# NAST: A Non-Autoregressive Generator with Word Alignment for Unsupervised Text Style Transfer

**Fei Huang, Zikai Chen, Chen Henry Wu, Qihan Guo, Xiaoyan Zhu, Minlie Huang**[*]
The CoAI group, DCST; Institute for Artificial Intelligence;
State Key Lab of Intelligent Technology and Systems;
Beijing National Research Center for Information Science and Technology;
Tsinghua University, Beijing 100084, China.
`f-huang18@mails.tsinghua.edu.cn` `natnstart@gmail.com` `henrychenwu98@gmail.com`
`gqh18@mails.tsinghua.edu.cn` `zxy-dcs@tsinghua.edu.cn` `aihuang@tsinghua.edu.cn`

## Abstract

Autoregressive models have been widely used in unsupervised text style transfer. Despite their success, these models still suffer from the content preservation problem that they usually ignore part of the source sentence and generate some irrelevant words with strong styles. In this paper, we propose a Non-Autoregressive generator for unsupervised text Style Transfer (NAST), which alleviates the problem from two aspects. First, we observe that most words in the transferred sentence can be aligned with related words in the source sentence, so we explicitly model word alignments to suppress irrelevant words. Second, existing models trained with the cycle loss align sentences in two stylistic text spaces, which lacks fine-grained control at the word level. The proposed non-autoregressive generator focuses on the connections between aligned words, which learns the word-level transfer between styles. For experiments, we integrate the proposed generator into two base models and evaluate them on two style transfer tasks. The results show that NAST can significantly improve the overall performance and provide explainable word alignments. Moreover, the non-autoregressive generator achieves over 10x speedups at inference. Our codes are available at `https://github.com/thu-coai/NAST`.

## 1 Introduction

Text style transfer aims at changing the text style while preserving the style-irrelevant contents, which has a wide range of applications, e.g., sentiment transfer (Shen et al., 2017), text formalization (Rao and Tetreault, 2018), and author imitation (Jhamtani et al., 2017). Due to the lack of parallel training data, most works focus on *unsupervised* text style transfer using non-parallel stylistic data.

The cycle consistency loss (Zhu et al., 2017), a.k.a. the back-translation loss (Lample et al., 2018, 2019), has been widely adopted by unsupervised text style transfer models (Dai et al., 2019; He et al., 2020; Yi et al., 2020). Specifically, the cycle loss minimizes the reconstruction error for the sentence transferred from style $\mathcal{X}$ to style $\mathcal{Y}$ and then back to $\mathcal{X}$, which aligns the sentences in two stylistic text spaces to achieve the transfer and preserve style-irrelevant contents. The cycle-loss-based models are trained in an end-to-end fashion, and thus can be easily applied to different datasets.

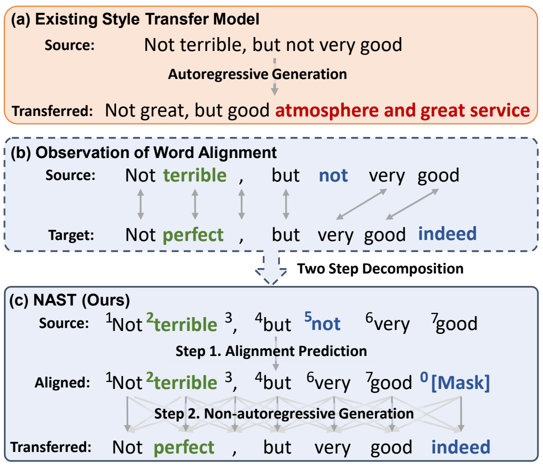

(a) Existing Style Transfer Model
Source: Not terrible, but not very good
Autoregressive Generation
Transferred: Not great, but good atmosphere and great service
(b) Observation of Word Alignment
Source: Not terrible , but not very good
Target: Not perfect , but very good indeed
Two Step Decomposition
(c) NAST (Ours)
Source: 1Not 2terrible 3, 4but 5not 6very 7good
Step 1. Alignment Prediction
Aligned: 1Not 2terrible 3, 4but 6very 7good 0[Mask]
Step 2. Non-autoregressive Generation
Transferred: Not perfect , but very good indeed

Figure 1: Sentiment transfer examples (negative to positive). (a) Existing models without word alignments may generate **words irrelevant to the source sentence**. (b) An example of word alignments between the source and target sentences. Arrows connect aligned words (identical or **relevant**), and **blue words are not aligned**. (c) NAST's generation process. Step 1: generate the index of aligned words. [Mask] is a placeholder for unaligned words. Step 2: generate the transferred sentence non-autoregressively.

Although cycle-loss-based models yield promising results, one of their major failure cases is to replace some part of the source sentence with irrelevant words that have strong styles, as shown in Fig 1(a). This problem degrades content preservation and can be alleviated from two perspectives. **First**, we observe that most words in the human-written

[*]Corresponding author: Minlie Huang.

transferred sentence can be aligned with those in the source sentence. As shown in Fig 1(b), we can align "*Not*" with "*Not*", "*terrible*" with "*perfect*", and leave only a few words unaligned. It shows that humans regard the alignments between words as a key aspect of content preservation, but they are not explicitly modeled by cycle-loss-based models yet. **Second**, existing models use the cycle loss to align sentences in two stylistic text spaces, which lacks control at the word level. For example, in sentiment transfer, "*tasty*" should be mapped to "*awful*" (because they both depict food tastes) but not "*expensive*". We utilize a non-autoregressive generator to model the word-level transfer, where the transferred words are predicted based on contextual representations of the aligned source words.

In this paper, we propose a Non-Autoregressive generator for unsupervised Style Transfer (NAST), which explicitly models word alignment for better content preservation. Specifically, our generation process is decomposed into two steps: first predicting *word alignments* conditioned on the source sentence, and then generating the transferred sentence with a *non-autoregressive* (NAR) decoder. Modeling word alignments directly suppresses the generation of irrelevant words, and the NAR decoder exploits the word-level transfer. NAST can be used to replace the autoregressive generators of existing cycle-loss-based models. In the experiments, we integrate NAST into two base models: StyTrans (Dai et al., 2019) and LatentSeq (He et al., 2020). Results on two benchmark datasets show that NAST steadily improves the overall performance. Compared with autoregressive models, NAST greatly accelerates training and inference and provides better optimization of the cycle loss. Moreover, we observe that NAST learns explainable word alignments. Our contributions are:

- We propose NAST, a Non-Autoregressive generator for unsupervised text Style Transfer. By explicitly modeling word alignments, NAST suppresses irrelevant words and improves content preservation for the cycle-loss-based models. To the best of our knowledge, we are the first to introduce a non-autoregressive generator to an unsupervised generation task.
- Experiments show that incorporating NAST in cycle-loss-based models significantly improves the overall performance and the speed of training and inference. In further analysis, we find that NAST provides better optimization of the cycle loss and learns explainable word alignments.

## 2 Related Work

### Unsupervised Text Style Transfer

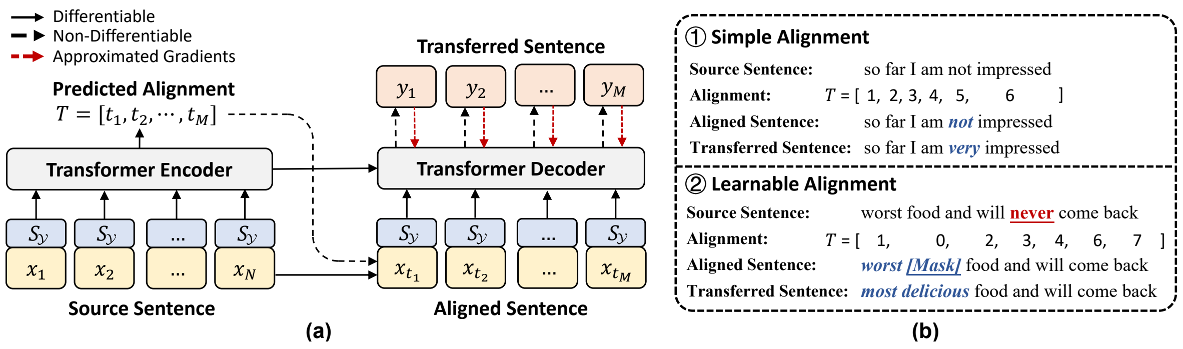
We categorize style transfer models into three types. The first type (Shen et al., 2017; Zhao et al., 2018; Yang et al., 2018; John et al., 2019) disentangles the style and content representations, and then combines the content representations with the target style to generate the transferred sentence. However, the disentangled representations are limited in capacity and thus hardly scalable for long sentences (Dai et al., 2019). The second type is the editing-based method (Li et al., 2018; Wu et al., 2019a,b), which edits the source sentence with several discrete operations. The operations are usually trained separately and then constitute a pipeline. These methods are highly explainable, but they usually need to locate and replace the stylist words, which hardly applies to complex tasks that require changes in sentence structures. Although our two-step generation seems similar to a pipeline, NAST is trained in an end-to-end fashion with the cycle loss. All transferred words in NAST are generated, not copied, which is essentially different from these methods. The third type is based on the cycle loss. Zhang et al. (2018); Lample et al. (2019) introduce the back translation method into style transfer, where the model is directly trained with the cycle loss after a proper initialization. The following works (Dai et al., 2019; Luo et al., 2019; He et al., 2020; Yi et al., 2020) further adopt a style loss to improve the style control.

A recent study (Zhou et al., 2020) explores the word-level information for style transfer, which is related to our motivation. However, they focus on word-level style relevance in designing novel objectives, while we focus on modeling word alignments and the non-autoregressive architecture.

### Non-Autoregressive Generation

Non-AutoRegressive (NAR) generation is first introduced in machine translation for parallel decoding with low latency (Gu et al., 2018). The NAR generator assumes that each token is generated independently of each other conditioned on the input sentence, which sacrifices the generation quality in exchange for the inference speed.

Most works on NAR generation focus on improving the generation quality while preserving the speed acceleration in machine translation. Gu et al. (2018) find the decoder input is critical to the gener-



① Simple Alignment

| | |
|---|---|
| **Source Sentence:** | so far I am not impressed |
| **Alignment:** | $T$ = [ 1, 2, 3, 4, 5, 6 ] |
| **Aligned Sentence:** | so far I am *not* impressed |
| **Transferred Sentence:** | so far I am *very* impressed |

② Learnable Alignment

| | |
|---|---|
| **Source Sentence:** | worst food and will never come back |
| **Alignment:** | $T$ = [ 1, 0, 2, 3, 4, 6, 7 ] |
| **Aligned Sentence:** | *worst [Mask]* food and will come back |
| **Transferred Sentence:** | *most delicious* food and will come back |

Figure 2: (a) Architecture of NAST transferring $X$ to $Y$. $S_\mathcal{Y}$ is the target style. The NAR decoder generates each word $y_i$ independently. (b) Examples of two alignment prediction strategies. **Simple Alignment**: each $y_i$ is aligned with $x_i$. **Learnable Alignment**: a network predicts the alignment, where $t_k = 0$ indicates a [Mask] placeholder.

ation quality. Several works (Bao et al., 2019; Ran et al., 2019) improve the decoder input by aligning source words with target words, which utilize a two-step generation process and inspire the design of NAST. To our knowledge, only a few works of NAR generation explore applications other than machine translation (Han et al., 2020; Peng et al., 2020). We are the first to apply NAR generators to an unsupervised text generation task, which surprisingly outperforms autoregressive models in transfer quality besides the acceleration.

## 3 Methods

In this paper, we formulate the unsupervised text style transfer as follows: for two non-parallel corpora with styles $\mathcal{X}$ and $\mathcal{Y}$ respectively, the task aims at training a style transfer model $G$. The model learns the transfer of two directions, $\mathcal{X} \to \mathcal{Y}$ and $\mathcal{Y} \to \mathcal{X}$, which can be denoted as $P_{G_\mathcal{Y}}(Y|X)$ and $P_{G_\mathcal{X}}(X|Y)$, respectively.

### 3.1 NAST

NAST is a non-autoregressive generator based on the observation of the word alignment: in style transfer tasks, most generated words can be aligned with the source words, where each pair of the aligned words is either identical or highly relevant. For simplicity, we only describe $G_\mathcal{Y}$, where $G_\mathcal{X}$ shares the architecture and parameters except style embeddings. Given the source sentence $X = [x_1, x_2, \cdots, x_N]$, the generation process of NAST is decomposed into two steps: predicting the alignment $T = [t_1, t_2, \cdots, t_M]$, and then generating the transferred sentence $Y = [y_1, y_2, \cdots, y_M]$. When $1 \le t_i \le N$, the generated word $y_i$ is aligned with the source word $x_{t_i}$. Otherwise, $y_i$ is not aligned with any source word, where we set $t_i$ to 0 and fill $x_{t_i}$ with a [Mask] placeholder. Formally, we regard $T$ as a latent variable, and the generation probability is formulated as

$$P_{G_\mathcal{Y}}(Y|X) = \sum_T P_{G_\mathcal{Y}}(Y|X,T)P_{G_\mathcal{Y}}(T|X), \qquad (1)$$

where $P_{G_\mathcal{Y}}(T|X)$ and $P_{G_\mathcal{Y}}(Y|X,T)$ are modeled by an alignment predictor and a non-autoregressive decoder, respectively, as shown in Fig 2.

#### 3.1.1 Alignment Predictor

The alignment predictor predicts the target length $M$ and the alignment $T$ conditioned on the source sentence $X$. We utilize a Transformer (Vaswani et al., 2017) to encode the source sentence and then explore two alternative strategies to predict $T$.

**Simple Alignment.** Simple Alignment assumes that the source and target sentences have the same length, and each generated word $y_i$ is exactly aligned with the source word $x_i$. Formally,

$$P_{G_\mathcal{Y}}(T|X) = \mathbb{I}[M = N]\prod_{i=1}^{M}\mathbb{I}[t_i = i],$$

where $\mathbb{I}[\cdot]$ is the indicator function. A similar strategy has been adopted by editing-based methods (Wu et al., 2019b; Helbig et al., 2020), where they simply replace several words in the source sentence. Although this strategy cannot alter the sentence length, it empirically works well on simple tasks, such as sentiment transfer.

**Learnable Alignment.** Inspired by Ran et al. (2019); Bao et al. (2019), we utilize a pointer network (Vinyals et al., 2015) on top of the encoder, which predicts the alignment $T$:

$$P_{G_\mathcal{Y}}(T|X) = \prod_{i=1}^{M} P_{G_\mathcal{Y}}(t_i|X, t_{<i}).$$

The pointer network is essentially an autoregressive generator, but it only generates the alignment $t_i$ pointing to a source word.

### 3.1.2 Non-autoregressive Decoder

The non-autoregressive decoder (Gu et al., 2018) is a Transformer that generates each word independently. Formally, we have

$$P_{G_\mathcal{Y}}(Y|X,T) = \prod_{i=1}^{M} P_{G_\mathcal{Y}}(y_i|X,T). \tag{2}$$

The Transformer decoder takes the aligned sentence $[x_{t_1}, x_{t_2}, \cdots, x_{t_M}]$ and the target style embedding $S_\mathcal{Y}$ as inputs. It also contains attention connections from the Transformer encoder.

### 3.1.3 Training

NAST is a generator that can be integrated into existing cycle-loss-based models. These models mainly utilize three losses, and the overall objective $\mathcal{L}$ is defined as $\alpha L_{self} + \beta L_{sty} + \gamma L_{cyc}$, where $\alpha, \beta, \gamma$ are hyper-parameters. The **self-reconstruction loss** $L_{self}$ aims at recovering sentences of both styles from their corrupted versions:

$$\begin{aligned}\mathcal{L}_{self} = -\mathbb{E}_{X\sim P_\mathcal{X}}\left[\log P_{G_\mathcal{X}}(X|\widetilde{X})\right] - \\ \mathbb{E}_{Y\sim P_\mathcal{Y}}\left[\log P_{G_\mathcal{Y}}(Y|\widetilde{Y})\right],\end{aligned} \tag{3}$$

where $\widetilde{X}$ and $\widetilde{Y}$ are constructed by word dropout, insertion, and masking (Lample et al., 2019), and $P_\mathcal{X}$ and $P_\mathcal{Y}$ are the data distributions of two styles. The **style loss** $L_{sty}$ is used to guide the style of generated sentences, which has various designs by existing works, e.g., adopting a style discriminator (Dai et al., 2019) or a language model (He et al., 2020). In our implementation, the style loss is determined by the base model. We simply present a general formulation:

$$\begin{aligned}\mathcal{L}_{sty} = -\mathbb{E}_{X\sim P_\mathcal{X}}\left[F(G_\mathcal{Y}(X),\mathcal{Y})\right] - \\ \mathbb{E}_{Y\sim P_\mathcal{Y}}\left[F(G_\mathcal{X}(Y),\mathcal{X})\right],\end{aligned} \tag{4}$$

where $F(X,\mathcal{X})$ indicates a score that shows to which extent the sentence $X$ has the style $\mathcal{X}$, and $G_\mathcal{Y}(X)$ is the generated sentence sampled from $P_{G_\mathcal{Y}}(Y|X)$ in two steps: $T_\mathcal{Y}(X) \sim P_{G_\mathcal{Y}}(T|X)$, $G_\mathcal{Y}(X) \sim P_{G_\mathcal{Y}}(Y|X, T_\mathcal{Y}(X))$. At last, the **cycle loss** $L_{cyc}$ is formulated as

$$\begin{aligned}\mathcal{L}_{cyc} = -\mathbb{E}_{X\sim P_\mathcal{X}}\left[\log P_{G_\mathcal{X}}(X|G_\mathcal{Y}(X))\right] - \\ \mathbb{E}_{Y\sim P_\mathcal{Y}}\left[\log P_{G_\mathcal{Y}}(Y|G_\mathcal{X}(Y))\right].\end{aligned} \tag{5}$$

However, there still exist two obstacles in optimization. **Firstly**, because of the non-differentiable problem, we cannot back-propagate the gradients through the discrete text $G_\mathcal{Y}(X)$ in Eq.(4)(5). As a common workaround, we adopt the Gumbel-Softmax trick (Jang et al., 2017) to approximate the gradients. Therefore, the gradients from $G_\mathcal{Y}(X)$ can be back-propagated through the decoder output (Fig 2(a)). However, the alignment $T_\mathcal{Y}(X)$ is remained discrete and non-differentiable, where we simply stop the gradients[1].

**Secondly**, the losses in Eq.(3)(5) are intractable for NAST because the generation probability, e.g. $P_{G_\mathcal{Y}}(Y|X)$, is summed over all alignments as defined in Eq.(1). We provide solutions for the two alignment strategies separately.

**For Simple Alignment.** There is only one valid alignment between $X$ and $Y$, so the generation probability is tractable as

$$\begin{aligned}\log P_{G_\mathcal{Y}}(Y|X) = \log P_{G_\mathcal{Y}}(Y|X,T^*), \\ \text{where } T^* = \arg\max_T P_{G_\mathcal{Y}}(T|X) = [1,2,\ldots,N].\end{aligned}$$

**For Learnable Alignment.** Inspired by Bao et al. (2019), we introduce a heuristic rule to obtain a pseudo alignment $T^*$:

$$\begin{aligned}T^* = \arg\max_T \sum_{i=1}^{M} cos(e(y_i), e(x_{t_i})) \\ s.t. \quad t_i = 0 \text{ or } t_i > t_j \quad \text{for } \forall\, 1 \le j < i \le M,\end{aligned}$$

where $e(\cdot)$ indicates the word embeddings. We can obtain the pseudo alignment by dynamic programming, and the details are presented in Appendix A. In the pseudo alignment, most words in $Y$ are aligned with identical or highly relevant words in $X$, which can be used as a good label to supervise our model. Next, we derive a tractable lower bound for the generation probability:

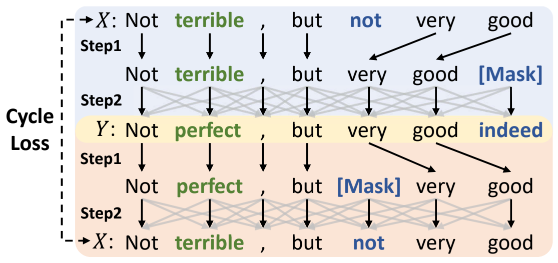
$$\log P_{G_\mathcal{Y}}(Y|X) \ge \log P_{G_\mathcal{Y}}(Y|X,T^*) + \log P_{G_\mathcal{Y}}(T^*|X). \tag{6}$$

On the right side, the first term trains the NAR decoder, and the second term trains the alignment predictor. By substituting Eq.(6) into Eq.(3)(5), we turn to optimize the upper bounds instead of the original intractable losses. The detailed training algorithm is shown in Appendix A.

## 3.2 Discussions

**Residual Connections and Multi-head Attention.** The aligned words in NAST are directly connected with the residual connections, and these connections form several chains in the cycle loss optimization, as shown in Fig 3. Most of these chains represent the word-level transfers and reconstructions, e.g., "*terrible*" is transferred to "*perfect*" and

[1] As a result, the alignment predictor (for Learnable Alignment) is not optimized following the gradients from $G_\mathcal{Y}(X)$, but with a pseudo label introduced later.



Cycle Loss
X: Not terrible , but not very good
Step1
Not terrible , but very good [Mask]
Step2
Y: Not perfect , but very good indeed
Step1
Not perfect , but [Mask] very good
Step2
X: Not terrible , but not very good

Figure 3: Connections of NAST in the cycle loss with the encoder omitted. The word alignments (step 1) and the residual connections (step 2) are in black.

then reconstructs "*terrible*". The reconstruction error is a part of the cycle loss, which is optimized to enhance the alignment in the word space. Besides the residual connections, the multi-head attention mechanism is also important for our model. The attention stops NAST from becoming a degenerate word-to-word dictionary and makes it possible to predict the unaligned words from the context.

**Exposure Bias in Autoregressive (AR) Models.** Exposure bias (Bengio et al., 2015) is a notorious problem in the AR generation. To obtain $P_{G_\mathcal{X}}(X|G_\mathcal{Y}(X))$ in the cycle loss, AR generators predict each word of $X$ based on the ground-truth prefix, which is an easy task even without information from $G_\mathcal{Y}(X)$. As a result, in inference, the model may fail in preserving the sentence meaning as it is trained to focus on its generated prefix. In contrast, NAST focuses on the source sentence since the ground-truth prefix is not given, which suppresses the problem of generating irrelevant words and improves content preservation. Moreover, the training and test are consistent in NAST[2], which alleviates the exposure bias problem.

## 4 Experiments

### 4.1 Experiment Settings

We conduct experiments on two style transfer tasks.

**Sentiment Transfer.** We use the YELP dataset (Li et al., 2018), which consists of two non-parallel corpora with positive and negative sentiments. For each sentence in the test set, multiple human references are provided by Luo et al. (2019).

**Text Formalization.** We use the family and relationship domain of the GYAFC dataset (Rao and Tetreault, 2018), which consists of paired corpora for formal and informal sentences. We do not use the paired data to supervise training.

[2]The claim only applies to NAST with Simple Alignment, because the pseudo alignment used in Learnable Alignment breaks the consistency.

We utilize several SOTA models as baselines, which include CrossAlign (Shen et al., 2017), DelRetrie (Li et al., 2018), Disent (John et al., 2019), StyIns (Yi et al., 2020), StyTrans (Dai et al., 2019), and LatentSeq (He et al., 2020). Our models are modified based on StyTrans and LatentSeq, where we replace their generators with NAST. For StyTrans, NAST adopts a Transformer of the same architecture as the original implementation. However, LatentSeq utilizes an LSTM generator. For a fair comparison, we first incorporate LatentSeq with a vanilla Transformer generator and then replace the generator with NAST of the same architecture. In inference, we use the greedy decoding strategy, i.e., we choose the top-1 candidate at each step in alignment prediction and sentence generation. More details are presented in Appendix B.

### 4.2 Automatic Evaluation

Following Luo et al. (2019); Dai et al. (2019), we utilize a pretrained classifier to evaluate the style accuracy (Acc), and adopt the BLEU-4 score comparing generated sentences with the source sentences (SelfB) or with the references (RefB) to evaluate content preservation. The classifier based on RoBERTa-base (Liu et al., 2019) achieves an accuracy of 97.6% and 90.1% on YELP and GYAFC, respectively. For each transfer direction, we calculate the geometric and harmonic mean of Acc and RefB and then report the average on two directions as G2 and H2, respectively. We further report the perplexity (PPL) of transferred sentences, which is evaluated by GPT2-base (Radford et al., 2019) fine-tuned on the training set.

The results are shown in Table 1. Compared with StyTrans and LatentSeq, NAST exhibits stable performance gains of G2 and H2 on both datasets. On the Yelp dataset, NAST remarkably improves content preservation (at least 6 points with RefB) but suffers a slight decline in Acc. We find that NAST can suppress irrelevant words with strong styles, which possibly leads to the decline in Acc. On the GYAFC dataset, NAST outperforms the base models mainly in Acc instead of RefB, which is affected by model selection strategies with the Acc-RefB trade-off. In Table 1, we choose the best model based on G2. A more comprehensive comparisons with trade-off curves will be discussed in the next section.

In terms of the alignment strategies, Learnable Alignment outperforms Simple Alignment on

| Model | Yelp PPL | Yelp Acc | Yelp SelfB | Yelp RefB | Yelp G2 | Yelp H2 | GYAFC PPL | GYAFC Acc | GYAFC SelfB | GYAFC RefB | GYAFC G2 | GYAFC H2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CrossAlign (Shen et al., 2017) | 105 | 74.0 | 20.3 | 17.9 | 31.8 | 28.8 | 47 | 63.8 | 2.3 | 3.2 | 14.1 | 6.1 |
| DelRetrie (Li et al., 2018) | 94 | 88.7 | 36.8 | 31.1 | 52.5 | 46.0 | 101 | 58.2 | 32.3 | 20.8 | 34.2 | 29.8 |
| Disent (John et al., 2019) | 27 | 92.2 | 8.3 | 13.8 | 35.6 | 24.0 | 27 | 68.4 | 4.8 | 8.0 | 23.4 | 14.4 |
| DualRL (Luo et al., 2019) | 73 | 88.6 | 59.0 | 55.2 | 68.6 | 67.0 | 91 | 58.9 | 50.1 | 40.3 | 43.9 | 39.2 |
| StyIns (Yi et al., 2020) | 98 | 91.5 | 53.2 | 49.0 | 66.9 | 63.7 | 72 | 65.6 | 62.6 | 45.5 | 52.6 | 50.0 |
| StyTrans (Dai et al., 2019) | 136 | **90.4** | 53.3 | 48.6 | 66.2 | 63.1 | 124 | 67.1 | 59.7 | 41.9 | 50.4 | 46.8 |
| **+ NAST (Simple)** | 117 | 88.9 | **63.3**** | **55.9**** | **70.4**** | **68.5**** | 130 | 67.6 | **63.7*** | 41.6 | 50.8 | 47.4 |
| **+ NAST (Learnable)** | **112*** | 87.4 | 62.0** | 54.6** | 69.0** | 67.1** | **119** | **72.9*** | 61.6 | **42.8** | **53.6**** | **49.9**** |
| LatentSeq (He et al., 2020) | 55 | **84.5** | 49.4 | 47.3 | 62.6 | 60.5 | 47 | 55.3 | **57.8** | 38.5 | 44.1 | 42.5 |
| LatentSeq w/ Transformer | **42** | **84.6** | 48.8 | 47.1 | 63.0 | 60.4 | **38** | 58.1 | 54.3 | 35.3 | 45.1 | 43.5 |
| **+ NAST (Simple)** | 73 | 81.2 | 65.2** | 57.6** | **68.1**** | **66.9**** | 56 | 60.4* | 57.0 | 38.2 | 47.4* | 45.6* |
| **+ NAST (Learnable)** | 70 | 79.6 | **65.5**** | **58.0**** | 67.7** | 66.7** | 53 | **64.1**** | 57.0 | **39.2** | **49.0**** | **46.6*** |

Table 1: Automatic evaluation results. **Simple** and **Learnable** indicate two alignment strategies. All values are averaged on two transfer directions. **Bold** denotes the best results for each base model, and underline denotes the best results in all models. * and ** indicate significant improvements over StyTrans or LatentSeq ($p < 0.05$ and $p < 0.01$ in t-test).

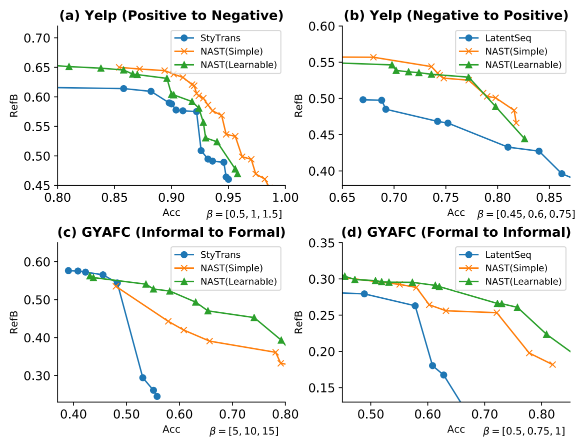


Figure 4: Trade-off curves between style control (Acc) and content preservation (RefB). (a)(c) use StyTrans as the base model, (b)(d) use LatentSeq as the base model. Each curve contains points from three runs with different style loss coefficients $\beta$, whose values for NAST are presented under sub-figures.

|  | #Param | Train (ms) | Inference (ms) |
|---|---|---|---|
| StyTrans | 31.1M | 857 (1.0x) | 249 (1.0x) |
| **+ NAST (Simple)** | 31.1M | **201 (4.3x)** | **8 (31.1x)** |
| **+ NAST (Learnable)** | 32.4M | 339 (2.5x) | 71 (3.5x) |
| LatentSeq w/ Trans. | 21.2M | 1282 (1.0x) | 266 (1.0x) |
| **+ NAST (Simple)** | 21.2M | **714 (1.8x)** | **23 (11.6x)** |
| **+ NAST (Learnable)** | 22.4M | 761 (1.7x) | 125 (2.1x) |

Table 2: Parameter size and the training and inference latency on GYAFC. The speedup of training LatentSeq is less significant, because the bottleneck is a language model used in the style loss, costing about 487ms.

GYAFC, but there is no significant difference on Yelp. We suppose that the sentiment transfer task is more straightforward than the text formalization, where the model can achieve a good transfer performance on Yelp without changing sentence structures.

Compared with all baselines, our best models set new SOTA results on two datasets in the overall performance of the transfer accuracy and content preservation (i.e., G2 and H2).

**Trade-Off Curves.** To investigate the trade-off between style control (shown by Acc) and content preservation (shown by RefB), we follow Fu et al. (2018) and evaluate the models with different hyper-parameters. To be specific, we select three different style loss coefficients $\beta$ around the best value. Please see Appendix B.2 for the search range and other details. Since the trade-off varies through the training, we evaluate the models and collect data points at every epoch. It is different from Fu et al. (2018), who only plot the metrics of the best model in each run. The curves are shown in Fig 4, where we only keep the outermost points of each model and remove the points dominated by at least one other point in both Acc and RefB.

The curves of NAST are generally above those of the base models, indicating that NAST achieves better content preservation when the style accuracy is kept at a similar level. In Fig 4 (c)(d), we find that the base model's RefB drops rapidly after Acc exceeds a certain value, which indicates that the cycle loss fails to preserve the sentence-level alignment, thereby leading to model collapse. By contrast, NAST largely alleviates the issue of model collapse. Moreover, we find that Learnable Alignment outperforms Simple Alignment on GYAFC, but performs equally or slightly worse on Yelp, due to the task differences discussed above.

**Training & Inference Speed.** Thanks to the parallel decoding of the NAR generator, NAST accelerates the model training and inference as shown in Table 2. For a fair comparison, NAST and the corresponding base model utilize the same Transformer architecture. The computation devices are detailed in Appendix B.3.

## 4.3 Human Evaluation

We follow Li et al. (2018) and conduct human evaluation experiments on the Yelp dataset. In addition to NAST and the base models, we choose three

| Model | Fluency | Style | Content |
|---|---|---|---|
| DelRetrie | 3.87 | 3.90 | 3.05 |
| DualRL | 4.38 | 4.25 | 4.24 |
| StyIns | 4.25 | 4.00 | 4.11 |
| StyTrans | 4.24 | **3.91** | 4.16 |
| **+ NAST(Simple)** | 4.34 | 3.87 | **4.41**** |
| **+ NAST(Learnable)** | **4.39*** | 3.87 | 4.38** |
| LatentSeq | 4.53 | 3.92 | 3.59 |
| **+ NAST(Simple)** | 4.41 | **3.93** | 4.43** |
| **+ NAST(Learnable)** | **4.57** | 3.82 | **4.48**** |

Table 3: Human evaluation results. **Bold** denotes the best results for each base model and underline denotes the best results among all models. * and ** indicate significant improvements over the base model ($p < 0.05$ and $p < 0.01$ in t-test). The Krippen-dorff's alpha of human rating is 0.72, indicating acceptable inter-annotator agreement.

| | Yelp | | | GYAFC | | |
|---|---|---|---|---|---|---|
| **Model** | **Acc** | **RefB** | **G2** | **Acc** | **RefB** | **G2** |
| LatentSeq(Trans.) | **84.6** | 47.1 | 63.0 | 58.1 | 35.3 | 45.1 |
| NAST(Simple) | 81.3 | 57.4 | **68.1** | 58.3 | 39.3 | **47.5** |
| w/o Aligned Sent. | 73.2 | 44.1 | 56.8 | 54.1 | 34.4 | 42.8 |
| w/o Multi-head Attn. | 57.0 | **62.8** | 59.6 | 22.0 | **41.1** | 29.7 |
| w/ Soft-Embedding | 44.3 | 44.6 | 43.6 | 63.4 | 26.0 | 40.3 |
| w/ Stop-Gradient | 80.5 | 50.1 | 63.5 | **64.2** | 26.3 | 40.6 |

Table 4: Ablation study of NAR decoder and gradient approximation methods. The base model is LatentSeq.

baselines with the highest G2. For each model, we sample 100 sentences (50 in each transfer direction), and 900 sentences are evaluated in total. For each sentence, three annotators are asked to rate from 1 (worst) to 5 (best) for fluency, style control, and content preservation.

The human evaluation results are shown in Table 3. Similar to the automatic evaluation results, NAST improves content preservation significantly. Moreover, we find that Learnable Alignment outperforms Simple Alignment in terms of fluency. It can be partially attributed to the fact that Learnable Alignment, which is able to remove or add words, is more flexible in generation.

## 4.4 Ablation Study

**NAR decoder.** Although NAST with Simple Alignment has a simple, straightforward design, it works surprisingly well compared with an AR generator. We conduct an ablation study to investigate the impact of different components in the NAR decoder. First, we remove the *aligned sentence* from the decoder input. Specifically, the decoder input is the positional encodings without the word embeddings. Second, we remove the *multi-head attention* in the decoder, and thus each output word is solely conditioned on its aligned word.

The results are shown in Table 4. After we remove the aligned sentence, the performance drops but still remains comparable. It shows that the multi-head attention over the source sentence learns reasonable transfer, while the performance can be largely improved by providing the decoder with the aligned sentence as input. After we remove the multi-head attention, the overall performance drops remarkably, especially on GYAFC. It shows that NAST utilizes multi-head attention to gather sentence-level information, and it is essentially *not* a word-to-word dictionary. Moreover, the contribution of the multi-head attention is larger on GYAFC than on Yelp. It further justifies that text formalization is less straightforward than sentiment transfer since it requires more sentence-level modifications.

| **Pseudo Alignment in Self-Reconstruction Loss** |
|---|
| S: that [Mask] *talk* , if are not happy *like* but you . |
| P: that **[Mask]** , if [Mask] are not happy but you [Mask] . |
| T: that **is** , if others are not happy but you are . |
| **Pseudo Alignment in Cycle Loss** |
| S: i leave your email on *exercise* , and see what happens . |
| P: **i** leave your email [Mask] on [Mask] **,** and see what happens **.** |
| T: **just** leave your email loged on accidentally **...** and see what happens **!** |

Table 5: Pseudo alignments on GYAFC. S = source, P = pseudo alignment, T = target. *Unaligned source words*, unaligned target words, and **non-identical aligned words** are marked in different colors.

**Gradient Approximation Methods.** The choice of gradient approximation methods is important for tackling the non-differentiable problem. Besides the Gumbel-Softmax trick used in our full model, we try two alternative methods. 1) The Soft-Embedding approximation (Dai et al., 2019) multiplies the softmax distribution by the word embedding matrix to get "soft" word embeddings. 2) The Stop-Gradient strategy (He et al., 2020) stops the gradient at the decoder output in the cycle loss. However, the style loss requires the output to be differentiable, so we still apply the Gumbel-Softmax trick for the style loss. Results in Table 4 show that the Gumbel-softmax trick outperforms the other methods, so we utilize the Gumbel-Softmax trick for NAST in other experiments.

**Learnable Alignment.** According to Eq.(3)(5)(6), the alignment predictor in Learnable Alignment is supervised by pseudo alignments when optimizing the upper bounds of the self-reconstruction loss and the cycle loss. For the former, the alignment predictor learns to align the corrupted $\widetilde{X}$ with $X$. For the latter, the alignment predictor learns to align the transferred sentence $G_{\mathcal{Y}}(X)$ with the original $X$. We show two cases in Table 5, where the pseudo alignments are of acceptable quality.

To investigate the effects of the pseudo align-

| Model | Acc | RefB | G2 | $\lvert\Delta\rvert$ | $std(\Delta)$ |
|---|---|---|---|---|---|
| NAST(Simple) | 66.5 | 41.6 | 50.4 | 0.00 | 0.00 |
| NAST(Learnable) | **73.0** | **43.5** | **54.2** | 0.80 | 1.26 |
| w/o Pseudo(Recon) | 66.1 | 41.5 | 50.3 | 0.02 | 0.28 |
| w/o Pseudo(Cyc) | 68.5 | 42.5 | 52.7 | 0.96 | 0.47 |

Table 6: Ablation study of NAST with Learnable Alignment on GYAFC. $\Delta$ is the length difference before and after the transfer. $|\Delta|$ and $std(\Delta)$ indicate the average absolute value and the standard deviation, respectively. All models use StyTrans as the base model.

| | **Yelp (Positive to Negative)** |
|---|---|
| Source | love this place and will keep coming back . |
| LatentSeq | *do n't waste your time* and wo n't be back . |
| StyTrans | avoid this place and *will keep coming back* . |
| **NAST(Simp.)** | **skip** this place and will **never** coming back . |
| **NAST(Lear.)** | **hate** this place and will not **be** coming back . |
| | **Yelp (Negative to Positive)** |
| Source: | i did n't even eat it . |
| LatentSeq: | i always *love their food and service* . |
| StyTrans: | i love the food eat it . |
| **NAST(Simp.):** | i **love it and** eat it . |
| **NAST(Lear.):** | i definitely **love** [DEL] **to** eat it . |
| | **GYAFC (Formal to Informal)** |
| Source | the world would be happier if men knew what women want . |
| LatentSeq | *the guy would be mad if they want* what women want . |
| StyTrans | the world would be *what if thing what girls want girl ur girl want* . |
| **NAST(Simp.)** | **and** world **'ll** be happier if men knew what women want . |
| **NAST(Lear.)** | **just** world would be happier ... if **guys** knew what women want [Del] |
| | **GYAFC (Informal to Formal)** |
| Source: | i do n't know ! ... i just want the points ... lol |
| LatentSeq: | i do not know . i just want the points . *however , i am not a good one .* |
| StyTrans: | i do not know ! |
| **NAST(Simp.):** | i do **not** know ! **.** i just want the points **. .** |
| **NAST(Lear.):** | i do **not** know ! [Del] i just want the points . [Del] |

Table 7: Transfer cases. *Red words* indicate irrelevant phrases or failed transfer in style. Non-trivial alignments and **non-identical aligned words** are marked in colors. [Del] indicates the source word is unaligned.

ments supervision, we remove $\log P_{G_y}(T^*|X)$ in Eq.(6) for the two losses separately. Results are shown in Table 6. Without the pseudo alignments supervision in the self-reconstruction loss, the model almost degenerates into Simple Alignment, because keeping the length unchanged is the easiest way to minimize the cycle loss. Without the pseudo supervision in the cycle loss, Learnable Alignment is slightly weaker than the full model but still outperforms Simple Alignment.

## 4.5 Case Study of Word Alignment

We present several transfer cases in Table 7. We observe that a major failure mode of the base models is generating irrelevant words. We also observe that NAST achieves better content preservation, and most words in NAST's prediction can be aligned with the source words. Focused on the alignment strategies, we observe that the outputs of NAST with Simple Alignment sometimes contain grammar errors (e.g., "*will never coming back*"), which can be attributed to its limitation of not changing the sentence length. In contrast, we observe that Learnable Alignment can add and remove words at appropriate positions.

| | **NAST(Simple) on Yelp (Negative to Positive)** | | | |
|---|---|---|---|---|
| **Src Word** | **Transferred Words** | | | |
| *helpful* | ***weird*** (100%) | | | |
| *fresh* | ***tasteless*** (61.5%) | ***overcooked*** (38.5%) | | |
| *definitely* | ***not*** (92.9%) | ***never*** (7.1%) | | |
| *nice* | ***rude*** (52.9%) | *no* (47.1%) | | |
| *best* | ***worst*** (96.8%) | *money* (3.2%) | | |
| *delicious* | ***bland*** (82.6%) | ***ok*** (13.0%) | ***frozen*** (4.4%) | |
| *love* | ***hate*** (63.6%) | *ordered* (18.2%) | ***skip*** (13.6%) | ***avoid*** (4.6%) |
| | **NAST(Learnable) on GYAFC (Informal to Formal)** | | | |
| **Src Word** | **Transferred Words** | | | |
| *'m* | ***am*** (100%) | | | |
| *n't* | ***not*** (98.5%) | *n't* (1.5%) | | |
| *guy* | ***man*** (98.4%) | *guy* (1.6%) | | |
| *u* | ***you*** (89.2%) | *[Del]* (10.8%) | | |
| *lol* | **.** (41.7%) | ***[Del]*** (41.7%) | *although* (16.7%) | |
| ... | ***[Del]*** (31.3%) | **,** (27.4%) | **.** (26.3%) | and other 7 words |
| *mean* | ***believe*** (50.0%) | *mean* (20.8%) | *am* (20.8%) | and other 2 words |
| *[Mask]* | **.** (55.2%) | *a* (12.0%) | ***?*** (4.8%) | and other 28 words |

Table 8: Cases of aligned word pairs generated by NAST. *[Del]* and *[Mask]* indicate an unaligned source word or an unaligned transferred word, respectively. ***Reasonable transfers are in blue.***

To understand the learned *word alignments* and the *word-level transfer*, we count the aligned word pairs based on the prediction of Learnable Alignment. Several cases are presented in Table 8. We observe the aligned word pairs are highly explainable. For example, NAST maps "*delicious*" to "*bland*" in sentiment transfer and maps "*guy*" to "*man*" in text formalization. These cases show that the model can learn fine-grained word-level transfer, where "*delicious*" and "*bland*" both depict food taste with different styles. Moreover, NAST with Learnable Alignment learns to add or remove words at reasonable positions, such as adding missing punctuation marks (".", "*?*") and removing redundant words ("...", "*lol*") in text formalization.

## 4.6 Analysis of Cycle Loss Optimization

The cycle loss plays a key role in unsupervised style transfer, which achieves style control and content preservation by aligning the sentences in two text spaces. However, the optimization is not straightforward due to the non-differentiable problem. In this section, we study how the cycle loss optimization is affected by the generator architecture and compare a NAR generator with an AR generator[3]. To remove the interference of other losses, we train the model solely with the cycle loss and report the BLEU-4 score of the cycle reconstruction.

The results are shown in Table 9. The NAR generator remarkably outperforms the AR generator with all gradient approximation methods. We provide two possible explanations for this observation. One reason is that word alignments can help the cycle loss align the text spaces. As discussed

[3]For a fair comparison, the target sentence length is provided to both models, where the AR generator does not need to predict the EOS token.

|  | Gumbel-Softmax | Stop-Gradient | Soft-Embedding |
|---|---|---|---|
| NAR | **94.4±0.4** | **67.6±4.2** | **33.6±13.8** |
| AR | 84.9±1.1 | 23.5±1.1 | 29.9±15.1 |

Table 9: BLEU-4 of the cycle reconstruction on the Yelp dataset. The values are reported with mean and standard deviation of three runs with different seeds.

in Sec 3.2, the residual connections directly connect aligned words, which exploits the word-level transfer and reconstruction. Compared with the AR generator that aligns the text spaces at the sentence level, aligning word pairs can be much easier. Another possible reason is the error accumulation caused by the gradient approximation methods. In each step of the AR generation, the gradient approximation methods are applied to the generated word, and the word is then fed into the model as the next input. As a result, gradients will be approximated multiple times in the back-propagation, and the error brought by the approximation may be accumulated and possibly lead to unstable optimization.

Our analysis provides a perspective to understand how NAST works, and reveals that the generator architecture can deeply affect the optimization in the non-differentiable problem. However, we should be cautious when generalizing the results to other settings. We notice inconsistent performance report for the gradient approximation methods (Dai et al., 2019; Tu et al., 2020; He et al., 2020), where the phenomenon needs further study.

## 5 Conclusion

In this paper, we propose NAST, a Non-Autoregressive generator for unsupervised text Style Transfer. It explicitly models word alignments to suppress irrelevant words and exploits the word-level transfer between different styles. Experiments show that NAST improves the overall performance, provides explainable word alignments, and largely speed up training and inference.

However, we should also notice a potential limitation: NAST relies on the assumption that word alignments exist between the source and target sentences. In a more complicated task that lacks word alignments, NAST may lose its advantage of exploiting the word-level transfer. In future work, we will improve NAST to tackle noisy word alignments in more challenging datasets and build explainable and faster models for a broader range of unsupervised text generation tasks.

## Acknowledgments

This work was partly supported by the NSFC projects (Key project with No. 61936010 and regular project with No. 61876096). This work was also supported by the Guoqiang Institute of Tsinghua University, with Grant No. 2019GQG1 and 2020GQG0005.

## References

Yu Bao, Hao Zhou, Jiangtao Feng, Mingxuan Wang, Shujian Huang, Jiajun Chen, and Lei Li. 2019. Non-autoregressive transformer by position learning. *CoRR*, abs/1911.10677.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style transformer: Unpaired text style transfer without disentangled latent representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5997–6007, Florence, Italy. Association for Computational Linguistics.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 663–670. AAAI Press.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Qinghong Han, Yuxian Meng, Fei Wu, and Jiwei Li. 2020. Non-autoregressive neural dialogue generation. *CoRR*, abs/2002.04250.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. A probabilistic formulation of unsupervised text style transfer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

David Helbig, Enrica Troiano, and Roman Klinger. 2020. Challenges in emotion style transfer: An exploration with a lexical substitution pipeline. In *Proceedings of the Eighth International Workshop on Natural Language Processing for Social Media*, pages 41–50, Online. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence to sequence models. In *Proceedings of the Workshop on Stylistic Variation*, pages 10–19, Copenhagen, Denmark. Association for Computational Linguistics.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Guillaume Lample, Sandeep Subramanian, Eric Michael Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

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.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the variance of the adaptive learning rate and beyond. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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.

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 2019, Macao, China, August 10-16, 2019*, pages 5116–5122. ijcai.org.

Kainan Peng, Wei Ping, Zhao Song, and Kexin Zhao. 2020. Non-autoregressive neural text-to-speech. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7586–7598. PMLR.

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

Qiu Ran, Yankai Lin, Peng Li, and Jie Zhou. 2019. Guiding non-autoregressive neural machine translation decoding with reordering information. *CoRR*, abs/1911.02215.

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality 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 129–140, New Orleans, Louisiana. Association for Computational Linguistics.

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 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6830–6841.

Lifu Tu, Richard Yuanzhe Pang, Sam Wiseman, and Kevin Gimpel. 2020. ENGINE: Energy-based inference networks for non-autoregressive machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2819–2826, Online. Association for Computational Linguistics.

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 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

Chen Wu, Xuancheng Ren, Fuli Luo, and Xu Sun. 2019a. A hierarchical reinforced sequence operation method for unsupervised text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4873–4883, Florence, Italy. Association for Computational Linguistics.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019b. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5271–5277. ijcai.org.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P. Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7298–7309.

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 2020*, pages 3801–3807. ijcai.org.

Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. 2018. Style transfer as unsupervised machine translation. *CoRR*, abs/1808.07894.

Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5897–5906. PMLR.

Chulun Zhou, Liangyu Chen, Jiachen Liu, Xinyan Xiao, Jinsong Su, Sheng Guo, and Hua Wu. 2020. Exploring contextual word-level style relevance for unsupervised style transfer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7135–7144, Online. Association for Computational Linguistics.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society.

**Algorithm 1** DP Algorithm for Pseudo Alignment $DP(X, Y)$

```
Require: Source sentence $X = [x_1, x_2, \cdots, x_N]$,
         Target sentence $Y = [y_1, y_2, \cdots, y_M]$.
 1: Initialize $f(0, j) = 0$ for $\forall j = 0, 1, \cdots, N$.
 2: Initialize $T(0, j)$ as empty lists for $\forall j = 0, 1, \cdots, N$.
 3: Calculate the similarity matrix:
        $sim_{i,j} = cos(e(y_i), e(x_j))$.
 4: for $i = 1, 2, \cdots, M$ do
 5:     for $j = 0, 1, 2, \cdots, N$ do
 6:         Calculate three choices of $f(i, j)$:
              $c_1 := f(i-1, j)$
              $c_2 := f(i-1, j-1) + sim_{i,j}$ only valid if $j > 0$
              $c_3 := f(i, j-1)$ only valid if $j > 0$
 7:         if $c_1$ is the maximum choice then
              ▷ $y_i$ is not aligned.
 8:             $f(i, j) := c_1$, $T(i, j) := T(i-1, j) \oplus [0]$
              ▷ $\oplus$ means list concatenation.
 9:         else if $c_2$ is the maximum choice then
              ▷ $y_i$ is aligned with $x_j$.
10:             $f(i, j) := c_2$, $T(i, j) := T(i-1, j-1) \oplus [j]$
11:         else if $c_3$ is the maximum choice then
              ▷ $y_i$ is aligned with $x_k$, where $k < j$.
12:             $f(i, j) := c_3$, $T(i, j) := T(i, j-1)$
13:         end if
14:     end for
15: end for
16: return $DP(X, Y) := T(M, N)$
```

## A Optimization of NAST with Learnable Alignment

Since the generation probability $P_{G_y}(Y|X)$ is intractable for NAST with Learnable Alignment, we introduce a pseudo alignment $T^*$. For $X = [x_1, x_2, \cdots, x_N]$ and $Y = [y_1, y_2, \cdots, y_M]$, the pseudo alignment $T^*$ is defined by a heuristic rule:

$$\begin{aligned} T^* &= \arg\max_T V(X, Y) \\ &= \sum_{i=1}^{M} cos(e(y_i), e(x_{t_i})) \\ s.t.\quad & t_i = 0 \text{ or } t_j < t_i \le N \quad \text{for } \forall\, 1 \le j < i \le M, \end{aligned}$$

where $e(\cdot)$ indicates the word embeddings. For the unaligned target words, we set $x_0$ to a [Mask] placeholder, and set the cosine similarity between the [Mask] placeholder and any other tokens to 0.

The pseudo alignments are obtained by dynamic programming. We introduce a 2-dim array $f(i, j)$ indicating the maximum value of the objective function $V(X, Y)$ if $Y = [y_1, y_2, \cdots, y_i]$ and $X = [x_1, x_2, \cdots, x_j]$. We further introduce a list $T(i, j)$ that records the best alignment for $f(i, j)$. The algorithm is presented in Algorithm 1. The time complexity is $O(NMd)$, where the bottleneck is calculating the similarity matrix, and $d$ is the dimension of word embeddings.

Based on the pseudo alignments, we derive tractable upper bounds of the losses, which are

**Algorithm 2** Training Algorithm for NAST with Learnable Alignment

```
Require: Non-parallel text distribution $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$.    Max number of batches: $max\_batch$.
1: for $iter = 1, 2, \cdots, max\_batch$ do
2:     Sample $X$ from $P_{\mathcal{X}}$, $Y$ from $P_{\mathcal{Y}}$.
3:     Construct $\widetilde{X}$ and $\widetilde{Y}$ from $X$ and $Y$.
4:     Use Algorithm 1 to obtain pseudo alignments for the self-reconstruction loss:
           $T^*_{X,self} = DP(\widetilde{X}, X), \quad T^*_{Y,self} = DP(\widetilde{Y}, Y).$
5:     Calculate the upper bound of the self-reconstruction loss.
           $\hat{\mathcal{L}}_{self} = -\log P_{G_{\mathcal{X}}}(X|\widetilde{X}, T^*_{X,self}) - \log P_{G_{\mathcal{X}}}(T^*_{X,self}|\widetilde{X})$
           $\quad - \log P_{G_{\mathcal{Y}}}(Y|\widetilde{Y}, T^*_{Y,self}) - \log P_{G_{\mathcal{Y}}}(T^*_{Y,self}|\widetilde{Y})$
6:     Generate transferred samples with gradient approximation methods:
           $T_{\mathcal{Y}} \sim P_{G_{\mathcal{Y}}}(T|X), \quad G_{\mathcal{Y}}(X) \sim P_{G_{\mathcal{Y}}}(Y|X, T_{\mathcal{Y}}).$
           $T_{\mathcal{X}} \sim P_{G_{\mathcal{X}}}(T|Y), \quad G_{\mathcal{X}}(Y) \sim P_{G_{\mathcal{X}}}(X|Y, T_{\mathcal{X}}).$
7:     Calculate the style loss.
           $\mathcal{L}_{sty} = -\mathbb{E}_{X\sim P_{\mathcal{X}}}[F(G_{\mathcal{Y}}(X), \mathcal{Y})] - \mathbb{E}_{Y\sim P_{\mathcal{Y}}}[F(G_{\mathcal{X}}(Y), \mathcal{X})].$
8:     Use Algorithm 1 to obtain pseudo alignments for the cycle loss:
           $T^*_{X,cyc} = DP(G_{\mathcal{Y}}(X), X), \quad T^*_{Y,cyc} = DP(G_{\mathcal{X}}(Y), Y).$
9:     Calculate the upper bound of the cycle loss.
           $\hat{\mathcal{L}}_{cyc} = -\log P_{G_{\mathcal{X}}}(X|G_{\mathcal{Y}}(X), T^*_{X,cyc}) - \log P_{G_{\mathcal{X}}}(T^*_{X,cyc}|G_{\mathcal{Y}}(X))$
           $\quad - \log P_{G_{\mathcal{Y}}}(Y|G_{\mathcal{X}}(Y), T^*_{Y,cyc}) - \log P_{G_{\mathcal{Y}}}(T^*_{Y,cyc}|G_{\mathcal{X}}(Y))$
10:    Update the model with the loss $\mathcal{L} = \alpha\hat{\mathcal{L}}_{self} + \beta\mathcal{L}_{sty} + \gamma\hat{\mathcal{L}}_{cyc}$.
11: end for
```

| Dataset | Styles | #Train | #Valid | #Test | $\|V\|$ | Avg Len |
|---|---|---|---|---|---|---|
| Yelp | Neg. | 177k | 2,000 | 500 | 9,943 | 9.55 |
| | Pos. | 266k | 2,000 | 500 | | 8.43 |
| GYAFC | Inf. | 52k | 2,788 | 1,332 | 26,790 | 13.06 |
| | For. | 52k | 2,247 | 1,019 | | 12.47 |

Table 10: Data statistics. Average length is calculated on the training set.

then optimized to train the model. The full training algorithm is presented in Algorithm 2.

## B Experiment Settings

### B.1 Dataset and Evaluation Metrics

We use the processed datasets provided by Luo et al. (2019), which can be downloaded at `https://github.com/luofuli/DualRL`. The data statistics are shown in Table 10.

The pretrained classifier is implemented based on the *transformers* package[4], and the BLEU-4 score is the corpus BLEU implemented in the *nltk* package[5]. All results in our paper are evaluated by our implemented codes. The reported results of NAST, StyTrans, and LatentSeq in Figure 1 are averaged over three runs with different random seeds.

[4]`https://github.com/huggingface/transformers`

[5]`https://www.nltk.org/`

### B.2 Network Architecture and Hyper-Parameters

NAST are implemented based on the base model, StyTrans (Dai et al., 2019) and LatentSeq (He et al., 2020). Their codes can be accessed at `https://github.com/fastnlp/style-transformer` and `https://github.com/cindyxinyiwang/deep-latent-sequence-model`.

For StyTrans, we follow their implementation and hyper-parameters for the Transformer architecture. We use 4 Transformer layers, 4 attention heads, and 256-dim hidden cells for both the encoder and the decoder. For the alignment predictor in Learnable Alignment, we utilize a one-layer Transformer decoder with the same number of attention head and dimension of hidden cells. Moreover, StyTrans utilizes a discriminator for the style loss, which is built on a 4-layer Transformer encoder with the same architecture above. The discriminator and the generator are trained adversarially. Following their implementation, in each iteration, the discriminator is trained for 10 steps and then the generator is trained for 5 steps. We utilize the Adam optimizer (Kingma and Ba, 2015) with the learning rate of $1e-4$ and the batch size of 64. We choose the gradient approximation method from the Gumbel-Softmax trick (Jang et al., 2017), the Soft-Embedding approximation (Dai et al., 2019), and the Stop-Gradient strategy

(He et al., 2020). We select the self-reconstruction loss weight $\alpha$ from $\{0.25, 0.5, 1\}$ and the cycle loss weight $\gamma$ from $\{0.25, 0.5, 1\}$. We find sometimes the transfer accuracy of one direction can be much higher than that of the other direction, so we separately tune the style loss weights for two directions. To be specific, the overall objective is defined as $\alpha L_{self} + \beta_1 L_{X,sty} + \beta_2 L_{Y,sty} + \gamma L_{cyc}$, where $L_{X,sty} = -\mathbb{E}_{X\sim P_{\mathcal{X}}}[F(G_{\mathcal{Y}}(X), \mathcal{Y})]$, and $L_{Y,sty} = -\mathbb{E}_{Y\sim P_{\mathcal{Y}}}[F(G_{\mathcal{X}}(Y), \mathcal{X})]$. We select $\beta_1$, $\beta_2$ from $\{0.5, 1, 1.5, 3, 5, 10, 15\}$.

For LatentSeq, LSTM is adopted as the generator in their original models. We first replace the LSTM with an autoregressive Transformer as a baseline, which also has 4 Transformer layers, 4 attention heads, and 256-dim hidden cells. Then we replace the autoregressive Transformer with an non-autoregressive Transformer with the same architecture. The alignment predictor is a one-layer Transformer decoder with the same architecture above. However, LatentSeq utilizes a language model for the style loss, which is a 512-dim LSTM. We preserve the implementation of the language model. For optimization, we utilize the RAdam optimizer (Liu et al., 2020) with the learning rate of $1e-3$ and the batch size of 64. We also try the three gradient approximation methods. We set the cycle-reconstruction loss weight $\gamma = 1$. Following their original implementation, the self-reconstruction weight $\alpha$ is annealed from 1 to 0 in the first 60k steps. Similar to NAST on StyleTrans, we tune the the style loss weight on two directions separately, where we select $\beta_1, \beta_2$ from $\{0.15, 0.3, 0.45, 0.6, 0.75\}$ for Yelp and $\{0.5, 0.75, 1, 1.25\}$ for GYAFC.

We manually tune the hyper-parameters and select the best model according the performance on the validation set. For the Yelp dataset, the validation set does not have reference answers, so we use the geometric mean of Acc and SelfB as the overall performance. For the GYAFC dataset, we use the geometric mean of Acc and RefB as the overall performance.

### B.3 Computing Devices and Running Time

In our experiment, each run uses approximately 4 Intel Xeon Gold 6226R CPUs at 2.90GHz, and 1 Nvidia Quadro RTX 6000 GPU. We present the max training step and the training time in Table 11. The best results usually appear in the first half of the training.

| Model | Yelp | GYAFC |
|---|---|---|
| StyTrans | | |
| + NAST(Simple) | 135k steps ($\sim$12h) | 135k steps ($\sim$16h) |
| + NAST(Learnable) | 135k steps ($\sim$16h) | 135k steps ($\sim$25h) |
| LatentSeq | | |
| + NAST(Simple) | 150k steps ($\sim$18h) | 75k steps ($\sim$16h) |
| + NAST(Learnable) | 150k steps ($\sim$24h) | 75k steps ($\sim$18h) |

Table 11: The max training step and the training time of our models.

| Yelp | | | | | | |
|---|---|---|---|---|---|---|
| | **Negative to Positive** | | | **Positive to Negative** | | |
| **Model** | **Acc** | **RefB** | **G2** | **Acc** | **RefB** | **G2** |
| DualRL | 85.4 | 49.6 | 65.1 | 85.8 | 60.8 | 72.3 |
| StyTrans | **87.5** | 45.4 | 63.0 | **93.1** | 51.6 | 69.3 |
| **+ NAST (Simple)** | 86.2 | **50.1** | **65.7** | 91.6 | **61.6** | **75.1** |
| **+ NAST (Learnable)** | 84.2 | 49.2 | 64.3 | 90.7 | 60.0 | 73.8 |
| LatentSeq | **82.3** | 42.8 | 59.3 | **86.7** | 51.8 | 67.0 |
| **+ NAST (Simple)** | **82.3** | 49.0 | **63.5** | 80.1 | **66.1** | **72.7** |
| **+ NAST (Learnable)** | 80.5 | **50.2** | **63.5** | 78.7 | 65.8 | 72.0 |

| GYAFC | | | | | | |
|---|---|---|---|---|---|---|
| | **Formal to Informal** | | | **Informal to Formal** | | |
| **Model** | **Acc** | **RefB** | **G2** | **Acc** | **RefB** | **G2** |
| DualRL | 86.6 | 24.7 | 46.2 | 31.2 | 55.9 | 41.7 |
| StyTrans | 87.2 | 28.4 | 49.7 | 46.9 | 55.5 | 51.0 |
| **+ NAST (Simple)** | 85.4 | 28.5 | 49.3 | 49.8 | 54.7 | 52.2 |
| **+ NAST (Learnable)** | **92.1** | **29.9** | **52.5** | **53.7** | **55.7** | **54.7** |
| LatentSeq | 56.8 | 24.7 | 37.2 | 49.8 | 52.4 | 51.1 |
| **+ NAST (Simple)** | 62.5 | **27.4** | 41.4 | 58.3 | 49.0 | 53.4 |
| **+ NAST (Learnable)** | **69.1** | 25.8 | **42.1** | **59.2** | **52.6** | **55.8** |

Table 12: Automatic evaluation results on two transfer directions.

## C Transfer Difficulties

In Table 12, we present the results on two transfer directions of Yelp and GYAFC. On the Yelp dataset, transferring a negative sentence to a positive one is more difficult than the other direction. One possible reason is that the negative sentences are euphemistic and need changes in sentence structures when transferring to the positive sentiment. In terms of G2, the text formalization is significantly more difficult than the sentiment transfer. The difficulties of two transfer directions vary across models on GYAFC. Transferring formal sentences to informal ones is harder for DualRL, while the other direction is harder for LatentSeq.

## D How to Count Aligned Word Pairs

In Section 4.5, we present cases of the word-level transfer. The aligned word pairs are counted based on the predict alignments $T$, following the rules below:

- If $1 \leq t_i \leq N$, we record a pair $x_{t_i} \to y_i$.
- If $t_i = 0$, the transferred word is unaligned, and we record a pair $[Mask] \to y_i$.
- If a source word $x_i$ is not aligned with any transferred word, we record a pair $x_i \to [Del]$.

We then collect all word pairs that have the same source word and calculate the proportion of different transferred words. The results shown in Table 8 is obtained on the test set of two datasets.