Syntactically-Informed Unsupervised Paraphrasing with Non-Parallel Data

Erguang Yang[1]*, **Mingtong Liu**[1], **Deyi Xiong**[2]*, **Yujie Zhang**[1], **Yao Meng**[3], **Changjian Hu**[3], **Jinan Xu**[1], **Yufeng Chen**[1]
[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]College of Intelligence and Computing, Tianjin University, Tianjin, China
[3]Lenovo Research AI Lab, Beijing, China
{egyang,mingtongliu,yjzhang,jaxu,chenyf}@bjtu.edu.cn,
dyxiong@tju.edu.cn, {yaomeng1,hucj1}@lenovo.com

Abstract

Previous works on syntactically controlled paraphrase generation heavily rely on large-scale parallel paraphrase data that are not easily available for many languages and domains. In this paper, we take this research direction to the extreme and investigate whether it is possible to learn syntactically controlled paraphrase generation with non-parallel data. We propose a syntactically-informed unsupervised paraphrasing model based on conditional variational auto-encoder (VAE) which can generate texts in a specified syntactic structure. Particularly, we design a two-stage learning method to effectively train the model using non-parallel data. The conditional VAE is trained to reconstruct the input sentence according to the given input and its syntactic structure. Furthermore, to improve the syntactic controllability and semantic consistency of the pre-trained conditional VAE, we fine-tune it using syntax controlling and cycle reconstruction learning objectives, and employ Gumbel-Softmax to combine these new learning objectives. Experiment results demonstrate that the proposed model trained only on non-parallel data is capable of generating diverse paraphrases with specified structures. Additionally, we further validate the effectiveness of our method for generating syntactically adversarial examples on a sentiment analysis task. Source codes are available at https://github.com/lanse-sir/sup.

1 Introduction

Paraphrases are texts or passages conveying the same meaning but with different surface realization. Paraphrase generation (PG) is a key technology of automatically generating a restatement for a given text, which has the potential use in many downstream tasks, such as question answering (Dong et al., 2017), machine translation (Zhou et al., 2019) and text summarization (Zhao et al., 2018).

Recent years have witnessed that learning controllable paraphrase generation (CPG) with specified styles is attracting intense research interests, e.g., satisfying particular syntactic templates (Iyyer et al., 2018) or exemplars (Chen et al., 2019; Kumar et al., 2020). As CPG can produce diverse paraphrases by exposing syntactic control, it can be also employed for adversarial example generation (Iyyer et al., 2018).

Existing syntactically controlled paraphrase networks (Iyyer et al., 2018) rely on large paraphrase parallel data for training. Unfortunately, paraphrase parallel corpora are not easily available for many languages, and are expensive to build. Conversely, non-parallel data is much easier to find, and many languages with limited parallel data still possess a huge amount of non-parallel data.

In this paper, we propose a **S**yntactically-informed **U**nsupervised **P**araphrasing (SUP) framework based on conditional variational auto-encoder (VAE) to generate syntactic paraphrases with specified syntactic skeletons, which does not require any parallel paraphrase data. The basic assumption behind SUP is that, given a sentence, there may exist many valid paraphrases with different syntactic structures. Specifically, as shown in Figure 1, SUP runs in two stages. At stage 1, we train a conditional VAE to reconstruct a given input sentence according to the sentence itself and its syntactic parse tree. The model trained at this stage is endowed with basic ability to generate texts of desired syntax structures (similar to a warmup procedure). At stage 2, to improve the syntactic controllability and semantic consistency of generated sentences, we fine-tune the model trained at stage 1 using carefully-designed objective functions involving syntax controlling and cycle reconstruction. After the conditional VAE model is fine-tuned, given an input sentence and a different syntactic structure, the model can generate a paraphrase according to the given structure.

*Corresponding Authors.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2594–2604
November 7–11, 2021.

We evaluate SUP on both syntactic paraphrase generation and adversarial example generation tasks. Experiments show that SUP outperforms previous unsupervised paraphrasing method SIVAE (Zhang et al., 2019). It is also capable of generating syntactically adversarial examples that have a significant impact on the performance of attacked neural models. We further show that augmenting training data with such examples can improve the robustness of target neural models.

In summary, the major contributions of this paper are as follows:

- We propose a syntactically-informed unsupervised paraphrasing model based on conditional VAE framework and use it to generate syntactically adversarial examples.
- To enable the model to generate syntactically-controlled paraphrases, we propose a novel tree encoder to effectively model structure information and a syntax controlling learning objective to further improve syntactic controllability. Meanwhile, we also introduce a cycle reconstruction learning objective to preserve the semantics of the input sentence.
- Experiments show that our model can successfully generate syntactically adversarial examples. By augmenting training data with such examples, we can improve the robustness of target neural models.

2 Related Work

Paraphrase Generation The task of paraphrase generation has recently received significant attention (Li et al., 2018, 2019; Liu et al., 2020a). Previous works mainly explore supervised paraphrasing methods, which require large corpora of parallel sentences for training. Due to the lack of parallel data, unsupervised paraphrasing has become an emerging research direction (Miao et al., 2018; Liu et al., 2020c). However, these methods mainly rely on lexical changes to generate paraphrases. Compared to these approaches, our work focus primarily on the syntactically controlled paraphrase generation, which is able to generate a paraphrase according to a given syntactic structure.

Controlled Text Generation Recent works on controlled generation aim at controlling attributes such as sentiment (Hu et al., 2017; John et al., 2019; Dai et al., 2019). These works use a categorical feature as a controlling signal. Different from them, we use a more complicated, non-categorical syntactic structure as a controlling signal. To ensure syntactic controllability, we design a tree encoder and syntax controlling loss to encourage the model to generate sentences that conform to given syntax.

We have also witnessed other works that attempt to control structural aspects of the generation, such as studies using a given syntactic form (Iyyer et al., 2018; Chen et al., 2019; Liu et al., 2020b). Our work is closely related to this category, and to the syntactically-controlled paraphrase networks (SCPN) proposed by Iyyer et al.(2018) in particular. They use the attentional seq2seq framework to build a parse generator and a paraphrase generator. A two-stage generation process is used. In the first stage, they generate full parse trees from syntactic templates, and then produce final generations in the second stage. Both parse and paraphrase generator require parallel data for training. Significantly different from their method, our model based on conditional VAE is an unsupervised method that does not require any parallel data for training.

Conditional Variational Autoencoder Our work is also related to syntax-infused text generation (Bao et al., 2019; Zhang et al., 2019). Their models use two variational autoencoders to introduce two latent variables which are designed to capture semantic and syntactic information. The variational autoencoder (VAE) network is proposed by Kingma and Welling (2014) for image generation. Bowman et al. (2016) successfully apply VAE in fluent sentence generation from a latent space. The conditional VAE is a modification of VAE to generate diverse images conditioned on certain attributes, e.g. generating different human faces given skin color (Sohn et al., 2015; Yan et al., 2016). Inspired by conditional VAE, we view the syntactic structure as the conditional attribute and adopt conditional VAE to generate syntactic paraphrases. Furthermore, to improve the syntactic controllability and semantic consistency of generated sentences, we use syntax controlling and cycle reconstruction objective functions to fine-tune the model.

Adversarial Example Generation To generate adversarial examples for NLP models, most previous works rely on injecting noise either at the character level (Ebrahimi et al., 2018; Gao et al., 2018) or at the word level by adding and deleting

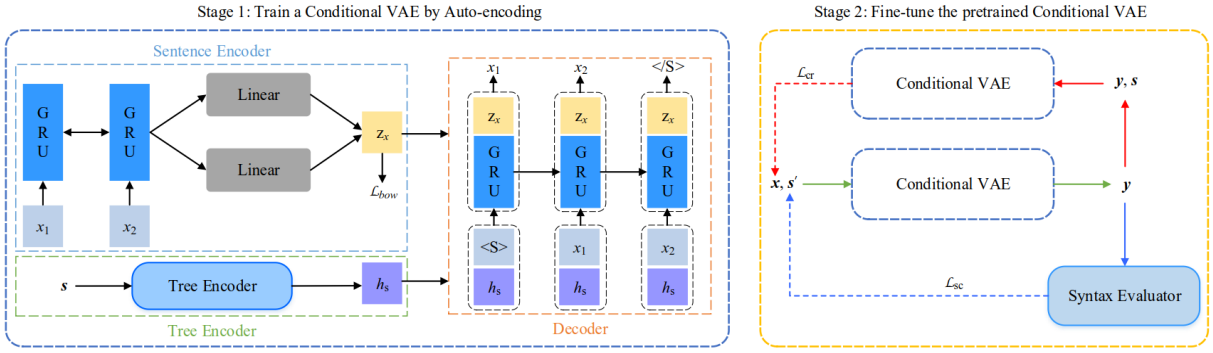

Figure 1: Architecture of the proposed syntactically-informed unsupervised paraphrasing model. **Stage 1**: Training a Conditional VAE model by reconstructing the input sentence given the sentence itself and its syntax structure. Here we simply take $\boldsymbol{x} = \{x_1, x_2\}$ as an example. **Stage 2**: Fine-tuning the model using novel objective functions. $\boldsymbol{x}$, $\boldsymbol{s}$, $\boldsymbol{s}'$(different from $\boldsymbol{s}$), and $\boldsymbol{y}$ denote the input sentence, its syntactic structure, other syntactic structure, and output sentence, respectively. $\mathcal{L}_*$ denote the loss terms.

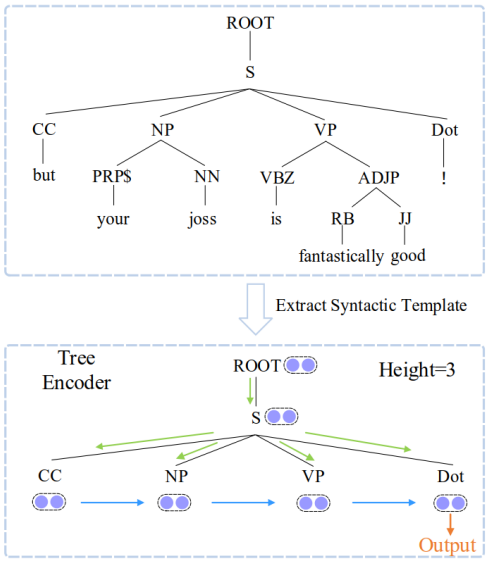

Figure 2: The upper part shows a constituency parse tree. The lower part visualizes the tree encoder that uses top-down (green line) and left-to-right (blue line) directions to encode a syntactic template (top three level).

words (Liang et al., 2017; Garg and Ramakrishnan, 2020). In this paper, we generate syntactically adversarial examples, which still remains an open challenge, as semantic meaning of these examples should be preserved despite of their substantial structural changes.

3 Approach

We use the constituency parse tree to provide syntactic information. Given a set of training instances $D = \{(\boldsymbol{x_i}, \boldsymbol{s_i})\}_{i=1}^{|D|}$, where $\boldsymbol{s_i}$ is the syntactic parse tree of the sentence $\boldsymbol{x_i}$, we aim to train a syntactic paraphrasing model which can produce more diverse paraphrases given arbitrary syntax.

However, using a full parse tree (the whole tree without leaf nodes) is too specific and poses the challenge of selecting such a tree for a given input as syntactic structures of two different sentences are not easily compatible to each other. Therefore, we mainly use a general template (the top 3 layers of a parse tree), as shown in Figure 2, as the controlling signal which is beneficial to generate meaningful paraphrases.

Specifically, we employ the conditional variational autoencoder (VAE) framework which have proven to be able to generate diverse texts conditioned on certain attributes. In this work, we view syntax as the conditional attribute. The training process consists of two stages. In the first stage, we train the model in an auto-encoding manner, while in the second stage, we use new objective functions to fine-tune it, as shown in Figure 1. The two stages will be described in detail below.

3.1 Stage 1: Training a Conditional VAE

At this stage, we pre-train the conditional VAE model. The model is required to reconstruct the input sentence given the sentence itself and its syntactic template. In doing so, the model acquires the preliminary ability to generate a desired sentence conditioning on given syntactic template, which makes the training in the subsequent stage easier.

Sentence Encoding Given a sentence $\boldsymbol{x}$, we first obtain the sentence hidden-state $\boldsymbol{h}_x = [\overrightarrow{\boldsymbol{h}}_{|\boldsymbol{x}|}; \overleftarrow{\boldsymbol{h}}_1]$ by the sentence encoder. For the semantic variable $\boldsymbol{z}_x$, we compute the mean and variance of $q(\boldsymbol{z}_x|\boldsymbol{x})$

from $\boldsymbol{h}_x$ as:

$$\begin{aligned} \boldsymbol{\mu}_x &= \boldsymbol{W}_x^{\mu}\boldsymbol{h}_x + \boldsymbol{b}_x^{\mu} \\ \log \boldsymbol{\sigma}_x^2 &= \boldsymbol{W}_x^{\sigma}\boldsymbol{h}_x + \boldsymbol{b}_x^{\sigma} \end{aligned} \tag{1}$$

where $\boldsymbol{W}_x^{\mu}$, $\boldsymbol{b}_x^{\mu}$, $\boldsymbol{W}_x^{\sigma}$, $\boldsymbol{b}_x^{\sigma}$ are trainable parameters.

Syntax Encoding This encoder provides the necessary syntactic guidance for the generation of paraphrases. Formally, let syntactic template $\boldsymbol{s} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ the set of edges.

As shown in Figure 2, we traverse the given syntactic template in a top-down (green line) and left-to-right (blue line) manner to obtain and model parent-child and sibling relationships, respectively. For the top-down (TD) direction, we encode each node in a depth-first manner. Specifically, the representation $\boldsymbol{h}_v$ of each node $v \in \mathcal{V}$ with the hidden-state representation of its parent node $pa(v)$ and its embedding as follows:

$$\boldsymbol{h}_v = GRU(\boldsymbol{e}(v), \boldsymbol{h}_{pa(v)}) \tag{2}$$

where $\boldsymbol{e}(v)$ is the embedding of the node v. Although we can obtain TD representations of all nodes of the syntactic template, only the TD representations of leaf nodes will be used for left-to-right encoding. For the particular example given in Figure 2, the TD representations of all leaf nodes are $H_{leaf}^{TD} = [\boldsymbol{h}_{CC}^{TD}, \boldsymbol{h}_{NP}^{TD}, \boldsymbol{h}_{VP}^{TD}, \boldsymbol{h}_{Dot}^{TD}]$.

For the left-to-right (LR) encoding, the encoder is a forward GRU network. We take the leaf nodes sequence $Leaf_{seq} = \{CC, NP, VP, Dot\}$ as input, and compute the LR representations of all leaf nodes $H_{leaf}^{LR} = [\boldsymbol{h}_{CC}^{LR}, \boldsymbol{h}_{NP}^{LR}, \boldsymbol{h}_{VP}^{LR}, \boldsymbol{h}_{Dot}^{LR}]$. Particularly, take the `NP` node as an example:

$$\boldsymbol{h}_{NP}^{LR} = GRU(\boldsymbol{h}_{NP}^{TD}, \boldsymbol{h}_{CC}^{LR}) \tag{3}$$

Then, we use the last hidden state of the syntactic encoder $\boldsymbol{h}_{Dot}^{LR}$ as the final syntax representation $\boldsymbol{h}_s$ for providing the syntactic signal to the decoder.

Decoding in the Training Phase We employ the reparameterization trick to obtain semantic variables $\boldsymbol{z}_x = \boldsymbol{\mu}_x + \boldsymbol{\sigma}_x \odot \varepsilon$, $\varepsilon \in \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Then at each time step, we concatenate the syntactic representation $\boldsymbol{h}_s$ with the previous word's embedding as the input to the decoder and concatenate the semantic variable $\boldsymbol{z}_x$ with the hidden-state output by the decoder for predicting the word at next time step, as shown in stage 1 in Figure 1. Note that the initial hidden state of the decoder is set to zero.

Decoding in the Test Phase Giving the same sentence but with a different syntactic template, the model can generate a syntactically controlled paraphrase. We obtain semantic variable $\boldsymbol{z}_x$ by the maximum a posteriori (MAP) inference. In this way, semantic information from the input sentence could be preserved as much as possible. After that, the decoding process is the same as the training phase.

The Objective Function To train the above model, we optimize the following objective function:

$$\mathcal{L}_1 = \mathcal{L}_{cvae} + \lambda_{bow}\mathcal{L}_{bow} \tag{4}$$

where $\mathcal{L}_{cvae}$ and $\mathcal{L}_{bow}$ denote the conditional VAE loss and bag-of-word loss, respectively. λ_{bow} is a hyper-parameters for balancing the two losses. .

Conditional VAE Loss: The loss is used to optimize the conditional VAE model by minimizing the reconstruction loss $\mathcal{L}_{rec}$, and meanwhile minimizing the KL loss $\mathcal{L}_{kl}$ to encourage the posterior $q(\boldsymbol{z}_x|\boldsymbol{x})$ to match the prior $p(\boldsymbol{z}_x)$:

$$\begin{aligned} \mathcal{L}_{cvae} =& -\lambda_{res}\mathcal{L}_{rec} + \lambda_{kl}\mathcal{L}_{kl} \\ =& -\lambda_{res}\log p(\boldsymbol{x}|\boldsymbol{z}_x, \boldsymbol{h}_s) \\ &+ \lambda_{kl}KL(q(\boldsymbol{z}_x|\boldsymbol{x}) \parallel p(\boldsymbol{z}_x)) \end{aligned} \tag{5}$$

where $p(\boldsymbol{z}_x)$ follows standard normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, $q(\boldsymbol{z}_x|\boldsymbol{x})$ takes the form $\mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2)$. Here, $\boldsymbol{\mu}_x$, $\boldsymbol{\sigma}_x$ are computed by Eq. (1). λ_* are balancing hyper-parameters.

Bag-of-Word Loss: We introduce the Bag-of-word loss to enhance content preservation during paraphrase generation. Specifically, we take $\boldsymbol{z}_x$ as input and predicts the Bag-of-Word distribution:

$$\boldsymbol{p}_b = sigmoid(\boldsymbol{W}_{bow}\boldsymbol{z}_x + \boldsymbol{b}_{bow}) \tag{6}$$

where $\boldsymbol{W}_{bow}$, $\boldsymbol{b}_{bow}$ are trainable parameters. The bag-of-word loss is computed as follows:

$$\mathcal{L}_{bow} = -\sum_{w \in V} t_w \log \boldsymbol{p}_b(w) \tag{7}$$

where V denotes the word vocabulary, $\boldsymbol{t}$ is the bag-of-word ground-truth distribution of the corresponding sentence.

3.2 Stage 2: Fine-tuning the Conditional VAE Model

During inference, we will give different syntactic structures for every input sentence to generate paraphrases. To encourage generalization on different

syntactic structures, we fine-tune the pre-trained conditional VAE in a cycle learning manner.

Specifically, as shown in stage 2 in Figure 1, given an input sentence $\boldsymbol{x}$, its syntactic template $\boldsymbol{s}$, and other syntactic template $\boldsymbol{s'}$, we feed $\boldsymbol{x}$ and $\boldsymbol{s'}$ into the conditional VAE model to generate sentence $\boldsymbol{y}$ (green line). We compute syntax controlling (blue line) and cycle reconstruction losses (red line), and then fine-tune the model to generate a better sentence that is formed in the syntactic structure of $\boldsymbol{s'}$ and preserves the semantic meaning of $\boldsymbol{x}$.

Syntax Controlling Loss First, we build a GRU-based seq2seq neural parser as the evaluator, which is pre-trained on the above mentioned training data D, with $\boldsymbol{x}$ as the input and the linearized syntactic template $\boldsymbol{s}$ as the decoding target. For example, the linearized syntactic template in Figure 2 is `(ROOT(S(CC)(NP)(VP)(Dot))).`

Second, we apply the pre-trained evaluator[1] to predict the linearized syntactic structure of the output sentence $\boldsymbol{y}$, where parameters of the conditional VAE are updated to encourage the target syntactic template $\boldsymbol{s'}$ to be predicted from the output sentence, i.e., minimizing the following term:

$$\mathcal{L}_{sc} = -\log p_{\texttt{eval}}(s'_l|GS(\boldsymbol{y})) \tag{8}$$

where s'_l is the linearized $\boldsymbol{s'}$. $GS(\boldsymbol{y})$ denotes a "softly" generated sentence based on Gumbel-Softmax distribution (Jang et al., 2016), where the representation of each word is defined as the weighted sum of word embeddings with the prediction probability at the current timestep. Please notice that the parameters of the evaluator are not updated in this step.

Cycle Reconstruction Loss However, only using the above syntax controlling loss will result in generating a sentence that conforms to the target syntactic structure but drifts away from the original meaning. To address this issue, we borrow the cycle reconstruction loss $\mathcal{L}_{cr}$ from style-transfer research (Hu et al., 2017; Dai et al., 2019) to encourage the generated sentence to preserve the meaning in the input sentence.

We feed the generated sentence $\boldsymbol{y}$ and the syntactic template $\boldsymbol{s}$ of $\boldsymbol{x}$ to the conditional VAE and update the model to reconstruct original input sentence $\boldsymbol{x}$ by minimizing the following term:

$$\mathcal{L}_{cp} = -\log p(\boldsymbol{x}|GS(\boldsymbol{y}), \boldsymbol{s}) \tag{9}$$

where $GS(\boldsymbol{y})$ is is the same as in Eq. (8).

The Objective Function The final loss function for fine-tuning is defined as follow:

$$\mathcal{L}_2 = \mathcal{L}_{cvae} + \lambda_{sc}\mathcal{L}_{sc} + \lambda_{cr}\mathcal{L}_{cr} \tag{10}$$

In our experiments, we still optimize $\mathcal{L}_{cvae}$ during the fine-tuning stage, which helps to stabilize the training process. λ_* are balancing hyper-parameters.

4 Experiments

In this section, we will answer the following questions:

- First, we investigate whether our model can generate syntactically controlled paraphrases.
- Second, we examine whether our model can generate syntactically adversarial examples for sentiment analysis.

4.1 Syntactically-Informed Paraphrase Generation

Given an input sentence, a syntactically-informed paraphrase is a sentence with the same meaning as the input sentence but in a different syntactic structure defined by a given syntactic structure.

4.1.1 Models for Comparison

We compared with the following unsupervised paraphrase models: 1) **VAE**: a vanilla variational autoencoder (Bowman et al., 2016) as a simple baseline; 2) **SIVAE**: a syntax-infused variational autoencoder (Zhang et al., 2019) that dutilizes additional syntax information to improve the quality of sentence generation and paraphrase generation, where syntax information is provided by a linearized parse tree.

We also compared against the supervised method SCPN (Iyyer et al., 2018) which uses an extended pointer-generator network (See et al., 2017) to encode input sentences and linearized parse trees to generate paraphrases.

4.1.2 Datasets

Quora. The dataset contains 140k pairs of paraphrase sentences and 260k pairs of non-paraphrase sentences. In the standard dataset split, there are 3k

[1]On the Quora and ParaNMT test set, the evaluator achieves 90% parsing accuracy.

Model	Quora					ParaNMT				
	ESM(↑)	i-BLEU(↑)	BLEU-ref (↑)	BLEU-ori (↓)	S-BERT(↑)	ESM(↑)	i-BLEU(↑)	BLEU-ref (↑)	BLEU-ori (↓)	S-BERT(↑)
Original Sentence	56.5	1.1	31.1	100	0.845	36.9	3.5	18.5	100	0.755
VAE	57.0	5.5	23.4	59.5	0.764	34.6	3.3	10.1	45.4	0.643
Using full parse tree										
SCPN-F (supervised)	**94.7**	**56.6**	**64.3**	**25.5**	**0.866**	**97.0**	**53.8**	**56.7**	**19.0**	**0.866**
SIVAE-F	81.7	23.9	32.6	29.0	0.760	82.6	18.2	21.4	20.9	0.708
Stage1: SUP-F	87.5	33.9	43.7	32.7	0.809	89.2	32.8	33.1	20.7	0.747
Using syntactic template										
SCPN-T (supervised)	**90.8**	**12.1**	23.74	**38.85**	0.711	71.6	**11.2**	**18.4**	47.5	0.708
SIVAE-T	65.6	3.3	**26.9**	78.7	**0.802**	39.3	3.3	16.4	87.4	**0.733**
Stage1: SUP-T	73.9	7.3	22.46	50.55	0.738	65.9	4.4	9.6	34.3	0.610
Stage2: + $\mathcal{L}_{cvae}$	73.9	7.1	23.29	53.73	0.753	63.9	5.1	11.0	39.1	0.652
Stage2: + $\mathcal{L}_{cvae}$ + $\mathcal{L}_{sc}$	80.7	7.4	22.18	49.06	0.728	**75.9**	4.3	8.9	**30.4**	0.597
Stage2: + $\mathcal{L}_{cvae}$ + $\mathcal{L}_{sc}$ + $\mathcal{L}_{cr}$	78.0	7.7	22.93	50.6	0.755	72.9	5.1	10.1	33.0	0.639

Table 1: Performance of syntactic paraphrase generation. The larger↑ (or lower↓), the better. S-BERT indicates Sentence-BERT. ESM denotes the rate of exact syntactic match. -F, -T means using full parse tree and syntactic template as controlling signal, respectively.

Model	Quora	ParaNMT
original	is it possible to lose weight without doing exercise?	you know anybody who might wan na harm your husband?
reference	how can i loose weight naturally without doing exercise?	do you know anyone who would want to hurt your husband?
VAE	is it possible to lose weight without doing exercise?	who might know you 're gon na kill anybody?
SCPN-F	how can i loose weight naturally without doing exercise?	do you know anyone who might want to hurt your husband?
SIVAE-F	how can i lose weight loss without doing exercise?	do you know anyone who might gon na harm your husband?
SUP-F	how can i lose weight just without doing exercise?	do you know anyone who might want to harm your husband?
SCPN-T	how do i reduce weight without doing exercise?	do you know who might want to hurt your husband?
SIVAE-T	how is it possible lose weight without doing exercise?	do you wan who might wan na harm your husband?
SUP-T	how can i lose weight without doing exercise?	do you know who might harm your husband?

Table 2: Example paraphrases generated by each model.

and 30k paraphrase pairs in the held-out validation and test set, respectively. We followed the same unsupervised setting as (Miao et al., 2018; Bao et al., 2019), using non-paraphrase sentences as training instances that do not appear in the validation and test set. For the supervised method SCPN (Iyyer et al., 2018), we used paraphrase sentences for training.

ParaNMT. For the unsupervised setting, we randomly selected 5 million reference sentences from the ParaNMT-50M dataset to train unsupervised methods. The manually annotated 800 sentence pairs created by Chen et al. (2019) were used as our test set, and 500 for the development set. For the supervised method SCPN (Iyyer et al., 2018), we used their trained model.[2]

4.1.3 Evaluation Metrics

We employed original sentences and syntactic templates (or full parse trees) obtained from references as input, which is convenient for evaluation. But in the application scenario, we can give any syntactic templates to the trained model.

For semantic evaluation, we computed BLEU (Papineni et al., 2002) scores against the reference and original sentence, denoted as BLEU-ref and BLEU-ori, respectively. Additionally, we used i-BLEU (Sun and Zhou, 2012) to measure the diversity of expressions. We also used the embedding-based evaluation method Sentence-BERT[3] (Reimers and Gurevych, 2019) to evaluate the semantic similarity between the generated sentence and the reference sentence.

For syntactic evaluation, we evaluated how often generated paraphrases completely conform to the target syntactic templates by computing the rate of *exact syntactic match* (ESM): a paraphrase g is deemed as an exact syntactic match to reference r only if the top three levels of its parse tree p_g exactly matches those of p_r. The tuning of all hyper-parameter was based on the BLEU-ref score on the validation set.

4.1.4 Implementation Details

We parsed all sentences in the training set, the reference sentences in the validation and test set using Stanford CoreNLP (Manning et al., 2014). We used the Adam optimizer (Kingma and Ba, 2014) for optimization. For the training of Stage 1 and Stage 2, we set the learning rate to 5e-4 and 1e-4, respectively. The word embedding layer was initialized by the publicly available GloVe 300-dimensional

[2] https://github.com/miyyer/scpn

[3] We used the paraphrase-distilroberta-base-v1, which is trained on large-scale paraphrase data. Available at: https://public.ukp.informatik.tu-darmstadt.de/reimers/sentence-transformers/v0.2/

embeddings.[4] We adopted the tricks of KL annealing and word dropout following (Bowman et al., 2016). We set λ_{res} to 5, λ_{bow} to 0.5, λ_{sc} to 2.5, and λ_{cr} to 1.

We reimplemented VAE and SIVAE, and set the same KL weights for fair comparison.

4.1.5 Results

As shown in Table 1, results in the first row are computed over original sentences, which show a BLEU-ori socre of 100. We can see that all models achieve strong results when using full parse trees as syntactic control. This is because full parse trees contain more fine-grained syntactic information which guides the model to correctly substitute words with equivalents. With the setting of full parse trees, SUP (stage 1) outperforms the existing unsupervised methods in all metrics; With syntactic templates, we beat them in ESM and i-BLEU metrics. VAE and SIVAE-T tend to copy the input sentence as the output and therefore get low ESM but high Ori-BLEU scores.

Among our models, the SUP-T obtains an ESM of 73.9% and 65.9% on Quora and ParaNMT dataset, respectively. This shows that it can generate sentences according to the given syntactic templates (compared to row 1). At the stage 2, adding the conditional VAE loss leads to improvements in semantic metrics. Using conditional VAE loss and syntax controlling loss, we observe that while the syntactic accuracy has been greatly improved, the semantic metrics has decreased. Adding all loss terms leads to gains across both the semantic and syntactic metric scores. These results demonstrate the effectiveness of the proposed fine-tuning methods.

Even without using any parallel data, our model is competitive to the supervised SCPN trained on parallel data in some metrics. Especially, SUP-T (stage 2) achieves a higher S-BERT score than SCPN on the Quora dataset, a higher ESM score than SCPN on the ParaNMT Dataset.

Table 2 shows several paraphrases generated by each model. More generation results are presented in Appendix A. We can observe that SUP-F can produce better results than SIVAE-F in terms of both semantics and syntax. VAE and SIVAE-T tend to copy the source sentences. SUP-T can generate paraphrases syntactically similar to the reference.

[4] https://nlp.stanford.edu/projects/glove/

Model	**2**	**1**	**0**	ESM-H
SCPN-T	55.0	15.0	30.0	80.0
SIVAE-T	57.3	30.0	12.7	37.3
SUP-T	35.0	30.0	35.0	62.6

Table 3: Human evaluation on the Quora dataset (percentages of paraphrases scored **0**, **1** and **2**). ESM-H denotes the percentage that generations follow given syntactic templates.

KL-weight	BLEU-ref	BLEU-ori	ESM	S-BERT
Original	31.13	100	56.5	0.845
0.1	22.71	53.89	70.8	0.750
0.3	22.46	50.55	73.9	0.738
0.5	22.37	48.66	75.7	0.731
0.7	22.43	47.07	77.1	0.723
1.0	22.09	44.49	78.3	0.707
1.3	21.87	41.96	79.9	0.691
1.5	21.41	39.29	80.7	0.673

Table 4: BLEU-ref, BLEU-ori, ESM, and S-BERT score with varying KL weights on the Quora test set.

4.1.6 Human Evaluation

We also conducted human evaluation to measure paraphrase quality in a blind fashion. Following previous work (Iyyer et al., 2018; Goyal and Durrett, 2020), Three annotators were asked to evaluate the 100 randomly selected generations from the Quora test set according to a three-point scale scoring system: **0** denotes that the generated sentence is not a paraphrase at all; **1** means that the generated sentence is a paraphrase containing grammatical errors; **2** indicates that the generated sentence is a grammatically good paraphrase. Additionally, we also asked annotators to evaluate syntactic controllability (ESM-H): whether generations follow given syntactic templates.

Table 3 shows the results of human evaluation which are somewhat consistent with the automatic metrics. We notice that the quality of generations from SIVAE-T is better than that of the SCPN-T model. The reason is that SIVAE-T tends to copy input sentences as outputs. It also means that SIVAE-T cannot generate meaningful paraphrases (only copying inputs) according to given syntactic templates. SUP-T obtains comparable results with the SCPN if we consider paraphrases scored 2 and 1 as meaningful paraphrases. Additionally, most generations from SUP-T follow given target syntax.

4.1.7 Influence of KL-Weight on Results

We also analyzed the influence of different KL-weights on the SUP-T (stage 1) model. We can see in Table 4 that BLEU and ESM are more or less

Model	Valid(↑)	No augmentation		With augmentation	
		Acc(↑)	Failure(↑)	Acc(↑)	Failure(↓)
SCPN	72.2	84.6	29.2	83.3	21.3
SUP	68.0	84.6	28.0	83.3	25.0

Table 5: Performances of adversarial example generation, which are reported as the mean over three runs.

contradictory to each other. Usually, a smaller KL weight makes the autoencoder less "variational" but more "deterministic," leading to a lower syntactic match but better content preservation. In this experiment, to trade-off the content preservation and syntactic controllability, we set the KL weight to 0.3.

4.2 Adversarial Example Generation

We further examined the utility of controlled paraphrase generation for adversarial example generation. Following previous work (Iyyer et al., 2018), we evaluated our syntactically adversarial examples on the Stanford Sentiment Treebank Dataset (SST) (Socher et al., 2013). We generated 10 syntactically different paraphrases for each instance using the top 10 frequent syntactic templates and add them to the SST training set. Since we cannot generate a valid paraphrase for each syntactic template, we filtered generated paraphrases using a threshold (BLEU, 1-3gram) to remove nonsensical outputs. In this experiment, we set the threshold to 0.5.

4.2.1 Evaluation Metrics

We evaluated this task with the following metrics:

1. Dev Failure (Failure). We assume a development instance x as a prediction *failure* if the original prediction is correct, but the prediction for at least one paraphrase is incorrect. Dev Failure is the percentage of instances on the development set, which become prediction failures after paraphrasing.

2. Validity (Valid). To measure the validity of our adversarial examples, we perform manual evaluation on randomly selected 100 adversarial examples. We ask three workers to choose the appropriate label (e.g., positive or negative) for a given sentence, and then compare the worker's judgment to the original sentiment label.

3. Test Accuracy (Acc). It is used to measure the performance of sentiment classification models on the test set.

4.2.2 Implementation Details

We first pre-trained our model on preprocessed 2.1M sentences from the One-Billion-Word Corpus[5], and then fine-tuned our model on the SST dataset. For the pre-trained classification models, we used the bidirectional LSTM baseline in (Tai et al., 2015). The word embedding layer was initialized by the publicly available GloVe 300-dimensional embeddings. We used the Adam optimizer (Kingma and Ba, 2014) for optimization, and set the learning rate to 1e-4.

4.2.3 Results

As shown by Table 5, we obtain a validity score of 68.0 and Dev Failure score of 28.0. By augmenting the training data with paraphrases generated by our model, we obtain a lower Dev Failure score of 25.0. These results suggest that our model could generate legitimate adversarial examples. We improve the robustness of models against syntactic adversaries with little effect on the test accuracy.

We also observe that SCPN obtains strong results. This is because the model is trained on large-scale parallel data, and generated paraphrases include lexical and syntactic variations. However, these advantages are due to the use of large-scale parallel corpus. Our unsupervised method could be very effective for low-resource languages where no parallel data are available.

4.2.4 Case Study

Table 6 lists some paraphrases generated by SUP with different syntactic templates. Table 7 shows adversarial examples generated by our model. We find that the generated sentences always conform to the target templates. These generation results show that our model could generate legitimate adversarial examples. We also observe that the generated paraphrases have only syntactic variations, not lexical variations. This is because it is difficult for the model to learn to substitute words with equivalents only using non-parallel data. We leave enabling word-level or phrase-level variations in our model for creating more diverse adversarial examples to our future work.

5 Conclusions

We have presented an unsupervised syntactically-informed paraphrasing model based on conditional

[5] http://www.statmt.org/lm-benchmark/

Template	Paraphrase
original	still, as a visual treat, the film is almost unsurpassed.
(S (S) (,) (CC) (S))	the film is a visual treat, but almost unsurpassed.
(S (PP) (,) (NP) (VP))	as a visual treat, the film is almost unsurpassed.
(S (ADVP) (,) (NP) (VP))	still , the film is almost unsurpassed as a film.
original	it proves quite compelling as an intense, brooding character study.
(S (PP) (,) (NP) (VP) (.))	as compelling, it proves quite an intense character study.
(S (NP) (ADVP) (VP) (.))	it still proves compelling as an intense character study.
(S (CC) (NP) (VP))	but it proves compelling as an intense character study.
(S (ADVP) (,) (NP) (VP) (.))	however, it proves quite compelling as an intense.

Table 6: Paraphrases generated by SUP with different templates.

Template	Paraphrase	P
original	though only 60 minutes long, the film is packed with information and impressions.	✓
(S (NP) (VP))	the film is only 60 minutes long and packed with information and impressions.	✗
(S (S) (,) (CC) (S))	only 60 minutes long , and the film is packed with information.	✗
original	this film seems thirsty for reflection , itself taking on adolescent qualities.	✓
(S (NP) (VP))	this film seems thirsty for taking on adolescent qualities.	✗
(S (CC) (NP) (VP))	but this film seems thirsty for taking on adolescent qualities.	✗

Table 7: Adversarial examples generated by our model. ✓indicates that the prediction of the sentiment classifier model is correct, ✗indicates that the prediction is incorrect. NP: Noun Phrase, CC: Coordinating Conjunction.

VAE and two-stage training process. We first train the conditional VAE model to generate sentences in desired syntactic structures. To further improve the syntactic controllability and semantic consistency of generated sentences, we introduce syntax controlling and cycle reconstruction objective functions to fine-tune the pre-trained model. Experiments show that our model achieves strong improvements over baselines on unsupervised setting and can generate syntactically controlled paraphrases. Furthermore, adversarial example generation experiments also validate that our model is able to generate syntactically adversarial examples for sentiment analysis, which can be used to improve the robustness of the sentiment classifier model via adversarial training.

Acknowledgements

The present research was supported by the National Nature Science Foundation of China (No. 61876198, 61976015, 61976016). Deyi Xiong was partially supported by the National Key Research and Development Program of China (Grant No. 2019QY1802) and Natural Science Foundation of Tianjin (Grant No. 19JCZDJC31400). We would like thank the three anonymous reviewers for their constructive suggestions and insightful comments.

References

Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xin-yu Dai, and Jiajun Chen. 2019. Generating sentences from disentangled syntactic and semantic spaces. In *ACL*, pages 6008–6019. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *SIGNLL*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *ACL*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

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

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *EMNLP*, pages 875–886. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *ACL*. Association for Computational Linguistics.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial

text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based Adversarial Examples for Text Classification. page arXiv:2004.01970.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252, Online. Association for Computational Linguistics.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596. PMLR.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *NAACL*, pages 1875–1885. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

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

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Computer Science*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha Talukdar. 2020. Syntax-guided controlled generation of paraphrases. *TACL*, 8:329–345.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2018. Paraphrase generation with deep reinforcement learning. In *EMNLP*, pages 3865–3878, Brussels, Belgium. Association for Computational Linguistics.

Zichao Li, Xin Jiang, Lifeng Shang, and Qun Liu. 2019. Decomposable neural paraphrase generation. In *ACL*, pages 3403–3414, Florence, Italy. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*.

Mingtong Liu, Erguang Yang, Deyi Xiong, Yujie Zhang, Yao Meng, Changjian Hu, Jinan Xu, and Yufeng Chen. 2020a. A learning-exploring method to generate diverse paraphrases with multi-objective deep reinforcement learning. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 2310–2321. International Committee on Computational Linguistics.

Mingtong Liu, Erguang Yang, Deyi Xiong, Yujie Zhang, Chen Sheng, Changjian Hu, Jinan Xu, and Yufeng Chen. 2020b. Exploring bilingual parallel corpora for syntactically controllable paraphrase generation. In *IJCAI-20*, pages 3955–3961. International Joint Conferences on Artificial Intelligence Organization. Main track.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020c. Unsupervised paraphrasing by simulated annealing. In *ACL*, pages 302–312, Online. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *ACL*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2018. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *AAAI*, pages 1875–1885.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *EMNLP-IJCNLP*, Hong Kong, China. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *ACL*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Kihyuk Sohn, Xinchen Yan, and Honglak Lee. 2015. Learning structured output representation using deep conditional generative models. In *NIPS*.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual SMT system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42, Jeju Island, Korea. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, pages 776–791, Cham. Springer International Publishing.

Xinyuan Zhang, Yi Yang, Siyang Yuan, Dinghan Shen, and Lawrence Carin. 2019. Syntax-infused variational autoencoder for text generation. In *ACL*, pages 2069–2078, Florence, Italy. Association for Computational Linguistics.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *EMNLP*, pages 3164–3173, Brussels, Belgium. Association for Computational Linguistics.

Zhong Zhou, Matthias Sperber, and Alexander Waibel. 2019. Paraphrases as foreign languages in multilingual neural machine translation. In *ACL*, pages 113–122. Association for Computational Linguistics.