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

Dinghan Shen¹, Asli Celikyilmaz², Yizhe Zhang², Liqun Chen¹, Xin Wang², Jianfeng Gao², Lawrence Carin¹

¹ Duke University ² Microsoft Research, Redmond ³ University of California, Santa Barbara
{dinghan.shen, liqun.chen, lcarin}@duke.edu
{aslicel, yizhe.zhang, jfgao}@microsoft.com, xwang@cs.ucsb.edu

Abstract

Variational autoencoders (VAEs) have received much attention recently as an end-to-end architecture for text generation with latent variables. In this paper, we investigate several multi-level structures to learn a VAE model to generate long and coherent text. In particular, we use a hierarchy of stochastic layers between the encoder and decoder networks to generate more informative latent codes. We also investigate a multi-level decoder structure to learn a coherent long-term structure by generating intermediate sentence representations as high-level plan vectors. Empirical results demonstrate that a multi-level VAE model produces more coherent and less repetitive long text compared to the standard VAE models and can further 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; Xu and Durrett, 2018). Most recent work has focused on generating relatively short sequences (e.g., a single sentence or multiple sentences up to around twenty words), while generating long-form text (e.g., a single or multiple paragraphs) with deep latent-variable models has been less explored.

Recurrent Neural Networks (RNNs) have been a cornerstone for many text generation models (Bahdanau et al., 2015; Chopra et al., 2016), including the standard VAE model (Bowman et al., 2015; Chopra et al., 2016; Bahdanau et al., 2015; Chopra et al., 2016), in-
latent code to initialize the RNN decoder directly, we first pass the code to a higher-level (sentence) RNN decoder, that outputs a new embedding for generating words with the lower-level RNN decoder. We found this to be an important feature of our architecture. Since during optimization of the loss, the word-level decoder network cannot simply fall back on autoregression, it gains a stronger reliance on the latent code to reconstruct the sequences.

Introducing long-term structure into a VAE model by a multi-level decoder structure, may not mitigate the “posterior collapse” issue, which is inherent in training VAEs with strong autoregressive decoders with a teacher-forcing scheme (Bowman et al., 2016; Yang et al., 2017; Goyal et al., 2017; Semeniuta et al., 2017; Shen et al., 2018b) when training. Bowman et al. (2016) has shown that the posterior distribution of latent codes tends to match the prior distribution regardless of the input sequence (the KL divergence between the two distributions is very close to zero). Consequently, the information from the latent variable is not leveraged by the generative network (Bowman et al., 2016) causing “posterior collapse.” Several strategies have been proposed (see optimization challenges in Section 4.2) 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. In this regard, 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 the typical VAE setting (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 sequences. Recent work has also explored extending latent codes to be more informative (Kim et al., 2018; Gu et al., 2018). Our approach, however, is conceptually simple and easy to implement.

In this paper, we propose a novel framework, multi-level variational autoencoders (ml-VAE), to enhance long and coherent text generation. We evaluate the proposed ml-VAE comprehensively on language modeling, generic (unconditional) text generation, and conditional generation. The proposed model demonstrates substantial improvement relative to several baseline methods, in terms of perplexity on language modeling and quality of generated samples (based on BLEU statistics and human evaluation). We further show that our network can be generalized for conditional-generation scenarios.

2 Variational Autoencoder (VAE)

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

$$
\mathcal{L}_{\text{vae}} = \mathbb{E}_{q_{\phi}(z|x)} \left[ \log \frac{p_{\theta}(x|z)p(z)}{q_{\phi}(z|x)} \right],
$$

$$
= \mathbb{E}_{q_{\phi}(z|x)}[\log p_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x)||p(z)),
$$

Although VAEs have been shown to be effective in a wide variety of text processing tasks (Bowman et al., 2016; Miao et al., 2016; Yang et al., 2017; Serban et al., 2017; Semeniuta 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; Shen et al., 2018a; Deng et al., 2018; Shah and Barber, 2018), there are two challenges associated with applying them for generating longer sequences: (i) they lack a long-term planning mechanism, which is critical for generating semantically-coherent long texts (Serdyuk et al., 2017); and (ii) they are characterized by posterior collapse. 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 $z$ entirely, resulting in an extremely small KL term (see Section 4.2).
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, with intermediate sentence representations. Instead of directly making word-level predictions only conditioned on the semantic information from $z$, a series of plan vectors are first generated based upon $z$ 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 both the words generated previously and from sentence-level representations.

Suppose an input paragraph consist of $M$ sentences, and each sentence $t$ has $N_t$ words, $t=1,\ldots,M$. To generate the plan vectors, the model first samples a latent code $z$ 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:

$$h_t^s = \text{LSTM}^\text{sent}(h_{t-1}^s, z),$$

The latent code $z$ can be considered as a paragraph-level abstraction, relating to information about the semantics of each generated subsequence. Therefore we input $z$ at each time step of the sentence-level LSTM, to predict the sentence representation. A schematic view of our single-latent-variable model is shown in Figure 2 in the Supplementaty 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, $h_t^s$, is concatenated with the word embedding of the previous word and fed to LSTM$^\text{word}$ at every time step $t$. Let $w_{t,i}$ denote the $i$-th token of the $t$-th sentence. This process can be expressed as (for $t = 1, 2, \ldots, M$ and $i = 1, 2, 3, \ldots, N_t$):

$$h_t^w = \text{LSTM}^\text{word}(h_{t-1}^w, h_t^s, W_e[w_{t,i-1}]),$$

$$p(w_{t,i}|w_{t,<i}, h_t^s) = \text{softmax}(V h_t^s),$$

The initial state $h_{1,0}^w$ of LSTM$^\text{word}$ is inferred from the corresponding plan vector via an MLP layer. $V$ represents the weight matrix for computing distribution over words, and $W_e$ are word embeddings to be learned. For each sentence, once the special _END token is generated, the word-level LSTM stops decoding $^{2}$. LSTM$^\text{word}$ 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). Specifically, 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

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

$^{2}$Each sentence is padded with an _END token at the preprocessing step.
Under the assumptions of (5) and (6), the vari-ables are assumed to be conditionally independent given the input \( x \). We can represent the joint posterior distribution of the two latent variables as

\[
q_\phi(z_1, z_2 | x) = q_\phi(z_2 | x) q_\phi(z_1 | x)
\]

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

\[
p_\theta(z_1, z_2 | x) = p_\theta(z_2 | x)p_\theta(z_1 | x)
\]

To optimize the parameters of the inference and generative networks, the second term in the VAE objective, \( D_{KL}(q_\phi(z_1, z_2 | x) || p(z)) \), can be regarded as the KL divergence between the joint posterior and prior distributions of the two latent variables. Under the assumptions of (5) and (6), the variational lower bound is:

\[
\mathcal{L}_{\text{vae}} = \mathbb{E}_{q(z_1 | x)}[\log p(x | z_1)] - D_{KL}(q(z_1, z_2 | x) || p(z_1, z_2))
\]

, where the functions \( p_\theta \) and \( q_\phi \) are abbreviated as \( p \) and \( q \) and:

\[
D_{KL}(q(z_1, z_2 | x) || p(z_1, z_2)) = \int q(z_2 | x) q(z_1 | x) \log \frac{q(z_2 | x) q(z_1 | x)}{p(z_1, z_2)} dz_1 dz_2 = \int q(z_2 | x) q(z_1 | x) \log \frac{q(z_1 | x)}{p(z_1)} dz_1 dz_2 + q(z_2 | x) q(z_1 | x) \log \frac{q(z_2 | x)}{p(z_2)} dz_1 dz_2
\]

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

Another important advantage of multi-layer latent variables in the VAE framework is related to the posterior collapse issue. With a single latent variable network, even with the multi-level LSTM decoder, the posterior collapse can still exist because the LSTM can still ignore the latent codes while decoding 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. As shown in the experiments, our network yields a larger KL loss term relative to \( \text{flat-VAEs} \), 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:

\[
q_\phi(z_1, z_2 | x) = q_\phi(z_2 | x) q_\phi(z_1 | x)
\]

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

\[
p_\theta(z_1, z_2 | x) = p_\theta(z_2 | x)p_\theta(z_1 | x)
\]

To optimize the parameters of the inference and generative networks, the second term in the VAE objective, \( D_{KL}(q_\phi(z_1, z_2 | x) || p(z)) \), can be regarded as the KL divergence between the joint posterior and prior distributions of the two latent variables. Under the assumptions of (5) and (6), the variational lower bound is:

\[
\mathcal{L}_{\text{vae}} = \mathbb{E}_{q(z_1 | x)}[\log p(x | z_1)] - D_{KL}(q(z_1, z_2 | x) || p(z_1, z_2))
\]

, where the functions \( p_\theta \) and \( q_\phi \) are abbreviated as \( p \) and \( q \) and:

\[
D_{KL}(q(z_1, z_2 | x) || p(z_1, z_2)) = \int q(z_2 | x) q(z_1 | x) \log \frac{q(z_2 | x) q(z_1 | x)}{p(z_1, z_2)} dz_1 dz_2 = \int q(z_2 | x) q(z_1 | x) \log \frac{q(z_1 | x)}{p(z_1)} dz_1 dz_2 + q(z_2 | x) q(z_1 | x) \log \frac{q(z_1 | x)}{p(z_2)} dz_1 dz_2
\]

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

### 3.3 Model Specifications

To abstract meaningful representations from the input paragraphs, we choose a hierarchical CNN architecture for the inference/encoder networks. Specifically, our model first applies a sentence-level CNN encoder to each sentence to obtain a fixed-length vector. Later, a paragraph-level CNN encoder is utilized to aggregate the vectors with respect to all sentences. Note that the inference networks parameterizing \( q(z_1 | x) \) and \( q(z_2 | x) \) share the parameters of the lower-level CNN.

The single-variable \( ml-\text{VAE-S} \) 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 \( ml-\text{VAE-D} \), the feature vector is further transformed with two MLP layers, and then is used to compute the mean and variance of the top-level latent variable.

### 4 Related Work

#### 4.1 VAE for text generation

The variational autoencoder, 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; Zhao et al., 2017). 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). Another type of deep generative model that has been widely adopted for text generation is the Generative Adversarial Networks (GANs) (Yu et al., 2017; Hu et al., 2017; Zhang et al., 2017; Fedus et al., 2018; Chen et al., 2018a). 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.

4.2 Optimization Challenges with Text-VAEs

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 further yields an intractable integral in the case where the latent variables are continuous. We deal with “posterior collapse” from two perspectives: i) more flexible priors are assumed over the latent variables (learned from the data); and ii) the hierarchical structure within a paragraph is taken into account, so that the latent variables can focus less on the local information (e.g., word-level coherence) and more on the global features.

4.3 Hierarchical Structures in NLP

Natural language is inherently organized in a hierarchical manner (characters form a word, words form a sentence, sentences form a paragraph, paragraphs from a document, etc.). In (Yang et al., 2016), multi-level LSTM encoders are used at the word- and sentence-level along with an attention mechanism to learn document representations. A hierarchical autoencoder is proposed in (Li et al., 2015a) to reconstruct long-paragraph text. Our approach is conceptually similar the model in (Serban et al., 2017), in which a stochastic latent variable is produced for each sentence during decoding. In contrast, our model encodes the entire paragraph into one single latent variable. As a result, 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. Therefore, their model is not suitable for tasks such as latent space interpolation.

Finally, our work is related to prior work that addresses plan-ahead capabilities in decoders. In (Park et al., 2018) a variational hierarchical conversational model (VHCR) model is proposed with global and local latent variables. The VHCR model generates its local/utterance variables from the global latent variable, while fixing the priors for the two sets of latent variables to be standard diagonal-covariance Gaussian. In contrast, both of our latent variables in ml-VAE-D are designed to contain global information. The prior of the bottom-level latent variable in our model is learned from the data (and is thus more flexible relative to a fixed prior), which exhibits promising results in terms of mitigating the issue of “posterior collapse” (see Table 2). Furthermore, in VHCR, the responses 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 ($z_1$). Therefore, the (global) latent variable learned by our model should contain 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 (Celikyilmaz et al., 2018). For the conditional-generation experiments, we consider the task of synthesizing a paper abstract (which typically includes several sentences) conditioned on the paper title (with the arXiv Abstracts dataset). More details of the dataset statistics and model architectures are provided in the Supplementary Materials.

Baselines For language modeling experiments, we implemented several 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).

For generic text generation, we further consider 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., 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., 2018a).

As shown in Table 2, on the Yelp dataset, the standard flat-VAE has a KL divergence term very close to zero, indicating that the generative model makes negligible use of the information from latent variable $z$. Consequently, flat-VAE model obtains slightly worse NNL and PPL relative to a flat LSTM-based language model. In contrast, with a 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 is helping in making 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 that with a single latent variable, indicating that more information from the latent variable has been utilized by the generative network. This may be attributed 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. More importantly, by effectively exploiting the sampled latent codes, 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 further 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. Specifically, we follow the strategy in (Yu et al., 2017; Zhang et al., 2017) and use the entire test set as the reference for each generated text, and get average BLEU scores over 1000 generated sentences for each model.

As shown 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 (Cifka et al., 2018). 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. Moreover, 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 the inference networks to extract semantic features from a paragraph, and thus yields more informative latent codes.

To further illustrate the capability of our model to extract global features, we visualize the learned latent variable. Using the arXiv dataset, we select the most frequent four article topics and retrain our ml-VAE-D model on the corresponding abstracts in an unsupervised way (no topic infor-
We randomly sample 1000 reviews from each model, and the corresponding results are shown in Table 4. Note that a small self-BLEU score must be accompanied with a large BLEU score to 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, further demonstrating the advantages of making both the generative networks and latent variables hierarchical. Concerning AAE and ARAE, although they exhibit better diversity according to both metrics, their corpus-level BLEU scores are much worse relative to ml-VAE-D. Thus, we leverage human evaluation for further comparison.

### Human Evaluation
We conducted human evaluation using Amazon Mechanical Turk to assess the coherence and non-redundancy of the texts generated from our models in comparison to the baselines, which is difficult to measure based on automated metrics. 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. Details of the evaluation are provided in the SM. As shown...
Title: Magnetic quantum phase transitions of the antiferromagnetic 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 7: Conditionally generated paper abstracts based upon a title (trained with the arXiv data).

| Model    | Grammaticality | Consistency | Non-Redundancy | Overall |
|----------|----------------|-------------|----------------|---------|
| ml-VAE   | 52.0           | 55.0        | 51.7           | 60.0    |
| flat-VAE | 30.0           | 33.0        | 27.7           | 32.3    |
| ml-VAE   | 75.5           | 86.0        | 76.7           | 86.0    |
| AAE      | 13.3           | 10.3        | 15.0           | 12.0    |
| Real data| 61.7           | 74.3        | 74.3           | 77.7    |
| ml-VAE   | 28.0           | 26.7        | 23.0           | 30.5    |
| Real data| 48.6           | 58.7        | 49.0           | 61.3    |

Table 8: A Mechanical Turk blind heads-up evaluation between pairs of models trained on the Yelp Reviews dataset.

The dataset is utilized, where when training the title and abstract are given as paired text sequences. To investigate the task of generating the 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), which is smaller that the test perplexity of ml-LM (58.1). This indicates that the information from the title has indeed been leveraged by the generative network to facilitate the decoding process. In Table 7 we show two generated samples from the ml-VAE-D model.

5.4 Conditional Paragraph Generation

We further evaluate the proposed VAE model on a conditional generation task. Specifically, we consider the task of generating the 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), which is smaller that the test perplexity of ml-LM (58.1). This indicates that the information from the title has indeed been leveraged by the generative network to facilitate the decoding process. In Table 7 we show two generated samples from the ml-VAE-D model.

Table 8: A Mechanical Turk blind heads-up evaluation between pairs of models trained on the Yelp Reviews dataset.

| Model    | Grammaticality | Consistency | Non-Redundancy | Overall |
|----------|----------------|-------------|----------------|---------|
| ml-VAE   | 52.0           | 55.0        | 51.7           | 60.0    |
| flat-VAE | 30.0           | 33.0        | 27.7           | 32.3    |
| ml-VAE   | 75.5           | 86.0        | 76.7           | 86.0    |
| AAE      | 13.3           | 10.3        | 15.0           | 12.0    |
| Real data| 61.7           | 74.3        | 74.3           | 77.7    |
| ml-VAE   | 28.0           | 26.7        | 23.0           | 30.5    |
| Real data| 48.6           | 58.7        | 49.0           | 61.3    |

5.5 Analysis

The Continuity of Latent Space Following (Bowman et al., 2016), we further measure the continuity of the learned latent space. Specifically, two points are randomly sampled from the prior latent space (denoted as A and B). Sentences are generated based on the equidistant intermediate points along the linear trajectory between A and B. As shown in Table 9, 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 further find the closest instance, among the entire training set, for each generated review. We demonstrate the details of the results in the SM (Table 13).

Attribute Vector Arithmetic To investigate the structure of the latent space, 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 sentiment 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 10, the original sentences
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 Lou Bouloulou. 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.

Asli Celikyilmaz, Antoine Bosselut, and Dingshan Shen. 2018. Arxgen: Corpus of arxiv articles for deep generative models. In https://github.com/Microsoft/ARXGEN/tree/master/arxiv.

Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinhua Shen, Yi Zhe 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 Čífa, 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.
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, Junghoo 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.

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.

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.
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. Seggan: 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, Xiajun 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.
A Datasets & Model Details

In the following, we provide details of data pre-processing and the experimental setups used in the experiments. For both Yelp Reviews and arXiv Abstracts datasets, we truncate the original paragraph to the first five sentences (split by punctuation marks including comma, period and point symbols), where each sentence contains at most 25 words. Therefore, each paragraph has at most 125 words. We further remove those sentences that contain less than 30 words. The statistics of both datasets are detailed in Table 11. Note that the average length of paragraphs considered here are much larger than previous generative models for text (Bowman et al., 2016; Yu et al., 2017; Hu et al., 2017; Zhang et al., 2017), since these works considered text sequences that contain only one sentence with at most twenty words.

| Dataset          | Train | Test  | Vocabulary | Aver. Length |
|------------------|-------|-------|------------|--------------|
| Yelp Reviews     | 244748| 18401 | 12461      | 48           |
| arXiv Abstracts  | 504268| 28016 | 32487      | 59           |

Table 11: Summary statistics for the datasets used in the generic text generation experiments.

The model is trained using Adam (Kingma and Ba, 2014) with a learning rate of $3 \times 10^{-4}$ for all parameters, with a decay rate of 0.99 for every 3000 iterations. Dropout (Srivastava et al., 2014) is employed on both word embedding and latent variable layers, with rates selected from $\{0.3, 0.5, 0.8\}$ on the validation set. We set the mini-batch size to 128. Following (Bowman et al., 2016) we adopt the KL cost annealing strategy to stabilize training: the KL cost term is increased linearly to 1 until 10,000 iterations. All experiments are implemented in Tensorflow (Abadi et al., 2016), using one NVIDIA GeForce GTX TITAN X GPU with 12GB memory.

B Additional Generated Samples from ml-VAE-D vs flat-VAE

We provide additional examples for the comparison between ml-VAE-D vs flat-VAE in Table 12, as a continuation of Table 1.

C Retrieved closest training instances of generated samples (Yelp Reviews Dataset)

We provide samples of retrieved instances from the Yelp Review training dataset which are closest to the generated samples. Table 13 shows the closest training samples of each generated Yelp review. The first column indicates the intermediate generated sentences produced from linear transition from a point $A$ to another point $B$ in the prior latent space. The second column on the right are the real sentences retrieved from the training set that are closest to the ones generated on the left (determined by BLEU-2 score). We can see that the retrieved training data is quite different from the generated samples, indicating that our model is indeed generating samples that it has never seen during training.

D Human evaluation setup and details

Some properties of the generated paragraphs, such as (topic) coherence or non-redundancy, can not be easily measured by automated metrics. Therefore, we further conduct human evaluation based on 100 samples randomly generated by each model (the models are trained on the Yelp Reviews dataset for this evaluation). We consider flat-VAE, adversarial autoencoders (AAE) and real samples from the test set to compare with our proposed ml-VAE-D model. The same hyperparameters are employed for the different model variants to ensure fair comparison. We evaluate the quality of these generated samples with a blind heads-up comparison using Amazon Mechanical Turk. 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 4 evaluation criteria: (1) fluency & grammar, the one that is more grammatically correct and fluent; (2) consistency, the one that depicts a sequence of topics and events that is more consistent; (3) non-redundancy, the one that is better at non-redundancy (if a review repeats itself, this can be taken into account); and (4) overall,
the one that more effectively communicates reasonable content. These different criteria help to quantify the impact of the hierarchical structures employed in our model, while the non-redundancy and consistency metrics could be especially correlated with the model’s plan-ahead abilities. The generated paragraphs are presented to the judges in a random order and they are not told the source of the samples. Each sample is rated by three judges and the results are averaged across all samples and judges.

**E  More Samples on Attribute Vector Arithmetic**

We provide more samples for sentiment manipulation, where we intend to alter sentiment of negative Yelp reviews with “attribute vector arithmetic”, as a continuation of Table 10.

**F  Comparison with the “utterance drop” strategy**

To resolve the “posterior collapse” issue of training textual VAEs, (Park et al., 2018) also introduced a strategy called **utterance drop** (u.d). Specifically, they proposed to weaken the autoregressive power of hierarchical RNNs by dropping the utterance encoder vector with a certain probability. To investigate the effectiveness of their method relative to our strategy of employing a hierarchy of latent variables, we further conduct a comparative study. Particularly, we utilize ml-VAE-S as the baseline model and apply the two strategies to it respectively. The corresponding results on language modeling (Yelp dataset) are shown in Table 15. Their u.d strategy indeed allows better usage of the latent variable (indicated by a larger KL divergence value). However, the NLL of the language model becomes even worse, possibly due to the weakening of the decoder during training (similar observations have also been reported in Table 2 of (Park et al., 2018)). In contrast, our hierarchical prior strategy yields larger KL terms as well as lower NNL value, indicating the advantage of our strategy to mitigate the “posterior collapse” issue.

| Model               | NLL | KL  | PPL |
|---------------------|-----|-----|-----|
| ml-VAE-S            | 160.8 | 3.6 | 46.6 |
| ml-VAE-S (with u.d) | 161.3 | 5.6 | 47.1 |
| ml-VAE-D            | 160.2 | 6.8 | 45.8 |

Table 15: Comparison with the **utterance drop** strategy.
| ml-VAE                                                                 | flat-VAE                                                                 |
|-----------------------------------------------------------------------|-------------------------------------------------------------------------|
| i would give this place zero stars if i could , the guy who was       | this is a great little restaurant in vegas , i had the shrimp scampi     |
| working the front desk was rude and unprofessional , i have to say    | and my wife had the shrimp scampi , and my husband had the shrimp scampi |
| that i was in the wrong place , and i m not sure what i was thinking ,| , it was delicious , i had the shrimp scampi which was                    |
| this is not a good place to go to .                                   | delicious and seasoned perfectly .                                       |
| my wife and i went to this place for dinner , we were seatied         | very good chinese food , very good chinese food , the service was very   |
| immediately , the food was good , i ordered the shrimp and grits ,    | slow , i guess that s what they were doing , very slow to get a quick   |
| which was the best part of the meal                                   | meal                                                                     |
| we got a gift certificate from a store , we walked in and were        | we go there for eakfast , i ve been here 3 times and it s always good ,  |
| greeted by a young lady who was very helpful and friendly , so we     | the hot dogs are delicious , and the hot dogs are delicious , i ve been  |
| decided to get a cut , i was told that they would be ready in 15      | there for eakfast and it is so good .                                    |
| minutes .                                                             | do not go here , their food is terrible , they were very slow , in my   |
| the place was packed , chicken was dry , tasted like a frozen         | opinion .                                                               |
| hot chocolate , others were just so so , i wouldn t recommend this    | the wynn is a great place to eat , the food was great and i had the     |
| place .                                                               | linguine and clams , ( i was so excited to try it ) .                   |
| went today with my wife , and received a coupon for a free            | i came here for a quick bite before heading to a friend s recommendation , |
| appetizer , we were not impressed , we both ordered the same thing ,  | the place was packed , but the food was delicious , i am a fan of the   |
| and we were not impressed .                                           | place , and the place is packed with a lot of people .                  |
| recently visited this place for the first time , i live in the area  | best haircut i ve had in years , friendly staff and great service .     |
| and have been looking for a good local place to eat , we stopped in  | he made sure that i was happy with my hair cut , just a little pricey   |
| for a quick bite and a few beers , always a nice place to sit and     | but worth it , she is so nice and friendly .                             |
| relax , wonderful and friendly staffs .                               | had a great experience here today , the delivery was friendly and       |
| best haircut i ve had in years , friendly staff and great service .   | efficient and the food was good , i would recommend this place to       |
| he made sure that i was happy with my hair cut , just a little pricey | anyone who will work in the future , will be back again .               |
| but worth it , she is so nice and friendly .                          | best place to get in vegas , ps the massage here is awesome , if you    |
| great place to go for a date night , first time i went here , service | want to spend your money , then go there , ps the massage is great .    |
| is good , the staff is friendly , 5 stars for the food .              |                                                                         |

Table 12: Samples randomly generated from ml-VAE-D and flat-VAE, which are both trained on the Yelp review dataset. The repetitive patterns within the generated reviews are highlighted.
### Generated samples vs Closest instance (in the training dataset)

| A | Generated samples | Closest instance (in the training dataset) |
|---|-------------------|-------------------------------------------|
| 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. | i've only been here once myself, and i wasn't impressed, the service was great, staff was very friendly and helpful, we waited for nothing. |
| 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. | i really love this place, red robin alone is a good place to eat, but the service here is great too not always easy to find, we were seated promptly,  ought drinks promptly and our orders were on point. |
| 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. | after a night of drinking, we were looking for a place to eat, the only place still open was the grad lux, its just like a cheesecake factory, the food was actually pretty good. |
| 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. | the food is very good, the margaritas hit the spot, and the service is great, the atmosphere is a little cheesy but overall it's a great place to go. |
| 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. | disliked this place, the hostess was so rude, when i asked for a booth, i got attitude, a major. |
| B | Generated samples | Closest instance (in the training dataset) |
| how is this place still in business, the staff is rude, no one knows what they are doing, they lost my business. | i can't express how awful this store is, don't go to this location, drive to any other location, the staff is useless, no one knows what they are doing. |

Table 13: Using the $ml$-VAE-D model trained on the Yelp Review dataset, intermediate sentences are produced from linear transition between two points (A and B) in the prior latent space. Each sentence in the left panel is generated from a latent point on a linear path, and each sentence on the right is the closest sample to the left one within the entire training set (determined by BLEU-2 score).

| Original | Transferred |
|----------|-------------|
| papa j's is expensive and inconsistent, the ambiance is nice but it doesn't justify the prices, there are better restaurants in carnegie. | love the food, the prices are reasonable and the food is great, it's a great place to go for a quick bite. |
| i had a lunch there once, the food is ok but it's on the pricey side, i don't think i will be back. | i had a great time here, the food is great and the prices are reasonable, i'll be back. |
| i have to say that i write this review with much regret, because i have always loved papa j's, but my recent experience there has changed my mind a bit, from the minute we were seated, we were greeted by a server that was clearly inexperienced and didn't know the menu. | i have to say, the restaurant is a great place to go for a date, my girlfriend and i have been there a few times, on my last visit, we were greeted by a very friendly hostess. |
| a friend recommended this to me, and i can't figure out why, the food was underwhelming and pricey, the service was fine, and the place looked nice. | a friend of mine recommended this place, and i was so glad that i did try it, the service was great, and the food was delicious. |
| this is a small, franchise owned location that caters to the low income in the area, selection is quite limited throughout the store with limited quantities on the shelf of the items they do carry, because of the area in which it is located, the store is not 24 hours as most giant eagle's seem to be. | this is a great little shop, easy to navigate, and they are always open, their produce is always fresh, the store is clean and the staff is friendly. |

Table 14: Sentiment transfer results with attribute vector arithmetic.