Syntax-Infused Variational Autoencoder for Text Generation

Xinyuan Zhang[1]*, **Yi Yang**[2]*, **Siyang Yuan**[1], **Dinghan Shen**[1], **Lawrence Carin**[1]

[1]Duke University
[2]ASAPP Inc.
`xy.zhang@duke.edu, yyang@asapp.com`

Abstract

We present a syntax-infused variational autoencoder (SIVAE), that integrates sentences with their syntactic trees to improve the grammar of generated sentences. Distinct from existing VAE-based text generative models, SIVAE contains two separate latent spaces, for sentences and syntactic trees. The evidence lower bound objective is redesigned correspondingly, by optimizing a joint distribution that accommodates two encoders and two decoders. SIVAE works with long short-term memory architectures to simultaneously generate sentences and syntactic trees. Two versions of SIVAE are proposed: one captures the dependencies between the latent variables through a conditional prior network, and the other treats the latent variables independently such that syntactically-controlled sentence generation can be performed. Experimental results demonstrate the generative superiority of SIVAE on both reconstruction and targeted syntactic evaluations. Finally, we show that the proposed models can be used for unsupervised paraphrasing given different syntactic tree templates.

1 Introduction

Neural language models based on recurrent neural networks (Mikolov et al., 2010) and sequence-to-sequence architectures (Sutskever et al., 2014) have revolutionized the NLP world. Deep latent variable modes, in particular, the variational autoencoders (VAE) (Kingma and Welling, 2014; Rezende et al., 2014) integrating inference models with neural language models have been widely adopted on text generation (Bowman et al., 2016; Yang et al., 2017; Kim et al., 2018), where the encoder and the decoder are modeled by long short-term memory

*Part of this work was done when the first two authors were at Bloomberg.
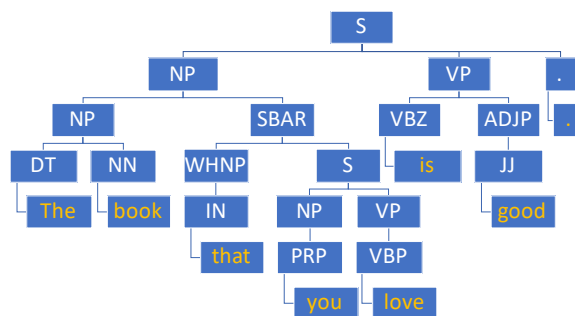

Figure 1: An example of a constituency tree structure.

(LSTM) networks (Chung et al., 2014). For a random vector from the latent space representing an unseen input, the decoder can generate realistic-looking novel data in the context of a text model, making the VAE an attractive generative model. Compared to simple neural language models, the latent representation in a VAE is supposed to give the model more expressive capacity.

Although syntactic properties can be implicitly discovered by such generative models, Shi et al. (2016) show that many deep structural details are still missing in the generated text. As a result of the absence of explicit syntactic information, generative models often produce ungrammatical sentences. To address this problem, recent works attempt to leverage explicit syntactic knowledge to improve the quality of machine translation (Eriguchi et al., 2016; Bastings et al., 2017; Chen et al., 2017) and achieve good results. Motivated by such success, we suggest that deep latent variable models for text generation can also benefit from the incorporation of syntactic knowledge. Instead of solely modeling sentences, we want to utilize augmented data by introducing an auxiliary input, a syntactic tree, to enrich the latent representation and make the generated sentences more grammatical and fluent. Syntactic trees can either be obtained from existing human-labeled

trees or syntactically parsed sentences using well-developed parsers. An example of a constituency tree is shown in Figure 1. In this work, we remove leaf nodes and linearize the bracketed parse structure into a syntactic tree sequence to simplify the encoding and decoding processes. For example, the syntactic tree sequence for the sentence "The book that you love is good." is `(S(NP(NP(DT)(NN))(SBAR(WHNP(IN))(S(NP(PRP))(VP(VBP)))))(VP(VBZ)(ADJP(JJ)))(.))`. Given such data, we aim to train a latent variable model that jointly encodes and decodes a sentence and its syntactic tree.

We propose a syntax-infused VAE model to help improve generation, by integrating syntactic trees with sentences. In contrast to the current VAE-based sentence-generation models, a key differentiating aspect of SIVAE is that we map the sentences and the syntactic trees into two latent representations, and generate them separately from the two latent spaces. This design decouples the semantic and syntactic representations and makes it possible to concentrate generation with respect to either syntactic variation or semantic richness. To accommodate the two latent spaces in one VAE framework, the evidence lower bound (ELBO) objective needs to be redesigned based on optimizing the joint log likelihood of sentences and syntactic trees. This new objective makes SIVAE a task-agnostic model, with two encoders and two decoders, so that it can be further used for other generative tasks.

Two variants of SIVAE that differ in the forms of the prior distributions corresponding to the syntactic tree latent variables are presented. SIVAE-c captures dependencies between two latent variables by making the syntax prior conditioned on the sentence prior. During generation, we first sample a latent variable from the sentence latent space and then sample the syntactic tree latent variable depending on the sampled sentence latent variable. This process resembles how humans write: think about substances like entities and topics first, then realize with a specific syntactic structure. We further propose SIVAE-i assuming the two priors are independent, and change the ELBO of the joint log likelihood correspondingly. This independence assumption manifests syntactically-controlled sentence generation as it allows to alter the syntactic structure, desirable for related tasks like paraphrase generation. Given a sentence and a syntactic tree template, the model produces a paraphrase of

the sentence whose syntax conforms to the template. Our SIVAE-based paraphrasing network is purely unsupervised, which makes it particularly suitable for generating paraphrases in low-resource languages or types of content.

The experiments are conducted on two datasets: one has trees labeled by humans and the other has trees parsed by a state-of-the-art parser (Kitaev and Klein, 2018). Other than employing the standard language modeling evaluation metrics like perplexity, we also adopt the targeted syntactic evaluation (Marvin and Linzen, 2018) to verify whether the incorporation of syntactic trees improves the grammar of generated sentences. Experiments demonstrate that the proposed model improves the quality of generated sentences compared to other baseline methods, on both the reconstruction and grammar evaluations. The proposed methods show the ability for unsupervised paraphrase generation under different syntactic tree templates.

Our contributions are four-fold: i) We propose a syntax-infused VAE that integrates syntactic trees with sentences, to grammatically improve the generated sentences. ii) We redesign the ELBO of the joint log likelihood, to accommodate two separate latent spaces in one VAE framework, for two SIVAE model variants based on different intuitions, which can be further used for other applications. iii) We evaluate our models on data with human-constituted trees or parsed trees, and yield promising results in generating sentences with better reconstruction loss and less grammatical errors, compared to other baseline methods. iv) We present an unsupervised paraphrasing network based on SIVAE-i that can perform syntactically controlled paraphrase generation.

2 Methodology

Given a sentence x and its corresponding syntactic tree y, the goal is to jointly encode x and y into latent representations $z_x \in \mathbb{R}^d$ and $z_y \in \mathbb{R}^d$, and then decode them jointly from the two latent spaces. We employ the VAE framework such that realistic-looking novel sentences can be generated with randomly sampled latent representations. However, current VAE-based language models cannot accommodate two separate latent spaces for z_x and z_y. To incorporate x, y, z_x, and z_y in one VAE framework, the objective needs to be redesigned to optimize the log joint likelihood $\log p(x, y)$. We propose two model variants of SIVAE. The first
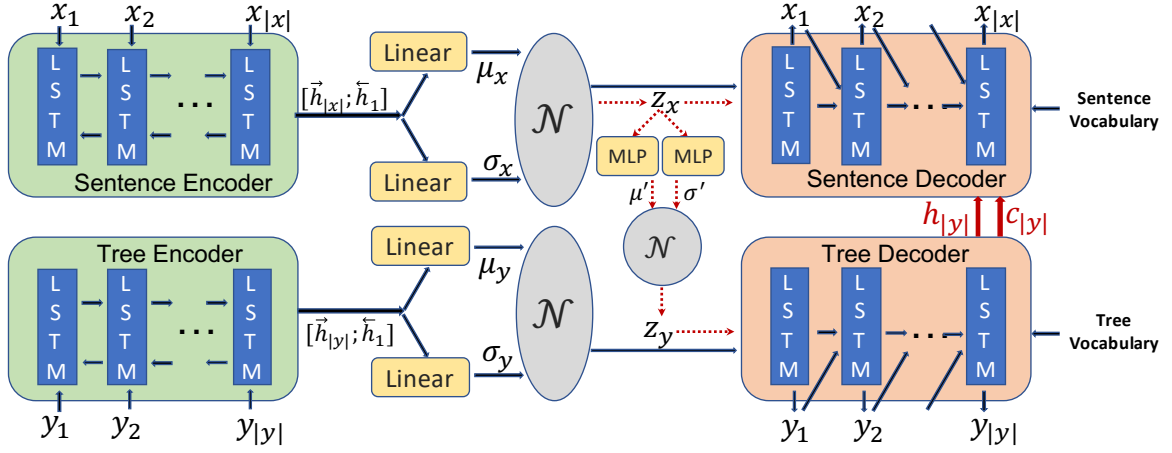
Figure 2: Block diagram of the proposed SIVAE model encoding and decoding sentences and their syntactic trees jointly. The prior network (dashed lines) is used only for the sampling stage of SIVAE-c.

model (SIVAE-c; Section 2.1), directly capturing the dependencies between z_x and z_y, presumes that semantic information should influence syntax structure. During the sampling stage, the prior for z_y is drawn based on z_x from a conditional prior network $p(z_y|z_x)$; z_x implicitly encodes the subject of the sentence, and z_y encodes the corresponding syntax. Although this model has robust performance on generation, it doesn't allow us to syntactically control the generated sentences by freely changing the syntactic tree template in z_y. Thus we propose SIVAE-i (Section 2.2), which generates sentences and syntactic trees assuming the priors $p(z_x)$ and $p(z_y)$ are independent. The entire architecture is shown in Figure 2.

2.1 Modeling Syntax-Semantics Dependencies

Since the syntax of a sentence is influenced by the semantics, especially when the content is long, we first propose a generative model to exploit the dependencies between z_x and z_y, through a conditional prior network $p_\psi(z_y|z_x)$. Formally, SIVAE-c models the joint probability of the sentence and its syntactic tree:

$$p(\boldsymbol{x}, \boldsymbol{y}) = \int_{d_{z_x}} \int_{d_{z_y}} p(\boldsymbol{x}|\boldsymbol{y}, z_x)p(\boldsymbol{y}|z_x, z_y)\cdot$$
$$p(z_y|z_x)p(z_x)d_{z_y}d_{z_x}, \quad (1)$$

where the prior over z_x is the isotropic Gaussian $p(z_x) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. We define $q(\cdot)$ to be the variational posterior distributions that approximate the true posterior distributions. The model is trained by

maximizing the lower bound of the log likelihood

$$\log p(\boldsymbol{x}, \boldsymbol{y}) \geq \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}; \theta, \phi, \psi) = \quad (2)$$
$$\mathbb{E}_{q_\phi(z_x|\boldsymbol{x})} \log p_\theta(\boldsymbol{x}|\boldsymbol{y}, z_x) - \mathrm{KL}[q_\phi(z_x|\boldsymbol{x})||p(z_x)]$$
$$+ \mathbb{E}_{q_\phi(z_y|\boldsymbol{y}, z_x)} \log p_\theta(\boldsymbol{y}|z_y)$$
$$- \mathrm{KL}[q_\phi(z_y|\boldsymbol{y}, z_x)||p_\psi(z_y|z_x)],$$

where ψ, ϕ, and θ are the parameters of the prior network, the recognition networks, and the generation networks, respectively. We apply the reparameterize trick to yield a differentiable unbiased estimator of the lower bound objective.

Conditional Prior Network The key to SIVAE-c is the conditional prior which is used to model the dependencies between the sentence latent variable z_x and the syntactic tree latent variable z_y. Given z_x, the prior for z_y is sampled from a conditional probability $p_\psi(z_y|z_x)$ modeled by a multivariate Gaussians $\mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\sigma}'^2\mathbf{I})$. The parameters of the Gaussian distribution are computed from z_x with a conditional prior network parameterized by ψ. In particular, $\boldsymbol{\mu}'$ and $\boldsymbol{\sigma}'^2$ are the outputs of multilayer perceptron (MLP) networks taking z_x as the input.

Recognition Networks To differentiate through the sampling stage $z \sim q_\phi(z|\boldsymbol{x})$, the VAE encoder $q_\phi(z_x|\boldsymbol{x})$ is also assumed to be a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$, where $\boldsymbol{\mu}(\boldsymbol{x})$ and $\mathrm{diag}(\boldsymbol{\Sigma}(\boldsymbol{x}))$ are the outputs of feedforward networks taking $\boldsymbol{x}$ as the input. The recognition network consists of a bidirectional LSTM encoder to produce a sentence embedding for $\boldsymbol{x}$ and two linear networks to transform the embedding to the Gaussian parameters. The Kullback-Leibler (KL) divergence

between $q_\phi(\boldsymbol{z}_x|\boldsymbol{x})$ and the isotropic Gaussian prior $p(\boldsymbol{z}_x)$ is

$$\mathrm{KL}(q_\phi(\boldsymbol{z}_x|\boldsymbol{x})\|p(\boldsymbol{z}_x)) = \frac{1}{2}[-\log|\boldsymbol{\Sigma}_x|$$
$$- d + \mathrm{tr}(\boldsymbol{\Sigma}_x) + \boldsymbol{\mu}_x^T\boldsymbol{\mu}_x]. \qquad (3)$$

So we only need to model $\boldsymbol{\mu}_x$ and the diagonal of $\boldsymbol{\Sigma}_x$ to compute the KL divergence.

To reconcile the conditional prior $p_\psi(\boldsymbol{z}_y|\boldsymbol{z}_x)$, the variational posterior $q_\phi(\boldsymbol{z}_y|\boldsymbol{y}, \boldsymbol{z}_x) = \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\sigma}_y^2\mathbf{I})$, also depends on the latent variable $\boldsymbol{z}_x$. $\boldsymbol{\mu}_y$ and $\boldsymbol{\sigma}_y^2$ are obtained from a recognition network that contains a bidirectional LSTM encoder, producing a syntactic tree embedding, and two linear networks, taking the embedding and $\boldsymbol{z}_x$ as inputs. The KL divergence is then given by

$$\mathrm{KL}(q_\phi(\boldsymbol{z}_y|\boldsymbol{y}, \boldsymbol{z}_x)\|p_\psi(\boldsymbol{z}_y|\boldsymbol{z}_x)) =$$
$$\frac{1}{2}[\log|\boldsymbol{\sigma}'^2\mathbf{I}| - \log|\boldsymbol{\sigma}_y^2\mathbf{I}| - d + \mathrm{tr}(\frac{\boldsymbol{\sigma}_y^2\mathbf{I}}{\boldsymbol{\sigma}'^2\mathbf{I}})$$
$$+ (\boldsymbol{\mu}' - \boldsymbol{\mu}_y)^T\boldsymbol{\sigma}'^{-2}\mathbf{I}(\boldsymbol{\mu}' - \boldsymbol{\mu}_y)]. \qquad (4)$$

Generation Networks   We employ an LSTM to generate $\boldsymbol{y}$ from $p_\theta(\boldsymbol{y}|\boldsymbol{z}_y)$. A word v_y is selected by computing the probability of $\boldsymbol{y}_t = v_y$ conditioned on previously generated words $\boldsymbol{y}_{-t}$ and $\boldsymbol{z}_y$

$$p(\boldsymbol{y}_t = v_y|\boldsymbol{y}_{-t}, \boldsymbol{z}_y) \propto \exp((\boldsymbol{v}_y^T\mathbf{W}^y\boldsymbol{h}_t^y)), \quad (5)$$

where $\boldsymbol{h}_t^y$ is the current hidden states of the LSTM tree decoder

$$\boldsymbol{h}_t^y = \mathrm{LSTM}(\boldsymbol{z}_y, \boldsymbol{e}(y_{t-1}), \boldsymbol{h}_{t-1}^y, \boldsymbol{c}_{t-1}^y). \qquad (6)$$

To generate $\boldsymbol{x}$ from $p_\theta(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{z}_x)$, we modify the generative model in GNMT (Shah and Barber, 2018). First, the last hidden states $\boldsymbol{h}_{|y|}^y$ and $\boldsymbol{c}_{|y|}^y$ in (6) are directly used as the generated syntactic tree $\boldsymbol{y}$, where $|y|$ is the length of $\boldsymbol{y}$. Then we use another LSTM for sentence generation,

$$\boldsymbol{h}_t^x = \mathrm{LSTM}(\boldsymbol{z}_x, \boldsymbol{e}(x_{t-1}), \boldsymbol{h}_{|y|}^y, \boldsymbol{h}_{t-1}^x, \boldsymbol{c}_{t-1}^x). \quad (7)$$

The conditional probabilities of $\boldsymbol{x}_t = v_x$ for $t = 1, \cdots, |x|$ are computed as

$$p(\boldsymbol{x}_t = v_x|\boldsymbol{x}_{-t}, \boldsymbol{z}_x, \boldsymbol{y}) \propto \exp((\boldsymbol{v}_x^T\mathbf{W}^x\boldsymbol{h}_t^x)). \quad (8)$$

In this way, the generated sentence is conditioned on $\boldsymbol{z}_x$ and the generated syntactic tree $\boldsymbol{y}$. SIVAE-c selects possible syntactic tree templates for a given sentence latent variable, but the syntactic tree template cannot be freely determined.

2.2 Syntactically-Controlled Sentence Generation

In order to freely change the syntactic tree template embedded in $\boldsymbol{z}_y$, we propose an alternative model assuming the independence of two priors. Let priors $\boldsymbol{z}_x$ and $\boldsymbol{z}_y$ be independent random variables drawn from $\mathcal{N}(0, \mathbf{I})$. The variational posteriors $q_\phi(\boldsymbol{z}_x|\boldsymbol{x})$ and $q_\phi(\boldsymbol{z}_y|\boldsymbol{y})$ follow Gaussian distributions parameterized by the outputs of feedforward networks, whose inputs are $\boldsymbol{x}$ and $\boldsymbol{y}$. The model is trained by maximizing the lower bound objective

$$\log p(\boldsymbol{x}, \boldsymbol{y}) \geq \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}; \theta, \phi) = \qquad (9)$$
$$\mathbb{E}_{q_\phi(\boldsymbol{z}_x|\boldsymbol{x})} \log p_\theta(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{z}_x) - \mathrm{KL}[q_\phi(\boldsymbol{z}_x|\boldsymbol{x})\|p(\boldsymbol{z}_x)]$$
$$+ \mathbb{E}_{q_\phi(\boldsymbol{z}_y|\boldsymbol{y})} \log p_\theta(\boldsymbol{y}|\boldsymbol{z}_y) - \mathrm{KL}[q_\phi(\boldsymbol{z}_y|\boldsymbol{y})\|p(\boldsymbol{z}_y)].$$

Since $\boldsymbol{y}$ and $\boldsymbol{z}_x$ are assumed to be independent when computing the joint probability $p(\boldsymbol{x}, \boldsymbol{y})$, we seek to minimize the mutual information $\mathbb{I}(\boldsymbol{y}; \boldsymbol{z}_x)$ during training.

The recognition networks and the generation networks of SIVAE-i are similar to those adopted in SIVAE-c, so we omit them for brevity.

3   Unsupervised Paraphrasing

Paraphrases are sentences with the same meaning but different syntactic structures. SIVAE allows us to execute syntax transformation, producing the desired paraphrases with variable syntactic tree templates. The syntactically controlled paraphrase generation is inspired by Iyyer et al. (2018); the difference is that our SIVAE-based syntactic paraphrase network is purely unsupervised. Unsupervised paraphrasing can be performed using both SIVAE-c and SIVAE-i.

One way to generate paraphrases is to perform syntactically controlled paraphrase generation using SIVAE-i. The latent representations of an input sentence $\boldsymbol{z}_x$ and a syntactic tree template $\boldsymbol{z}_y$ are fed into SIVAE-i, and the syntax of the generated sentence conforms with the explicitly selected target template. However, linearized syntactic sequences are relatively long (as shown in Table 1) and long templates are more likely to mismatch particular input sentences, which may result in nonsensical paraphrase outputs. Therefore, we use simplified syntactic sequences as templates, by taking the top two levels of the linearized constituency trees.

The paraphrase generative process is:

1. Encode the original sentence to $\boldsymbol{z}_x$;

Dataset	Train	Test	Valid	Ave_s	Max_s	Voc_s	Tree Type	Ave_t	Max_t	Voc_t
PTB	39366	4921	4921	25	271	24699	Golden	113	1051	1272
wiki90M	71952	8995	8994	28	318	28907	Parsed	119	1163	387

Table 1: Statistics of the two datasets used in this paper. Ave_s/ Ave_t, Max_s/ Max_t, and Voc_s/ Voc_t denote the average length, maximum length, and vocabulary size for sentences/ tree sequences correspondingly.

2. Select and encode a syntactic template into z_y;

3. Generate the reconstructed syntactic sequence y from $p(y|z_y)$;

4. Generate the paraphrase of the original sentence that conforms to y from $p(x|y, z_x)$.

We can also use a trained SIVAE-c to generate paraphrases. The paraphrase generation process is similar to sampling from a standard VAE with various tempera. The difference is that SIVAE-c first selects possible syntactic tree templates using the conditional prior network $p_\psi(z_y|z_x)$ then generates paraphrases based on the syntactic template and the latent variable.

4 Related Work

Syntax-Aware Neural Text Generation The ability to generate sentences is core to many NLP tasks, such as machine translation (Bahdanau et al., 2015), summarization (Rush et al., 2015), and dialogue generation (Vinyals and Le, 2015). Recent works have shown that neural text generation can benefit from the incorporation of syntactic knowledge (Shen et al., 2018; Choe and Charniak, 2016). Sennrich and Haddow (2016) propose to augment each source word representation with its corresponding part-of-speech tag, lemmatized form and dependency label; Eriguchi et al. (2016) and Bastings et al. (2017) utilize a tree-based encoder and a graph convolutional network encoder respectively to embed the syntactic parse trees as part of the source sentence representations; Chen et al. (2017) model source-side syntactic trees with a bidirectional tree encoder and tree-coverage decoder; Eriguchi et al. (2017) implicitly leverage linguistic prior by treating syntactic parsing as an auxiliary task. However, most of these syntax-aware generation works only focus on neural machine translation.

Deep Latent Variable Models Deep latent variable models that combine the complementary strengths of latent variable models and deep learning have drawn much attention recently. Generative adversarial networks (Goodfellow et al., 2014) and variational autoencoders (Kingma and Welling, 2014) are the two families of deep generative models that are widely adopted in applications. As VAEs allow discrete generation from a continuous space, they have been a popular variant for NLP tasks including text generation (Bowman et al., 2016; Yang et al., 2017; Xu and Durrett, 2018; Shen et al., 2019; Wang et al., 2019). The flexibility of VAEs also enables adding conditions during inference to perform controlled language generation (Hu et al., 2017; Zhao et al., 2017). Divergent from these VAE-based text generation models, our work decouples the latent representations corresponding to the sentence and its syntactic tree respectively.

Paraphrase Generation Due to the similarity between two tasks, neural machine-translation-based models can often be utilized to achieve paraphrase generation (Hasan et al., 2016; Mallinson et al., 2017). Recently, Iyyer et al. (2018) proposed to syntactically control the generated paraphrase and Gupta et al. (2018) generate paraphrases in a deep generative architecture. However, all these methods assume the existence of some parallel paraphrase corpora while unsupervised paraphrase generation has been little explored.

5 Experiments

We conduct our experiments on two datasets: sentence-level Penn Treebank (Marcus et al., 1993) with human-constituted parse trees and a 90 million word subset of Wikipedia (Gulordava et al., 2018) with parsed trees. When the decoder is too strong, VAE suffers from posterior collapse where the model learns to ignore the latent variable (Bowman et al., 2016). To avoid posterior collapse, KL-term annealing and dropping out words during decoding are employed for training in this work. We also tried an advanced method replacing Gaussian priors with von Mises-Fisher priors (Xu and Durrett, 2018) to prevent KL collapse, but the results

Model	PTB						wiki90M					
		Standard			Inputless			Standard			Inputless	
	PPL	NLL	KL	PPL	NLL	KL	PPL	NLL	KL	PPL	NLL	KL
KN5	145	132	-	593	169	-	141	141	-	588	182	-
LSTM-LM	110	124	-	520	165	-	105	133	-	521	179	-
VAE	112	125	2	317	153	13	106	133	5	308	164	22
SIVAE-c	98(**1.6**)	121(**53**)	5(0.5)	286(**2.4**)	150(**99**)	17(1.3)	94(**1.6**)	130(**56**)	12(1.0)	278(**2.3**)	161(**99**)	29(2.4)
SIVAE-i	**90**(1.7)	**119**(60)	9(1.0)	**261**(2.6)	**147**(108)	24(2.5)	**89**(1.7)	**128**(63)	16(1.9)	**256**(2.4)	**158**(104)	36(5.1)

Table 2: Language modeling results on testing sets of PTB and wiki90M. For two SIVAE models, the syntactic tree sequence reconstruction scores are shown in parenthesis alongside the sentence reconstruction scores. Lower is better for PPL and NLL. The best results are in bold.

are about the same.

To discover whether the incorporation of syntactic trees is helpful for sentence generation, we compare our two versions of SIVAE with three baselines that do not utilize syntactic information: a 5-gram Kneser-Ney language model (KN5) (Heafield et al., 2013), an LSTM language model (LSTM-LM) (Sundermeyer et al., 2012), and a standard VAE (Bowman et al., 2016) using an LSTM-based encoder and decoder. Experimental results of language modeling are evaluated by the reconstruction loss using perplexity and the targeted syntactic evaluation proposed in (Marvin and Linzen, 2018). In section 5.3, we show the unsupervised paraphrase generation results.

Datasets We use two datasets in this paper. For sentence-level Penn Treebank (PTB), the syntactic trees are labeled by humans (*i.e.* "gold-standard" trees). For Wikipedia-90M (wiki90M), which does not contain human-generated trees, we first feed the sentences into a state-of-the-art constituency parser (Kitaev and Klein, 2018), and then use the parsed trees as syntactic information for our model. Further, we replace (low-frequency) words that appear only once in both datasets with the <unk> token. Statistics about the two datasets are shown in Table 1. As we can see, the linearized sequences are much longer than sentences. The vocabulary of trees sequences is much smaller than the vocabulary of sentences; and golden trees have larger vocabulary than parsed trees.

Settings The parameters are fine-tuned on the validation set. Our implementation of SIVAE uses one-layer bi-directional LSTM architectures for both encoders, and one-layer unidirectional LSTM architectures for both decoders. The size of hidden units in the LSTM is 600 and the size of word embeddings is 300. The latent variable size is set to 150 for both sentences and their syntactic trees. The hidden units size of the MLP in the conditional

prior network is 400. We also tried to use different model sizes for sentences and syntactic trees but the results are about the same and the performance even get worse when the difference of the model sizes is too big. We use SGD for optimization, with a learning rate of 0.0005. The batch size is 32 and the number of epochs is 10. The word dropout rate during decoding is 0.4. For KL annealing, the initial weights of the KL terms are 0, and then we gradually increase the weights as training progresses, until they reach the KL threshold of 0.8; the rate of this increase is set to 0.5 with respect to the total number of batches.

5.1 Language Modeling Results

We explore two settings for the decoders: standard and inputless. In the standard setting, the input to the LSTM decoder is the concatenation of the latent representation z and the previous ground truth word. A powerful decoder usually results in good reconstruction in this setting but the model may ignore the latent variable. In the inputless setting, the decoder purely relies on the latent representations without any use of prior words, so that the model is driven to learn high-quality latent representations of the sentences and syntactic trees.

The language-modeling results, on testing sets evaluated by negative log likelihood (NLL) and perplexity (PPL), are shown in Table 2. SIVAEs outperform all other baselines on both datasets, demonstrating the explicit incorporation of syntactic trees helps with the reconstruction of sentences. The performance boost on the wiki90M dataset also shows that syntactic trees parsed by a well-developed parser can serve the same function as human-constituted trees, for our model to utilize syntactic information; this underscores how mature parser technology may be leveraged in text generation. Between the two proposed methods, SIVAE-i is better at reconstructing sentences while SIVAE-c is better at reconstructing syntactic trees. In the

standard setting, VAE performs almost the same as the LSTM language model, possibly because the strong LSTM decoder plays a dominant role when it uses prior words, so the VAE becomes similar to an LSTM language model. Furthermore, the KL divergence of the proposed models indicate that SIVAE is better at avoiding posterior collapse, so the LSTM sentence decoder can take advantage of the encoded latent variable as well as the previously generated syntactic tree. In the inputless setting, we see that VAE contains a significantly larger KL term and shows substantial improvement over KN5 and LSTM language models. SIVAEs further reduces PPL from 317 to 261 on PTB and from 308 to 256 on wiki90M, compared to VAE.

5.2 Targeted Syntactic Evaluation

We adopt targeted syntactic evaluation (Marvin and Linzen, 2018) to examine whether the proposed methods improve the grammar of generated sentences. The idea is to assign a higher probability for generating the grammatical sentence than the ungrammatical one, given a pair of sentences that only differ in grammar. There are three types of sentence pairs used in this work.

Subject-verb agreement (SVA): Third-person present English verbs need to agree with the number of their subjects.

For example, *simple SVA*:
(a). The author laughs.
(b). *The author laugh.

Reflexive anaphoras (RA): A reflective pronoun such as *himself* needs to agree in number (and gender) with its antecedent.

For example, *simple RA*:
(a). The senators embarrassed themselves.
(b). *The senators embarrassed herself.

Negative polarity items (NPI): Words like *any* and *ever* that can only be used in the scope of negation are negative polarity items.

For example, *simple NPI*:
(a). No students have ever lived here.
(b). *Most students have ever lived here.

In the above examples, we expect the probability of generating (a) to be higher than the probability of generating (b). However, it is trivial to identify these simple test pairs with simple syntax. Thus we include complex longer test pairs with greater

Model	SVA		RA		NPI	
	S	C	S	C	S	C
Humans	0.96	0.85	0.96	0.87	0.98	0.81
KN5	0.79	0.50	0.50	0.50	0.50	0.50
LSTM-LM	0.94	0.56	0.83	0.55	0.50	0.50
VAE	0.94	0.57	0.84	0.57	0.51	0.50
SIVAE-c	**0.97**	**0.75**	**0.89**	**0.64**	**0.57**	**0.52**
SIVAE-i	0.95	0.71	0.88	0.63	0.56	0.52

Table 3: Accuracy of targeted syntactic evaluation for each grammar test case. S and C denote simple testing pairs and complex testing pairs. The total number of test sentences is 44800. Models are trained on wiki90M. The best results are in bold.

depth in relative clauses, identifying which requires more understanding of the syntactic structure.

The accuracy per grammar test case of each method is shown in Table 3. Human scores on these test pairs in (Marvin and Linzen, 2018) are also shown for reference. SIVAE outperforms other baselines on grammar testing cases, demonstrating the explicit incorporation of syntactic trees helps with the grammar of generated sentences. For simple SVA testing pairs, SIVAE-c has a better score than humans. Even for a difficult grammar test like NPI, our methods still makes significant progress compared to other baselines, whose scores show no syntactic understanding of these sentences. From Table 3, note that KN5 can only identify simple SVA pairs. In addition, VAE has similar syntactic performance as a LSTM language model, which verifies the results in reconstruction. Between the two proposed methods, SIVAE-i makes more grammar mistakes than SIVAE-c, although it has better perplexity in Table 2. This is because SIVAE-c considers the dependency between the sentence prior and the syntactic tree prior, so it can more efficiently prevent the mismatch between two latent variables. In other words, SIVAE-c learns more robust syntactic representations, but this advantage is not reflected on the reconstruction evaluation.

5.3 Unsupervised Paraphrasing Results

The proposed method is used for generating paraphrases by implicitly selecting (SIVAE-c) or explicitly changing (SIVAE-i) the syntactic tree templates. Our model is not trained on a paraphrase corpora, which makes it a purely unsupervised paraphrasing network.

Syntactically Controlled Paraphrasing SIVAE-i as the syntactically controlled para-

Template	Paraphrase
original	the discovery of dinosaurs has long been accompanied by a legend .
(SBARQ (NP) (VP) (,) (SQ) (?))	the discovery of dinosaurs has been a legend , is it ?
(S (") (NP) (VP) (") (NP) (VP) (.))	" the discovery of dinosaurs is a legend " he said .
(S (VP) (,) (NP) (.))	having been accompanied , the unk lengend .
original	in 1987 a clock tower and a fountain were erected at council unk monument .
(S (PP) (PP) (NP) (VP) (.))	in 1987 at council a fountain was erected .
(S (VP) (NP) (CC) (NP) (PP) (.))	build a clock and a fountain at council unk unk .
(S (NP) (;) (S) (PP) (.))	a clock p ; he shops everything on the fountain at unk unk .

Table 4: Examples of syntactically controlled paraphrases generated by SIVAE-i. We show two successful and one failed (in blue) generations with different templates for each input sentence.

Ori	the new york times has been one of the best selling newspapers in america .
Gen1	the new york times also has been used as american best selling newspaper .
Gen2	the new york times also has been used as a " unk " that sells in america .
Gen3	the new york times also has been used as the best " unk " selling in america .

Table 5: An example of paraphrases generated by SIVAE-c.

Model	PTB			wiki90M		
	Rele	Read	Div	Rele	Read	Div
VAE	2.63	3.07	2.77	3.03	3.20	2.60
SIVAE-c	2.93	3.47	2.80	3.27	3.67	2.73
SIVAE-i	3.00	3.30	3.13	3.37	3.53	3.20

Table 6: Human evaluation results on Relevance, Readability, and Diversity of generated paraphrases.

phrasing network is trained on sentences and their simplified syntactic sequences of PTB and wiki90M dataset. Table 4 shows some example paraphrases generated by SIVAE-i using different syntactic templates. We see that SIVAE-i has the ability to syntactically control the generated sentences that conform to the target syntactic template. The examples are well-formed, semantically sensible, and grammatically correct sentences that also preserve semantics of the original sentences. However, the model can generate nonsensical outputs, like the failed cases in Table 4, when the target template mismatches the input sentence.

Paraphrasing with Different Tempera We further perform paraphrasing using SIVAE-c with different tempera. Table 5 shows example paraphrases generated by SIVAE-i. We see that SIVAE-c can generate grammatical sentences that are relevant to the original sentence. However, the generated paraphrases are very similar, indicating that the variance of the conditional prior network is small. In other words, given a sentence latent representation, the range for SIVAE-c selecting a possible syntactic tree representation is small, so it tends to generate similar paraphrases.

Qualitative Human Evaluation We adopt similar human evaluation metrics as in (Gupta et al., 2018) for generated paraphrases. For 20 original sentences, we collect 5 paraphrases for each sentence (100 in total) generated by SIVAE-c or SIVAE-i using 5 different syntactic templates. The models are trained on PTB and wiki90M. Three aspects are verified in human evaluation: *Relevance* with the original sentence, *Readability* w.r.t. the syntax of generated sentences, and *Diversity* of different generations for the same original sentence. Three human evaluators assign a score on a scale of 1-5 (higher is better) for each aspect per generation.

The human evaluation results for unsupervised paraphrase generation using standard VAE, SIVAE-i and SIVAE-c are shown in Table 6. SIVAE-c has the best scores and standard VAE has the worst scores at the readability of generated sentences, which further verifies that syntactic information is helpful for sentence generation. Paraphrases generated by SIVAE-i are more diverse under different syntactic templates, compared to SIVAE-c and standard VAE. All three models show better paraphrasing performance on the wiki90M dataset.

5.4 Continuity of Latent Spaces

We further test the continuity of latent spaces in our model. Two vectors z_A and z_B are randomly sampled from the sentence latent space of SIVAE-c. Table 7 shows generated sentences based on intermediate points between z_A and z_B. We see the transitions are smooth and the generations are grammatical, verifying the continuity of the sen-

A	in january 2014 , the unk announced that one player would be one of the first two heroes .
•	in january 2014 , he was one of the first two players to be the most successful .
•	until the end of the first half of the series , he has played the most reported time .
•	until the end of world war i , he was the first player in the united states .
•	there are also a number of other members in the american war association .
B	there are also a number of other american advances , such as the unk unk of the american association .

Table 7: Intermediate sentences are generated between two random points in the latent space of SIVAE-c.

tence latent space. The syntactic structure remains consistent in neighborhoods along the path, indicating the continuity in the syntactic tree latent space.

6 Conclusion

We present SIVAE, a novel syntax-infused variation autoencoder architecture for text generation, leveraging constituency parse tree structure as the linguistic prior to generate more fluent and grammatical sentences. The new lower bound objective accommodates two latent spaces, for jointly encoding and decoding sentences and their syntactic trees. The first version of SIVAE exploits the dependencies between two latent spaces, while the second version enables syntactically controlled sentence generation by assuming the two priors are independent. Experimental results demonstrate the incorporation of syntactic trees is helpful for reconstruction and grammar of generated sentences. In addition, SIVAE can perform unsupervised paraphrasing with different syntactic tree templates.

Acknowledgments

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. *Proceedings of the Association for Computational Linguistics (ACL)*.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Neural Information Processing Systems (NIPS) Workshop on Deep Learning*.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Neural Information Processing Systems (NIPS)*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Ankush Gupta, Arvind Agarwal, Prawaan Singh, and Piyush Rai. 2018. A deep generative framework for paraphrase generation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Sadid A Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, Oladimeji Farri, et al. 2016. Neural paraphrase generation with stacked residual lstm networks. In *Proceedings the International Conference on Computational Linguistics (COLING)*.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H Clark, and Philipp Koehn. 2013. Scalable modified kneserney language model estimation. In *Proceedings of the Association for Computational Linguistics (ACL)*, volume 2.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Yoon Kim, Sam Wiseman, Andrew C Miller, David Sontag, and Alexander M Rush. 2018. Semi-amortized variational autoencoders. *Proceedings of the International Conference on Machine Learning (ICML)*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. *Proceedings of the Association for Computational Linguistics (ACL)*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*.

Mitchell P Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2).

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. *Proceedings of the International Conference on Machine Learning (ICML)*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*.

Harshil Shah and David Barber. 2018. Generative neural machine translation. In *Neural Information Processing Systems (NIPS)*.

Dinghan Shen, Asli Celikyilmaz, Yizhe Zhang, Liqun Chen, Xin Wang, Jianfeng Gao, and Lawrence Carin. 2019. Towards generating long and coherent text with multi-level latent variable models. *arXiv preprint arXiv:1902.00154*.

Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural mt learn source syntax? In *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Neural Information Processing Systems (NIPS)*.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *Proceedings of the International Conference on Machine Learning (ICML) Workshop on Deep Learning*.

Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Topic-guided variational autoencoders for text generation. *arXiv preprint arXiv:1903.07137*.

Jiacheng Xu and Greg Durrett. 2018. Spherical latent spaces for stable variational autoencoders. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the Association for Computational Linguistics (ACL)*.