Province, 266237, China; Xuemeng Song, [sxmusc@gmail.com](mailto:sxmustc@gmail.com), Shandong University, No. 72 Binhai Road, Jimo, Qingdao, Shandong Province, 266237, China; Xuming Lin, [xuming.lxm@alibaba-inc.com](mailto:xuming.lxm@alibaba-inc.com), Alibaba Group, No. 969 Wenyi West Road, Yuhang, Hangzhou, Zhejiang Province, 311121, China; Zhongzhou Zhao, [zhongzhou.zhaozz@alibaba-inc.com](mailto:zhongzhou.zhaozz@alibaba-inc.com), Alibaba Group, No. 969 Wenyi West Road, Yuhang, Hangzhou, Zhejiang Province, 311121, China; Wei Zhou, [fayi.zw@alibaba-inc.com](mailto:fayi.zw@alibaba-inc.com), Alibaba Group, No. 969 Wenyi West Road, Yuhang, Hangzhou, Zhejiang Province, 311121, China; Liqiang Nie, [nieliqiang@gmail.com](mailto:nieliqiang@gmail.com), Harbin Institute of Technology (Shenzhen), Nanshan, Shenzhen, Guangdong Province, 518055, China.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

© 2022 Association for Computing Machinery.

0004-5411/2022/8-ART111 \$15.00

<https://doi.org/XXXXXXX.XXXXXXX>

**ACM Reference Format:**

Liqiang Jing, Xuemeng Song, Xuming Lin, Zhongzhou Zhao, Wei Zhou, and Liqiang Nie. 2022. Stylized Data-to-Text Generation: A Case Study in the E-Commerce Domain. *J. ACM* 37, 4, Article 111 (August 2022), 24 pages. <https://doi.org/XXXXXXX.XXXXXXX>

**1 INTRODUCTION**

Data-to-text generation, which benefits many real-world applications, aims to convert the nonlinguistic data [13], such as knowledge triples, tables, syntax trees, and attribute-value pairs, to the coherent natural language text. One promising application of data-to-text generation is the automatic advertising article generation for products in e-commerce. Apparently, the application can not only be used to raise e-commerce platforms' sales but also reduce the labor costs of hiring professionals to manually write advertising articles for a large number of products.

Due to its great practical value, data-to-text generation has attracted increasing attention from academia and industry. Early data-to-text generation models have mainly adopted the pipeline architecture with three key modules: content planning [10], sentence planning [7], and surface realization [28]. With advancements in deep learning, recent studies have utilized deep neural networks and solved tasks in an end-to-end manner [40, 51]. For example, Wiseman *et al.* [47] devised a hidden semi-markov model-based neural generation system, where several templates are introduced to enhance the model interpretability. To promote the diversity of the generated text, Shao *et al.* [40] designed a planning-based hierarchical variational model with the gated recurrent units (GRUs) [5] to capture inter-sentence coherence and generate diversified texts.

Despite the compelling success achieved by previous efforts, these studies have mainly focused on general data-to-text generation. In other words, the linguistic quality of the generated text was emphasized as well as its semantic consistency with the given data. However, the fact that in different application scenarios, texts of different styles may be needed was overlooked. As shown in Figure 1, given the product data, *i.e.*, a set of attribute-value pairs, formal advertising text is needed for presenting on a formal e-commerce platform, such as Weitaos<sup>2</sup>, but informal advertising text is desired for colloquially broadcasting in a live-streaming e-commerce scenario.

Motivated by this, we define a new task: stylized data-to-text generation, in which the goal is to generate coherent text for given nonlinguistic data according to a specific style. In a sense, the target text should maintain semantic consistency with the given data and style consistency with the specific style. As a pioneer study, we focus on nonlinguistic data in the form of attribute-value pairs, as shown in Figure 1. To represent the desired style, instead of using fixed or learnable style embeddings, we employ a reference text of the desired style as the anchor to deliver the style information, because the linguistic style has been reported as a highly complex concept encompassing various intricate linguistic features [50].

In fact, our defined stylized data-to-text task faces three main challenges.

- **C1: The Logic of the Generated Text.** A coherent text for the given data in terms of attribute-value pairs, as shown in Figure 1, should maintain a logical flow, *i.e.*, the attributes of the data should be described in a certain logical order, making it natural and easy to interpret. Consequently, guaranteeing the logic of the generated text is a key challenge.
- **C2: Unstructured Style Reference.** The given style reference text is unstructured and expresses both the semantic and style information. Accordingly, how to accurately derive the style information without the semantic content from the unstructured style reference text is another crucial challenge.

<sup>2</sup><https://tinyurl.com/4h9fznkn>.

<b>Data</b>	功效:去除黑头; 卖点:轻轻按压; 起泡程度:泡沫; 成分原料:氨基酸; 卖点:自带打泡器; 功效:预防白头; 主题:赫丽尔斯洗面奶。 efficacy: blackheads removal; selling point: gently press; foaming degree: foam; ingredients: amino acids; selling point: its own bubbler; efficacy: whiteheads prevention; title: HELIUS facial cleanser.
<b>Formal Text</b>	<b>赫丽尔斯洗面奶, 自带打泡器</b> 设计, 在洁面时, 只需要 <b>轻轻按压</b> , 就可以产生丰富的 <b>泡沫</b> , 无须再购买打泡器。 <b>氨基酸</b> 活性配方, 长期使用可以 <b>去除黑头</b> 并能 <b>预防白头</b> 。 <b>HELIUS facial cleanser</b> is designed with <b>its own foam beater</b> . When cleaning our face, we only need to <b>gently press</b> it to produce rich <b>foam</b> . At the same time, the facial cleanser adopts the active formula of <b>amino acids</b> , which can <b>remove blackheads</b> and <b>prevent white head comedone</b> after long-term use.
<b>Informal Text</b>	我们家的这款 <b>赫丽尔斯洗面奶</b> , 非常非常的好用, 它本身呢, <b>自带打泡器</b> 。我们在洗脸的时候呢, 只需要呢, <b>轻轻按压</b> , 就能够起很多的 <b>泡沫</b> 。同时呢, 我们家的这个宝贝它还富含 <b>氨基酸</b> , 能够起到 <b>去除黑头</b> 和 <b>预防白头</b> 的作用, 我自己在用的时候有被这款产品圈粉到。 Our <b>HELIUS facial cleanser</b> has a <b>bubble device</b> , very excellent. When wash the face, we produce lots of <b>bubbles</b> by <b>pressing gently</b> . At the same time, our product is also rich in <b>amino acids</b> , it can <b>remove blackheads</b> and <b>prevent white head comedone</b> . When I used this product, I was completely surprised by its efficacy.

Fig. 1. Illustration of stylized data-to-text generation in e-commerce, where both formal and informal advertising texts are desired. The mentioned attribute-value pairs of the given data in each text are highlighted in bold. The English texts are translated from Chinese texts, where some colloquial terms are hard to translate.

- **C3: Biased Training Samples.** It is unrealistic to simultaneously obtain the corresponding text for given data for each style (*e.g.*, formal and informal styles). Thus, optimizing the model with biased training samples may lead to the biased generation of text, *i.e.*, the stylized text generator may be misled and generate the target text based only on the given data without considering the style information. Therefore, how to achieve unbiased stylized text generation is a vital challenge.

To tackle these challenges, we propose a novel stylized data-to-text generation model for product advertising generation, StyleD2T. As shown in Figure 2, StyleD2T consists of three components, namely, logic planning-enhanced data embedding, mask-based style embedding, and unbiased stylized text generation. In particular, for **C1**, we design a graph-guided logic planner in the first component, where a logic graph is introduced to characterize the logical relation among attribute-value pairs of the given data of the product and plan the attribute organization for generating the target text with the GRU. We also introduce the data transformer encoder to obtain the context embedding of the planned attribute. Based on the concern that some words may simultaneously contain style and content information, to tackle **C2**, we propose the feature-level mask-based style embedding component, as opposed to adopting high-level or low-level masks as in existing methods. Regarding **C3**, we introduce pseudo triplet augmentation in the third component, whereby a multi-condition-based confidence assignment function is devised to ensure the quality of pseudo samples. To verify the effectiveness and generality of our model, we collect a real-world dataset named TaoStyle, from Taobao, a very large Chinese online shopping site. This dataset supports stylized data-to-text

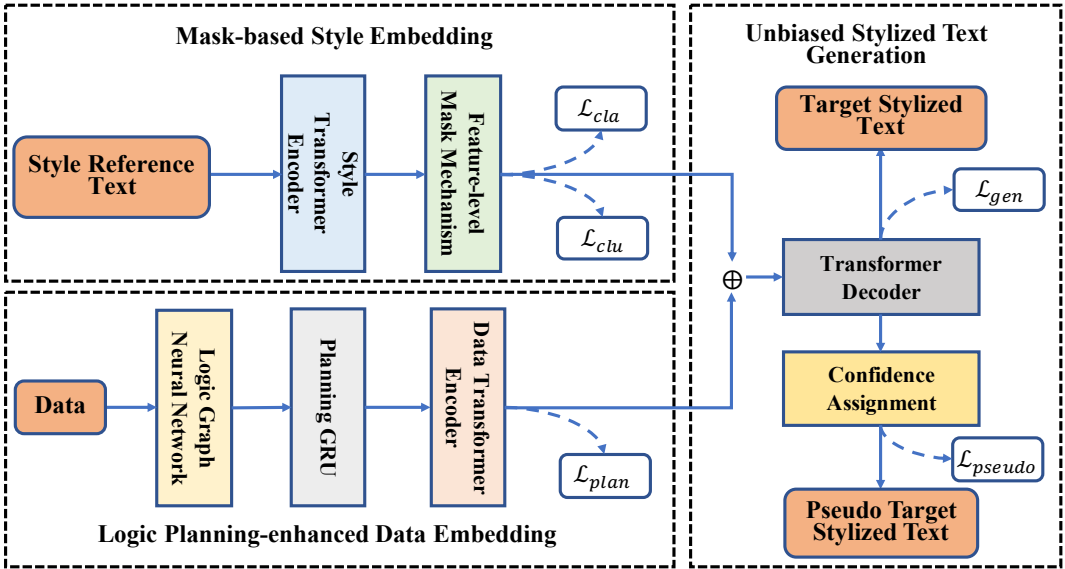


Fig. 2. Illustration of the proposed scheme StyleD2T, which consists of three components: logic planning-enhanced data embedding, mask-based style embedding, and unbiased stylized text generation. The concatenation operation is denoted as " $\oplus$ ".

generation for the informal and formal styles. Our model has yielded superior performance compared with state-of-the-art methods on this real-world dataset.

Our contributions can be summarized as follows:

- We define a new research task, stylized data-to-text generation, which is motivated by realistic application scenarios that indeed require different writing styles. This new task emphasizes that the generated text should be not only coherent and semantically consistent with the given data but also written in a manner that conforms to the desired style.
- We propose a novel stylized data-to-text generation framework comprising three key components: logic planning-enhanced data embedding, mask-based style embedding, and unbiased stylized text generation, which can effectively fulfill the proposed task.
- To verify our StyleD2T model, we construct a real-world dataset from the popular Chinese e-commerce platform Taobao. Extensive experiments on this dataset show the effectiveness of our model. In addition, the codes have been released as a byproduct<sup>3</sup>.

The remainder of this article is as follows. We first briefly review the related work in Section 2. Next, we formulate the proposed task and detail our StyleD2T in Section 3. Then, the experimental setup and the analysis of the results are presented in Section 4. Finally, the conclusion and future research directions are described in Section 5.

## 2 RELATED WORK

Our work is related to the studies of data-to-text generation, stylized generation, graph neural networks, and transformer-based models for neural language processing.

<sup>3</sup><https://github.com/LiqiangJing/StyleD2T>

## 2.1 Data-to-Text Generation

Early data-to-text generation methods [19, 27] mainly adopted the pipeline architecture with separate components, such as content planning, sentence planning, and surface realization. Content planning aims to determine what information should appear in the generated text and the structure of the text. For example, Duboue *et al.* [10] proposed a method to automatically acquire the content selection rules from a corpus of text and its associated semantics. Sentence planning is used to determine the structure and lexical content of each sentence in the generated text. For example, Dalians *et al.* [7] presented eight aggregation strategies to avoid repetition of the selected material during text generation. The aim of surface realization is to convert the sentence plan to a natural language string, which can be classified into template-based and grammar-based approaches. Template-based approaches [28, 44] utilize templates defined by knowledge experts to convert the sentence plan to the text. In contrast, in grammar-based approaches [1, 11], a grammatical system to construct surface strings is usually built.

Recently, with the evolution of deep neural networks, several end-to-end data-to-text generation models have emerged. For example, Jain *et al.* [15] proposed an end-to-end mixed hierarchical attention-based encoder-decoder model to generate summaries of table data, while Nema *et al.* [31] introduced an end-to-end fused bifocal attention model with a gated orthogonalization mechanism for exploiting task-specific characteristics. To enhance the content fidelity of the generated texts with the input attribute-based data, Yuan *et al.* [51] devised a dual-copy mechanism, which copies tokens from the textual product description and product attributes to the generated text. To deal with a large vocabulary, Lebre *et al.* [21] extended the conditional neural language model to mix a fixed vocabulary with the copy mechanism so that sample-specific words can be transferred from the input database to the generated sentence. Moreover, to promote the logic of the generated text, Sha *et al.* [39] devised a table field linking mechanism to capture the relationship between different fields in the table and used this relationship to enable the model to better plan what to output first and what to output next. Noticed the practical demand of diversified expression, Shao *et al.* [40] proposed an end-to-end planning-based hierarchical variational approach to diversify the content plan and hence generate diversified texts. In addition, Ye *et al.* [49] proposed a variational template machine, which utilizes diverse templates to generate diverse texts.

Although existing methods have achieved great success, their focus was on general data-to-text generation. The fact that generated text should be in accordance with certain styles in many application scenarios was overlooked, which is the major concern of this work.

## 2.2 Stylized Generation

Stylized generation refers to the generation of text with a specified style. Because stylized generation can be applied to various application scenarios, it has been explored in multiple research tasks. For example, some studies have explored the classic stylized generation task called text style transfer, in which the aim is to generate new text by changing the style of the original text while keeping its content unchanged. Specifically, existing studies of text style transfer can be roughly divided into two categories: disentanglement approaches and attention-based approaches. In the former approaches, the content and style from the original text are explicitly disentangled and then separated content and style representations are incorporated to generate the target text [16, 41]. In the latter approaches, a trainable or fixed style embedding is utilized as a style signal and the model is encouraged to focus on style-independent words [6, 14]. With respect to dialogue systems, Zhou *et al.* [56] proposed an emotional chatting machine that can generate responses with a specified emotion. Later, Kong *et al.* [20] introduced the task of stylized story generation, in which the goal is to generate a coherent story with a specified style given the first sentence as the leading context in story generation.

Inspired by these efforts, we define a new task of stylized data-to-text generation, which differs from conventional data-to-text generation by requiring the generated text to have a specific style. This task can be applied to more scenes compared to general data-to-text generation.

### 2.3 Graph Convolutional Networks

Recently, studies on graph convolutional networks (GCNs) have received increasing attention because of their superior ability to model structured data, *i.e.*, graphs [33, 43, 48, 54]. The original GCN algorithm proposed by Kipf *et al.* [18] learns hidden representations of nodes by aggregating neighbor information. Due to the great power of the GCN, it has been widely applied in multiple fields, including natural language processing (NLP), computer vision, and recommendation systems. For example, in the NLP domain, Chen *et al.* [4] devised a fine-grained privacy detection network that explored the semantic correlations among personal aspects with a GCN. In addition, in the computer vision domain, Caramalau *et al.* [2] presented a novel sequential GCN to learn node representations and distinguish sufficiently different unlabeled examples from labeled examples for active learning, and Zhang *et al.* [53] devised a multimodal interaction GCN to jointly explore the complex intra-modal relations and inter-modal interactions for temporal language localization in videos. Zheng *et al.* [55] integrated disentangled item representations into a GCN to adaptively propagate the fine-grained compatibility relationships among items for outfit compatibility modeling, Wang *et al.* [46] developed a novel neural graph collaborative filtering method that integrated user-item interactions into a user embedding process for recommendation systems.

According to the abovementioned studies, the GCN has demonstrated its superiority in learning the structure information among nodes in the graph, which inspires us to utilize the GCN for logic correlation modeling among attribute-value pairs of the given input.

### 2.4 Transformer-based Models for Natural Language Processing

The transformer [45] model, which consists of an encoder and a decoder, was initially proposed for machine translation. The encoder learns interactions between the left and right context words with the bidirectional attention mechanism, while the decoder is designed to generate the target translation text using the unidirectional attention mechanism. Due to the remarkable capability of attention mechanisms, many transformer-based models have emerged in the NLP domain. According to the architecture, current transformer-based models for NLP can be roughly classified into three groups: transformer encoder-based models, transformer decoder-based models, and full transformer-based models [34]. The typical transformer encoder-based models are BERT [9] and RoBERTa [26], both of which adopt the transformer encoder as the backbone and are trained by the masked language modeling technique. Regarding the transformer decoder-based model, a classic example is GPT [35], which is developed based on the transformer decoder and achieves comparable performance with the above two models (*i.e.*, BERT and RoBERTa) on many natural language understanding tasks. Regarding the full transformer-based model, BART [22] and T5 [36] are two representative models that were proposed for tackling natural language generation tasks.

To date, these transformer-based models have achieved compelling success in various NLP tasks, such as movie script generation [57], personalized answer generation [8], and query suggestion [30]. Inspired by their remarkable performance, we utilize the transformer encoder and decoder for our text representation and generation, respectively.

Table 1. Summary of the key notations.

Symbol	Explanation
$\mathcal{D}$	The set of training triplets.
$\mathcal{P}_j$	The input data ( <i>i.e.</i> , a set of attribute-value pairs) of the $j$ -th triplet.
$p_j^k$	The $k$ -th attribute-value pair of the input data $\mathcal{P}_j$ .
$X_j$	The style reference text of the $j$ -th triplet.
$Y_j$	The target text of the $j$ -th triplet.
$\hat{Y}_j$	The generated target text of the $j$ -th triplet.
$N$	The total number of training triplets.
$N_s$	The total number of styles.
$\mathbf{g}_j$	The ground-truth one-hot style label vector of the $j$ -th triplet.
$\hat{\mathbf{g}}_j$	The predicted style vector of the $j$ -th triplet.
$\mathbf{E}^k$	The token embedding matrix of the $k$ -th attribute-value pair.
$C$	The corpus containing all the target stylized texts in the training dataset.
$r_i^Y$	The ground-truth ranking of the attribute-value pair $p^i$ in the text $Y$ .
$L$	The ground-truth attribute-value pair organization plan.
$\hat{L}$	The predicted attribute-value pair organization plan.
$H_o$	The planning-enhanced representation for the input data.
$E_X$	The embedding of the style reference text $X$ .
$H_X$	The encoded representation of the style reference text $X$ .
$M$	The feature-level mask matrix.
$s$	The style embedding derived from the style reference text $X$ .
$\alpha, \beta, \gamma, \delta$	The hyperparameters for loss functions.

### 3 METHOD

In this section, the problem formulation is first introduced, and then the three components of our proposed StyleD2T model are detailed. Finally, the training and inference processes are presented.

#### 3.1 Problem Formulation

Suppose that we have a set of  $N$  training triplets  $\mathcal{D} = \{(\mathcal{P}_1, X_1, Y_1), (\mathcal{P}_2, X_2, Y_2), \dots, (\mathcal{P}_N, X_N, Y_N)\}$ .  $\mathcal{P}_j = \{p_j^1, p_j^2, \dots, p_j^{K_j}\}$ ,  $j = 1, \dots, N$ , is the input data of the  $j$ -th triplet, which is represented by a set of  $K_j$  attribute-value pairs, where  $p_j^k$  is the  $k$ -th attribute-value pair,  $k = 1, 2, \dots, K_j$ .  $p_j^k = \{w_j^{k,1}, w_j^{k,2}, \dots, w_j^{k,m}\}$  and  $w_j^{k,i}$  is the  $i$ -th token in  $p_j^k$ .  $X_j$  is the style reference text of the  $j$ -th triplet, which shares the same style as the target stylized text  $Y_j$ . In particular,  $N_s$  refers to the number of possible styles that we can specify. To facilitate style embedding learning, we assign each  $X_j$  a one-hot label vector  $\mathbf{g}_j = [g_j^1, g_j^2, \dots, g_j^{N_s}] \in \{0, 1\}^{N_s}$ , where  $g_j^q = 1$  if  $X_j$  belongs to the  $q$ -th style; otherwise,  $g_j^q = 0$ .  $Y_j$  is the target text of the  $j$ -th triplet, which meets the content fidelity with the given data  $\mathcal{P}_j$  and the style consistency with the style reference text  $X_j$ . Formally, we use  $C = \{Y_1, Y_2, \dots, Y_N\}$  to denote the whole corpus. Table 1 summarizes the key notations. Ultimately, our goal is to learn a model  $\mathcal{M}$  that is able to take  $(\mathcal{P}, X)$  as the input and generate the target stylized text  $\hat{Y}$  as follows:

$$\hat{Y} = \mathcal{M}(\mathcal{P}, X | \Theta), \quad (1)$$

where  $\Theta$  refers to the parameters to be learned. For simplicity, we temporarily omit the index (*i.e.*, the subscript  $j$ ) of each training triplet.

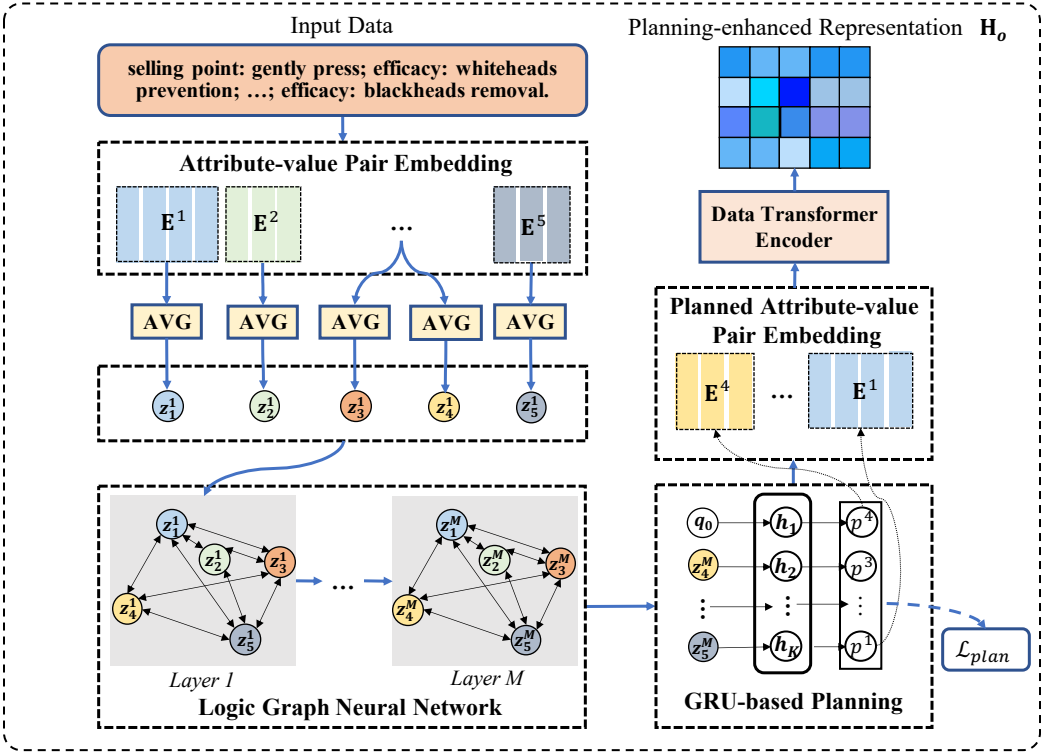


Fig. 3. Illustration of the logic planning-enhanced data embedding component, where  $K = 5$ .

### 3.2 Logic Planning-enhanced Data Embedding

As shown in Figure 3, the logic planning-enhanced data embedding component works on the data embedding, *i.e.*, the attribute-value pair embedding, and organizes the attribute-value pairs in the target text.

**Attribute-value Pair Embedding.** Suppose that the input data  $\mathcal{P} = \{p^1, p^2, \dots, p^K\}$  is represented by  $K$  attribute-value pairs. We first embed each attribute-value pair with the embedding layer of the pretrained language model BART [22] as,

$$E^k = \text{BART\_Embedding}(p^k), \quad (2)$$

where  $E^k \in \mathbb{R}^{t_k \times d_1}$  refers to the token embedding matrix of the  $k$ -th attribute-value pair of the data  $\mathcal{P}$ , whose  $t$ -th row refers to the embedding of the  $t$ -th token of the attribute-value pair  $p^k$ .  $t_k$  is the total number of tokens in  $p^k$ , and  $d_1$  is the dimension of token embedding.

**Graph-guided Logic Planner.** Since each input data has multiple attribute-value pairs, we need to generate a long text (*i.e.*, multiple sentences) to comprehensively describe it. One way to generate long text for data that involves a set of attribute-value pairs is to learn how to arrange the order of the attribute-value pairs to be described. To ensure that the transition flow of the generated long text is logical, we devise a graph-guided logic planner, which 1) first builds a logic graph for each data to characterize the logical relation among all attribute-value pairs, 2) then employs the GCN [18] to refine the embedding of each attribute-value pair by absorbing its logical context information, and 3) finally adopts the GRU [5] to fulfill the attribute-value pair planning.



*Logic Graph Construction.* To comprehensively capture the logical relation among all attribute-value pairs of the given data, we build a graph  $\mathcal{G} = \{\mathcal{E}, \mathcal{R}\}$ , where  $\mathcal{E} = \{n_1, n_2, \dots, n_K\}$  denotes the set of nodes, and node  $n_i$  corresponds to the  $i$ -th attribute-value pair of the given data, *i.e.*,  $p^i$ . Each node  $n_i$  is associated with a hidden vector  $z_i$ , which is updated at each iteration of the information propagation over the graph.  $\mathcal{R} = \{(n_i, n_j) | i, j \in [1, \dots, K]\}$  represents the set of edges among these nodes, reflecting the logical relation among all attribute-value pairs. In particular, we utilize the statistics of the relative positions of each two attribute-value pairs in texts of our corpus  $C = \{Y_1, \dots, Y_N\}$  to indicate their logical relations. Specifically, we assign each edge  $(n_i, n_j)$  a weight  $e_{i,j}$ , which is defined as follows:

$$e_{i,j} = \sum_{Y \in C} \mathbb{1}(r_j^Y > r_i^Y > 0) / (r_j^Y - r_i^Y), \quad (3)$$

where  $\mathbb{1}(\cdot)$  is the indicator function.  $r_i^Y \in \{0, 1, \dots, K\}$  indicates that the ranking of the attribute-value pair  $p^i$  appears in the text  $Y$ . Notably,  $r_i^Y = 0$  when the attribute-value pair  $p^i$  fails to appear in the text  $Y$ . For example, the rankings for the attribute-value pair “selling point: its own bubbler” and “selling point: gently press” in the formal text shown in Figure 1 are 1 and 2, respectively. Intuitively, the attribute-value pairs with the closer relative positions, *i.e.*, the smaller  $(r_j^Y - r_i^Y)$ , have the stronger logical relation. An instance of a logic graph is shown in Figure 4.

*Logical Context Propagation.* Due to its remarkable performance in nonsequential data representation learning, we adopt the GCN to refine each attribute-value pair embedding by logical context propagation as follows:

$$\begin{cases} z_i^{l+1} = \eta \left[ W_c^l \left( \sum_{n_j \in \mathcal{N}_i} \frac{e_{i,j} W_a^l z_j^l}{e_{i,*}} + W_b^l z_i^l \right) + b_c^l \right], \\ e_{i,*} = \sum_{n_{j'} \in \mathcal{N}_i} e_{i,j'} \end{cases} \quad (4)$$

where  $\mathcal{N}_i$  is the set of neighbor nodes of node  $n_i$ .  $W_a^l \in \mathbb{R}^{d_1 \times d_1}$ ,  $W_b^l \in \mathbb{R}^{d_1 \times d_1}$ ,  $W_c^l \in \mathbb{R}^{d_1 \times d_1}$  and  $b_c^l \in \mathbb{R}^{d_1}$  are the learnable parameters for the  $l$ -th propagation layer, where  $l = 1, 2, \dots, M$ .  $M$  is the total number of propagation layers.  $\eta$  denotes the tanh activation function.  $z_j^l$  is the input hidden state vector for node  $n_j$  of the  $l$ -th layer, where we set  $z_i^1 = \text{AVG}(E^i)$ .  $\text{AVG}(\cdot)$  is the average pooling operation. Finally, we regard the last layer output of the GCN (*i.e.*,  $z_i^{M+1}$ ) as the refined embedding of the  $i$ -th attribute-value pair, which is denoted as  $a_i$ .

*GRU-based Planning.* Intuitively, since we can derive the ground-truth plan from the ground-truth stylized text  $Y$ , we can directly use this plan to reorganize the refined embeddings of all the attribute-value pairs of the given data. In particular, we employ the GRU to learn to logically organize the attribute-value pairs based on the refined attribute-value pair embeddings that have absorbed the logical context. Let  $L = [L_1, L_2, \dots, L_K] = [p^{m_1}, p^{m_2}, \dots, p^{m_K}]$  denote the ground-truth attribute-value pair organization plan for the given data  $\mathcal{P}$ , which can be derived from the ground-truth target text  $Y$ .  $L_t = p^{m_t}$  refers to the  $t$ -th mentioned attribute-value pair in the text  $Y$ , where  $m_t \in [1, K]$  is the original index of  $L_t$  in the data  $\mathcal{P}$ . We then arrange the attribute-value pairs of  $\mathcal{P}$  in a sequence according to  $L$  and feed them into the GRU as follows:

$$\mathbf{o}_t, \mathbf{h}_t = \text{GRU}(q_{t-1}, \mathbf{h}_{t-1}), \quad (5)$$

where  $\mathbf{h}_t \in \mathbb{R}^{d_1}$  is the hidden state vector at step  $t$  that encodes the historical logical context.  $\mathbf{o}_t \in \mathbb{R}^{d_1}$  is the output of the GRU that is utilized to predict the  $t$ -th attribute-value pair. The first hidden state is initialized by the average of all the refined node embeddings, *i.e.*,  $\mathbf{h}_0 = \text{AVG}(a_1, \dots, a_K)$ .  $\mathbf{q}_{t-1} = \mathbf{a}_{m_{t-1}}$  is the input vector for predicting the  $t$ -th attribute-value pair and refers to the refined

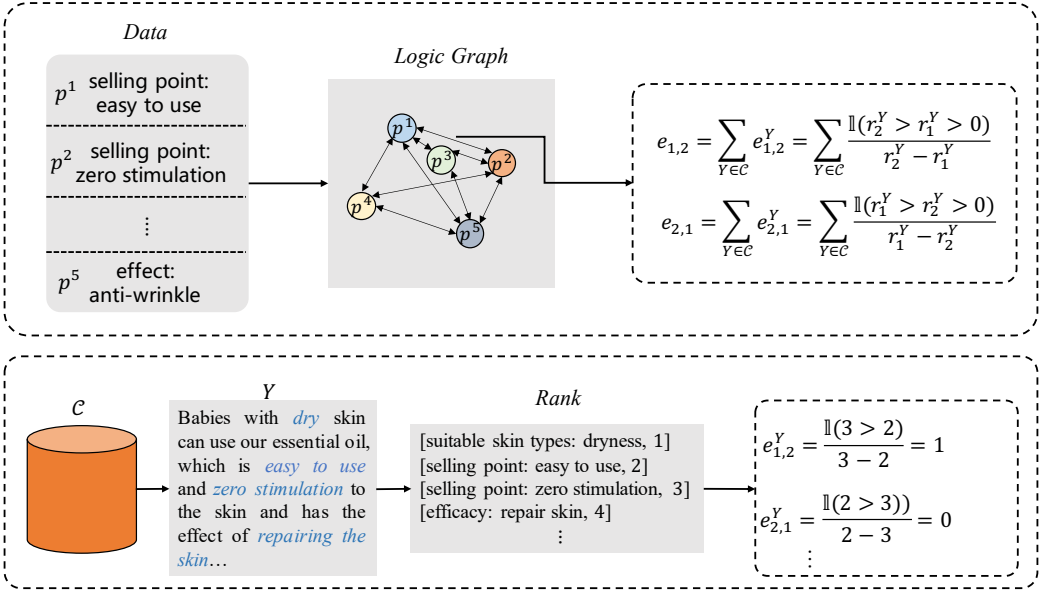


Fig. 4. Illustration of the logic graph construction.  $\mathcal{C}$  denotes all the stylized texts existing in the training set.  $Y$  is a stylized text. The top part is the instance of computing edges between node “selling point: easy to use” (i.e.,  $p^1$ ) and node “selling point: zero stimulation” (i.e.,  $p^2$ ). The bottom part shows the computation of edges  $e_{1,2}^Y$  and  $e_{2,1}^Y$  for the stylized text  $Y$  existing in  $\mathcal{C}$ .

logical embedding of the  $(t - 1)$ -th attribute-value pair described in the ground-truth stylized text  $Y$  when  $t = 2, \dots, K$ .  $q_0$  is a learnable parameter for predicting the first attribute-value pair  $p^{m_1}$  to be described. Finally, given  $\mathcal{P}$ , we can define the probability of the ground-truth plan  $L$  as follows:

$$\begin{cases} P(L|\mathcal{P}) = \prod_{t=1}^K P(L_t|L_{<t}, \mathcal{P}), \\ P(L_t|L_{<t}, \mathcal{P}) = \text{softmax}_{L_t}(\mathbf{o}_t^\top \mathbf{W}_L \mathbf{A}), \end{cases} \quad (6)$$

where  $L_t$  and  $L_{<t}$  are the  $t$ -th attribute-value pair and all its previous attribute-value pairs in  $L$ , respectively.  $P(L_t|L_{<t}, \mathcal{P})$  refers to the probability that the  $t$ -th predicted attribute-value pair is  $L_t$  given the previous  $t - 1$  attribute-value pairs of plan  $L$ .  $\mathbf{A} = [\mathbf{a}_{m_1}; \dots; \mathbf{a}_{m_K}] \in \mathbb{R}^{d_1 \times m_K}$ , where  $[\cdot; \cdot]$  is the concatenation operation.  $\mathbf{W}_L \in \mathbb{R}^{d_1 \times d_1}$  is a mapping matrix to be learned, and  $\text{softmax}_{L_t}(\cdot)$  denotes the  $t$ -th element of the softmax normalized predicted probability distribution over all possible attribute-value pairs. Finally, in order to ensure correct planning for the given data, we use the following loss function:

$$\mathcal{L}_{\text{planning}} = -\log P(L|\mathcal{P}). \quad (7)$$

**Planning-enhanced Data Representation.** To utilize the logic relationship existing in the input data, we directly use the plan predicted by the graph-guided logic planner rather than the original data to obtain a planning-enhanced data representation. Specifically, we first obtain the planned attribute-value pair embeddings of the training data  $\mathcal{P}$  according to the predicted plan  $\hat{L} = [p^{m_1}, p^{m_2}, \dots, p^{m_K}]$  and then derive the data’s planning-enhanced representation with a transformer encoder [45] due to its great success in many NLP tasks [24]. Formally, we have

$$\mathbf{H}_o = \text{Transformer}_o([\mathbf{E}^{m_1}; \dots; \mathbf{E}^{m_K}]), \quad (8)$$

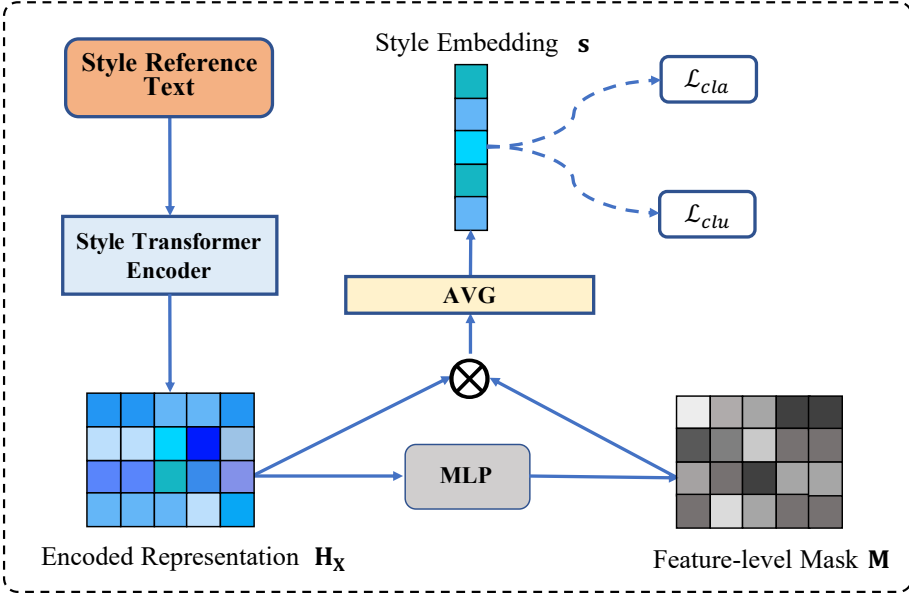


Fig. 5. Illustration of the mask-based style embedding module.  $\otimes$  denotes the elementwise multiplication operation.

where  $E^{m_k}$  denotes the token embedding matrix of the  $k$ -th attribute-value pair in the predicted plan, which is obtained by Eqn. (2).  $H_o \in \mathbb{R}^{T \times d_1}$  is the planning-enhanced representation for the data  $\mathcal{P}$ , and  $T = \sum_{k=1}^K t_k$  is the total number of tokens of  $\mathcal{P}$ .

### 3.3 Mask-based Style Embedding

Existing studies on stylized generation mainly extract the style information from the high-level sentence representation or learn the style embedding directly by omitting certain low-level unnecessary word representations. Nevertheless, we argue that each word in the style reference text could simultaneously contain both the content and style information. Thus, previous style extraction methods suffer from style information loss. Accordingly, we propose a derivation of the style information with an intermediate feature-level mask, as shown in Figure 5. In particular, we first embed the style reference text  $X$  by the pretrained embedding layer of BART,

$$\mathbf{E}_X = \text{BART\_Embedding}(X), \quad (9)$$

where  $\mathbf{E}_X \in \mathbb{R}^{Q \times d_1}$  denotes the embedding of the style reference text  $X$ .  $Q$  is the total number of tokens in text  $X$ , and  $d_1$  is the dimension for each token embedding. Similar to the data embedding, we also employ a transformer encoder as our style encoder as follows:

$$H_X = \text{Transformer}_X(\mathbf{E}_X), \quad (10)$$

where  $H_X \in \mathbb{R}^{Q \times d_1}$  denotes the encoded representation of  $X$ . Obviously, the extracted feature  $H_X$  contains both content and style information, but the content is not needed by the generation model. Additionally, content may interfere with the generation model, causing content inconsistency between the generated text and input data. Therefore, we should remove the content of the input text and disentangle its style. We then introduce the feature-level mask  $M \in \mathbb{R}^{Q \times d_1}$  to derive the style

embedding as follows:

$$\begin{cases} M = \sigma(H_X W_m + b_m), \\ s = AVG(M \otimes H_X), \end{cases} \quad (11)$$

where  $\sigma(\cdot)$  is the sigmoid function and  $W_m \in \mathbb{R}^{d_1 \times d_1}$  and  $b_m \in \mathbb{R}^{d_1}$  are learnable parameters.  $\otimes$  is the elementwise product operation, and  $s \in \mathbb{R}^{d_1}$  is the style embedding extracted from the reference text  $X$ . To achieve accurate style embedding, we introduce two learning constraints for optimizing the parameters of the mask-based style embedding module.

*Style Discrimination.* Here, we introduce the style classifier to guide style embedding learning,

$$\hat{g} = softmax(W_s^2 (tanh(W_s^1 s + b_s^1) + b_s^2), \quad (12)$$

where  $W_s^1 \in \mathbb{R}^{d_1 \times d_1}$ ,  $W_s^2 \in \mathbb{R}^{d_1 \times N_s}$ ,  $b_s^1 \in \mathbb{R}^{d_1}$  and  $b_s^2 \in \mathbb{R}^{N_s}$  are trainable parameters.  $\hat{g} = [\hat{g}_1, \hat{g}_2, \dots, \hat{g}_{N_s}]$  is the predicted style probability distribution. We then use the cross-entropy loss as the objective function,

$$\mathcal{L}_{cla} = - \sum_{i=1}^{N_s} g_i \log \hat{g}_i. \quad (13)$$

*Mutual Information Maximization.* From the information theory perspective and to encourage the learned style embedding  $s$  to be representative, it is desirable to maximize the mutual information between the style embedding  $s$  and the input style reference text  $X$ . Therefore, we have the following loss function according to [52]:

$$\mathcal{L}_{clu} = ||s - s^u||^2 + \frac{1}{N_s} \sum_{u,v \neq u} exp(-||s - s^v||^2), \quad (14)$$

where we suppose that the given style reference text belongs to the  $u$ -th style, and  $s^u = AVG_{j:g_j^u=1} (s_j)$  denotes the center embedding of the  $u$ -th style.

### 3.4 Unbiased Stylized Text Generation

As shown in Figure 6, based on the logic planning-enhanced data representation  $H_o$  and the style embedding  $s$ , we generate the target text  $Y$  with a transformer decoder as follows:

$$\hat{y}_t = Transformer_g([H_o; s], E_{Y_{<t}}), \quad (15)$$

where  $Y_{<t} = \{Y_1, \dots, Y_{t-1}\}$  refers to the previous  $t-1$  tokens, and  $E_{Y_{<t}}$  denotes the embedding of the previous tokens derived by Eqn. (2).  $\hat{y}_t \in \mathbb{R}^{d_1}$  is the output hidden state vector encoding the style, planning, and previously generated token information utilized for predicting the  $t$ -th token.

For optimization, we introduce the following loss function for stylized data-to-text generation:

$$\begin{cases} \mathcal{L}_{gen} = -\log P(Y|\mathcal{P}, X) = -\log \prod_t P(Y_t|Y_{<t}, H_o, s), \\ P(Y_t|Y_{<t}, H_o, s) = softmax_{Y_t}(\mathbf{W}_Y \hat{y}_t + \mathbf{b}_Y), \end{cases} \quad (16)$$

where  $Y_t$  is the  $t$ -th generated token of the target text, while  $\mathbf{W}_Y \in \mathbb{R}^{|\mathcal{V}| \times d_1}$  and  $\mathbf{b}_Y \in \mathbb{R}^{|\mathcal{V}|}$  are parameters to be learned.  $\mathcal{V}$  is the token vocabulary, and  $|\mathcal{V}|$  denotes its size.  $P(Y_t|Y_{<t}, H_o, s)$  represents the probability that the  $t$ -th generated token is the ground-truth token  $Y_t$ .

*Pseudo Triplet Augmentation.* In fact, given data, it is more likely that we only have text corresponding to a single style. For example, given a product, we may only have its formal text and not its informal text. Therefore, optimizing the model with these types of biased training samples may lead to biased text generation, *i.e.*, a stylized text generator that relies only on the content of the given data to generate the target text and overlooks the specified style information. In other words, once our generator encounters the same testing data, it tends to generate the text of a specific style

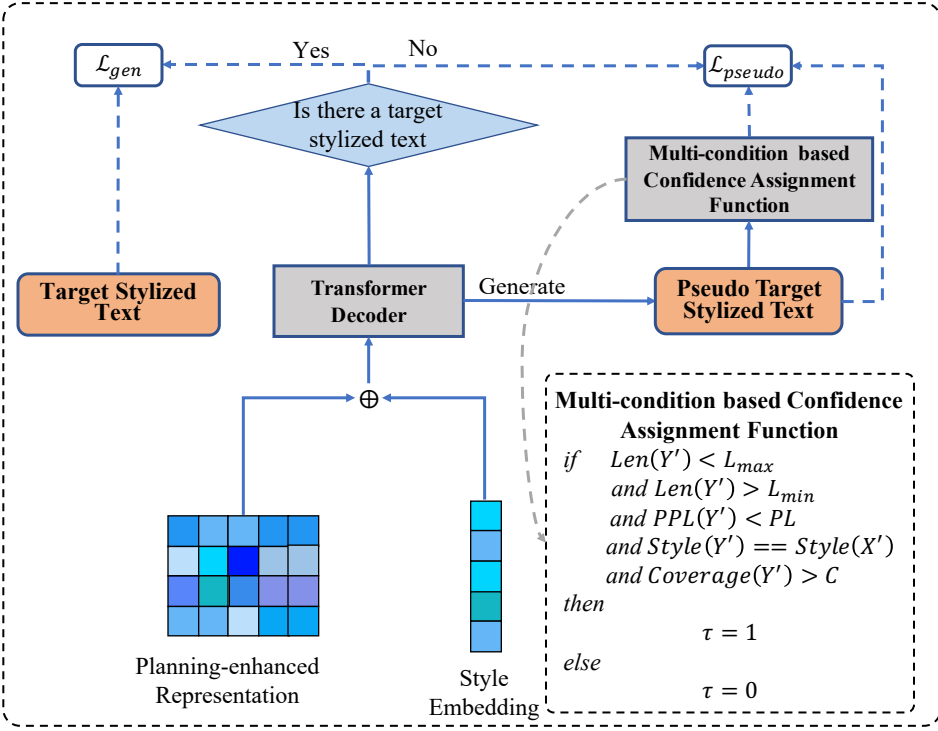


Fig. 6. Illustration of the unbiased stylized text generation module. " $\oplus$ " denotes the concatenation operation.

according to the training data, regardless of the given style reference text. To address this issue, for each data, we augment a pseudo-stylized text for its missing style to promote the diversity of the training dataset. Suppose that we have the target stylized text  $Y$  of style  $s$  for the given data  $\mathcal{P}$  but no stylized text  $Y'$  of style  $s'$ . We then select a style reference text  $X'$  of the style  $s'$  and feed it together with the data  $\mathcal{P}$  to our transformer decoder in Eqn. (15) to obtain a target text candidate  $Y'$  according to Eqn. (16).

Actually, the augmented texts with the model may be of low quality. Therefore, blindly adding these augmented data will bring noise and degrade the performance of the model. Firstly, the style of augmented text should be accurate. Second, the content consistency should be high. Next, the augmented texts should be fluent. Last but not least, the length of the augmented text should in the distribution of the origin training set. Based on these requirements, we choose style, coverage, perplexity, and text length as four metrics to test whether the augmented data is high quality. To be specific, we define a multi-condition-based confidence assignment function as follows:

$$\tau = \mathbb{1}(Len(Y') < L_{max} \text{ and } Len(Y') > L_{min} \text{ and } PPL(Y') < PL \text{ and } Style(Y') == Style(X') \text{ and } Coverage(Y', \mathcal{P}) > C), \quad (17)$$

where  $\tau \in \{0, 1\}$  is the binary confidence for the generated text  $Y'$ , which depends on the length, perplexity (*i.e.*, PPL) [3], style and coverage [40] of the generated text  $Y'$ .  $L_{max}$ ,  $L_{min}$ ,  $PL$ , and  $C$  are the predefined thresholds for the corresponding conditions.

Once the generated text meets all the conditions in Eqn. (17), we deem the triplet  $(\mathcal{P}, X', Y')$  as a reliable pseudo triplet. Accordingly, we introduce the additional loss function for the pseudo triplet

**Algorithm 1** Training Process

**Input:** Training triplets  $\mathcal{D}$ , model  $\mathcal{M}$ , the batch size  $B$ , the number of epochs  $N_{epoch}$ , the number of iterations  $N_{iter}$ , hyperparameters  $L_{max}$ ,  $L_{min}$ ,  $PL$ ,  $C$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$ , the number of styles  $N_s$ .

**Require:** The parameters to be learned  $\Theta$ .

```

1: Initialize parameters  $\Theta$ .
2: for  $e = 1$  to  $N_{epoch}$  do
3:   for  $t = 1$  to  $N_{iter}$  do
4:     for  $i = 1$  to  $N_s$  do
5:       Randomly sample  $B$  examples from  $\mathcal{D}$  for the  $i$ -th style.
6:     end for
7:     Compute the loss function of planning according to Eqn. (7).
8:     Compute  $\mathcal{L}_{cla}$  and  $\mathcal{L}_{clu}$  according to Eqn. (13) and Eqn. (14), respectively.
9:     Compute the generation loss function according to Eqn. (16).
10:    for each training triplet  $(\mathcal{P}, X, Y)$  in the batch do
11:      Generate a pseudo triplet  $(\mathcal{P}, X', Y')$ 
12:      Compute confidence  $\tau$  for the generated pseudo triplet according to Eqn. (17).
13:    end for
14:    Compute the loss function for pseudo triplets according to Eqn. (18).
15:    Update the parameters  $\Theta$  of the model  $\mathcal{M}$  by Eqn. (19).
16:  end for
17: end for
18: return The learned model  $\mathcal{M}$ .

```

as follows:

$$\mathcal{L}_{pseudo} = -\tau \log P(Y'|P, X'). \quad (18)$$

### 3.5 Training and Inference

For training, we combine all the loss functions as follows:

$$\mathcal{L} = \mathcal{L}_{gen} + \alpha \mathcal{L}_{plan} + \beta \mathcal{L}_{cla} + \gamma \mathcal{L}_{clu} + \delta \mathcal{L}_{pseudo}, \quad (19)$$

where  $\alpha$ ,  $\beta$ ,  $\gamma$ , and  $\delta$  are hyperparameters used for balancing the effect of each component on the entire training process. The overall optimization procedure is briefly summarized in Algorithm 1. During the inference phase, given a testing sample, we first employ the well-trained logic planner to predict its attribute organization plan as  $\hat{L}$ . Then, according to  $\hat{L}$ , we can derive the planning-enhanced data representation  $\hat{H}_o$  by Eqn. (8). Finally, we adopt the greedy search algorithm to generate the target text  $\hat{Y}$  according to Eqn. (16).

## 4 EXPERIMENTS

To justify our model, we conduct extensive experiments on a real-world dataset collected from the largest Chinese e-commerce platform Taobao and answer the following research questions:

**RQ1.** Can existing methods be used to solve the proposed task?

**RQ2.** How do the components of the StyleD2T model affect its performance?

**RQ3.** What is the qualitative performance of the StyleD2T model?

Table 2. The statistics of TaoStyle dataset. #Total Attr/#Total Val: the total number of attributes / attribute values. #Avg Attr-val: the average number of input attribute-value pairs. #Avg Len: the average length of target stylized text.

Style	Train	Valid	Test	#Total Attr	#Total Val	#Avg Attr-val	#Avg Len
Informal	9,728	1,000	1,000	108	26,916	8.1	115
Formal	18,000	1,000	1,000	210	9,567	4.6	151

## 4.1 Dataset

To verify the effectiveness of StyleD2T, we collected a Chinese stylized data-to-text dataset named TaoStyle from TaoBao, which contains 20,000 formal advertising texts of products written by qualified experts on the Weitao platform and 11,728 informal advertising transcripts exacted from live broadcast video streams on TaoBao using automatic speech recognition (ASR)<sup>4</sup>. In fact, some noisy transcripts may be introduced by ASR in the dataset construction process. Thus we conducted manual corrections for the noisy transcripts. The to-be-corrected problems include broken sentences, word errors, and word repetition. Meanwhile, during the data collection process, we ignored the text whose length is less than 10 or more than 300. Notably, each stylized text is linked to a product. Accordingly, for each stylized text, we can derive a set of attribute-value pairs as the nonlinguistic input data by using the corresponding product’s attribute properties from its webpage. For keeping the consistency between the input data and the target text, we only retain the attribute-value pairs from the product’s webpage that appear in the target stylized text. Then, for each data and its corresponding target text, we randomly sample stylized text which has the same style with the target text as the style reference to make the (data, style reference text, target text) triplets.

Ultimately, we randomly split our corpus and derive a training set, a validation set and a test set, comprising 27,728, 2,000, and 2,000 triplets, respectively. Table 2 summarizes the statistics of our dataset. In addition, the average token count of the target text and the average number of input attribute-value pairs are 115 and 8.1 for the informal style and 151 and 4.6 for the formal style, respectively. The total numbers of attributes and values are 108 and 26,915 for the formal style and 210 and 9,567 for the informal style, respectively. To intuitively obtain the word distribution difference between informal and formal texts, we show the word clouds over two styles of texts in Figure 7. As seen, the words “remove”, “would”, and “relatively” most frequently appear in informal texts, while “devise”, “use” and “ingredient” often appear in formal texts. In addition, we also visualize the text length distributions of the two styles of texts in Figure 8. Compared with informal texts, the text length distribution of formal texts is more concentrated. This suggests that the informal texts are looser, as compared with formal texts.

## 4.2 Evaluation Metrics.

For the comprehensive evaluation, we conducted both quantitative and qualitative evaluations.

**4.2.1 Quantitative Evaluation.** To quantitatively evaluate the quality of generated texts, we used the following four widely used metrics: (1) **Style Accuracy** refers to the style accuracy of the generated text<sup>5</sup>, (2) **Coverage** [40] measures the average proportion of input attribute-value pairs covered by the generated text, (3) **ROUGE-L** [25] evaluates the similarity between the generated text and the target text based on their longest common subsequences, and (4) **BLEU-4** [32] measures the difference between the generated text and the ground truth based on the 4-gram.

For the unbiased stylized data-to-text generation, although we do not simultaneously have both the ground-truth formal and informal texts for each testing data, we still generate a formal text and

<sup>4</sup>[https://help.aliyun.com/document\\_detail/90727.html](https://help.aliyun.com/document_detail/90727.html).

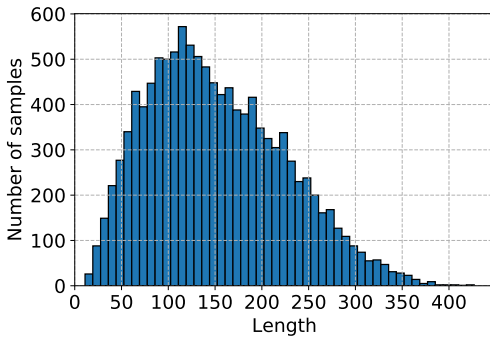
<sup>5</sup>Similar to [6], we train a style classifier with the training set using fastText [17].



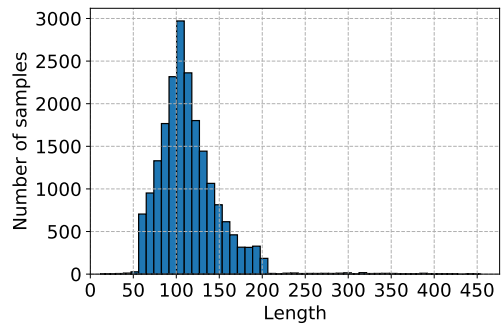
(a) Informal texts.

(b) Formal texts.

Fig. 7. Word clouds of two styles of texts.



(a) Informal texts.



(b) Formal texts.

Fig. 8. The distributions of text length for two styles of texts.

an informal text for each input data. Then, we compute BLEU-4 and ROUGE-L, which require ground-truth texts based on both the generated and ground-truth texts, while the other metrics are derived only by the generated texts.

**4.2.2 Qualitative Evaluation.** Apart from the quantitative evaluation, we also consider the qualitative evaluation, as it has been reported [12, 29, 38] that the quantitative evaluation is unreliable in generation tasks. In this part, we first select 100 testing samples for evaluation. Then, we employ 40 volunteers to manually evaluate the quality of the generated texts of different models for the testing samples. Specifically, for each testing sample, we generate both formal and informal texts, and we have 200 cases for evaluation. Every case is annotated 5 times. These cases are randomly distributed to the 40 volunteers. Namely, each volunteer is asked to evaluate 25 cases. For each case, the volunteer needs to score all the generated texts of different models according to the following three dimensions: (1) **Style Consistency** is used for judging whether the style of the generated text meets the given requirements, (2) **Content Consistency** is used for evaluating the consistency between the input data and generated text, and (3) **Text Fluency** is used for assessing the readability of generated texts. The score range for each metric is {1, 2, 3, 4, 5}, where 1 refers to “very bad” and 5 corresponds to “very good”. Notably, volunteers, who were all native Chinese speakers, did not know which model was used for generating the text.



Table 3. Quantitative evaluation performance comparison with baselines. The bold font indicates the best result and the second best result is underlined. \* denotes that the  $p$ -value of the significant test between our result and the best baseline result is less than 0.01. “Improvement.  $\uparrow$ ” refers to the relative improvement by our model over the best baseline result.

Model	Style Accuracy	Coverage	ROUGE-L	BLEU-4
Seq2seq+StyleT	68.90	72.84	17.44	0.30
Seq2seq+NAST	87.35	68.00	17.44	0.30
BART+StyleT	71.03	88.07	25.38	5.28
BART+NAST	<u>89.48</u>	88.07	25.38	5.28
PHVM+StyleT	50.08	58.49	17.67	1.05
PHVM+NAST	68.52	38.42	17.67	1.05
PlanGen+StyleT	67.18	<u>88.40</u>	<u>25.35</u>	<u>5.31</u>
PlanGen+NAST	85.62	68.25	<u>25.35</u>	<u>5.31</u>
StyleD2T	<b>97.49*</b>	<b>93.44*</b>	<b>26.45*</b>	<b>5.77*</b>
Improvement. $\uparrow$	8.95%	5.70%	4.34%	8.66%

### 4.3 Implementation Details

We train our model on a Tesla V100 GPU, and the batch size is set to 8 for each style. We use Adam as the optimizer, and the learning rate is set to  $1e-4$ . Similar to BART, the number of layers of all transformer-based models (*i.e.*, the style transformer encoder, data transformer encoder and transformer decoder) is set to 6. Their hidden size, embedding size, and positional embedding size are all set to 768, while the number of attention heads is set to 12.  $d_1$  and  $d_2$  are set to 768 and 256. Our embedding layer (*i.e.*,  $BART\_Embedding(\cdot)$ ) is also trainable.  $|\mathcal{V}|$  is 21, 128.  $M$  is set to 1. Since our dataset has texts of two styles,  $N_s$  is set to 2. Hyperparameters  $\alpha$ ,  $\beta$ ,  $\gamma$ , and  $\delta$  are set to 1, 1, 1, and 0.1, respectively. We used the grid search strategy to find the optimal hyperparameters. Specifically, hyperparameters  $\alpha$ ,  $\beta$ ,  $\gamma$ , and  $\delta$  are searched within  $[0, 1]$  with a step size of 0.1. Because ninety-nine percent of the length of texts in our dataset is in the range  $[60, 160]$ ,  $L_{min}$  and  $L_{max}$  are set to 60 and 160.  $PL$  and  $C$  in the multi-condition-based confidence assignment function are set to 50 and 0.95, respectively. To obtain the perplexity and style of the generated text in Eqn. (17), similar to the text style transfer studies [6, 23], the 5-gram language model is trained on our training dataset by KenLM<sup>6</sup> and the style classifier is trained on our training dataset with fastText<sup>7</sup> to implement  $PPL(\cdot)$  and  $Style(\cdot)$ .

### 4.4 On Model Comparison (RQ1)

Since there is no existing method for the stylized data-to-text task, we stack existing general data-to-text generation methods and text style transfer methods to derive our baselines. The data-to-text method can generate a base text for the given data, while the text style transfer model can further render the generated base text into the desired style. In particular, we adopt four data-to-text methods as follows:

(1) **Seq2Seq** [37], an LSTM-based model with an attention mechanism. This method utilizes a local attention-based model to generate a target text of each word from the input text.

(2) **BART** [22], a denoising autoencoder for pretraining sequence-to-sequence models. It is trained by reconstructing the original text of corrupted text with an arbitrary noising function. It has achieved state-of-the-art performance in many natural language generation tasks.

<sup>6</sup><https://kheffield.com/code/kenlm/>.

<sup>7</sup><https://fasttext.cc/docs/en/supervised-tutorial.html>.

Table 4. The average scores of the qualitative evaluation results for three metrics. The best results are in bold. \* denotes that the  $p$ -value of the significant test between our model result and the best baseline result is less than 0.01.

Model	Style Consistency	Content Consistency	Text Fluency
BART+StyleT	3.50	3.80	3.22
BART+NAST	3.50	4.18	3.89
PlanGen+StyleT	3.43	3.71	3.20
PlanGen+NAST	3.38	4.12	4.10
StyleD2T	<b>4.21*</b>	<b>4.31*</b>	<b>4.28*</b>

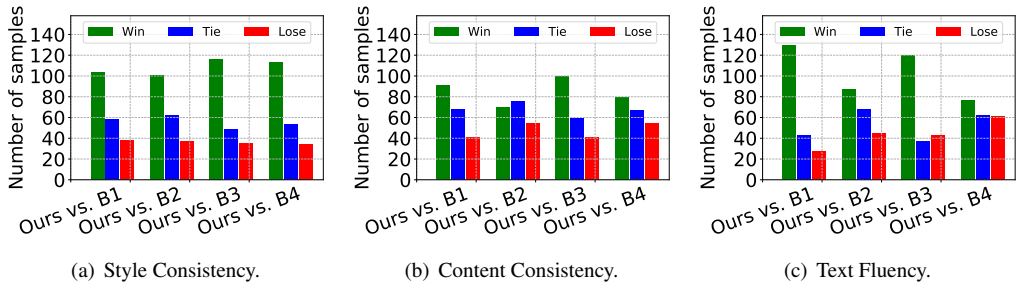


Fig. 9. Comparison results between StyleD2T and the baseline models for three qualitative evaluation metrics. “B1”, “B2”, “B3”, and “B4” denote the baseline BART+StyleT, BART+NAST, Plangen+StyleT, and Plangen+NAST, respectively. “Win”, “Tie”, and “Lose” indicate whether the score of StyleD2T is higher than, equal to, and lower than the baseline models, respectively.

(3) **PHVM** [40], a planning-based hierarchical variational model for data-to-text generation. It first selects the input items contained in a sentence and then generates the corresponding sentence.

(4) **PlanGen** [42], a plan-then-generate framework to improve the controllability of data-to-text models. It is a multistage framework and utilizes the slot keys from a table as a planning signal.

In addition, we used two text style transfer methods:

(a) **StyleT** [6], a transformer-based style transfer model for unpaired text. This is the first work to apply the transformer architecture to style transfer tasks.

(b) **NAST** [14], a nonautoregressive generator for unsupervised text style transfer tasks. It mainly focuses on the connections between aligned words and learns the word-level transfer between styles.

Finally, based on these methods, we derive the following eight baselines: (1) **Seq2seq+StyleT**, (2) **Seq2seq+NAST**, (3) **BART+StyleT**, (4) **BART+NAST**, (5) **PHVM+StyleT**, (6) **PHVM+NAST**, (7) **PlanGen+StyleT**, and (8) **PlanGen+NAST**.

**Quantitative Evaluation.** Table 3 shows the performance comparison between our model and the baselines. As shown by this table, our StyleD2T model exceeds all baselines on all metrics. In particular, coverage is significantly improved (an increase of 5.70%) when using our StyleD2T model compared with the best baseline, which suggests the superiority of our model in maintaining the semantic consistency of the generated text for the given data. Meanwhile, the style accuracy is improved from 89.48% using the best baseline model to 97.49% using our model, which indicates that StyleD2T achieves superior performance in the context of biased training samples. In addition, our StyleD2T also achieves improvements on BLEU and ROUGE-L due to the three modules that we designed. These observations confirm the superiority of our model over the existing methods.

**Qualitative Evaluation.** For the qualitative evaluation, we choose StyleD2T and the four best baselines (*i.e.*, BART+StyleT, BART+NAST, Plangen+StyleT, and Plangen+NAST). Table 4 shows the average scores of Style Consistency, Content Consistency, and Text Fluency for all testing

Table 5. Efficiency comparison with baselines. **Time** is the average generation time consumption of samples in the testing set.

<b>Model</b>	<b>Time</b>	<b>Model</b>	<b>Time</b>	<b>Model</b>	<b>Time</b>
Seq2seq+StyleT	2.6s	BART+NAST	0.4s	PlanGen+StyleT	1.2s
Seq2seq+NAST	1.9s	PHVM+StyleT	2.1s	PlanGen+NAST	0.5s
BART+StyleT	1.6s	PHVM+NAST	1.5s	StyleD2T	0.6s

Table 6. Ablation study results. The best results are in bold. \* denotes that the  $p$ -value of the significant test between our model result and the best model's variant result is less than 0.01.

<b>Model</b>	<b>Style Accuracy</b>	<b>Coverage</b>	<b>ROUGE-L</b>	<b>BLEU-4</b>
w/o-Style	89.67	90.25	25.50	5.25
w/o-StyleCon	89.83	92.27	25.44	5.10
w/o-Planner	93.23	86.95	25.83	5.16
w/o-Graph	94.70	90.46	26.28	5.34
w/o-Pseudo	88.75	90.50	26.37	5.60
w/o-Weight	94.89	89.30	25.65	5.23
w/o-GRU	93.76	90.02	25.44	5.31
w-Fixed	94.07	89.30	25.28	5.24
w-Learnable	96.05	90.94	25.94	5.46
StyleD2T	<b>97.49*</b>	<b>93.44*</b>	<b>26.45*</b>	<b>5.77*</b>

samples of each model. We observed that our StyleD2T model outperforms the baselines in all three dimensions (*i.e.*, Style Consistency, Content Consistency and Text Fluency), which is consistent with the aforementioned quantitative evaluation results. This further indicates that our StyleD2T model has a more powerful stylized generation ability than the baselines. It is worth noting that our model achieves more improvement on the Style Consistency metric over the baselines compared with the other two metrics (*i.e.*, Content Consistency and Text Fluency). This suggests that our model is more robust towards unbiased stylized text generation and the necessity of introducing pseudo target texts.

To gain a more intuitive understanding of the qualitative evaluation results, we also report the comparison results between our StyleD2T model and the four baseline models in Figure 9. In particular, for each pair of model comparisons and each metric, we show the number of samples where our model achieves higher (denoted as “Win”), equal (denoted as “Tie”), and lower scores (denoted as “Lose”) compared with the baselines. As seen, StyleD2T outperforms all baselines across different evaluation metrics, as the number of “Win” cases is always significantly larger than that of “Lose” cases in each pair of model comparisons. In addition, on average, the number of “Win” cases is the largest for the Style Consistency metric compared with the other two metrics. This is consistent with the results in Table 4.

**Efficiency Comparison.** To learn the efficiency of our model, we compared the inference speed of our model and baselines in Table 5. Notably, our model significantly surpasses the most efficient two baselines in terms of the style accuracy and coverage, as shown in Table 3.

#### 4.5 On Ablation Study (RQ2)

To justify each module of our model and the superiority of stylized text input over using fixed or learnable vectors, we devise the following variants of StyleD2T.

- w/o-Style. In this variant, we discarded the mask-based style embedding by setting  $\mathbf{s} = \text{AVG}(\mathbf{H}_X)$  in Eqn. (11);
- w/o-StyleCon. In this variant, we removed the two constraints for style embedding by setting  $\beta = 0$  and  $\gamma = 0$ ;

Case 1	<i>Data</i>	卖点:赋予肌肤 卖点:润泽保护 卖点:天然滋润 适合肤质:敏感肌 主题:面膜
		selling point: empowerment skin; selling point: moist protect; selling point: natural moist; suitable skin: sensitive skin; title: mask.
	<i>Formal Text</i>	这款面膜采用了天然滋润成分, 赋予肌肤润泽保护, 为敏感肌肤带来更好的呵护。 This <b>mask</b> utilizes <b>natural moisturizing</b> ingredients which can <b>moisturize and protect the skin</b> , and bring better care for <b>sensitive skin</b> .
<i>Informal Text</i>	这款面膜, 它的质地比较的天然滋润, 赋予肌肤润泽保护。而且敏感肌也可以用的。	
	This <b>mask</b> , its texture is rather <b>natural moisture</b> , it can <b>protect your skin to keep moisture</b> . And <b>sensitive skin</b> can also use it.	
Case 2	<i>Data</i>	卖点:护理头皮 卖点:深层清洁 卖点:保湿补水 卖点:弹性十足 卖点:有效护理 卖点:盈亮秀发 卖点:柔顺有光泽 卖点:有效改善 主题:威娜滋养修护洗发水250ml(细软-正常发质)男女滋养修护干枯毛躁
		selling point: hair care; selling point: deep cleaning; sell point: hydrating; selling point: elastic; selling point: effective care; selling point: full of bright hair; selling point: supple and shiny; selling point: effectively improve; title: Weina nourishing and repairing shampoo, 250 ml, fine and normal hair, for men and women, nourishing and repairing dry and flabby hair.
	<i>Formal Text</i>	这款洗发水有效改善头皮干燥, 保湿补水, 令秀发柔顺有光泽,还能有效护理头皮, 深层清洁头皮污垢, 盈亮秀发, 令秀发弹性十足。 This <b>shampoo effectively improves</b> dry scalp and <b>moisturizes</b> hair, <b>making it soft and shiny</b> . It is also <b>effective in treating the scalp</b> , <b>deeply cleaning</b> the dirt on the scalp, and <b>leaving the hair full, shiny, and elastic</b> .
<i>Informal Text</i>	这款洗发水呢, 可以有效改善头发干枯毛躁的问题, 保湿补水, 盈亮秀发的, 而且可以让秀发柔顺有光泽的, 还可以帮助我们有效护理头皮的, 可帮助你去保湿, 可深层清洁头皮, 让头发更加的弹性十足的。	
	This <b>shampoo</b> , well, it can <b>effectively help</b> solve the dry hair problem and wild hair, <b>moisturize</b> your hair, and make your hair <b>full, soft, and shiny</b> . It can also help <b>take care of your scalp</b> , help you to moisturize, <b>effectively clean</b> your scalp, and make your hair more <b>elastic</b> .	

Fig. 10. Generated cases of StyleD2T on the TaoStyle dataset. The English texts are translated from the Chinese texts, where some colloquial terms are difficult to translate.

- w/o-Planner. In this variant, we disabled the logic planner by changing Eqn. (8) to  $H_o = \text{Transformer}_o([E^1, \dots, E^K])$ ;
- w/o-Graph. In this variant, we directly fed the original attribute-value pair embeddings of the data to the GRU rather than the enhanced ones;
- w/o-Pseudo. In this variant, we disabled the pseudo triplet augmentation by setting  $\delta = 0$ .

- w/o-Weight. To explore the effect of weight in the logic graph, we set all the weights to 1, *i.e.*,  $e_{i,j} = 1$  in Eqn. (3).
- w/o-GRU. To show the benefit of the GRU-based planning, we removed it by setting  $H_o = \text{Transformer}_o([E^m; \dots; E^K])$  in Eqn. (8)
- w-Fixed. In this variant, we replaced the style embedding  $s$  in our model with a fixed vector.
- w-Learnable. Similar to w-Fixed, in this variant, we replaced the style embedding  $s$  with a learnable vector.

Table 6 shows the ablation study results, from which we have made the following observations. 1) Removing any module of our model decreases its performance, which demonstrates the necessity of each module. 2) Removing the whole mask-based style embedding module (*i.e.*, w/o-Style) or only the two constraints for style embedding (*i.e.*, w/o-StyleCon) significantly decreases the style accuracy, implying the effectiveness of the feature-level mask design and the two constraints in style information extraction from unstructured style reference text. 3) Disabling the logic planner (*i.e.*, w/o-Planner) would largely decrease coverage, which suggests that a good planner can facilitate the subsequent target text generation. 4) Our model exceeds its variants w/o-Graph, w/o-Weight, and w/o-GRU, especially in the coverage metric, which suggests that logic graph, GRU-based planning, and the weight modeling of the logic graph are vital for the planner. 5) Our model substantially improves the style accuracy compared with w/o-Pseudo, which verifies its benefit in the unbiased stylized generation. 6) Our model outperforms w-Fixed and w-Learnable, demonstrating the advantage of using flexible style reference text to deliver the style information.

#### 4.6 On Case Study (RQ3)

We show two cases generated by our StyleD2T model in Figure 10. Based on the generated texts, our model can maintain the semantic consistency well for the given data. For example, in Case 1, the attribute values of the given data are all covered by the formal and informal texts that are generated. Meanwhile, we noticed that the texts generated by our StyleD2T model in these two cases meet the requirement of style consistency. For example, in Case 2, the formal text contains certain formal expressions, such as “It is also effective in treating the scalp, deeply cleaning the dirt on the scalp, and leaving the hair full, shiny, and elastic”, while the informal text has some colloquial expressions, such as “This shampoo, well, it can effectively help solve the dry hair problem and wild hair, moisturize your hair, and make your hair soft and shiny”. Overall, our StyleD2T model can generate coherent text for the given attribute-value pairs according to different styles.

### 5 CONCLUSION AND FUTURE WORK

In this paper, we define a new stylized data-to-text generation task and conduct a case study in the e-commerce domain. In particular, we present a novel framework, StyleD2T, which consists of three key components: logic planning-enhanced data embedding, mask-based style embedding, and unbiased stylized text generation. Meanwhile, we collect a real-world dataset named TaoStyle, which consists of 31,728 triplets, to evaluate the stylized data-to-text generation for formal and informal styles. Extensive experiments on the dataset demonstrate the superiority of our model over existing state-of-the-art methods and verify the effectiveness of each key module of our model. In particular, we find that the graph-guided logic planner helps to improve the coverage of the generated text; the mask-based style embedding indeed boosts the style information extraction, while the pseudo triplet augmentation benefits the unbiased text generation.

In the future, we plan to explore more diverse input data forms, such as tables and English text. As a pioneer study, we also expect that more studies can be conducted in this research direction.

## ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China, No.: U1936203; the Major Basic Research Project of Natural Science Foundation of Shandong Province, No.: ZR2021ZD15; the Shandong Provincial Natural Science Foundation, No.:ZR2022YQ59, and Alibaba Group through Alibaba Innovative Research Program.

## REFERENCES

- [1] John A. Bateman. 1997. Enabling technology for multilingual natural language generation: the KPML development environment. *Nat. Lang. Eng.* 3, 1 (1997), 15–55.
- [2] Razvan Caramalau, Binod Bhattarai, and Tae-Kyun Kim. 2021. Sequential Graph Convolutional Network for Active Learning. In *IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 9583–9592.
- [3] Stanley F Chen, Douglas Beeferman, and Roni Rosenfeld. 1998. Evaluation metrics for language models. (1998).
- [4] Xiaolin Chen, Xuemeng Song, Ruiyang Ren, Lei Zhu, Zhiyong Cheng, and Liqiang Nie. 2020. Fine-Grained Privacy Detection with Graph-Regularized Hierarchical Attentive Representation Learning. *ACM Trans. Inf. Syst.* 38, 4 (2020), 37:1–37:26.
- [5] Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1724–1734.
- [6] Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style Transformer: Unpaired Text Style Transfer without Disentangled Latent Representation. In *Proceedings of the Conference of the Association for Computational Linguistics*. Association for Computational Linguistics, 5997–6007.
- [7] Hercules Dalianis and Eduard H. Hovy. 1993. Aggregation in Natural Language Generation. In *Trends in Natural Language Generation, An Artificial Intelligence Perspective, European Workshop (Lecture Notes in Computer Science, Vol. 1036)*. Springer, 88–105.
- [8] Yang Deng, Yaliang Li, Wenxuan Zhang, Bolin Ding, and Wai Lam. 2022. Toward Personalized Answer Generation in E-Commerce via Multi-Perspective Preference Modeling. *ACM Transactions on Information Systems* 40, 4 (2022), 87:1–87:28.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 4171–4186.
- [10] Pablo Ariel Duboué and Kathleen R. McKeown. 2003. Statistical Acquisition of Content Selection Rules for Natural Language Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- [11] Dominic Espinosa, Michael White, and Dennis Mehay. 2008. Hypertagging: Supertagging for Surface Realization with CCG. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. The Association for Computer Linguistics, 183–191.
- [12] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style Transfer in Text: Exploration and Evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence, the innovative Applications of Artificial Intelligence, and the AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 663–670.
- [13] Albert Gatt and Emiel Kraemer. 2018. Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.* 61 (2018), 65–170.
- [14] Fei Huang, Zikai Chen, Chen Henry Wu, Qihan Guo, Xiaoyan Zhu, and Minlie Huang. 2021. NAST: A Non-Autoregressive Generator with Word Alignment for Unsupervised Text Style Transfer. In *Findings of the Association for Computational Linguistics*. Association for Computational Linguistics, 1577–1590.
- [15] Parag Jain, Anirban Laha, Karthik Sankaranarayanan, Preksha Nema, Mitesh M. Khapra, and Shreyas Shetty. 2018. A Mixed Hierarchical Attention Based Encoder-Decoder Approach for Standard Table Summarization. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 622–627.
- [16] Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled Representation Learning for Non-Parallel Text Style Transfer. In *Proceedings of the Conference of the Association for Computational Linguistics*. Association for Computational Linguistics, 424–434.
- [17] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the Conference of the European Chapter of the Association for Computational*

- Linguistics*. Association for Computational Linguistics, 427–431.
- [18] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*. OpenReview.net.
- [19] Ravi Kondadadi, Blake Howald, and Frank Schilder. 2013. A Statistical NLG Framework for Aggregated Planning and Realization. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. The Association for Computational Linguistics, 1406–1415.
- [20] Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021. Stylized Story Generation with Style-Guided Planning. In *Findings of the Association for Computational Linguistics*. Association for Computational Linguistics, 2430–2436.
- [21] Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. The Association for Computational Linguistics, 1203–1213.
- [22] 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 Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 7871–7880.
- [23] 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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1865–1874.
- [24] Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained Language Model for Text Generation: A Survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. ijcai.org, 4492–4499.
- [25] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.
- [26] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019).
- [27] Susan Weber McRoy, Songsak Channarukul, and Syed S. Ali. 2000. YAG: A Template-Based Generator for Real-Time Systems. In *Proceedings of the International Natural Language Generation Conference*. The Association for Computational Linguistics, 264–267.
- [28] Susan Weber McRoy, Songsak Channarukul, and Syed S. Ali. 2003. An augmented template-based approach to text realization. *Nat. Lang. Eng.* 9, 4 (2003), 381–420.
- [29] Remi Mir, Bjarke Felbo, Nick Obradovich, and Iyad Rahwan. 2019. Evaluating Style Transfer for Text. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 495–504.
- [30] Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2022. On the Study of Transformers for Query Suggestion. *ACM Transactions on Information Systems* 40, 1 (2022), 18:1–18:27.
- [31] Preksha Nema, Shreyas Shetty, Parag Jain, Anirban Laha, Karthik Sankaranarayanan, and Mitesh M. Khapra. 2018. Generating Descriptions from Structured Data Using a Bifocal Attention Mechanism and Gated Orthogonalization. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1539–1550.
- [32] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 311–318.
- [33] Ruihong Qiu, Zi Huang, Jingjing Li, and Hongzhi Yin. 2020. Exploiting Cross-session Information for Session-based Recommendation with Graph Neural Networks. *ACM Transactions on Information Systems* 38, 3 (2020), 22:1–22:23.
- [34] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained Models for Natural Language Processing: A Survey. *CoRR* abs/2003.08271 (2020).
- [35] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2022. Improving Language Understanding by Generative Pre-Training.
- [36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21 (2020), 140:1–140:67.
- [37] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. The Association for Computational Linguistics, 379–389.
- [38] Natalie Schluter. 2017. The limits of automatic summarisation according to ROUGE. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics,

41–45.

- [39] Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2018. Order-Planning Neural Text Generation From Structured Data. In *Proceedings of the AAAI Conference on Artificial Intelligence, the innovative Applications of Artificial Intelligence, and the AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 5414–5421.
- [40] Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and Diverse Text Generation with Planning-based Hierarchical Variational Model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 3255–3266.
- [41] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2017. Style Transfer from Non-Parallel Text by Cross-Alignment. In *Advances in Neural Information Processing Systems: Annual Conference on Neural Information Processing Systems*. 6830–6841.
- [42] Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-Generate: Controlled Data-to-Text Generation via Planning. In *Findings of the Association for Computational Linguistics*. Association for Computational Linguistics, 895–909.
- [43] Zhiqiang Tian, Yezheng Liu, Jianshan Sun, Yuanchun Jiang, and Mingyue Zhu. 2022. Exploiting Group Information for Personalized Recommendation with Graph Neural Networks. *ACM Transactions on Information Systems* 40, 2 (2022), 27:1–27:23.
- [44] Kees van Deemter, Mariët Theune, and Emiel Krahmer. 2005. Real versus Template-Based Natural Language Generation: A False Opposition? *Comput. Linguistics* 31, 1 (2005), 15–24.
- [45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*. 5998–6008.
- [46] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 165–174.
- [47] Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2018. Learning Neural Templates for Text Generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 3174–3187.
- [48] Jun Yang, Weizhi Ma, Min Zhang, Xin Zhou, Yiqun Liu, and Shaoping Ma. 2022. LegalGNN: Legal Information Enhanced Graph Neural Network for Recommendation. *ACM Transactions on Information Systems* 40, 2 (2022), 33:1–33:29.
- [49] Rong Ye, Wenxian Shi, Hao Zhou, Zhongyu Wei, and Lei Li. 2020. Variational Template Machine for Data-to-Text Generation. In *International Conference on Learning Representations*. OpenReview.net.
- [50] 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.org, 3801–3807.
- [51] Peng Yuan, Haoran Li, Song Xu, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. On the Faithfulness for E-commerce Product Summarization. In *Proceedings of the International Conference on Computational Linguistics*. International Committee on Computational Linguistics, 5712–5717.
- [52] Siyang Yuan, Pengyu Cheng, Ruiyi Zhang, Weituo Hao, Zhe Gan, and Lawrence Carin. 2021. Improving Zero-Shot Voice Style Transfer via Disentangled Representation Learning. In *International Conference on Learning Representations*. OpenReview.net.
- [53] Zongmeng Zhang, Xianjing Han, Xuemeng Song, Yan Yan, and Liqiang Nie. 2021. Multi-Modal Interaction Graph Convolutional Network for Temporal Language Localization in Videos. *IEEE Trans. Image Process.* 30 (2021), 8265–8277.
- [54] Minghao Zhao, Qilin Deng, Kai Wang, Runze Wu, Jianrong Tao, Changjie Fan, Liang Chen, and Peng Cui. 2022. Bilateral Filtering Graph Convolutional Network for Multi-relational Social Recommendation in the Power-law Networks. *ACM Transactions on Information Systems* 40, 2 (2022), 31:1–31:24.
- [55] Na Zheng, Xuemeng Song, Qingying Niu, Xue Dong, Yibing Zhan, and Liqiang Nie. 2021. Collocation and Try-on Network: Whether an Outfit is Compatible. In *The ACM International Conference on Multimedia*. ACM, 309–317.
- [56] Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional Chatting Machine: Emotional Conversation Generation with Internal and External Memory. In *Proceedings of the AAAI Conference on Artificial Intelligence, the innovative Applications of Artificial Intelligence, and the AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 730–739.
- [57] Yutao Zhu, Ruihua Song, Jian-Yun Nie, Pan Du, Zhicheng Dou, and Jin Zhou. 2022. Leveraging Narrative to Generate Movie Script. *ACM Transactions on Information Systems* 40, 4 (2022), 86:1–86:32.