Controlled Text Generation Using Dictionary Prior in Variational Autoencoders

Xianghong Fang[1], Jian Li[2]*, Lifeng Shang[2], Xin Jiang[2], Qun Liu[2], Dit-Yan Yeung[1]
[1]The Hong Kong University of Science and Technology
[2]Huawei Noah's Ark Lab
xfangam@connect.ust.hk, dyyeung@cse.ust.hk
{lijian703, shang.lifeng, jiang.xin, qun.liu}@huawei.com

Abstract

While variational autoencoders (VAEs) have been widely applied in text generation tasks, they are troubled by two challenges: insufficient representation capacity and poor controllability. The former results from the posterior collapse and restrictive assumption, which impede better representation learning. The latter arises as continuous latent variables in traditional formulations hinder VAEs from interpretability and controllability. In this paper, we propose Dictionary Prior (DPrior), a new data-driven prior that enjoys the merits of expressivity and controllability. To facilitate controlled text generation with DPrior, we propose to employ contrastive learning to separate the latent space into several parts. Extensive experiments on both language modeling and controlled text generation demonstrate the effectiveness of the proposed approach.

1 Introduction

As one of the representative deep generative models, variational autoencoders (VAEs) (Kingma and Welling, 2014) have been widely applied in text generation tasks, such as dialog generation (Wu et al., 2020; Zhao et al., 2017), machine translation (Shah and Barber, 2018; McCarthy et al., 2020; Sheng et al., 2020) and poetry generation (Li et al., 2018b; Yi et al., 2020). Despite the success, VAEs still suffer from two challenges: insufficient representation capacity and poor controllability.

The challenge of insufficient representation capacity in variational models arises from two aspects. One is the posterior collapse, a notorious issue that generally exists in VAEs especially serious in autoregressive text generation (Bowman et al., 2016), which leads to degenerate local optimums during the training of VAEs (He et al., 2019). Another is the restrictive assumption for priors and variational posteriors (Ding and Gimpel, 2021), which generally follow Gaussian distribution and spherical Gaussian distributions with diagonal co-variance matrices, respectively (Higgins et al., 2017; He et al., 2019; Li et al., 2019a). Such predefined forms would hinder VAEs from larger optimization space (Fang et al., 2019), thus restricting the expressivity of the model (Ding and Gimpel, 2021) and further leading to the posterior collapse (Fang et al., 2019). Therefore, a potential solution is to try more expressive distribution forms for priors and variational posteriors to improve the representation capacity (Fang et al., 2019; Tomczak and Welling, 2018; Ding and Gimpel, 2021).

Attributes	Samples
Positive	this is followed by good movies, great food.
Negative	for me it looks crappy and understaffed.
Present	this restaurant has an excellent view.
Past	i was able to get the delicious sushi!

Table 1: Examples of controlled text generation in second column where sentence attributes indicated by colored words are consistent with user-specified attributes in the first column.

Another challenge of VAEs is poor controllability. The challenge is rooted in the continuous latent variables that hinder VAEs from interpreting the discrete attributes like sentiments or topics (Zhao et al., 2018; Shi et al., 2020). Thus it is difficult to generate text with user-specified attributes, as the examples in Table 1. To approach controlled text generation in variational models, Hu et al. (Hu et al., 2017) propose to disentangle the latent representations by separately modeling discrete attribute and continuous content representations. Nevertheless, it is hard to completely disentangle attribute and attribute-independent content, resulting in poor readability in text generation (Wang et al., 2019; Higgins et al., 2017). A natural choice is to employ discrete representations as each of them could well correspond to one of the discrete attributes. Recent studies also reveal learned discrete representations by K-means and self-organization map (Kohonen,

* Corresponding author. This work was done when Xianghong was an intern at Huawei Noah's Ark Lab.

Findings of the Association for Computational Linguistics: ACL 2022, pages 97 - 111
May 22-27, 2022 ©2022 Association for Computational Linguistics

1995) display great clustering performance and interpretability (van den Oord et al., 2017; Fortuin et al., 2019), showing the potential to be manipulated and split latent space for controlled text generation.

In this paper, we follow the practice of learning discrete representations and propose a new data-driven prior that enjoys the merits of expressivity and controllability. Specifically, we deploy a set of learnable vectors and interpolate the learnable vectors to form the prior, which we call *Dictionary Prior (DPrior)*. Each learnable vector is dubbed an atom in the dictionary. To facilitate generative models with DPrior, dual-form KL-divergence (Dai et al., 2018) is employed to make the prior distribution spanned by dictionary atoms approximate the posterior distribution. Our DPrior is model-agnostic and could be combined with pre-trained models such as BERT/GPT to enrich posterior representations (Li et al., 2020a). To enforce controllability to the DPrior, we separate the dictionary atoms into several disjoint subsets according to the natural language attributes. Then, we propose to employ contrastive learning to incorporate the attribute information, which can cluster different subsets of dictionary atoms into different semantic subspaces.

We demonstrate the superiority of DPrior against recent VAE variants on the language modeling task. We also validate our DPrior in controlled text generation where DPrior shows its effectiveness over several advanced counterparts. The main contributions of this paper can be summarized as:

- We propose an expressive Dictionary Prior (DPrior) within VAEs framework, which consists of learnable dictionary atoms and interpolating the atoms as latent variables.
- DPrior is model-agnostic and can be combined with pre-trained language models. By doing so, DPrior achieves SOTA language modeling performance on four benchmarks.
- We enforce controllability to DPrior by separating dictionary atoms into disjoint subsets and applying contrastive learning to incorporate attribute information.

2 Related Work

Controlled Text Generation Controlled text generation is a task aiming to generate realistic sentences with desired attributes, e.g., sentiments or topics. Most efforts for controlled text generation are based on conditional pre-trained language models (Keskar et al., 2019; Dathathri et al., 2020). CTRL (Keskar et al., 2019) employs a GPT-2 like pre-trained language model and trains it from scratch on a large corpus containing various control codes. Subsequently, controlled generation is accomplished by using the control codes as prompting words. PPLM (Dathathri et al., 2020) seeks to avoid the further training process and combines the GPT-2 model with several simple attribute classifiers whose gradients can update the latent representations.

Another line of work tries to explore limited labeled data via learning latent representations (Hu et al., 2017). Hu et al. (Hu et al., 2017) propose to approach controlled text generation by learning disentangled latent representations including independent content and attribute parts. In this paper, we learn entangled latent representations and approach controlled text generation by separating prior space into several parts.

Expressive Prior and Posterior In VAEs VAEs usually employ simple Gaussian distribution as the prior and spherical Gaussian distributions with diagonal co-variance matrices as the variational posteriors (Higgins et al., 2017; He et al., 2019; Fu et al., 2019). Such predefined forms in traditional formulations hinder VAEs from the expressivity of the model (Ding and Gimpel, 2021), thus further inducing the posterior collapse (Fang et al., 2019).

To improve the representation capacity, some efforts try more expressive priors. MoG-VAE (Ding and Gimpel, 2021) considers a uniform mixture of Gaussians as the prior, Vamp-VAE (Tomczak and Welling, 2018) introduces "Variational Mixture of Posteriors" prior (VampPrior). APo-VAE (Dai et al., 2021) adopts VampPrior to learn a hyperbolic latent space. FlowPrior (Ding and Gimpel, 2021) tries a new prior through normalizing flows. It is noted that VQ-VAE (van den Oord et al., 2017) introduces an auto-regressive prior via learning discrete representations, which enjoys the merits of learnability and expressivity. Nevertheless, the auto-regressive prior has low generation efficiency and no ability of latent manipulation (Fang et al., 2021). In this paper, we propose a data-driven prior via learning discrete representations but have same generation efficiency and the ability of latent variable manipulation to traditional VAEs.

Another line of work is to seek more expressive posteriors. Fang et al. (Fang et al., 2019) adopts implicit posterior representation. APo-VAE (Dai et al., 2021) and our DPrior also employ the implicit posterior representations to match the flexible priors thus further improve representation capacity.

3 Methodology

In this section, we first review the basics of deep generative models in Section 3.1, then introduce Dictionary Prior (DPrior) in Section 3.2 which is built on a set of learnable vectors. We further approach controlled text generation in Section 3.3. The overall illustration of our proposed approach is shown in Figure 1. More details will be explained in the following sections.

3.1 Deep Generative Models

VAEs are one of the most representative deep generative models for language modeling. Given a text $\mathbf{x} = x_{1:T}$ with length T, VAEs seek to infer latent variable $\mathbf{z}$ that explains the observation. Towards this end, VAEs maximize the marginal log-likelihood $\log p_\theta(\mathbf{x})$, which is usually intractable due to the complex true posterior $p(\mathbf{z}|\mathbf{x})$. Consequently an approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ (i.e. the *encoder*) is introduced, and the evidence lower bound (ELBO) of the marginal likelihood is maximized as follows:

$$\begin{aligned}\log p_\theta(\mathbf{x}) \geq & \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \\ & - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})), \quad (1)\end{aligned}$$

where $p_\theta(\mathbf{x}|\mathbf{z})$ represents likelihood function conditioned on $\mathbf{z}$, also known as the *decoder*.

VAEs usually adopt simple Gaussian distribution as the prior and spherical Gaussian distributions with diagonal co-variance matrices as the variational posterior. However, predefined distribution forms in traditional formulations of VAEs restrict representation capacity. As discussed before, we turn to learning an expressive prior via discrete representations instead of predefined prior.

3.2 Data-driven Dictionary Prior

We define the prior via a set of learnable vectors, i.e., $\psi = \{\mathbf{e}_1, ..., \mathbf{e}_m\}$, and each vector is dubbed as a dictionary atom. Intuitively, we could sample one dictionary atom and feed it to the decoder, i.e., $p_\theta(\mathbf{x}|\mathbf{e})$. However, the generation capacity is always restricted by the dictionary size m. To facilitate larger generation capacity, we further introduce a continuous random variable $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_m)^\top$ that follows the Dirichlet distribution parameterized by an m-dimensional vector $\boldsymbol{\gamma}$:

$$\boldsymbol{\pi} \sim Dir(\boldsymbol{\pi}|\boldsymbol{\gamma}) = \frac{\Gamma(\sum_k \gamma_k)}{\Pi_k \Gamma(\gamma_k)} \Pi_k \pi_k^{\gamma_k - 1}. \quad (2)$$

Then we interpolate all dictionary atoms with $\boldsymbol{\pi}$ to form the latent variable: $\mathbf{z} = \sum_i \pi_i \cdot \mathbf{e}_i$. Although atoms in ψ are discrete and finite, the latent variable $\mathbf{z}$ is continuous and has infinitely possible realizations via sampling $\boldsymbol{\pi}$ according to the Dirichlet distribution. We call the prior defined on these dictionary atoms as Dictionary Prior (DPrior), or $p_\psi(\mathbf{z}|\boldsymbol{\gamma})$. Note that $\boldsymbol{\gamma}$ is a hyper-parameter and we set $\boldsymbol{\gamma}$ the same in each dimension. Dirichlet distribution would approximate one hot distribution when $\gamma_k \to 0$, and approximate uniform categorical distribution when $\gamma_k \to \infty$. In general, The smaller γ_k, produces more diverse text from our proposed DPrior.

As part of the network parameters, the dictionary ψ would be differentially updated according to various training samples. Such a data-driven prior would produce larger optimization space, enforcing to learn better representations.

Dual Form of KL divergence It is intractable to deploy vanilla KL divergence to train DPrior as in Equation 1, since learnable discrete atoms in ψ make it difficult to explicitly estimate the density of $p_\psi(\mathbf{z}|\boldsymbol{\gamma})$. To address the issue, we propose to employ its dual form based on Fenchel duality theorem (Rockafellar et al., 1966), which can effectively narrow the distribution gap between the prior $p_\psi(\mathbf{z}|\boldsymbol{\gamma})$ and posterior $q_\phi(\mathbf{z})$ when the density of the priors and/or variational posterior are unknown (Fang et al., 2019; Dai et al., 2021).

Specifically, we follow (Fang et al., 2019) and introduce an auxiliary dual function $v(\cdot)$, parameterized by a neural network with weights φ, to optimize the KL divergence as:

$$\begin{aligned}L_D^{\phi,\psi,\varphi} &= D_{KL}(q_\phi(\mathbf{z})\|p_\psi(\mathbf{z}|\boldsymbol{\gamma})) \quad (3)\\ &= \max_\varphi \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} v_\varphi(\mathbf{z}) - \mathbb{E}_{\mathbf{z}\sim p_\psi(\mathbf{z}|\boldsymbol{\gamma}))} \exp(v_\varphi(\mathbf{z})),\end{aligned}$$

where $q_\phi(\mathbf{z}) = \int q(\mathbf{x}) q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{x}$ is the *aggregated posterior*. To make the posterior match the expressive DPrior, we also employ implicit posterior representations as (Fang et al., 2019). Specially, we adopt white noise $\epsilon_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and concatenate it with hidden representations of $\mathbf{x}$ to obtain i-th latent variable as $\mathbf{z}_i = G_\phi(\mathbf{x}, \epsilon_i)$.

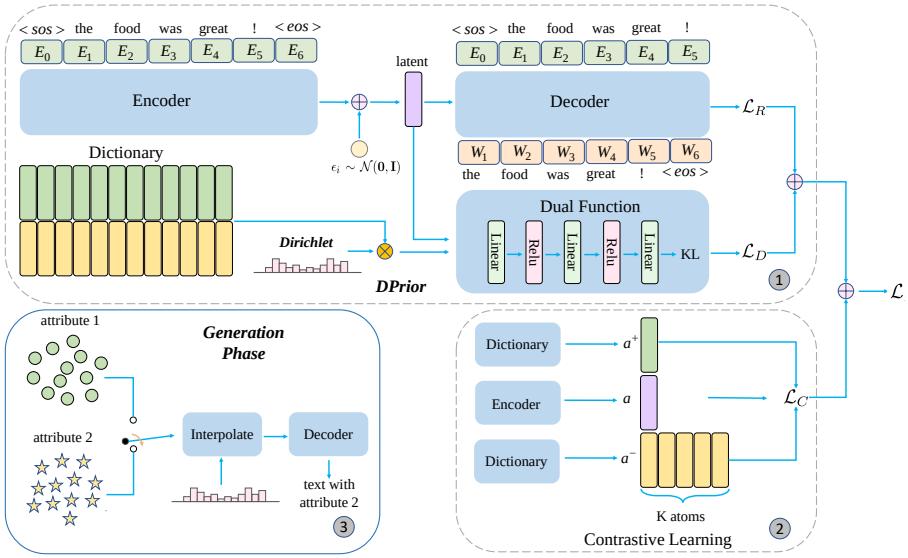


Figure 1: The overall illustration of our proposed method, which consists of an encoder-decoder network, a learnable dictionary, and a deep dual function network. $\otimes$ and $\oplus$ represent interpolation and sum operator respectively. Different colors in the dictionary denote different attributes. Block 1 represents the training process of DPrior, consisting of the reconstruction loss $\mathcal{L}_R$ and the dual-form KL divergence $\mathcal{L}_D$. Block 2 denotes contrastive learning applied to the dictionary with the contrastive loss $\mathcal{L}_C$. Block 3 denotes the controlled text generation after training.

During training, we choose γ near 0 as it consistently performs better than other values in our experiments. Together with the reconstruction loss, i.e., $\mathcal{L}_R^{\phi,\theta} = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z})$, the objective function of DPrior for language modeling can be summarized as:

$$\min_{\phi,\psi,\theta} \max_{\varphi} \mathcal{L}_R^{\phi,\theta} + \beta_1 * \mathcal{L}_D^{\varphi,\phi,\psi}, \tag{4}$$

where β_1 is a regularization parameter.

Combined with Pre-trained Models Our DPrior is model-agnostic and could be combined with various neural networks such as LSTM (Hochreiter and Schmidhuber, 1997) and Transformer (Vaswani et al., 2017). To improve representation capacity, we propose the combination of DPrior and a large-scale pre-trained deep latent variable model, i.e., OPTIMUS (Li et al., 2020a), which adopts the pre-trained BERT and GPT-2 as the encoder and decoder, respectively. Since extra large-scale text corpus was exploited, more diverse and even out-of-domain sentences that exploit more words are able to be generated.

3.3 DPrior for Controlled Text Generation

In this section, we enforce interpretability and controllability to DPrior to approach controlled text generation. Specifically, we separate the dictionary ψ into L disjoint subsets, i.e. $\psi_1, \psi_2, ..., \psi_L$, given L different attributes in the dataset. For example, we have two subsets to represent positive and negative sentiments as in Figure 1. The number of atoms in each subset is set according to the attribute proportion in the dataset. To accomplish controlled text generation, we can then choose a certain dictionary subset and interpolate atoms in this subset as the latent variable $\mathbf{z}$ for decoder generation.

To effectively incorporate the attribute information into dictionary atoms, we propose to employ *contrastive learning* such that sentences generated from a certain subset accurately correspond to the desired attribute. During the training of DPrior, The semantic space of the dictionary could be gradually clustered into several parts according to the natural language attributes.

Contrastive Learning for DPrior Given a latent variable $\mathbf{z}$ from encoder $q_\phi(\mathbf{z}|\mathbf{x})$ with its attribute label $c \in \{1, ..., L\}$, we denote $\mathbf{z}$ as an anchor a. Therefore, atoms in the subset ψ_c with the same attribute constitute *positive samples* (denoted as a^+) of anchor a, and atoms in other subsets $\psi_{\{-c\}}$ are *negative samples* (denoted as a^-) of anchor a. A contrastive loss (van den Oord et al., 2018; Hoffer and Ailon, 2015) is a distance metric to enforce the anchor a to be similar to positive samples a^+ and dissimilar to negative samples a^-. With the supervised attribute information contained in anchor a, the positive samples would learn to cluster into the same semantic subspace with the anchor while negative samples into other seman-

tic subspaces. The contrastive loss will gradually enlarge the gap among different subspaces.

As shown in Block 2 of Figure 1, we employ InfoNCE loss (van den Oord et al., 2018) where we randomly sample one positive sample from ψ_c and K negative samples from $\psi_{\{-c\}}$ for each anchor a. Then the objective is to produce the log loss of a (K+1)-way softmax-based classifier that tries to classify a as a^+:

$$\mathcal{L}_C^{\phi,\psi} = -\mathbb{E}_S \log \frac{e^{\tau \cdot a \cdot a^+}}{e^{\tau \cdot a \cdot a^+} + \sum_{i=1}^{K} e^{\tau \cdot a \cdot a_i^-}}, \quad (5)$$

where $S = \{a, a^+, a^-\}$ and τ is a temperature hyper-parameter and we set $\tau = 1$ in all experiments. Together with the loss function of DPrior introduced in Equation 4, the overall objective for controlled text generation can be summarized as:

$$\min_{\phi,\psi,\theta} \max_{\varphi} \mathcal{L}_R^{\phi,\theta} + \beta_1 * \mathcal{L}_D^{\varphi,\phi,\psi} + \beta_2 * \mathcal{L}_C^{\phi,\psi}, \quad (6)$$

where $\mathcal{L}_R$ denotes the reconstruction loss, $\mathcal{L}_D$ denotes the dual-form KL-divergence, $\mathcal{L}_C$ denotes the contrastive loss, β_1 and β_2 are the hyper-parameters.

Controlled Text Generation from DPrior After the training phase of DPrior, as Block 3 of Figure 1, given any attribute label $c \in \{1, ..., L\}$, we select all atoms from the corresponding subset ψ_c, sample $\boldsymbol{\pi}$ from the Dirichlet distribution, interpolate these atoms with $\boldsymbol{\pi}$ to produce a latent variable $\mathbf{z}$, and finally feed it to the decoder for text generation. In this way, controlled text generation with the user-specified attributes can be achieved.

4 Experiments

In this section, we apply DPrior model to two tasks: (i) language modeling, where DPrior shows its advantage in expressive prior in comparison with state-of-the-art VAE methods. (ii) controlled text generation, where DPrior shows its superiority in controllability with desired attributes. We also conduct a series of analyses and visualizations to shed more light on the proposed approach.

4.1 Language Modeling

Following (Li et al., 2020a), we consider four benchmark datasets of language modeling for evaluation: Penn Tree (Marcus et al., 1993), SNLI (Bowman et al., 2015), Yahoo Answers (Xu and Durrett, 2018) and Yelp corpora (Yang et al., 2017). A summary of dataset statistics is shown in Appendix A.

Baselines We compare the proposed DPrior against following baselines: (i) auto-regressive models such as LSTM-LM (Mikolov et al., 2010) and GPT-2 (Radford et al., 2019). (ii) VAE (Kingma and Welling, 2014) with simple Gaussian prior, and its advanced variants for bettering training and avoiding posterior collapse, including Annealing VAE (Bowman et al., 2016), Free Bits (FB)-VAE (Kingma et al., 2017), Lag-VAE (He et al., 2019), and AE-FB (Li et al., 2019a). (iii) VAEs with expressive prior choices, including MoG-VAE (Ding and Gimpel, 2021), Vamp-VAE (Tomczak and Welling, 2018), APo-VAE (Dai et al., 2021), FlowPrior (Ding and Gimpel, 2021). (iv) iVAE (Fang et al., 2019) considers implicit posterior representation instead of explicit form. (v) OPTIMUS (Li et al., 2020a), a large-scale pre-trained VAE model.

Metrics We evaluate language modeling from two perspectives: Generation capacity measured by *perplexity* (PPL) and representation learning capacity measured by Active Units (AU) of $\mathbf{z}$ and its Mutual Information (MI). Note that LSTM-LM and GPT-2 has exactly PPL, while VAEs do not. Following (Fang et al., 2019), our calculation of PPL is slightly different from exact PPL in two ways: (i) we approximate $\log p(\mathbf{x})$ to report PPL; (ii) the KL term in the bound is estimated via samples, rather than a closed-form. We also report results with ELBO, KL, and Reconstruction in Appendix B.

Main Results As the results shown in Table 2, our proposed DPrior achieves state-of-the-art language modeling performance in terms of PPL and MI in all datasets. In comparison with vanilla VAE and its variants in the middle block that employ explicit posterior representations, iVAE, APo-VAE, and DPrior that adopt implicit posterior representations achieve better performance, indicating the importance of expressive posterior representations. Moreover, our DPrior achieves further improvements upon iVAE, which we attribute to the proposed data-driven prior and improving the representation capacity.

In comparison with VAEs implemented by LSTM layers in the middle block of Table 2, VAEs based on the OPTIMUS framework in the bottom block achieve impressive results by large margins.

	Dataset	PTB			Yelp			Yahoo			SNLI		
	Methods	LM PPL↓	Repr. MI↑	Repr. AU↑	LM PPL↓	Repr. MI↑	Repr. AU↑	LM PPL↓	Repr. MI↑	Repr. AU↑	LM PPL↓	Repr. MI↑	Repr. AU↑
	LSTM-LM†	100.47	-	-	42.60	-	-	60.75	-	-	21.44	-	-
	GPT-2†	24.23	-	-	23.40	-	-	22.00	-	-	19.68	-	-
LSTM	VAE§	101.39	0.01	0	40.56	0.00	0	61.52	0.00	0	21.67	0.03	1
LSTM	Annealing-VAE†	101.40	0.00	0	40.39	0.13	1	61.21	0.00	0	21.50	1.45	2
LSTM	Lag-VAE†	99.83	0.83	4	39.84	2.16	12	59.77	2.90	19	21.16	1.38	5
LSTM	FB-VAE$^{\S}(\lambda = 5.0)$	101.42	4.80	4				62.78	5.00	3	21.58	4.95	6
LSTM	AE-FB$^{\S}(\lambda = 5.0)$	96.86	5.31	32	47.97	7.89	32	59.28	8.08	32	21.64	7.71	32
LSTM	MoG-VAE$^{\diamondsuit}$	97.50	0.68	32				64.60	0.00	0	28.05	0.41	1
LSTM	Vamp-VAE$^{\diamondsuit}$	97.83	0.72	32				74.81	0.00	0	25.98	0.00	0
LSTM	Flow-Prior$^{\diamondsuit}$	93.58	2.83	31				68.29	0.61	25	26.19	3.16	32
LSTM	APo-VAE*	53.02	4.50	32	32.91	6.20	32	46.61	4.90	32			
LSTM	iVAE‡	53.44	12.20	32	36.88	11.00	32	47.93	10.70	32	7.40	9.93	32
LSTM	***DPrior (Ours)***	**46.08**	**12.59**	32	**32.79**	**11.35**	32	**45.18**	**10.93**	32	**6.44**	**10.02**	32
OPTIMUS	AE-FB$^{\dagger}(\lambda = 1.0)$	35.53	8.18	32	24.59	9.13	32	24.92	9.18	32	29.63	9.20	32
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.5)$	26.69	7.64	32	22.79	7.67	32	23.11	8.85	32	16.67	8.89	32
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.05)$	23.58	3.78	32	21.99	2.54	32	22.34	5.34	32	13.47	3.49	32
OPTIMUS	iVAE	15.49	15.86	32	15.44	15.07	32	15.04	12.52	32	5.65	14.28	32
OPTIMUS	***DPrior (Ours)***	**14.74**	**15.96**	32	**14.52**	**17.05**	32	**14.67**	**12.99**	32	**5.54**	**14.42**	32

Table 2: Language modeling performance comparison on PTB, Yelp, Yahoo, and SNLI datasets. "LSTM" indicates autoencoder architectures are built with two-layer LSTMs, while "OPTIMUS" employs pre-trained BERT and GPT-2 as the encoder and decoder. †: results from (Li et al., 2020a). ‡: results from (Fang et al., 2019). §: results from (Li et al., 2019a). *: results from (Dai et al., 2021). $^{\diamondsuit}$: results from (Ding and Gimpel, 2021). "-" indicates the models are improper to report these values. Empty cells indicate the results were not reported in the literature.

A potential explanation is that the latter could incorporate natural language understanding knowledge into generation tasks, and then learn a more structured semantic latent space with the combination of strengths of VAE, BERT, and GPT-2. Overall, DPrior achieves the lowest PPL and highest MI among all datasets based on the OPTIMUS framework, which further verifies the superiority of the data-driven prior via learnable dictionary atoms.

Analysis We conduct a set of analyses including the influence of the dictionary size, atom analysis, latent interpolation, and sentence transfer. We find that the results on the PTB dataset are insensitive to the size of the dictionary. To gain a comprehensive understanding of the prior, we also conduct atoms analysis. Specifically, we randomly choose an atom from the dictionary and search top-9 nearest atoms via euclidean distance to this atom. Then we feed the sampled atom and top-9 nearest atoms to the decoder for sentence generation. The results are illustrated by the red and blue sentences in Table 3. We conduct latent interpolation to demonstrate DPrior could learn a smooth latent space. We also conduct sentence transfer to imply DPrior has great ability of high-level sentence editing in latent space. More details are illustrated in Appendix C.

	a dog is running on the plant
1	a chicken is chasing off animals.
2	a girl flings a dog on water.
3	a dog is on athletic grounds.
4	a small white dog runs under the grass.
5	a dog goes alone from his village.
6	a dog plays with a play on a grassy field.
7	the brown dog is attacking other people.
8	three puppies are eating right inside.
9	a black pup on monkey jump.

Table 3: Atom analysis on SNLI dataset.

4.2 Controlled Text Generation

In this section, we conduct controlled text generation on the Yelp (Li et al., 2018a) and Arxiv (Sergio, 2019) datasets. Yelp dataset (Yelp-s) consists of business reviews that are labeled as either positive or negative according to their sentiment. To gain the tense attributes (present or past) from Yelp, we use the Stanford Parser to extract the main verb from a sentence to constitute a new dataset (Yelp-t). We also consider the combination of sentiment and tense attributes (Yelp-st) for multi-set controlled text generation. Arxiv dataset extracts the abstract from `arxiv` articles regarding three topics: artificial intelligence, computer vision, and natural language process. Appendix A shows the detailed dataset statistics.

Baselines We compare the proposed DPrior with constrastive loss (denoted as DPrior+c) against several baselines:(i) **CVAE**, the conditional-VAE model (Sohn et al., 2015) where each attribute is

	Methods	Yelp-s			Yelp-t			Yelp-st		
		Acc↑	PPL↓	Dist↑	Acc↑	PPL↓	Dist↑	Acc↑	PPL↓	Dist↑
Transformer	CVAE	85.2	5.87	**0.384**	86.9	5.66	0.350	75.6	5.75	0.270
	CVAE+c	96.9	5.72	0.354	98.3	5.73	**0.368**	96.6	5.69	0.263
	Semi-VAE	96.8	5.82	0.375	98.2	5.66	0.351	94.2	5.77	0.282
	Disentangle	97.7	5.81	0.377	98.5	5.63	0.343	94.5	5.82	**0.297**
	DPrior+c	**99.2**	**5.45**	0.313	**99.9**	**5.63**	0.298	**98.4**	**5.54**	0.195
	Reference	98.4	6.01	0.552	99.5	5.94	0.560	98.0	5.93	0.481
Pre-train	GPT-2	96.4	**5.00**	0.436	97.7	**4.93**	0.422	93.2	**5.05**	0.359
	CVAE+c	95.1	6.02	**0.629**	96.0	5.95	**0.633**	88.8	5.94	**0.556**
	DPrior+c	**98.6**	5.82	0.498	**99.4**	5.92	0.467	**95.1**	5.96	0.489

Table 4: Automatic evaluation results of controlled text generation on Yelp dataset. "Transformer" indicates autoencoder architectures are built with transformer layers, while "pre-train" employs pre-trained models such as GPT-2 or OPTIMUS. Reference represents samples from the test dataset. ↑/↓ means the larger/smaller the better.

Methods	sentiment				tense			
	Acc↑	Agree↑	Flu↑	Agree↑	Acc↑	Agree↑	Flu↑	Agree↑
Reference	4.32	80.3%	4.30	67.2%	4.88	96.3%	4.43	68.4%
GPT-2	4.15	78.5%	4.12	64.2%	4.74	92.3%	4.19	65.1%
CVAE+c	4.30	79.8%	4.08	64.3%	4.76	92.8%	3.89	65.5%
DPrior+c	**4.51**	**82.6%**	**4.23**	**66.6%**	**4.90**	**97.2%**	**4.39**	**69.2%**

Table 5: Human evaluation results of controlled text generation on Yelp dataset in terms of sentiment and tense attributes. Reference represents samples from the test dataset. ↑ means the larger the better.

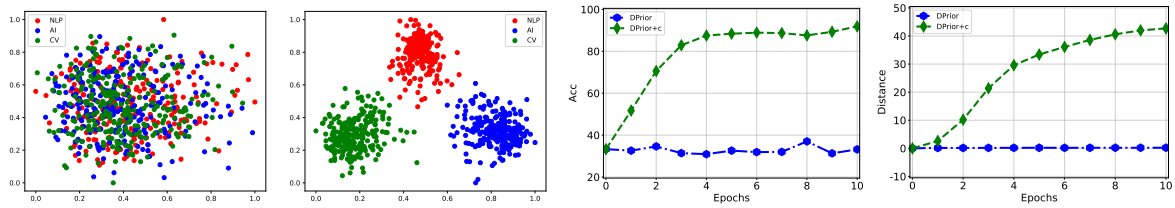
Methods	Arxiv		
	Acc↑	PPL↓	Dist↑
Reference	86.2	3.79	0.556
GPT-2	81.8	**3.08**	0.377
CVAE+c	95.8	4.39	0.555
DPrior+c	**98.7**	4.28	**0.575**

Table 6: Automatic evaluation results of controlled text generation on Arxiv dataset. Reference represents samples from the test dataset. ↑/↓ means the larger/smaller the better.

represented by a separated Gaussian distribution. (ii) **CVAE+c**, which applies constrastive loss as DPrior+c to the conditional-VAE model. (iii) **Disentagle** (Hu et al., 2017), which disentangles the latent representations into content and attribute parts for controlled text generation; (iv) **Semi-VAE** (Kingma et al., 2014), semi-supervised VAE model with independent discrete and continuous latent variables; (v) a fine-tuned **GPT-2** (Radford et al., 2019) model using attribute labels as the the prompt. We deploy the test dataset as **Reference** for comparison. To demonstrate the influence of constrative loss, we also consider an ablation where no constrative loss is applied on DPrior. Implementation details are discussed in Appendix D.

Metrics We evaluate the performance of controlled text generation from three aspects, i.e., *controllability*, *fluency* and *diversity*. For controllability, we fine-tune a BERT classifier (Devlin et al., 2019) on the training data as attribute predictor, which measures the accuracy (Acc) of correctly generated sentences with desired attributes. Note that the BERT classifier achieves an accuracy of 98.4%, 99.5%, 98.0%, and 86.2% on Yelp-s, Yelp-t, Yelp-st, and Arxiv respectively, being a good automatic evaluator. For fluency, we adopt a pre-trained GPT-2 model (Radford et al., 2019) as the fluency evaluator, which takes the generated sentences as input and returns the corresponding perplexity scores (PPL). For diversity, distinct metric (Dist) is employed which calculates the number of distinct bigrams in generated sentences (Li et al., 2016). A better-controlled generation generally has higher Acc, lower PPL, and higher Dist.

Main Result The results are listed in Table 4, 5, and 6, including automatic evaluation and human evaluation. From the results, we can conclude that: (i) in terms of **controllability**, our proposed DPrior+c consistently achieves the best generation accuracies (Acc) on all four datasets via either automatic evaluation or human evaluation. (ii) In terms of **fluency**, there is no doubt that GPT-2 produces the best PPL scores since it is pre-trained on language modeling tasks. Though not the best, our



(a) DPrior (b) DPrior+c (c) Accuracy of items (d) Distance between subsets

Figure 2: Illustration of subspace separations on the Arxiv dataset.

DPrior+c also achieves better PPL scores against other methods. Note that fluency is a very subjective metric, and the use of the GPT-2 PPL score may not be a reliable measurement. We also conduct human evaluation, reported in Table 5, and our DPrior+c always achieves the best fluency excluding the reference. (iii) In terms of **diversity**, our DPrior+c can also attain comparable distinct metrics (Dist) against other methods. Note that DPrior+c achieves the best distinct metrics (Dist) in the Arxiv dataset, as shown in Table 6. With the help of pre-trained OPTIMUS, DPrior+c could generate more diverse long sentences with more words exploited in the vocabulary.

In comparison with DPrior+c, DPrior always attains the worst controllability as shown in the top block of Table 4, which can be explained that dictionary atoms cannot receive supervised information without contrastive learning. We also find that Transformer-based models always achieve a little better controllability but worse diversity compared with pretrain-based models, as shown in Table 4. A possible explanation is that pretrain-based models can always leverage extra large-scale text corpus and generate out-of-domain sentences that exploit more words, even their attributes cannot be distinguished by the BERT classifier.

Visualizations To gain a better understanding of how contrastive learning benefits the prior subspace separations, we visualize dictionary atoms with different attributes. Specifically, we focus on the Arxiv dataset and sample all atoms from DPrior and DPrior+c models. We reduce the dimensionality from 32 to 2 using PCA and plot them in Figure 2. As shown in Figure 2(a), the subspace for AI, CV, and NLP parts are highly overlapped without contrastive loss. This can also explain the poor controllability of DPrior in Table 4. By contrast, DPrior+c model clearly separates the prior space into the AI, CV, and NLP parts, as shown in Figure 2(b), indicating that the contrastive loss could effectively enlarge the gap among different subspaces. Therefore, text generated from the interpolation of the disjoint dictionary subsets will be highly consistent with the desired attributes.

We further analyze the advantages of contrastive learning from two perspectives: the accuracy of dictionary atoms, where we directly feed all atoms to the decoder and measure the accuracy of predicted attributes by the BERT classifier; and the distance between the mean of the three disjoint subsets. As shown in Figure 2(c) and Figure 2(d), when no contrastive loss is applied, the atom accuracy and subset distance keep almost unchanged, i.e., 33% and 0 respectively. By contrast, when contrastive learning is deployed, the atom accuracy quickly increases to 91.9%, and the distance gradually enlarges during the model training.

Other Analysis We show some sampled sentences from DPrior+c including short controlled text generation trained on Yelp dataset in terms of sentiment, tense, and the combination of them, and long controlled text generation trained on Arxiv dataset. All samples can be found in Appendix E.

We also analyze the influence of Dirichlet distribution for text generation in terms of controllability, fluency, and diversity. Details can be found in Appendix G.

5 Conclusion

In this paper, we propose the Dictionary Prior (DPrior), a new data-driven prior that enjoys the merits of expressivity and controllability. The proposed prior deploys a set of learnable vectors dubbed as dictionary atoms and interpolate the atoms to form the prior. We apply dual-form KL-divergence to make the prior distribution spanned by dictionary atoms approximate the posterior distribution. Contrastive learning is further deployed to the disjoint dictionary subsets to enable controllability and interpretability. Empirical results on benchmark datasets demonstrate the superiority of our approach in both language modeling and controlled text generation.

Nevertheless, the proposed approach has limitations. While the Gaussian distribution employed in standard VAEs has an infinite support region, the support region of DPrior is finite as it corresponds to the convex hull of the dictionary atoms. Therefore, future work considers extending our framework to the more general infinite support region. We will also apply DPrior to more text generation tasks like poetry generation (Yi et al., 2020) and machine translation (Li et al., 2020b, 2019b).

6 Acknowledgements

We thank the anonymous reviewers for their insightful comments.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

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

Bo Dai, H. Dai, Niao He, Weiyang Liu, Z. Liu, Jianshu Chen, Lin Xiao, and Le Song. 2018. Coupled variational bayes via optimization embedding. In *NeurIPS*.

Shuyang Dai, Zhe Gan, Yu Cheng, Chenyang Tao, L. Carin, and Jingjing Liu. 2021. Apo-vae: Text generation in hyperbolic space. In *NAACL*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, J. Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *ICLR*.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Xiaoan Ding and Kevin Gimpel. 2021. Flowprior: Learning expressive priors for latent variable sentence models. In *NAACL*.

Le Fang, C. Li, Jianfeng Gao, Wen Jun Dong, and Changyou Chen. 2019. Implicit deep latent variable models for text generation. In *EMNLP*.

Xianghong Fang, Haoli Bai, Jian Li, Zenglin Xu, Michael Lyu, and Irwin King. 2021. Discrete autoregressive variational attention models for text modeling. In *IJCNN*, pages 1–8.

Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and G. Rätsch. 2019. Somvae: Interpretable discrete representation learning on time series. In *ICLR*.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Çelikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. In *NAACL-HLT*.

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

I. Higgins, Loïc Matthey, A. Pal, Christopher P. Burgess, Xavier Glorot, M. Botvinick, S. Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*.

E. Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *SIMBAD*.

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

N. Keskar, Bryan McCann, L. Varshney, Caiming Xiong, and R. Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Diederik P. Kingma, S. Mohamed, Danilo Jimenez Rezende, and M. Welling. 2014. Semi-supervised learning with deep generative models. In *NeurIPS*.

Diederik P. Kingma, Tim Salimans, and Max Welling. 2017. Improved variational inference with inverse autoregressive flow. In *NeurIPS*.

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

Teuvo Kohonen. 1995. Self-organizing maps. In *Springer Series in Information Sciences*.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019a. A surprisingly effective fix for deep latent variable modeling of text. In *EMNLP*.

Chunyuan Li, Xiang Gao, Yuan Li, Xiujun Li, Baolin Peng, Yizhe Zhang, and Jianfeng Gao. 2020a. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *EMNLP*.

J. Li, Michel Galley, Chris Brockett, Jianfeng Gao, and W. Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*.

Jian Li, Xing Wang, Baosong Yang, Shuming Shi, Michael R Lyu, and Zhaopeng Tu. 2020b. Neuron interaction based representation composition for neural machine translation. In *AAAI*.

Jian Li, Baosong Yang, Zi-Yi Dou, Xing Wang, Michael R Lyu, and Zhaopeng Tu. 2019b. Information aggregation for multi-head attention with routing-by-agreement. In *NAACL-HLT*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018a. Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *NAACL-HLT*.

Juntao Li, Yan Song, Haisong Zhang, Dongmin Chen, Shuming Shi, Dongyan Zhao, and Rui Yan. 2018b. Generating classical chinese poems via conditional variational autoencoder and adversarial training. In *EMNLP*.

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

Arya D. McCarthy, Xian Li, Jiatao Gu, and Ning Dong. 2020. Addressing posterior collapse with mutual information for improved variational neural machine translation. In *ACL*.

Tomas Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*.

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

R Tyrrell Rockafellar et al. 1966. Extension of fenchel'duality theorem for convex functions. *Duke mathematical journal*.

Gwenaelle Cunha Sergio. 2019. Arxivabstitledataset: Extracting abstract and title dataset from arxiv articles. *Github repository*.

Harshil Shah and David Barber. 2018. Generative neural machine translation. In *NeurIPS*.

Xin Sheng, Linli Xu, Junliang Guo, Jingchang Liu, Ruoyu Zhao, and Yinlong Xu. 2020. Introvnmt: An introspective model for variational neural machine translation. In *AAAI*.

Wenxian Shi, Hao Zhou, Ning Miao, and Lei Li. 2020. Dispersed exponential family mixture vaes for interpretable text generation. In *ICML*.

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

Jakub M. Tomczak and M. Welling. 2018. Vae with a vampprior. In *AISTATS*.

Aäron van den Oord, Y. Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv*.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *NeurIPS*.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *NeurIPS*.

B. Wu, Mengyuan Li, Z. Wang, Yifu Chen, Derek Wong, Qihang Feng, Junhong Huang, and Baoxun Wang. 2020. Guiding variational response generator to exploit persona. In *ACL*.

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

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

Xiaoyuan Yi, Ruoyu Li, C. Yang, Wenhao Li, and Maosong Sun. 2020. Mixpoet: Diverse poetry generation via learning controllable mixed latent space. In *AAAI*.

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. 2018. Unsupervised discrete sentence representation learning for interpretable neural dialog generation. In *ACL*.

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

A Dataset Statistics

We list the data statistics of all experiments in Table 7. PTB, Yelp, Yahoo, and SNLI datasets are used in the language modeling experiments in Section 4.1. Yelp-s, Yelp-t, Yelp-st, and Arxiv datasets are used in the controlled text generation experiments in Section 4.2.

B Language Modeling Results

The language modeling performance was evaluated by perplexity(PPL), Mutual Information(MI), Active Units(AU), Evidence Lower Bound(ELBO), KL divergence(KL), and Reconstruction(Rec) on PTB, SNLI, Yelp, and Yahoo datasets are shown in Table 8 and 9.

C Analysis on Language Modeling

The Influence of Dictionary Size To analyze how the dictionary size m influences the language modeling performance, we vary $m = 2^k, k \in \{8, 9, 10, 11, 12, 13, 14, 15\}$, and conduct experiments on the PTB dataset. The curves shown in Figure 3 present slight fluctuations in terms of PPL, MI, ELBO, and Rec, indicating the experiment results are insensitive to the size of the dictionary. We set m to 2048 for all language modeling experiments in Table 2, 8 and 9 for the highest MI.

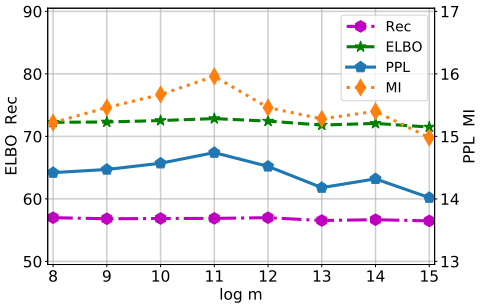


Figure 3: Influence of various dictionary sizes m for language modeling on PTB dataset.

Atoms Analysis To gain a better understanding of the prior space, we conduct atoms analysis on the SNLI dataset, i.e., we randomly choose an atom from the dictionary and search top-9 nearest atoms via euclidean distance to this atom, and then feed the sampled atom and top-9 nearest atoms to the decoder to obtain red and blue sentences respectively, as shown in Table 3 and 10, which show similar semantics, grammar and text length are well clustered in the prior space.

Latent Interpolation To demonstrate DPrior can learn a smooth latent space that captures sentence semantics, we implement linear interpolation between latent vectors on the SNLI dataset, i.e., we take two sentences $\mathbf{x}_1$ and $\mathbf{x}_2$, and use their posterior as the latent features $\mathbf{z}_1$ and $\mathbf{z}_2$, respectively. We interpolate a path $\mathbf{z}_\tau = \mathbf{z}_1 \cdot (1 - \tau) + \mathbf{z}_2 \cdot \tau$ with τ increases from 0 to 1 by a step size of 0.1. As shown in Table 11, the interpolated sentences using greedy decoding conditioned on $\mathbf{z}_\tau$ exhibit smooth semantic evolution.

Sentence Transfer To testify the ability of high-level sentence editing in latent space, we also conduct a one arithmetic latent vector operation on the SNLI dataset. Specially, given source sentence $\boldsymbol{x}_A$ and target sentence $\boldsymbol{x}_B$, the goal is to re-write the input sentence $\boldsymbol{x}_C$ as output in analogy to the transition from $\boldsymbol{x}_A$ to $\boldsymbol{x}_B$. We take encoded latent features $\mathbf{z}_A, \mathbf{z}_B, \mathbf{z}_C$ from $\boldsymbol{x}_A, \boldsymbol{x}_B, \boldsymbol{x}_C$, then apply the arithmetic operator $\mathbf{z}_D = \mathbf{z}_B - \mathbf{z}_A + \mathbf{z}_C$, and generate $\boldsymbol{x}_D$ conditioned $\mathbf{z}_D$ using greedy decoding. As shown in Table 12, two style transitions are well achieved, i.e., from singular to plural subject and from daily-life activity to sport, indicating DPrior can well support the sentence editing.

D Implementations for Controlled Text Generation

We implement all the baselines on our own under the same protocols as there is hardly any reference code for controlled text generation. For transformer-based models, reported in the top block of Table 4, all encoders and decoders are stacked by two transformer layers. These models share the same hyper-parameter settings, including the dimension of latent space, word embedding, and self-attention module. The dimension of latent variable and dictionary atom is set to 32. Adam (Kingma and Ba, 2015) optimizer is employed with an initial learning rate of 0.001. Among pretrained-based models in the bottom block, CVAE+c and DPrior+c adopt OPTIMUS framework (Li et al., 2020a) that employs BERT as the encoder and GPT-2 as the decoder with an initial learning rate of 1e-5. GPT-2 model is fine-tuned on the above datasets with an initial learning rate of 1e-5 directly. We prepend the attribute label words (e.g., positive, negative) to each sentence such that GPT-2 learns to treat them as prompt words. For Yelp-s, Yelp-t, and Yelp-st datasets, the size of the subset for each attribute in the dictionary is set to 2048, and $\gamma = 1/2^9$, and

we sample 1000 sentences each attribute for automatic evaluation. Similarly, the size of each subset in the dictionary is set to 256 for Arxiv dataset, and $\gamma = 1/2$, and we sample 200 sentences each attribute for automatic evaluation.

E Case Study on Controlled Text Generation

We show some sampled sentences from DPrior+c trained on the Yelp dataset in terms of sentiment and tense, and the combination of them. Each attribute is paired with two sentences, and we highlight the corresponding salient words in Table 13. We also choose three long controlled text generations from DPrior+c trained on Arxiv dataset in Table 14.

F Human evaluation for controlled text generation

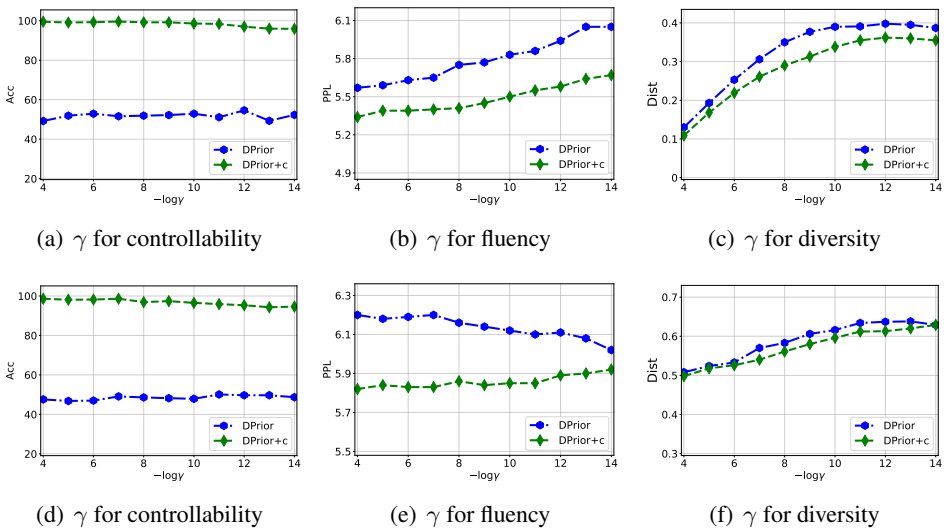
We also conduct human evaluation for the controlled text generation besides automatic evaluation. Due to the limited budgets, here we only compare DPrior+c with Reference, GPT-2, CVAE+c, as shown in Table 5. And we experiment on the Yelp-s and Yelp-t datasets in terms of sentiment and tense attributes. We randomly select 50 samples for each attribute, so there is a total of 200 sentences from each model.

Four annotators with well linguistic background were invited to assess each sentence with desired attributes in a blind manner. The evaluation is on a scale of 1-5 regarding two criteria: accuracy and fluency. Better controlled generation would come with higher accuracy and higher fluency. For example, given a generated sentence *"the price is great and i recommend them!"* with desired *"positive"* sentiment, the accuracy scores [5, 5, 5, 5] were annotated as the sentiment of the sentence could be easily assessed. When it is hard to determine the sentiment of the sentence, annotators might differ their opinions. An example is that [3, 2, 3, 4] were annotated for the sentence *"this was absolutely the first time for me."* with desired *"negative"* sentiment. The fluency scores were assessed in the same manner. Each sentence was reviewed by four judges and the average scores are reported in Table 5. We can see that our DPrior+c achieves the best accuracy, as well as best fluency score except for the Reference. We also set an agreement metric on accuracy and fluency via the percentage of the scale that most annotators agree with. For annotated scores [5, 5, 5, 5] and [3, 2, 3, 4], the agreement would be 100% and 50%, respectively. As seen, humans have a higher agreement when the model performance is high.

G Influence of Dirichlet Distribution

As γ in Equation 2 determines the density of the Dirichlet distribution which further determines the interpolation coefficients $\boldsymbol{\pi}$, here we analyze its influences on text generation from three aspects, i.e., controllability, fluency, and diversity as in the main results in Section 4.2. We vary $\gamma = 1/2^j, j \in \{4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$, and conduct controlled text generation on the Yelp-s dataset on the transformer-based architecture. We sample 2000 sentences for each γ and employ metrics introduced in Section 4.2 for automatic evaluation. As shown in Figure 4(a), when we set a comparatively large value to γ, the DPrior+c model achieves great performance on controllability, while DPrior gains very poor accuracy, indicating the importance of contrastive learning in our framework. We also take generation fluency into consideration which is measured by GPT-2 PPL score. As in Figure 4(b), the PPL score increases gradually on both models when γ declines, showing larger γ would lead to more fluent generations. Finally, the influence of γ on generation diversity is depicted in Figure 4(c). We can see the two models have similar trends, i.e., the diversity evaluated by Dist increases rapidly when γ decreases from $1/2^4$ to $1/2^{12}$, then diversity has a slight decline. Comprehensively considering the controllability, fluency, and diversity of text generation, we set $\gamma = 1/2^9$ for all experiments on Table 4.

We also analyze the influence of Dirichlet distribution on the OPTIMUS-based architecture that could leverage extra large-scale text corpus. The most salient change is that the diversity measured by Dist significantly increases from 0.1 to 0.5 when γ equals $1/2$, as shown in Figure 4(c) and Figure 4(f), indicating the combination of DPrior and the pre-trained model could generate out-of-domain sentences that exploit more words. In terms of controllability, the OPTIMUS-based architecture exhibits the same trend but slightly lower controllability, as illustrated in Figure 4(a) and Figure 4(d). In terms of fluency, shown in Figure 4(e), OPTIMUS-based architecture presents more similar fluency to the test dataset as reported in Table 4.



(a) γ for controllability (b) γ for fluency (c) γ for diversity

(d) γ for controllability (e) γ for fluency (f) γ for diversity

Figure 4: Influence of Dirichlet distribution on text generation controllability, fluency and diversity. (a) (b) (c) are transformer-based, (d) (e) (f) are OPTIMUS-based.

Dataset	Attributes	#Train	#Dev	#Test	#Vocab	Max-Length	Mean-Length
PTB (Marcus et al., 1993)	None	42068	3370	3761	10000	82	21.1
Yelp (Yang et al., 2017)	None	100000	10000	10000	19994	200	96.0
Yahoo (Yang et al., 2017)	None	100000	10000	10000	19998	200	78.8
SNLI (Bowman et al., 2015)	None	100000	10000	10000	9987	70	9.7
Yelp-s (Li et al., 2018a)	Negative	177218	2,000	500	9355	15	8.9
	Positive	266041	2,000	500			
Yelp-t (Li et al., 2018a)	Present	298524	2594	577	9355	15	8.8
	Past	133460	1290	394			
Yelp-st (Li et al., 2018a)	Negative Present	96944	1091	244	9355	15	8.8
	Negative Past	76153	860	244			
	Positive Present	201580	1503	333			
	Positive Past	57307	430	150			
Arxiv (Sergio, 2019)	AI	9981		200	162239	567	139.3
	CV	14382		200			
	NLP	14314		200			

Table 7: Data Statistics

	Dataset	PTB						SNLI					
	Method	PPL↓	MI↑	AU↑	-ELBO↓	KL↑	Rec↓	PPL↓	MI↑	AU↑	-ELBO↓	KL↑	Rec↓
	LSTM-LM†	100.47	-	-	-	-	-	21.44	-	-	-	-	-
	GPT-2†	24.23	-	-	-	-	-	20.24	-	-	-	-	-
LSTM	VAE§	101.39	0.01	0	101.27	0.00	101.27	21.67	0.03	1	33.12	0.04	33.08
LSTM	Annealing-VAE†	101.40	0.00	0	101.28	0.00	101.28	21.50	1.42	2	33.07	1.42	31.66
LSTM	Lag-VAE†	99.83	0.83	4	101.19	0.93	100.26	21.16	1.38	5	32.95	1.42	31.53
LSTM	FB-VAE$^{\S}(\lambda = 5.0)$	101.42	4.80	4	102.21	5.10	97.12	21.58	4.95	6	33.49	5.10	28.38
LSTM	AE-FB$^{\S}(\lambda = 5.0)$	96.86	5.31	32	102.41	6.54	95.87	21.64	7.71	32	34.47	9.53	24.94
LSTM	MoG-VAE$^{\diamondsuit}$	97.50	0.68	32	101.79	2.35	99.44	28.05	0.41	1	41.40	0.44	40.96
LSTM	Vamp-VAE$^{\diamondsuit}$	97.83	0.72	32	101.84	2.31	99.53	25.98	0.00	0	41.35	0.00	41.35
LSTM	Flow-Prior$^{\diamondsuit}$	93.58	2.83	31	106.41	7.21	99.20	26.19	3.16	32	51.15	7.59	43.56
LSTM	APo-VAE*	53.02	4.50	32	87.00	8.90	78.10						
LSTM	iVAE‡	53.44	12.20	32	87.20	12.51	74.69	7.40	9.93	32	21.54	**10.19**	11.35
LSTM	***DPrior (Our)***	**46.08**	**12.59**	32	**83.95**	**12.62**	**71.33**	**6.44**	**10.02**	32	**20.04**	10.04	**10.00**
OPTIMUS	AE-FB$^{\dagger}(\lambda = 1.0)$	35.53	8.18	32	77.65	**28.50**	77.65	29.63	9.20	32	47.35	**28.96**	18.39
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.5)$	26.69	7.64	32	96.82	15.72	81.09	16.67	8.89	32	38.50	16.35	22.14
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.05)$	23.58	3.78	32	91.31	4.88	86.43	13.47	3.49	32	33.08	3.92	29.17
OPTIMUS	iVAE	15.49	15.86	32	74.19	16.07	58.11	5.65	14.28	32	19.54	14.30	5.24
OPTIMUS	***DPrior (Our)***	**14.74**	**15.96**	32	**72.84**	15.96	**56.88**	**5.54**	**14.42**	32	**19.33**	14.42	**4.90**

Table 8: Language modeling performance comparison on PTB and SNLI datasets. "LSTM" indicates autoencoder architectures are built with two-layer LSTMs, while "OPTIMUS" employs pre-trained BERT and GPT-2 as the encoder and decoder. †: results from (Li et al., 2020a). ‡: results from (Fang et al., 2019). §: results from (Li et al., 2019a). *: results from (Dai et al., 2021). $^{\diamondsuit}$: results from (Ding and Gimpel, 2021). "-" indicates the models are improper to report these values. Empty cells indicate the results were not reported in the literature.

	Dataset	Yelp						Yahoo					
	Method	PPL↓	MI↑	AU↑	-ELBO↓	KL↑	Rec↓	PPL↓	MI↑	AU↑	-ELBO↓	KL↑	Rec↓
	LSTM-LM†	42.60	-	-	-	-	-	60.75	-	-	-	-	-
	GPT-2†	23.40	-	-	-	-	-	22.00	-	-	-	-	-
LSTM	VAE§	40.56	0.00	0	357.90	0.00	357.90	61.52	0.00	0	329.10	0.00	329.10
LSTM	Annealing-VAE†	40.39	0.13	1	357.76	0.14	357.62	61.21	0.00	0	328.80	0.00	328.80
LSTM	Lag-VAE†	39.84	2.16	12				59.77	2.9	19	328.40	5.70	322.70
LSTM	FB-VAE$^{\S}(\lambda = 0.5)$							62.78	5.00	3	331.32	5.07	326.26
LSTM	AE-FB$^{\S}(\lambda = 5.0)$	47.97	7.89	32				59.28	8.08	32	329.31	10.76	318.55
LSTM	MoG-VAE$^{\diamondsuit}$							64.60	0.00	0	332.90	0.00	332.90
LSTM	Vamp-VAE$^{\diamondsuit}$							74.81	0.00	0	344.61	0.00	344.61
LSTM	Flow-Prior$^{\diamondsuit}$							68.29	0.61	25	356.67	10.99	345.68
LSTM	APo-VAE*	32.91	6.20	32				46.61	4.90	32			
LSTM	iVAE‡	36.88	11.00	32	348.70	**11.60**	337.10	47.93	10.70	32	309.10	**11.40**	297.70
LSTM	***DPrior (Our)***	**32.79**	**11.35**	32	**337.35**	11.36	**325.99**	**45.18**	**10.93**	32	**304.34**	10.94	**293.40**
OPTIMUS	AE-FB$^{\dagger}(\lambda = 1.0)$	24.59	9.13	32	353.67	**27.89**	325.77	24.92	9.18	32	301.21	**30.41**	270.80
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.5)$	22.79	7.67	32	344.10	15.09	329.01	23.11	8.85	32	293.34	17.45	275.89
OPTIMUS	AE-FB$^{\dagger}(\lambda = 0.05)$	21.99	2.54	32	337.41	3.09	334.31	22.34	5.34	32	282.70	6.97	282.84
OPTIMUS	iVAE	15.44	15.07	32	294.55	15.35	279.19	15.04	12.52	32	246.26	12.95	233.31
OPTIMUS	***DPrior (Our)***	**14.52**	**17.05**	32	**287.92**	17.05	**270.87**	**14.67**	**12.99**	32	**244.01**	13.00	**231.01**

Table 9: Language modeling performance comparison on Yelp and Yahoo datasets. "LSTM" indicates autoencoder architectures are built with two-layer LSTMs, while "OPTIMUS" employs pre-trained BERT and GPT-2 as the encoder and decoder. †: results from (Li et al., 2020a). ‡: results from (Fang et al., 2019). §: results from (Li et al., 2019a). *: results from (Dai et al., 2021). $^{\diamondsuit}$: results from (Ding and Gimpel, 2021). "-" indicates the models are improper to report these values. Empty cells indicate the results were not reported in the literature.

	a man in white shirt is jogging on an iron horse in a women's path.
1	a man in a green and white outfit is racing a motocross machine.
2	a man in a white shirt in his silk blue robe with animals.
3	a skier in blue jeans is jousting on a pier in a city.
4	a man in blue shirts is holding up cans and laying at a skateboard drawing.
5	a male basketball players is led by another male as the waves on the beach.
6	a man wearing a shirt does his legstand on a hovering horse.
7	a man dressed in a white shirt and black hat is using sticks.
8	the man in the men's pants and helmet beats a rugby on a wave at their race.
9	the blond man will race two dogs back to shore in their same boat.

Table 10: Atom analysis on SNLI dataset.

0.0	a young woman with a black hairbrush brushes her teeth while a man in a white shirt watches.
0.1	a young woman with a black hairnet brushes her teeth while a man in a gray shirt watches her.
0.2	a young woman with a black shirt brushes her teeth in a house while a family watches.
0.3	a young woman with a black shirt cuts her teeth in a yard while a man watches.
0.4	a young man in a blue shirt with a black hair grabs a rag on her shoulder while other people work in the background.
0.5	a young man in a gray shirt holds a bottle of food with his two dogs in a distance.
0.6	a man in a brown shirt is holding a blue bag with a body of water in front of him.
0.7	a man in a blue shirt is holding a small dog with a bag in the grass.
0.8	a man in a blue shirt is holding a small dog in a area of grass.
0.9	a man in a blue shirt is holding a bag of food in a grassy area.
1.0	a man in a blue shirt is holding a bag of food in a small area of grass.

Table 11: Interpolating latent space $\mathbf{z}_\tau = \mathbf{z}_1 \cdot (1-\tau) + \mathbf{z}_2 \cdot \tau$. Each row shows τ, and the generated sentence (in blue) conditioned on $\mathbf{z}_\tau$.

Source $\boldsymbol{x}_A$	**Target** $\boldsymbol{x}_B$
a girl makes a silly face	two soccer players are playing soccer
Input $\boldsymbol{x}_C$	**Output** $\boldsymbol{x}_D$
• a girl poses for a picture	• two soccer players are posing
• a girl in a blue shirt is taking pictures of a microscope	• two boys are wearing soccer uniforms in a soccer field
• a woman with a red scarf looks at the stars	• two men in green jerseys are at rugby
• a boy is taking a bath	• two players are running
• a little boy is eating a bowl of soup	• two soccer boys are playing a soccer ball

Table 12: Sentence transfer via arithmetic operator $\mathbf{z}_D = \mathbf{z}_B - \mathbf{z}_A + \mathbf{z}_C$. The output sentences are in blue.

Types	Attributes	Samples
sentiment	positive	[s] this is followed by good movies, great food. [/s] [s] for sure the burrito is amazing and affordable. [/s]
	negative	[s] for me it looks crappy and understaffed. [/s] [s] i must have seen the disgusting and overpriced boxes. [/s]
tense	present	[s] this restaurant has an excellent view. [/s] [s] plus this place is clean and genuine customer service. [/s]
	past	[s] i was able to get the delicious sushi! [/s] [s] plus my car was messed up but our expectations were extremely low. [/s]
multi-set	positive present	[s] drinks are excellent as well as wine. [/s] [s] the haircut is completely worth the price! [/s]
	positive past	[s] the environment was awesome and friendly. [/s] [s] finally got a perfect haircut with great customer service. [/s]
	negative present	[s] well in my opinion it is a waste of calories. [/s] [s] probably the worst haircut they have ever had. [/s]
	negative past	[s] to my surprise, the plate was empty. [/s] [s] it might have been worst haircut you called or even asked for. [/s]

Table 13: DPrior+c case study on the Yelp dataset. Red and blue words indicate the sentiment and tense of sentences respectively.

Attributes	Samples
NLP	[s] the paper studies the use of generative adversarial networks (gans) for natural language parsing applications. upon retrieval of natural text digits, with a gan fixed-sized dictionary and a small set of rules, contextual grammar is generated for a given input group. this contextual grammar offers various incremental mechanism for gans to capture context, including a violation-theoretic scheme for the recognition rate of contextual grammars, exacerbated by accounts of its integration with quantitative metrics such as ver studies or globally-confluent grammars. our approach is primarily agnostic to concepts. furthermore, with real world examples, we show that with just a simple implementation we can expect to improve word parsing performance, carry out a state-of-the-art sequence learning algorithm, and finally generate an effective lexical prop grounding from its trace on the text data. [/s]
CV	[s] the topic of computer vision that attempts to predict gestures (i.e., hands) using probability distributions is rapidly gaining popularity. additionally, binary constraints lead to efficient finite state machine (fsm) composition strategies that tend to preserve image correspondences, since intuitive expressions of the departing fsm mechanisms only require a few trace steps from a given fsm state. we introduce a general cnn architecture that efficiently processes images with probabilistic hand model elements. we present a novel classification setting where the fsm parameters only need to be confirmed at a small level of training and test to improve the classification performance. we perform experiments (toads, limitation, handdisc) on datasets with numbers varying from about 320k samples to a relatively small amount of activity on a held-out dataset of collections of well-known hand gestures. through experiments, we have validated the effectiveness of our architecture; and we discovered that our gated knuckle-less fsm constraints selectively preserve image correspondences. [/s]
AI	[s] one of the problems in real-world monte carlo tree search problems (mcts) is the generation of promising algorithms and performing efficient learning of mcts parameters. parameters distributional constraints induced by a large number of observations are difficult to generate and therefore a way to overcome this issue is posed in this paper. through an empirical analysis of a prototype mct based on the control-box machine learning (cbm) and kleywagatoff-lofert satisfiability problems, we advocate deep belief learning (dl), a procedure with epistemic discretization to kickstart training. dl operates through an abstraction tree which enables better reasoning, language understanding, and preference of trained models. we introduce a number of different psychometric specifications to infer behavioral potentials. as a remedy, we propose an approach that starts with belief processes simultaneously. we present dl mouth-to-teeth behaviors that show considerably better soundness and recall compared to the current state-of-the-art mct based approaches as well as artificial neural networks (anns), and that satisfactorily generates better algorithms. [/s]

Table 14: DPrior+c case study on the Arxiv dataset. Blue words indicate the attributes.