Towards Generating Long and Coherent Text with Multi-Level Latent Variable Models

Dinghan Shen[1]*, **Asli Celikyilmaz**[2], **Yizhe Zhang**[2], **Liqun Chen**[1], **Xin Wang**[3], **Jianfeng Gao**[2], **Lawrence Carin**[1]

[1] Duke University [2] Microsoft Research, Redmond
[3] University of California, Santa Barbara
dinghan.shen@duke.edu

Abstract

Variational autoencoders (VAEs) have received much attention recently as an end-to-end architecture for text generation with latent variables. However, previous works typically focus on synthesizing relatively short sentences (up to 20 words), and the posterior collapse issue has been widely identified in text-VAEs. In this paper, we propose to leverage several multi-level structures to learn a VAE model for generating *long*, and *coherent* text. In particular, a hierarchy of stochastic layers between the encoder and decoder networks is employed to abstract more informative and semantic-rich latent codes. Besides, we utilize a multi-level decoder structure to capture the coherent long-term structure inherent in long-form texts, by generating intermediate sentence representations as high-level *plan vectors*. Extensive experimental results demonstrate that the proposed multi-level VAE model produces more coherent and less repetitive long text compared to baselines as well as can mitigate the posterior-collapse issue.

1 Introduction

The variational autoencoder (VAE) for text (Bowman et al., 2016) is a generative model in which a stochastic latent variable provides additional information to modulate the sequential text-generation process. VAEs have been used for various text processing tasks (Semeniuta et al., 2017; Zhao et al., 2017; Kim et al., 2018; Du et al., 2018; Hashimoto et al., 2018; Shen et al., 2018a; Xu and Durrett, 2018; Wang et al., 2019). While most recent work has focused on generating relatively short sequences (*e.g.,* a single sentence or multiple sentences up to around twenty words), generating long-form text (*e.g.,* a single or multiple paragraphs) with deep latent-variable models has been less explored.

***flat*-VAE** (standard)	***multilevel*-VAE** (our model)
i went here for a grooming and a dog . it was very good . **the owner is very nice and friendly** . **the owner is really nice and friendly** . i don t know what they are doing .	i have been going to this nail salon for over a year now . the last time i went there . the stylist was nice . but the lady who did my nails . she was very rude and did not have the best nail color i once had .
the staff is very friendly and helpful . **the only reason i** can t give them 5 stars . **the only reason i** am giving the ticket is because of the ticket . **can t help but** the staff is so friendly and helpful . **can t help but** the parking lot is just the same .	i am a huge fan of this place . my husband and i were looking for a place to get some good music . this place was a little bit pricey . but i was very happy with the service . the staff was friendly .

Table 1: Comparison of samples generated from two generative models on the Yelp reviews dataset. The standard model struggles with repetitions of the same context or words (in blue), yielding non-coherent text. A hierarhical decoder with multi-layered latent variables eliminates redundancy and yields more coherent text planned around focused concepts.

Recurrent Neural Networks (RNNs) (Bahdanau et al., 2015; Chopra et al., 2016) have mainly been used for most text VAE models (Bowman et al., 2016). However, it may be difficult to scale RNNs for *long-form* text generation, as they tend to generate text that is repetitive, ungrammatical, self-contradictory, overly generic and often lacking coherent long-term structure (Holtzman et al., 2018). Two samples of text generated using standard VAE with an RNN decoder is shown in Table 1.

In this work, we propose various multi-level network structures for the VAE model (*ml*-VAE), to address coherency and repetitiveness challenges associated with long-form text generation. To generate globally-coherent long text sequences, it is desirable that both the *higher-level* abstract features (*e.g.*, topic, sentiment, etc.) and *lower-level* fine-granularity details (*e.g.*, specific word choices) of long text can be leveraged by the generative network. It's difficult for a standard

* This research was carried out during an internship at Microsoft Research.

RNN to capture such structure and learn to *plan-ahead*. To improve the model's plan-ahead capability for capturing long-term dependency, following (Roberts et al., 2018), our first multi-level structure defines a hierarchical RNN decoder as the generative network that learns *sentence-* and *word-level* representations. Rather than using the latent code to initialize the RNN decoder directly, we found it more effective when first passing the latent code to a higher-level (sentence) RNN decoder, that outputs an embedding for the lower-level (word) RNN decoder that generates words. Since the low-level decoder network cannot fall back on autoregression, it gains a stronger reliance on the latent code to reconstruct the sequences.

Prior work has found that VAE training often suffers from *posterior collapse*, in which the model ignores the latent code (Bowman et al., 2016). This issue is related to the fact that the decoder network is usually parametrized with an autoregressive neural network, such as RNNs with teacher forcing scheme (Bowman et al., 2016; Yang et al., 2017; Goyal et al., 2017; Semeniuta et al., 2017; Shen et al., 2018b). Several strategies have been proposed (see optimization challenges in Section 4) to make the decoder less autoregressive, so less *contextual information* is utilized by the decoder network (Yang et al., 2017; Shen et al., 2018b). We argue that learning more informative latent codes can enhance the generative model without the need to lessen the contextual information. We propose leveraging a hierarchy of latent variables between the convolutional inference (encoder) networks and a multi-level recurrent generative network (decoder). With multiple stochastic layers, the prior of bottom-level latent variable is inferred from the data, rather than fixed as a standard Gaussian distribution as in typical VAEs (Kingma and Welling, 2013). The induced *latent code* distribution at the bottom level can be perceived as a Gaussian mixture, and thus is endowed with more flexibility to abstract meaningful features from the input text. While recent work has explored structures for more informative latent codes (Kim et al., 2018; Gu et al., 2018), *ml*-VAE is conceptually simple and easy to implement.

We evaluate *ml*-VAE on language modeling, unconditional and conditional text generation tasks. We show substantial improvements against several baseline methods in terms of *perplexity* on language modeling and quality of generated samples based on BLEU statistics and human evaluation.

2 Variational Autoencoder (VAE)

Let $\boldsymbol{x}$ denote a text sequence, which consists of L tokens, *i.e.*, $x_1, x_2, ..., x_L$. A VAE encodes the text $\boldsymbol{x}$ using a recognition (encoder) model, $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, parameterizing an approximate posterior distribution over a continuous latent variable $\boldsymbol{z}$ (whose prior is typically chosen as standard diagonal-covariance Gaussian). $\boldsymbol{z}$ is sampled stochastically from the posterior distribution, and text sequences $\boldsymbol{x}$ are generated conditioned on $\boldsymbol{z}$, via a generative (decoder) network, denoted as $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$. A variational lower bound is typically used to estimate the parameters (Kingma and Welling, 2013):

$$\begin{aligned}\mathcal{L}_{\text{vae}} &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right], \qquad (1)\\ &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})),\end{aligned}$$

This lower bound is composed of a reconstruction loss (first term) that encourages the inference network to encode information necessary to generate the data and a KL regularizer (second term) to push $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ towards the prior $p(\boldsymbol{z})$.

Although VAEs have been shown to be effective in a wide variety of text processing tasks (see related work), there are two challenges associated with generating longer sequences with VAEs: (i) they lack a long-term planning mechanism, which is critical for generating *semantically-coherent* long texts (Serdyuk et al., 2017); and (ii) *posterior collapse* issue. Concerning (ii), it was demonstrated in (Bowman et al., 2016) that due to the autoregressive nature of the RNN, the decoder tends to ignore the information from $\boldsymbol{z}$ entirely, resulting in an extremely small KL term (see Section 4).

3 Multi-Level Generative Networks

3.1 Single Latent Variable (*ml*-VAE-S:)

Our first multi-level model improves upon standard VAE models by introducing a *plan-ahead* ability to sequence generation. Instead of directly making word-level predictions only conditioned on the semantic information from $\boldsymbol{z}$, a series of *plan vectors* are first generated based upon $\boldsymbol{z}$ with a *sentence-level* LSTM decoder (Li et al., 2015b). Our hypothesis is that an explicit design of (inherently hierarchical) paragraph structure can capture sentence-level coherence and potentially mitigate repetitiveness. Intuitively, when predicting each token, the decoder can use information from

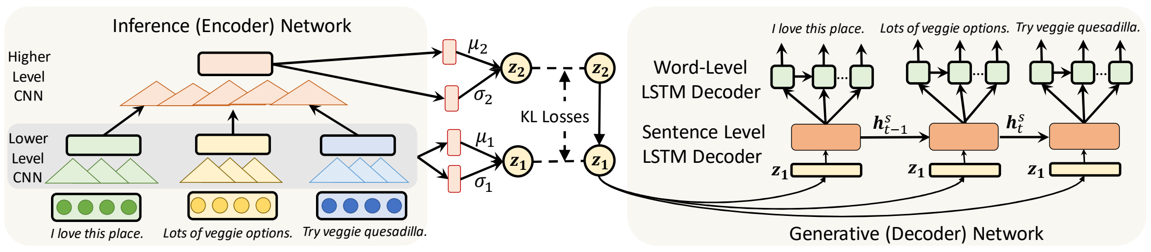

Figure 1: The proposed *multi-level* VAE with *double* latent variables (*ml*-VAE-D).

both the words generated previously and from sentence-level representations.

An input paragraph consist of M sentences, and each sentence t has N_t words, t=1,. . . , M. To generate the plan vectors, the model first samples a latent code $\boldsymbol{z}$ through a one-layer multi-layered perceptron (MLP), with ReLU activation functions, to obtain the starting state of the sentence-level LSTM decoder. Subsequent sentence representations, namely the *plan vectors*, are generated with the sentence-level LSTM in a sequential manner:

$$\boldsymbol{h}^s_t = \text{LSTM}^{\text{sent}}(\boldsymbol{h}^s_{t-1}, \boldsymbol{z}), \tag{2}$$

The latent code $\boldsymbol{z}$ can be considered as a paragraph-level abstraction, relating to information about the semantics of each generated subsequence. Therefore we input $\boldsymbol{z}$ at each time step of the sentence-level LSTM, to predict the sentence representation. Our single-latent-variable model sketched in Figure 3 of supplementary material.

The generated sentence-level plan vectors are then passed onto the word-level LSTM decoder to generate the words for each sentence. To generate each word of a sentence t, the corresponding *plan vector*, $\boldsymbol{h}^s_t$, is concatenated with the word embedding of the previous word and fed to LSTM^{word} at every time step [1]. Let $w_{t,i}$ denote the i-th token of the t-th sentence This process can be expressed as (for $t = 1, 2, ..., M$ and $i = 1, 2, 3, ..., N_t$):

$$\boldsymbol{h}^w_{t,i} = \text{LSTM}^{\text{word}}(\boldsymbol{h}^w_{t,i-1}, \boldsymbol{h}^s_t, \boldsymbol{W_e}[w_{t,i-1}]), \tag{3}$$

$$p(w_{t,i}|w_{t,<i}, \boldsymbol{h}^s_t) = \text{softmax}(\boldsymbol{V}\boldsymbol{h}^w_{t,i}), \tag{4}$$

The initial state $\boldsymbol{h}^w_{t,0}$ of LSTM^{word} is inferred from the corresponding plan vector via an MLP layer. $\boldsymbol{V}$ represents the weight matrix for computing distribution over words, and $\boldsymbol{W_e}$ are word embeddings to be learned. For each sentence, once the special _END token is generated, the word-level LSTM stops decoding [2]. LSTM^{word} decoder parameters are shared for each generated sentence.

3.2 Double Latent Variables (*ml*-VAE-D):

Similar architectures of our single latent variable *ml*-VAE-S model have been applied recently for multi-turn dialog response generation (Serban et al., 2017; Park et al., 2018), mainly focusing on short (one-sentence) response generation. Different from these works, our goal is to generate *long text* which introduces additional challenges to the hierarchical generative network. We hypothesize that with the two-level LSTM decoder embedded into the VAE framework, the load of capturing global and local semantics are handled differently than the *flat*-VAEs (Chen et al., 2016). While the multi-level LSTM decoder can capture relatively detailed information (*e.g.*, word-level (local) coherence) via the word- and sentence-level LSTM networks, the latent codes of the VAE are encouraged to abstract more global and high-level semantic features of multiple sentences of long text.

Our double latent variable extension, *ml*-VAE-D, is shown in Figure 1. The inference network encodes upward through each latent variable to infer their posterior distributions, while the generative network samples downward to obtain the distributions over the latent variables. The distribution of the latent variable at the bottom is inferred from the top-layer latent codes, rather than fixed (as in a standard VAE model). This also introduces flexibility to the model to abstract useful high-level features (Gulrajani et al., 2016), which can then be leveraged by the multi-level LSTM network. Without loss of generality, here we choose to employ a two-layer hierarchy of latent variables, where the bottom and top layers are denoted as $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$, respectively, which can be easily extended to multiple latent-variable layers.

Another important advantage of multi-layer la-

[1]We use teacher-forcing during training and greedy decoding at test time.

[2]Each sentence is padded with an _END token.

tent variables in the VAE framework is related to the *posterior collapse* issue. Even though the single latent variable model (*ml*-VAE-S) defines a multi-level LSTM decoder, the posterior collapse can still exist since the LSTM decoder can still ignore the latent codes due to its autoregressive property. With the hierarchical latent variables, we propose a novel strategy to mitigate this problem, by making less restrictive assumptions regarding the prior distribution of the latent variable. Our model yields a larger KL loss term relative to *flat*-VAEs, indicating more informative latent codes.

The posterior distributions over the latent variables are assumed to be conditionally independent given the input x. We can represent the joint posterior distribution of the two latent variables as [3]:

$$q_{\phi}(\boldsymbol{z_1}, \boldsymbol{z_2}|\boldsymbol{x}) = q_{\phi}(\boldsymbol{z_2}|\boldsymbol{x})q_{\phi}(\boldsymbol{z_1}|\boldsymbol{x}) \tag{5}$$

Concerning the generative network, the latent variable at the bottom is sampled conditioned on the one at the top. Thus, we have:

$$p_{\boldsymbol{\theta}}(\boldsymbol{z_1}, \boldsymbol{z_2}) = p_{\boldsymbol{\theta}}(\boldsymbol{z_2})p_{\boldsymbol{\theta}}(\boldsymbol{z_1}|\boldsymbol{z_2}) \tag{6}$$

$D_{KL}(q_{\phi}(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))$, the second term of the VAE objective, then becomes the KL divergence between joint posterior and prior distributions of the two latent variables. Under the assumptions of (5) and (6), the variational lower bound yields:

$$\begin{aligned}\mathcal{L}_{\text{vae}} = \mathbb{E}_{q(\boldsymbol{z_1}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z_1})] \\ - D_{\text{KL}}(q(\boldsymbol{z_1}, \boldsymbol{z_2}|\boldsymbol{x})||p(\boldsymbol{z_1}, \boldsymbol{z_2}))\end{aligned} \tag{7}$$

Abbreviarting $p_{\boldsymbol{\theta}}$ and $q_{\boldsymbol{\phi}}$ with p and q, we get:

$$\begin{aligned}&D_{\text{KL}}(q(\boldsymbol{z_1}, \boldsymbol{z_2}|\boldsymbol{x})||p(\boldsymbol{z_1}, \boldsymbol{z_2})) \\ &= \int q(\boldsymbol{z_2}|\boldsymbol{x})q(\boldsymbol{z_1}|\boldsymbol{x}) \log \frac{q(\boldsymbol{z_2}|\boldsymbol{x})q(\boldsymbol{z_1}|\boldsymbol{x})}{p(\boldsymbol{z_2})p(\boldsymbol{z_1}|\boldsymbol{z_2})} d\boldsymbol{z_1} d\boldsymbol{z_2} \\ &= \int_{\boldsymbol{z_1}, \boldsymbol{z_2}} [q_{\phi}(\boldsymbol{z_2}|\boldsymbol{x})q_{\phi}(\boldsymbol{z_1}|\boldsymbol{x}) \log \frac{q_{\phi}(\boldsymbol{z_1}|\boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{z_1}|\boldsymbol{z_2})} \\ &\quad + q(\boldsymbol{z_2}|\boldsymbol{x})q(\boldsymbol{z_1}|\boldsymbol{x}) \log \frac{q(\boldsymbol{z_2}|\boldsymbol{x})}{p(\boldsymbol{z_2})}] d\boldsymbol{z_1} d\boldsymbol{z_2} \\ &= \mathbb{E}_{q(\boldsymbol{z_2}|\boldsymbol{x})}[D_{KL}(q(\boldsymbol{z_1}|\boldsymbol{x})||p(\boldsymbol{z_1}|\boldsymbol{z_2}))] \\ &\quad + D_{KL}(q(\boldsymbol{z_2}|\boldsymbol{x})||p(\boldsymbol{z_2}))\end{aligned} \tag{8}$$

The left-hand side of (8) is the abbreviation of $D_{\text{KL}}(q_{\phi}(\boldsymbol{z_1}, \boldsymbol{z_2}|\boldsymbol{x})||p(\boldsymbol{z_1}, \boldsymbol{z_2}))$. Given the Gaussian assumption for both the prior and posterior distributions, both KL divergence terms can be written in closed-form.

[3]We assume $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ to be independent on the encoder side, since this specification will yield a closed-form expression for the KL loss between $p_{\boldsymbol{\theta}}(\boldsymbol{z_1}, \boldsymbol{z_2})$ and $q_{\phi}(\boldsymbol{z_1}, \boldsymbol{z_2}|\boldsymbol{x})$.

To abstract meaningful representations from the input paragraphs, we choose a hierarchical CNN architecture for the inference/encoder networks for both single and double latent variable models. We use sentence-level CNNs to encode each sentence into a fixed-length vector, which are then aggregated and send to paragraph-level CNN encoder. The inference networks parameterizing $q(\boldsymbol{z_1}|\boldsymbol{x})$ and $q(\boldsymbol{z_2}|\boldsymbol{x})$ share the same parameters as the lower-level CNN.

The single-variable *ml*-VAE-S model feeds the paragraph feature vector into the linear layers to infer the mean and variance of the latent variable z. In the double-variable model *ml*-VAE-D, the feature vector is transformed with two MLP layers, and then is used to compute the mean and variance of the top-level latent variable.

4 Related Work

VAE for text generation. VAEs trained under the neural variational inference (NVI) framework, has been widely used for generating text sequences: (Bowman et al., 2016; Yang et al., 2017; Semeniuta et al., 2017; Miao et al., 2016; Serban et al., 2017; Miao et al., 2017; Zhao et al., 2017; Shen et al., 2017; Guu et al., 2018; Kim et al., 2018; Yin et al., 2018; Kaiser et al., 2018; Bahuleyan et al., 2018; Chen et al., 2018b; Deng et al., 2018; Shah and Barber, 2018).

By encouraging the latent feature space to match a prior distribution within an encoder-decoder architecture, the learned latent variable could potentially encode high-level semantic features and serve as a global representation during the decoding process (Bowman et al., 2016). The generated results are also endowed with better diversity due to the sampling procedure of the latent codes (Zhao et al., 2017). Generative Adversarial Networks (GANs) (Yu et al., 2017; Hu et al., 2017; Zhang et al., 2017; Fedus et al., 2018; Chen et al., 2018a), is another type of generative models that are commonly used for text generation. However, existing works have mostly focused on generating one sentence (or multiple sentences with at most twenty words in total). The task of generating relatively longer units of text has been less explored.

Optimization Challenges. The "posterior collapse" issue associated with training text-VAEs was first outlined by (Bowman et al., 2016). They used two strategies, *KL divergence annealing* and *word dropout*, however, none of them help to improve the perplexity compared to a plain neural language model. (Yang et al., 2017) argue that the small KL term relates to the strong autoregressive nature of an LSTM generative network, and they proposed to utilize a dilated CNN as a decoder to improve the informativeness of the latent variable. (Zhao et al., 2018b) proposed to augment the VAE training objective with an additional mutual information term. This yields an intractable integral in the case where the latent variables are continuous. Recent work (He et al., 2019; Fu et al., 2019) has shown that advanced scheduling can mitigate the posterior collapse issue. We instead introduce more flexible priors and hierarchical encoder and decoder structures to deal with posterior collapse.

Hierarchical Structures. Natural language is inherently hierarchical (characters form a word, words form a sentence, sentences form a paragraph, paragraphs from a document, *etc*.). Previous work used multi-level LSTM encoders (Yang et al., 2016) or hierarchical autoencoders (Li et al., 2015a) to learn hierarchical representations for long text or defined a stochastic latent variable for each sentence at decoding time (Serban et al., 2017). In contrast, our model encodes the entire paragraph into one *single* latent variable. The latent variable learned in our model relates more to the global semantic information of a paragraph, whereas those in (Serban et al., 2017) mainly contain the local information of a specific sentence.

Park et al.(Park et al., 2018) introduced a variational hierarchical conversational model (VHCR) with global and local latent variables. They generate local/utterance variables condintioned on the global latent variable, assuming standard dialog-covariance Gaussian for both latent variables. In contrast, both our latent variables in *ml*-VAE-D are designed to contain global information. *ml*-VAE learns the prior of the bottom-level latent variable from the data, yielding more flexible prior relative to a fixed prior and promising results in mitigating the issue of "posterior collapse" in the experiments. The responses in VHCR are generated conditionally on the latent variables and context, while our *ml*-VAE-D model captures the underlying data distribution of the entire paragraph in the bottom latent variable (z_1), so the global latent variable contains more information.

5 Experiments

5.1 Experimental Setup

Datasets We conducted experiments on both generic (unconditional) long-form text generation and conditional paragraph generation (with additional text input as auxiliary information). For the former, we use two datasets: Yelp Reviews (Zhang et al., 2015) and arXiv Abstracts. For the conditional-generation experiments, we consider the task of synthesizing a paper abstract conditioned on the paper title (with the arXiv Abstracts dataset)[4]. Details on dataset statistics and model architectures are provided in the supplementary material.

Baselines We implement the following langauge modeling baselines: language model with a flat LSTM decoder (*flat*-LM), VAE with a flat LSTM decoder (*flat*-VAE), and language model with a multi-level LSTM decoder (*ml*-LM)[5].

For generic text generation, we build models using two recently proposed generative models as baselines: Adversarial Autoencoders (AAE) (Makhzani et al., 2015) and Adversarially-Regularized Autoencoders (ARAE) (Zhao et al., 2018a). Instead of penalizing the KL divergence term, AAE introduces a discriminator network to match the prior and posterior distributions of the latent variable. AARE model extends AAE by introducing Wassertein GAN loss (Arjovsky et al., 2017) and a stronger generator network. We build two variants of our multi-level VAE models: single latent variable *ml*-VAE-S and double latent variable *ml*-VAE-D. Our code will be released to encourage future research.

5.2 Language Modeling Results

We report negative log likelihood (NLL) and perplexity (PPL) results on Yelp and arXiv datasets. Following (Bowman et al., 2016; Yang et al., 2017; Kim et al., 2018), we use the KL loss term to measure the extent of *"posterior collapse"*.

[4]Our goal is to analyze if the proposed architecture can discover different concepts with the hierarchical decoding and latent code structures, thus we use the arxiv dataset with indicated domains for demonstration purposes. We leave the common summarization datasets for future research.

[5]We only experimented with state of the art models with similar architectures to our models, since our goal is to investigate the impact of hiararhical VAE structure on the text generation. More efficient new encoder and decoder architectures such as non-autoregressive models is a direction for extending this work.

Model	Yelp			arXiv		
	NLL	KL	PPL	NLL	KL	PPL
flat-LM	162.6	-	48.0	218.7	-	57.6
flat-VAE	$\leq$ 163.1	0.01	$\leq$ 49.2	$\leq$ 219.5	0.01	$\leq$ 58.4
ml-LM	162.4	-	47.9	219.3	-	58.1
ml-VAE-S	$\leq$ 160.8	3.6	$\leq$ 46.6	$\leq$ 216.8	5.3	$\leq$ 55.6
ml-VAE-D	$\leq$ **160.2**	6.8	$\leq$ **45.8**	$\leq$ **215.6**	12.7	$\leq$ **54.3**

Table 2: Language modeling results on Yelp and arXiv data. Upper block are baselines, and lower are our models.

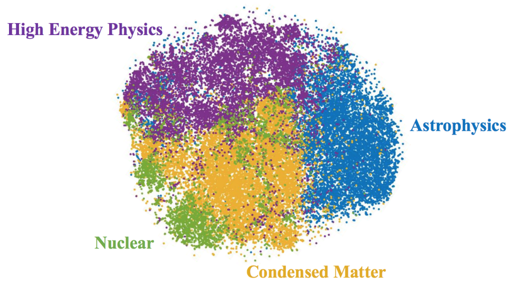

Figure 2: *t*-SNE visualization of the learned latent codes.

As shown in Table 2, the standard *flat*-VAE on Yelp dataset yields a KL divergence term very close to zero, indicating that the generative model makes negligible use of the information from latent variable z. The *flat*-VAE model obtains slightly worse NNL and PPL relative to a flat LSTM-based language model. With multi-level LSTM decoder, our *ml*-VAE-S yields increased KL divergence, demonstrating that the VAE model tends to leverage more information from the latent variable in the decoding stage. The PPL of *ml*-VAE-S is also decreased from 47.9 to 46.6 (compared to *ml*-LM), indicating that the sampled latent codes improve word-level predictions.

Our double latent variable model, *ml*-VAE-D, exhibits an even larger KL divergence cost term (increased from 3.6 to 6.8) than single latent variable model, indicating that more information from the latent variable has been utilized by the generative network. This may be due to the fact that the latent variable priors of the *ml*-VAE-D model are inferred from the data, rather than a fixed standard Gaussian distribution. As a result, the model is endowed with more flexibility to encode informative semantic features in the latent variables, yet matching their posterior distributions to the corresponding priors. *ml*-VAE-D achieves the best PPL results on both datasets (on the arXiv dataset, our hierarchical decoder outperforms the *ml*-LM by reducing the PPL from 58.1 down to 54.3).

5.3 Unconditional Text Generation

We evaluate the quality of generated paragraphs as follows. We randomly sample 1000 latent codes and send them to all trained generative models to generate text. We use corpus-level BLEU score (Papineni et al., 2002) to quantitatively evaluate the generated paragraphs. Following strategy in (Yu et al., 2017; Zhang et al., 2017) we use the entire test set as the reference for each generated text, and get the average BLEU scores[6] over 1000 generated sentences for each model.

The results are in Table 3. VAE tends to be a stronger baseline for paragraph generation, exhibiting higher corpus-level BLEU scores than both AAE and ARAE. This observation is consistent with the results in (Cífka et al., 2018) in Table 3. The VAE with multi-level decoder demonstrates better BLEU scores than the one with a flat decoder, indicating that the plan-ahead mechanism associated with the hierarchical decoding process indeed benefits the sampling quality. *ml*-VAE-D exhibits slightly better results than *ml*-VAE-S. We attribute this to the more flexible prior distribution of *ml*-VAE-D, which improves the ability of inference networks to extract semantic features from a paragraph, yielding more informative latent codes.

We visualize the learnt latent variables to analyze if our models can extract global features. Using the arXiv dataset, we select the most frequent four article topics and re-train our *ml*-VAE-D model on the corresponding abstracts in an unsupervised way (no topic information is used). We sample bottom-level latent codes from the learned model and plot them with *t*-SNE in Figure 2. Each point indicates one paper abstract and the color of each point indicates the topic it belongs to. The embeddings of the same label are very close in the 2-D plot, while those with different labels are relatively farther away from each other. The embeddings of the *High Energy Physics* and *Nuclear* topic abstracts are meshed, which is expected since these two topics are semantically highly related. The inference network can extract meaningful global patterns from the input paragraph.

In Table 1 two samples of generations from *flat*-VAE and *ml*-VAE-D are shown. Compared to our hierarchical model, a flat decoder with a flat VAE

[6]Being interested in longer text generation, we evaluate our models on the n-gram reconscturion ability (where n>1).

Model	Yelp			arXiv		
	B-2	**B-3**	**B-4**	**B-2**	**B-3**	**B-4**
ARAE	0.684	0.524	0.350	0.624	0.475	0.305
AAE	0.735	0.623	0.383	0.729	0.564	0.342
flat-VAE	0.855	0.705	0.515	0.784	0.625	0.421
ml-VAE-S	0.901	0.744	0.531	0.821	**0.663**	0.447
ml-VAE-D	**0.912**	**0.755**	**0.549**	**0.825**	0.657	**0.460**

Table 3: Evaluation results for generated sequences by our models and baselines on corpus-level BLEU scores (**B-n** denotes the corpus-level *BLEU-n* score.)

exibits repetitions as well as suffers from uninformative sentences. The hierarchical model generates reviews that contain more information with less repetitions (word or semantic semantic repetitions), and tend to be semantically-coherent.

Diversity of Generated Paragraphs We evaluate the diversity of random samples from a trained model, since one model might generate realistic-looking sentences while suffering from severe mode collapse (*i.e.*, low diversity). We use three metrics to measure the diversity of generated paragraphs: Self-BLEU scores (Zhu et al., 2018), unique n-grams (Fedus et al., 2018) and the entropy score (Zhang et al., 2018). For a set of sampled sentences, the Self-BLEU metric is the BLEU score of each sample with respect to all other samples as the reference (the numbers over all samples are then averaged); the unique score computes the percentage of *unique* n-grams within all the generated reviews; and the entropy score measures how evenly the empirical n-gram distribution is for a given sentence, which does not depend on the size of testing data, as opposed to unique scores. In all three metrics, lower is better. We randomly sample 1000 reviews from each model.

The results are shown in Table 5. A small self-BLEU score together with a large BLEU score can justify the effectiveness of a model, *i.e.*, being able to generate realistic-looking as well as diverse samples. Among all the VAE variants, *ml*-VAE-D shows the smallest BLEU score and largest unique n-grams percentage, demonstrating the effectiveness of hieararhically structured generative networks as well as latent variables. Even though AAE and ARAE yield better diversity according to both metrics, their corpus-level BLEU scores are much worse relative to *ml*-VAE-D. We leverage human evaluation for further comparison.

we study the effect of disorder on the dynamics of a two-dimensional electron gas in a two-dimensional optical lattice , we show that the superfluid phase is a phase transition , we also show that , in the presence of a magnetic field , the vortex density is strongly enhanced .
in this work we study the dynamics of a colloidal suspension of frictionless , the capillary forces are driven by the UNK UNK , when the substrate is a thin film , the system is driven by a periodic potential, we also study the dynamics of the interface between the two different types of particles .

Table 4: Generated arXiv abstracts from *ml*-VAE-D model.

Model	Yelp						
	B-2	**B-3**	**B-4**	**2gr**	**3gr**	**4gr**	**Etp-2**
ARAE	0.725	0.544	0.402	36.2	59.7	75.8	7.551
AAE	0.831	0.672	0.483	33.2	57.5	71.4	6.767
flat-VAE	0.872	0.755	0.617	23.7	48.2	69.0	6.793
ml-VAE-S	0.865	0.734	0.591	28.7	50.4	70.7	6.843
ml-VAE-D	0.851	0.723	0.579	30.5	53.2	72.6	6.926

Table 5: Evaluation of diversity of 1000 generated sentences on *self-BLEU scores* (**B-n**), *unique n-gram percentages* (**ngr**), 2-*gram entropy score*.

Human Evaluation We conducted human evaluations using Amazon Mechanical Turk to assess the coherence and non-redundancy properties of our proposed models. Given a pair of generated reviews, the judges are asked to select their preferences (no difference between the two reviews is also an option) according to the following four evaluation criteria: *fluency & grammar*, *consistency*, *non-redundancy*, and *overall*. We compare generated text from our *ml*-VAE-D againts flat-VAE, AAE and real samples from the test set. Details are provided in the supplementary material.

As shown in Table 7, *ml*-VAE generates superior human-looking samples compared to *flat*-VAE on the Yelp Reviews dataset. Even though both models underperform when compared against the ground-truth real reviews, *ml*-VAE was rated higher in comparison to *flat*-VAE (raters find *ml*-VAE closer to human-generated than the *flat*-VAE) in all the criteria evaluation criteria. When compared against AAE baseline models using the same data preprocessing steps and hyperparameters, *ml*-VAE again produces more grammatically-correct and semantically-coherent samples. The human evaluations correlate with the automatic metrics, which indicate that our *ml*-VAE is ac-

Title: *Magnetic quantum phase transitions of the anti-ferromagnetic - Heisenberg model*
We study the phase diagram of the model in the presence of a magnetic field, The model is based on the action of the Polyakov loop, We show that the model is consistent with the results of the first order perturbation theory.
Title: *Kalman Filtering With UNK Over Wireless UNK Channels*
The Kalman filter is a powerful tool for the analysis of quantum information, which is a key component of quantum information processing, However, the efficiency of the proposed scheme is not well understood .

Table 6: Conditionally generated arXiv paper abstracts from *ml*-VAE-D model based on a given title.

Model	Grammar.	Cons.	Non-Red.	Overall
ml-VAE	52.0	55.0	53.7	60.0
flat-VAE	30.0	33.0	27.7	32.3
ml-VAE	75.3	86.0	76.7	86.0
AAE	13.3	10.3	15.0	12.0
flat-VAE	19.7	18.7	14.3	19.0
Real data	61.7	74.7	74.3	77.7
ml-VAE	28.0	26.3	25.0	30.3
Real data	48.6	58.7	49.0	61.3

Table 7: Human evaluations on Yelp Reviews dataset. Each block is a head-to-head comparison of two models on grammatically, consistency, and non-redundancy.

tually generating more coherent stories than the baseline models. We leave further evaluations using embedding based metrics as a possible extension to our work.

5.4 Conditional Paragraph Generation

We consider the task of generating an abstract of a paper based on the corresponding title. The same arXiv dataset is utilized, where when training the title and abstract are given as paired text sequences. The title is used as input of the inference network. For the generative network, instead of reconstructing the same input (*i.e.*, title), the paper abstract is employed as the target for decoding. We compare the *ml*-VAE-D model against *ml*-LM. We observe that the *ml*-VAE-D model achieves a test perplexity of 55.7 (with a KL term of 2.57), smaller that the test perplexity of *ml*-LM (58.1), indicating that the information from the title is used by the generative network to facilitate the decoding process. Generated abstract samples from *ml*-VAE-D model are shown in Table 6.

A	the service was great, the receptionist was very friendly and the place was clean, we waited for a while, and then our room was ready .
•	same with all the other reviews, this place is a good place to eat, i came here with a group of friends for a birthday dinner, we were hungry and decided to try it, we were seated promptly.
•	this place is a little bit of a drive from the strip, my husband and i were looking for a place to eat, all the food was good, the only thing i didn t like was the sweet potato fries.
•	this is not a good place to go, the guy at the front desk was rude and unprofessional, it s a very small room, and the place was not clean.
•	service was poor, the food is terrible, when i asked for a refill on my drink, no one even acknowledged me, they are so rude and unprofessional.
B	how is this place still in business, the staff is rude, no one knows what they are doing, they lost my business .

Table 8: Intermediate sentences are produced from linear transition between two points in the latent space and sending them to the generator network.

5.5 Analysis

The Continuity of Latent Space Following (Bowman et al., 2016), we measure the continuity of the learned latent space. We randomly sample two points from the prior latent space (denoted as A and B) and generate sentences based on the equidistant intermediate points along the linear trajectory between A and B. As shown in Table 8, these intermediate samples are all realistic-looking reviews that are syntactically and semantically reasonable, demonstrating the smoothness of the learned VAE latent space. Interestingly, we even observe that the generated sentences gradually transit from positive to negative sentiment along the linear trajectory. To validate that the sentences are not generated by simply retrieving the training data, we find the closest instance, among the entire training set, for each generated review. Details of the results can be found in the supplementary material (Table 12).

Attribute Vector Arithmetic We conduct an experiment to alter the sentiments of reviews with an *attribute vector*. We encode the reviews of the Yelp Review training dataset with positive sentiment and sample a latent code for each review and measure the mean latent vector. The mean latent vector of the negative reviews are computed in the same way. We subtract the negative mean vector from the positive mean vector to obtain the "sentiment attribute vector". Next, for evaluation, we randomly sample 1000 reviews with negative sen-

Original: you have no idea how badly i want to like this place, they are incredibly vegetarian vegan friendly , i just haven t been impressed by anything i ve ordered there , even the chips and salsa aren t terribly good , i do like the bar they have great sangria but that s about it .
Transferred: this is definitely one of my favorite places to eat in vegas , they are very friendly and the food is always fresh, i highly recommend the pork belly , everything else is also very delicious, i do like the fact that they have a great selection of salads .

Table 9: An example sentiment transfer result with attribute vector arithmetic. More examples can be found in the supplementary material (Table 13).

timent and add the "sentiment attribute vector" to their latent codes. The manipulated latent vectors are then fed to the hierarchical decoder to produce the transferred sentences, hypothesizing that they will convey positive sentiment.

As shown in Table 9, the original sentences have been successfully manipulated to positive sentiment with the simple attribute vector operation. However, the specific contents of the reviews are not fully retained. One interesting future direction is to decouple the style and content of long-form texts to allow *content-preserving* attribute manipulation. We employed a CNN sentiment classifier to evaluate the sentiment of manipulated sentences. The classifier is trained on the entire training set and achieves a test accuracy of 94.2%. With this pre-trained classifier, 83.4% of the transferred reviews are predicted as positive-sentiment, indicating that "attribute vector arithmetic" consistently produces the intended manipulation of sentiment.

6 Conclusion

We introduce a hierarchically-structured variational autoencoder for long text generation. It consists of a multi-level LSTM generative network to model the semantic coherence at both the word- and sentence-levels. A hierarchy of stochastic layers is employed, where the priors of the latent variables are learned from the data. Consequently, more informative latent codes are manifested, and the generated samples also exhibit superior quality relative to those from several baseline methods.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*.

Martin Arjovsky, Soumith Chintala, and Lon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. in international conference on learning representations. *ICLR*.

Hareesh Bahuleyan, Lili Mou, Olga Vechtomova, and Pascal Poupart. 2018. Variational attention for sequence-to-sequence models. In *COLING*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *CoNLL*.

Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. 2018a. Adversarial text generation via feature-mover's distance. In *Advances in Neural Information Processing Systems*, pages 4671–4682.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018b. Variational sequential labelers for semi-supervised learning. In *EMNLP*.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. *NAACL*.

Ondřej Cífka, Aliaksei Severyn, Enrique Alfonseca, and Katja Filippova. 2018. Eval all, trust a few, do wrong to none: Comparing sentence generation models. *arXiv preprint arXiv:1804.07972*.

Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander M. Rush. 2018. Latent alignment and variational attention. *CoRR*, abs/1807.03756.

Jiachen Du, Wenjie Li, Yulan He, Ruifeng Xu, Lidong Bing, and Xuan Wang. 2018. Variational autoregressive decoder for neural response generation. In *EMNLP*.

William Fedus, Ian Goodfellow, and Andrew M Dai. 2018. Maskgan: Better text generation via filling in the _. *arXiv preprint arXiv:1801.07736*.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigate kl vanishing. *NAACL*.

Anirudh Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. 2017. Z-forcing: Training stochastic recurrent networks. In *NIPS*.

Xiaodong Gu, Kyunghyun Cho, Jungwoo Ha, and Sunghun Kim. 2018. Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. *arXiv preprint arXiv:1805.12352*.

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. 2016. Pixelvae: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013*.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *TACL*, 6:437–450.

Tatsunori B Hashimoto, Kelvin Guu, Yonatan Oren, and Percy S Liang. 2018. A retrieve-and-edit framework for predicting structured outputs. In *Advances in Neural Information Processing Systems*, pages 10052–10062.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. *ICLR*.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. *ACL*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *ICML*.

Lukasz Kaiser, Aurko Roy, Ashish Vaswani, Niki Parmar, Samy Bengio, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *ICML*.

Yoon Kim, Sam Wiseman, Andrew C. Miller, David A Sontag, and Alexander M. Rush. 2018. Semi-amortized variational autoencoders. In *ICML*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Jiwei Li, Minh-Thang Luong, and Daniel Jurafsky. 2015a. A hierarchical neural autoencoder for paragraphs and documents. In *ACL*.

Jiwei Li, Thang Luong, and Dan Jurafsky. 2015b. A hierarchical neural autoencoder for paragraphs and documents. In *ACL*, volume 1, pages 1106–1115.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. 2015. Adversarial autoencoder. *CoRR*.

Yishu Miao, Edward Grefenstette, and Phil Blunsom. 2017. Discovering discrete latent topics with neural variational inference. In *ICML*.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *ICML*, pages 1727–1736.

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 on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Yookoon Park, Jaemin Cho, and Gunhee Kim. 2018. A hierarchical latent structure for variational conversation modeling. In *NAACL-HLT*.

Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. *arXiv preprint arXiv:1803.05428*.

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2017. A hybrid convolutional variational autoencoder for text generation. In *EMNLP*.

Iulian Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*.

Dmitriy Serdyuk, Nan Rosemary Ke, Alessandro Sordoni, Christopher Joseph Pal, and Yoshua Bengio. 2017. Twin networks: Using the future as a regularizer. *CoRR*, abs/1708.06742.

Harshil Shah and David Barber. 2018. Generative neural machine translation. *arXiv preprint arXiv:1806.05138*.

Dinghan Shen, Qinliang Su, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Lawrence Carin, and Ricardo Henao. 2018a. Nash: Toward end-to-end neural architecture for generative semantic hashing. In *ACL*.

Dinghan Shen, Yizhe Zhang, Ricardo Henao, Qinliang Su, and Lawrence Carin. 2018b. Deconvolutional latent-variable model for text sequence matching. In *AAAI*.

Xiaoyu Shen, Hui Su, Yanran Li, Wenjie Li, Shuzi Niu, Yang Zhao, Akiko Aizawa, and Guoping Long. 2017. A conditional variational framework for dialog generation. In *ACL*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Yingxu Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Topic-guided variational autoencoders for text generation. *NAACL*.

Jiacheng Xu and Greg Durrett. 2018. Spherical latent spaces for stable variational autoencoders. In *EMNLP*.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *ICML*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. Structvae: Tree-structured latent variable models for semi-supervised semantic parsing. In *ACL*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*, pages 649–657.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. Generating informative and diverse conversational responses via adversarial information maximization. In *NIPS*.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. 2017. Adversarial feature matching for text generation. In *ICML*.

Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. 2018a. Adversarially regularized autoencoders. In *ICML*.

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. 2018b. Unsupervised discrete sentence representation learning for interpretable neural dialog generation. *arXiv preprint arXiv:1804.08069*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *arXiv preprint arXiv:1802.01886*.