FlowPrior: Learning Expressive Priors for Latent Variable Sentence Models

Xiaoan Ding
University of Chicago
Chicago, IL, 60637, USA
xiaoanding@uchicago.edu

Kevin Gimpel
Toyota Technological Institute at Chicago
Chicago, IL, 60637, USA
kgimpel@ttic.edu

Abstract

Variational autoencoders (VAEs) are widely used for latent variable modeling of text. We focus on variations that learn expressive prior distributions over the latent variable. We find that existing training strategies are not effective for learning rich priors, so we add the importance-sampled log marginal likelihood as a second term to the standard VAE objective to help when learning the prior. Doing so improves results for all priors evaluated, including a novel choice for sentence VAEs based on normalizing flows (NF). Priors parameterized with NF are no longer constrained to a specific distribution family, allowing a more flexible way to encode the data distribution. Our model, which we call FlowPrior, shows a substantial improvement in language modeling tasks compared to strong baselines. We demonstrate that FlowPrior learns an expressive prior with analysis and several forms of evaluation involving generation.

1 Introduction

Variational autoencoders (VAEs; Kingma and Welling, 2014) have been widely applied to many natural language processing tasks (Bowman et al., 2016; Zhang et al., 2016; Shen et al., 2017; Kim et al., 2018; Fang et al., 2019; Chen et al., 2019). VAEs provide statistical transparency in describing observations in a latent space and flexibility when used in applications that require directly manipulating the learned representation (Hu et al., 2017). Recent work (Li et al., 2020) has combined VAEs with BERT/GPT in representation learning and guided generation. However, the representation capacity of VAEs is still limited for modeling sentences due to two main reasons.

One is known as the *posterior collapse* problem, in which the posterior "collapses" to the prior and the generator learns to ignore the latent variable (Bowman et al., 2016). Many methods have been developed to address it: annealing (Fu et al., 2019), weakening the capacity of the generator (Semeniuta et al., 2017; Yang et al., 2017), manipulating training objectives (Burda et al., 2016; Higgins et al., 2017; Zhao et al., 2017), including the use of *free bits* (FB) (Kingma et al., 2016; Li et al., 2019), and changing training (He et al., 2019).

The other reason is the *restrictive assumption* of the parametric forms for the prior and approximate posterior. While these forms are computationally efficient, they limit the expressivity of the model. The main existing solutions (Kingma et al., 2016; Tomczak and Welling, 2018; Razavi et al., 2019) focus on enriching the variational posterior, while other work focuses on learning an expressive prior (Tomczak and Welling, 2018; Serban et al., 2017; Chen et al., 2017).

In this paper, we follow the latter line of research and draw upon methods in building and learning expressive priors. We first show empirically that the original VAE objective, the evidence lower bound (ELBO), is not effective when learning priors. The issue is not solely due to posterior collapse since it is not resolved by using modifications based on free bits. To address this issue, we propose using a combined objective, adding to the ELBO a second objective (denoted $\mathrm{M_{IS}}$) which is a different lower bound on the log marginal likelihood obtained using importance sampling (Burda et al., 2016).

Using the combination of the ELBO and $\mathrm{M_{IS}}$, we compare multiple choices for the prior, including a mixture of Gaussians, a prior based on a variational mixture of posteriors (VampPrior; Tomczak and Welling, 2018), and a prior based on normalizing flows (NF), specifically real NVP transformations (Dinh et al., 2016). Using a real NVP prior entails creating an invertible mapping from a simple base distribution to the prior distribution of the latent variable in a VAE. This choice allows a flexible prior distribution that is not constrained to a specific parametric family. The hope is that it would be better at modeling the data distribution.

We perform an empirical evaluation of priors and objective functions for training VAE sentence models on four standard datasets. We find the best performance overall when using the flow-based prior and the combined objective in the training objective. We refer to this setting as *FlowPrior*. The generation of prior samples with FlowPrior comports to the training distribution while maintaining a higher diversity than competing models in our quantitative and qualitative evaluation.

To summarize, this paper contributes: (1) a strategy for improved training of sentence VAEs based on combining multiple lower bounds on the log marginal likelihood; (2) the first results applying real NVP to model the prior in sentence VAEs; and (3) comprehensive evaluation and analysis with three expressive priors and training objective variations.

2 Background

Variational autoencoders (VAEs; Kingma and Welling, 2014) are a popular framework for learning latent variable models with continuous latent variables. Let x be the observed variable and z the latent variable. The model factorizes the joint distribution over x and z into a prior $p_\psi(z)$ and a generator $p_\theta(x \mid z)$. Maximizing the log marginal likelihood $\log p(x)$ is intractable in general, so VAEs introduce an approximate posterior $q_\phi(z \mid x)$ parameterized using a neural network (i.e., an "inference network"), and replace the log marginal likelihood with the *evidence lower bound* (ELBO):

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

Maximizing the right-hand side of the equation above can be viewed as a regularized autoencoder in which the first term is the negative reconstruction error and the second is the negative KL divergence between the approximate posterior $q_\phi(z|x)$ and the latent variable prior $p_\psi(z)$. It is common in practice to fix the prior $p_\psi(z)$ to be a standard Gaussian distribution and only learn θ and ϕ (Bowman et al., 2016; Yang et al., 2017; Shen et al., 2017).

While constraining the prior to be a fixed standard Gaussian is common, it is not necessary for tractability. Researchers have found benefit from using richer priors and posteriors (Rezende and Mohamed, 2015; Kingma et al., 2016; Chen et al., 2017; Ziegler and Rush, 2019; Ma et al., 2019). In this paper, we consider investigating alternative priors while still using the standard Gaussian form for the approximate posterior.

3 Choices for Prior Families

We now describe the three kinds of priors we will compare in our experiments. The first two are based on Gaussian mixtures (Sec. 3.1) and the third is based on normalizing flows (Sec. 3.2). We take these three prior families into consideration because they represent the three main categories of work in learning priors: simple Gaussian mixtures (usually as baselines), defining the prior as a function of the approximate posterior (Tomczak and Welling, 2018; Chen et al., 2018), and flow-based priors (Chen et al., 2017; Ziegler and Rush, 2019; Ma et al., 2019; Lee et al., 2020). Note that we do not make any changes to the approximate posterior distribution. That is, the approximate posterior follows a Gaussian distribution with a diagonal covariance matrix as in standard VAEs.

3.1 Gaussian Mixture Priors

Our first choice is a uniform mixture of K Gaussians (MoG):

$$p_\psi(z) = \frac{1}{K}\sum_{k=1}^{K} f(z; \mu_k, \mathrm{diag}(\sigma_k^2)) \quad (2)$$

where $f(z; \mu, \Sigma)$ is the density function of a d-dimensional Gaussian with mean μ and covariance matrix Σ. The μ_k and σ_k are learnable parameter vectors with dimensionality d (which is 32 in our experiments). This prior was used as a baseline by Tomczak and Welling (2018). We refer to a VAE that uses this prior as *MoG-VAE*.

Tomczak and Welling (2018) extend MoG-VAE to a "Variational Mixture of Posteriors" prior (VampPrior). This approach parameterizes the prior using a mixture of Gaussians with components given by a variational posterior conditioned on learnable "pseudo-inputs":

$$p_\psi(z) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(z \mid u_k) \quad (3)$$

where K is the number of pseudo-inputs, each of which is denoted u_k. Pelsmaeker and Aziz (2020) applied this idea to text modeling and we follow their strategy for defining pseudo-inputs. That is, each u_k consists of a sequence of embeddings that have the same dimensionality as word embeddings. For each component k, the lengths of pseudo-inputs

can vary; they are sampled based on the statistics of the lengths in the training set. We refer to a VAE with this prior as *Vamp-VAE*.

3.2 Flow-based Priors

Our third choice for a prior distribution is to leverage normalizing flows (NF). A normalizing flow is a sequence of invertible, deterministic transformations. By repeatedly applying the rule for change of variables (see the Appendices for details), the base density is transformed into a more complex one. Networks parameterized using NF can be trained through exact maximum log-likelihood computation. Exact sampling is performed by drawing a sample from the base distribution and performing the chain of transformations. This allows a flexible prior and is expected to have more expressive latent components compared to those based on Gaussian mixtures.

Computing the Jacobian of functions with high dimension and the determinants of large matrices (i.e., the two main computations in NF) are very expensive. Our flow-based prior uses *real-valued non-volume preserving* (real NVP) transformations (Dinh et al., 2016) which are efficient in both training and sampling. The transformations are based on *scale* and *translation* operations.[1] It is worth noting that these two operations are not used in computing the Jacobian determinant and inverse. So one can design arbitrarily complex operations that allow a flexible transformation without incurring large computational cost.

More specifically, we apply real NVP as a prior by creating an invertible mapping between a base distribution $p_0(z_0)$ (in our case, $z_0 \sim \mathcal{N}(0, \mathrm{I})$) and the prior distribution $p_\psi(z_L)$ in the VAE:

$$z_L = f_L \circ f_{L-1} \circ ... \circ f_1(z_0) \tag{4}$$

where z_L is the sentence latent variable and $f_1, f_2, ..., f_L$ are all bijective functions.

Using the change-of-variables theorem, given a latent variable z_L, we can compute the exact density under the prior with the "image" z_0 acquired by inverting the transformation:

$$z_0 = f_1^{-1} \circ ... \circ f_{L-1}^{-1} \circ f_L^{-1}(z_L) \tag{5}$$

$$\log p_\psi(z_L) = \log p_0(z_0) - \sum_{l=1}^{L} \log \left|\det\left(\frac{\partial f_l(z_{l-1})}{\partial z_{l-1}}\right)\right| \tag{6}$$

[1] More details about normalizing flows and real NVP are in the Appendices.

We refer to a VAE with a real NVP prior as real NVP-VAE. We find our best setting to consist of a real NVP prior and the combined objective in Section 4.1 and we refer to this setting as *FlowPrior*.

4 Objectives for Learning Priors in VAEs

ELBO. Our preliminary experiments found that, when training with the standard ELBO, using more sophisticated priors does not improve perplexity compared to standard Gaussian priors (Table 3). Though these priors could potentially be highly multimodal, the learned prior parameters yield approximately unimodal forms (Figure 1, left).

Several approaches have been proposed to mitigate or avoid collapse in the approximate posterior. One method that we include in our experiments is a variation of KL divergence known as "free bits" (FB) KL (Li et al., 2019; Kingma et al., 2016). Posterior collapse is mitigated, but the VAE models still do not benefit much from expressive priors (Tables 1-2). Pelsmaeker and Aziz (2020) made similar observations with an improved FB objective. We speculate that these undesirable results are due to the lack of learning signal for the prior parameters.

Marginal Likelihood via Importance Sampling. In the ELBO, the prior distribution only appears in the KL term. As a consequence, the prior parameters receive a limited amount of learning signal. The posterior network, by contrast, receives gradient updates from both the reconstruction and KL terms. When minimizing the KL term the potentially expressive prior density can "collapse" to a unimodal form, as this may facilitate minimizing the KL divergence between the approximate posterior and prior.

We consider optimizing another objective, a different lower bound on the log marginal likelihood obtained using importance sampling (Burda et al., 2016):

$$\log \frac{1}{N} \sum_{i=1}^{N} \frac{p_\theta(x|z^{(i)}) p_\psi(z^{(i)})}{q_\phi(z^{(i)}|x)}, \text{s.t. } z^{(i)} \sim q_\phi(z|x)$$

where x is an input in the training data and N is the number of samples in use. This objective was proposed as the training objective in the importance-weighted autoencoder (IWAE; Burda et al., 2016), and was shown to be a tighter lower bound on the log marginal likelihood than the ELBO. In this paper, we denote this objective by M_{IS}.

In addition to providing a tighter lower bound, M_{IS} also increases the flexibility of the approximate posterior, as shown by Cremer et al. (2017). By increasing N, the approximate posterior has an implicit complex distribution that approaches the true posterior, which may also be beneficial in learning an expressive prior.

Combination of the Two. However, M_{IS} is not necessarily optimal by itself for training VAEs. Rainforth et al. (2018) prove that using M_{IS} with a large value of N is detrimental in learning the posterior, which is also shown in our empirical evaluation in Table 3. If we only have M_{IS}, the approximate posterior q only appears in the denominator so learning seeks to make samples from the posterior q less likely under q, which could cause q to become a poor proposal distribution. The ELBO, with its reconstruction loss, appears helpful in learning a better posterior. Therefore, we optimize the sum of the ELBO and M_{IS}, which was proposed by Rainforth et al. (2018).

4.1 Combined Training Objective

Our combined training objective then contains three terms: M_{IS}, reconstruction, and sample-based KL. We draw N samples from $q_\phi(z|x)$, and compute the three terms using the same samples:

$$\mathcal{L}(\theta, \phi, \psi; x) = \log \frac{1}{N} \sum_{i=1}^{N} \frac{p_\theta(x|z^{(i)}) p_\psi(z^{(i)})}{q_\phi(z^{(i)}|x)} + \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(x|z^{(i)}) - \mathrm{KL}_{\phi,\psi}(x, \{z^{(i)}\}_{i=1}^{N}) \quad \text{s.t. } z^{(i)} \sim q_\phi(z|x) \quad (7)$$

When training with the ELBO alone, one typically uses a single sample from $q_\phi(z|x)$. However, since we draw multiple samples anyway in order to compute M_{IS}, we use those same samples for the reconstruction term, which can lead to more robust gradients of that term than the standard approach of using one sample.

The reason we use sample-based estimates for the KL divergence is because our choices for the prior preclude the possibility of a closed form for the KL. We consider two different approaches when computing sample-based KLs: *standard KL* and a modified one inspired by free bits (Li et al., 2019; Pelsmaeker and Aziz, 2020; Kingma et al., 2016), which we refer to as *FB KL*.

For standard KL, we use Monte Carlo estimation in computing the KL divergence with N samples:

$$\mathrm{KL}_{\phi,\psi}(x, \{z^{(i)}\}_{i=1}^{N}) = \frac{1}{N} \sum_{i=1}^{N} (\log q_\phi(z^{(i)}|x) - \log p_\psi(z^{(i)})) \quad (8)$$

For the FB KL, we follow prior work (Kingma et al., 2016) that replaces the KL with a hinge loss term in each latent dimension:

$$\text{FB KL}_{\phi,\psi}(x, \{z^{(i)}\}_{i=1}^{N}) = \sum_{j=1}^{d} \max(\lambda, \mathrm{KL}^{j}_{\phi,\psi}(x, \{z^{(i)}\}_{i=1}^{N})) \quad (9)$$

where $\mathrm{KL}^{j}_{\phi,\psi}$ denotes the KL computed only for dimension j of the latent variable, and λ is the "target rate" hyperparameter.

4.2 Training Procedure

We describe our training procedure below for VampPrior, which combines a real NVP prior with the objective in Eq. 7. For simplicity, our description only uses one input x. In practice, we use mini-batches with a stochastic gradient based optimizer. All the parameters (θ, ϕ, ψ) are updated simultaneously during training.

1. Draw N samples $z_L^{(1)}, z_L^{(2)}, ..., z_L^{(N)}$ from the inference network using the reparameterization trick.
2. Perform the inverse transformation to get the image of each point under the base distribution: $z_0^{(1)}, z_0^{(2)}, ..., z_0^{(N)}$.
3. Compute the exact log likelihood of the sample prior with change of variable theorem (Eq. 6).
4. Compute and backpropagate the loss (Eq. 7).

When using the other priors (standard Gaussian, MoG, and VampPrior), we do not need steps 2 and 3 above because those priors can be computed directly without the inverse transformation or change of variable theorem.

5 Experiments

5.1 Datasets

We consider four widely-used, publicly available English datasets: the Penn Treebank (PTB) (Marcus et al., 1993; Bowman et al., 2016), Yahoo (Yang et al., 2017; He et al., 2019), Yelp sentiment (Shen et al., 2017), and SNLI (Bowman et al., 2015).

5.2 Baselines

Our baselines include *standard VAE* with linear KL annealing (Bowman et al., 2016); *Cyc-VAE* (Fu et al., 2019) in which the KL term is reweighted with a cyclical annealing schedule; *Lag-VAE* (He et al., 2019) which updates the encoder multiple times before each decoder update; *VAE+FB* (Kingma et al., 2016; Chen et al., 2017) which replaces the standard KL with FB (i.e., Eq. 9 with $N = 1$); and *Pre-VAE+FB* (Li et al., 2019) which initializes the VAE with a pretrained autoencoder and replaces standard KL with FB. We evaluated these baselines using their open source implementations.[2]

In addition, we include two prior-learning baselines: MoG-VAE (Eq. 2) and Vamp-VAE (Eq. 3). We follow Pelsmaeker and Aziz (2020) and set 100 components/pseudo-inputs. Unlike the earlier baselines, for which we used open source codebases, we implemented the MoG-VAE and Vamp-VAE models on top of our standard VAE implementation, which was also used for FlowPrior.

5.3 Implementation and Training Details

Across all the experiments for our implemented baselines (i.e., standard VAE, MoG-VAE, Vamp-VAE) and our proposed model FlowPrior, we follow prior work (Kim et al., 2018; He et al., 2019; Li et al., 2019) and use a single-layer LSTM encoder and decoder with a 32-dimensional latent variable. We use a batch size of 32 and train using SGD.[3]

5.4 Evaluation Metrics

Our evaluation measures language modeling performance, the use of the latent variable, and the quality and diversity of generations from the prior and posterior. The metrics are listed below:

PPL: We estimate log marginal likelihood using importance sampling (Burda et al., 2016) and calculate perplexity on the test set.[4]

KL: We report the KL term in the ELBO on the test set. When training with FB KL, we still report standard KL. For standard VAE, we compute KL with its closed-form expression. Otherwise, we report the KL estimated with samples.

[2]The links to their implementations are in the Appendix.

[3]We use the open source implementations for other baselines. All models are trained with the simple linear annealing schedule, with same hyperparameter search space. We run each setting with 5 random seeds and report the medians.

[4]We use 1000 samples which appears to be more than sufficient for estimation; Ziegler and Rush (2019) found that using more than 50 samples did not even show much difference.

Model	PPL(↓)	Recon(↓)	KL	AU(↑)	MI(↑)
VAE	101.40	101.28	0.00	0	0.00
Cyc-VAE	107.73	101.17	2.01	5	1.24
Lag-VAE	100.25	100.41	1.04	3	0.79
VAE + FB	101.56	99.84	4.46	32	0.90
Pre-VAE + FB	96.35	94.52	8.15	32	6.30
MoG-VAE	98.22	100.54	0.00	0	0.00
MoG-VAE + FB	97.50	99.44	2.35	32	0.68
Vamp-VAE	98.27	100.56	0.00	0	0.00
Vamp-VAE + FB	97.83	99.53	2.31	32	0.72
FlowPrior	94.72	98.46	3.28	2	2.25
FlowPrior + FB	93.58	99.20	7.21	31	2.83

Table 1: Language modeling results on PTB dataset.

MI: We follow Hoffman and Johnson (2016) and report estimated mutual information between the observation and its latent variable.

AU: A dimension z in the latent variable is considered "active" if $\mathrm{Cov}_x(\mathbb{E}_{z\backsim q(z|x)}[z]) > 10^{-2}$. AU is then the number of active latent dimensions (Burda et al., 2016).

F-PPL and R-PPL: These metrics measure the correspondence between generated sentences from the model and the training corpus. We evaluate both F-PPL and R-PPL by estimating 5-gram language models using the KenLM toolkit (Heafield, 2011) with its default smoothing method. For F-PPL, we estimate language models from the actual text and compute the perplexity of the generated samples. For R-PPL, we estimate language models from the generated samples and compute the perplexity of the actual text.[5]

Self-BLEU: The self-BLEU metric is one measure of the diversity of a set of samples (Zhu et al., 2018). It is calculated by averaging the BLEU scores computed between all pairs of samples.

6 Results

6.1 Language Modeling

We first perform language modeling tasks to characterize models' efficacy at modeling texts in terms of modeling the distribution of language data and making use of the latent variable. We refer to our model as **FlowPrior**, which uses the training objective in Eq. 7 which includes M_{IS} and the standard KL (Eq. 8). We use **FlowPrior + FB** to refer to our model with the FB KL (Eq. 9).

[5]Our R-PPL is slightly different from that in Fang et al. (2019). For R-PPL, we always concatenate the training set vocabulary (one word per line) to the set of samples from the models to ensure LMs have seen the entire vocabulary.

Model	PPL(↓)	Recon(↓)	KL	AU(↑)	MI(↑)
Yahoo					
VAE	65.77	333.17	0.00	0	0.00
MoG-VAE	64.60	332.90	0.00	0	0.00
Vamp-VAE	74.81	344.61	0.01	0	0.00
FlowPrior	62.49	331.57	1.43	4	1.62
FlowPrior + FB	68.29	345.68	10.99	25	0.61
Yelp					
VAE	35.10	35.18	0.00	0	0.00
MoG-VAE	35.18	35.20	0.01	0	0.00
Vamp-VAE	34.99	35.15	0.00	0	0.00
FlowPrior	31.82	30.25	4.15	2	2.46
FlowPrior + FB	39.03	36.87	10.13	32	2.57
SNLI					
VAE	25.97	41.34	0.00	0	0.00
MoG-VAE	28.05	40.96	0.44	1	0.41
Vamp-VAE	25.98	41.35	0.00	0	0.00
FlowPrior	22.41	37.89	3.83	3	0.97
FlowPrior + FB	26.19	43.56	7.59	32	3.16

Table 2: Language modeling results on other datasets.

Prior	PPL(↓)	KL	AU(↑)
PTB			
Standard	101.8 / 101.4 / 98.4	0.0 / 0.0 / 3.2	0 / 0 / 2
MoG	101.9 / 98.2 / 96.7	0.0 / 0.0 / 0.0	0 / 0 / 0
Vamp	101.7 / 98.3 / 96.1	0.0 / 0.0 / 3.1	0 / 0 / 4
Real NVP	102.5 / 98.4 / 94.7	0.0 / 0.0 / 3.3	0 / 0 / 2
Yahoo			
Standard	65.6 / 65.8 / 63.9	0.0 / 0.0 / 2.7	0 / 0 / 1
MoG	65.6 / 64.6 / 62.7	0.0 / 0.0 / 0.5	0 / 0 / 1
Vamp	78.5 / 74.8 / 62.9	0.0 / 0.0 / 1.5	0 / 0 / 2
Real NVP	65.6 / 65.8 / 62.5	0.0 / 0.0 / 1.4	0 / 0 / 4
Yelp			
Standard	35.4 / 35.1 / 33.2	0.0 / 0.0 / 2.9	0 / 0 / 2
MoG	36.0 / 35.2 / 34.9	0.0 / 0.0 / 0.0	0 / 0 / 0
Vamp	38.0 / 35.0 / 33.7	0.0 / 0.0 / 4.1	0 / 0 / 1
Real NVP	35.6 / 35.1 / 31.8	0.0 / 0.0 / 4.2	0 / 0 / 2
SNLI			
Standard	27.4 / 26.0 / 25.3	0.0 / 0.0 / 1.2	0 / 0 / 3
MoG	27.2 / 28.1 / 24.3	0.0 / 0.4 / 4.2	0 / 1 / 5
Vamp	27.6 / 26.0 / 23.7	0.0 / 0.0 / 2.8	0 / 0 / 2
Real NVP	27.7 / 26.1 / 22.4	0.0 / 0.0 / 3.8	0 / 0 / 3

Table 3: Comparing training objectives with several choices for priors. Each cell has three results: training with M_{IS} only, ELBO only, and the combination of ELBO + M_{IS}. The combination consistently improves performance across models and datasets.

Comparison to baselines. Table 1 shows results on the PTB dataset for several VAEs from prior work and our implemented models. Since our contributions lie in learning the prior instead of changing the training procedure or manipulating the KL term, we set the baselines as standard VAE, MoG, and VampPrior for the rest of the paper. We report the performance of FlowPrior and those baselines on Yahoo, Yelp, and SNLI in Table 2.

From Tables 1 and 2, we observe that FlowPrior consistently outperforms the baselines in test set perplexity, sometimes by large margins. This is not surprising since the M_{IS} term in our training objective directly targets the perplexity metric because the expressions are identical (differing only in the number of samples used). While FB typically improves models on PTB, and helps FlowPrior to reach a higher AU and KL on the other datasets, it does not lead to better test PPL and reconstruction. We report additional results on measuring the impact of FB in the Appendix.

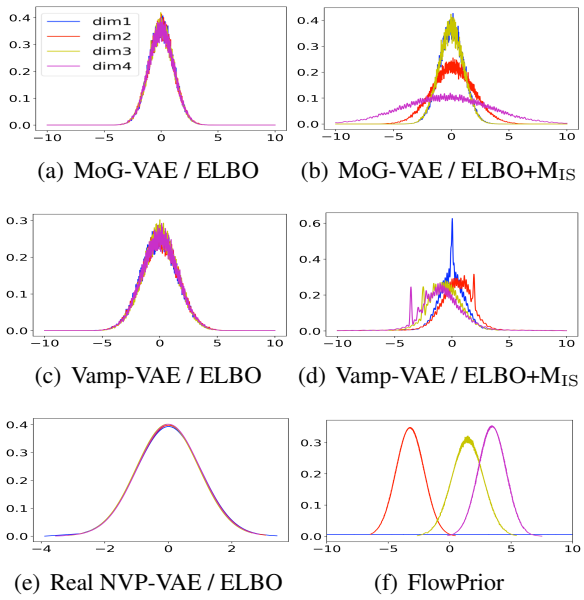
Another finding is that simply enriching the parametric family of the prior is not sufficient to improve our evaluation metrics. Tables 1 and 2 show mixed results when moving from the VAE with its standard Gaussian prior to the MoG- or Vamp-VAE. Though these priors have the potential to be multimodal, they could still be unimodal after training. For example, the MoG-VAE might learn a mixture in which all Gaussians have the same location and scale. Also, the complexity of the prior learned by the Vamp-VAE is dependent upon the inference network, so if the inference network does not learn anything useful, the learned prior may not be useful either.

Impact of selection of objectives. The learned prior baselines (MoG-VAE and Vamp-VAE) fail to learn to use the latent variable, as shown by the small numbers (nearly zero) for the AU and MI metrics in Tables 1-2. Similar observations were made by Pelsmaeker and Aziz (2020). We argue that only improving the prior might not be sufficient, as the ELBO objective is difficult to optimize and little information may be learnable for the prior from the ELBO alone. To measure the utility of the M_{IS} term, we introduce this term to standard-VAE, MoG-VAE, and Vamp-VAE and evaluate the improved models under the same language model metrics.

Table 3 compares models trained with M_{IS}, the ELBO, and the combined training objective (Eq. 7). The combined objective is beneficial to all metrics for all priors and datasets. Our results are consistent with the observations of Rainforth et al. (2018) that tighter bounds are preferable for training the gener-

Vamp-VAE + M_{IS}
Three people are sitting on a bench .
People are walking down the street .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
Women in a white dress and a man in a black shirt are standing in front of a microphone .
Women in a white dress and a man in a black shirt are standing in front of a microphone .
two men are playing soccer
two men are playing basketball
Two men are playing a game of chess .
Two men are playing a game of chess .
FlowPrior
The dog is running through the snow .
Two young boys are playing in the snow .
There is a man in a blue shirt and a woman in a black shirt and black pants .
Three people are sitting on a bench .
two men are standing on a bench
A girl is sitting on a bench .
A young girl is sitting on a bench .
A young man is sitting on a bench .
A woman in a black shirt is sitting on a bench .
A woman is sitting on a bench .

Table 4: Interpolation from the prior on SNLI dataset. In each cell, the first and last sentences correspond to two sampled latent codes and between are linearly interpolated samples.



(a) MoG-VAE / ELBO (b) MoG-VAE / ELBO+M_{IS}

(c) Vamp-VAE / ELBO (d) Vamp-VAE / ELBO+M_{IS}

(e) Real NVP-VAE / ELBO (f) FlowPrior

Figure 1: Densities of 4 dimensions of learned priors (SNLI dataset).

ative network, while looser bounds are preferable for training the inference network. Still, FlowPrior (real NVP + M_{IS}) performs the best in PPL and MI compared to other models, showing the flexibility and the power of the real NVP architecture.

For the "Standard" setting in Table 3, the prior is fixed and not learned while in the other three settings the prior is learned. The combination of ELBO and M_{IS} is helpful across all settings.[6]

6.2 Interpolations Between Prior Samples

One appealing aspect of VAEs for sentence modeling is the potential for learning a smooth, interpretable space for sentences. A qualitative way to explore the latent space is to interpolate between samples from the prior distribution. We randomly sample two latent vectors from the prior and linearly interpolate between them with evenly divided intervals (Bowman et al., 2016).[7] We use greedy decoding in generation.[8] Table 4 shows linear interpolation between prior samples in FlowPrior and Vamp-VAE + M_{IS} (i.e., Vamp-VAE with the combined training objective). We observe substantial improvement with FlowPrior, as it can generate sentences with smooth semantic evolution while maintaining plausible generations in terms of fluency. This semantic evolution may reflect the complex structure in the learned prior distribution. Interpolations with MoG-VAE + M_{IS} and Vamp-VAE + M_{IS} have more repetitions and do not transit smoothly from one to the other. (Results with MoG-VAE are in the appendix.)

6.3 Visualization of Learned Priors

We randomly select 4 dimensions from the learned priors per model and plot their densities in Fig. 1.

In MoG-VAE, each dimension is a Gaussian mixture with 100 components. When only using the ELBO for training (Fig. 1(a)), the four visualized components all have similar shapes. After adding M_{IS} (Fig. 1(b)), different dimensions have similar locations but different scales.

Vamp-VAE permits relatively complex components because the means and variances are acquired from the inference network applied to learned

[6]For the MoG setting, we also performed experiments with setting the number of Gaussian components $K = 1$ and observed comparable or slightly worse test PPL under all 3 choices of training loss than Standard setting.

[7]FlowPrior is slightly different. Instead of directly sampling from the latent variable of VAE (in MoG-VAE and Vamp-VAE), FlowPrior samples from the base distribution of real NVP, interpolates in the base distribution, and maps to the latent with Eq. 4. We also experiment with interpolating the two samples after mapping, namely interpolating in the VAE latent space, and find similar results.

[8]We additionally tried various sampling methods for decoding. This leads to more noise and becomes harder to interpret. Generations can be found in the Appendices.

MoG-VAE	**MoG-VAE + M_{IS}**
The man is wearing a black shirt . A man is standing in front of a building . A man is standing in front of a building .	An older gentleman in a white shirt is walking in a parking lot . A woman is walking in a field . A young girl in a red shirt is playing with a toy .
Vamp-VAE	**Vamp-VAE + M_{IS}**
A man is playing a guitar . A man is playing a guitar . A man is playing a guitar .	Man in a blue shirt and jeans is sitting on a bench . The man is wearing a black shirt . People are walking down the street .
VAE	**FlowPrior**
A man is sitting on a bench . A man is sitting on a bench . A man is sitting on a bench .	Man in a blue shirt and blue jeans is sitting on a rock with a hammer . Two young boys are playing in the snow . A dog is running through the snow .

Table 5: Generations from prior samples with greedy decoding (SNLI dataset).

pseudo-inputs. Fig. 1(c) shows that Vamp-VAE trained without M_{IS} does not show much difference compared to MoG-VAE. However, when training with M_{IS} (Fig. 1(d)), the distributions in several dimensions appear to be multimodal.

The real NVP prior learns little information when training without M_{IS}, as all dimensions are akin to standard normal distributions. When training with M_{IS}, different dimensions show distinct placement and shape. The prior in FlowPrior is highly multimodal overall and smooth in each dimension.

6.4 Generations from Prior Samples

Sampling from Prior. To measure the expressiveness of the prior and the richness of the learned latent variable, we randomly sample 5000 times from the prior distribution and evaluate their greedy-decoded generations qualitatively and quantitatively. Table 5 shows greedy generations from prior samples. We observe substantial improvements in term of generation diversity when adding M_{IS} in the training objective. Note that these diverse samples are achieved with a purely deterministic decoding. A diverse set of samples implies that (1) richer latent codes and a highly multimodal distribution is learned by the model; (2) and the generator is trained to attend to the latent codes.

Sample Mundanity and Coverage. A strongly-performing generative model should be able to generate samples that comport to the training data distribution. We use the forward and reverse PPL to estimate the similarity between the training data and samples. We can consider F-PPL as a *generation "precision"* as it reflects the amount of information in the samples that is relevant to the actual text. Analogously, we can consider R-PPL as a *generation "recall"* as it reflects how much the samples as a whole provide coverage of the actual text. Moreover, both F-PPL and R-PPL reflect whether the decoder is able to attend to the latent variable in generation.

	Yelp			**SNLI**		
	F-PPL	R-PPL	SB	F-PPL	R-PPL	SB
VAE	4	30248	96	4	51127	100
VAE+M_{IS}	5	10818	30	4	19047	73
Vamp-VAE	4	32504	100	4	56050	100
Vamp-VAE+M_{IS}	7	5280	10	5	8420	29
FlowPrior	209	1677	3	42	5725	13

Table 6: Forward PPL (F-PPL), Reverse PPL (R-PPL), and Self-BLEU (SB) of greedy-decoded prior samples.

Table 6 shows the F-PPL and R-PPL with greedy generation from prior samples. While Fang et al. (2019) treats a lower F-PPL as an indicator of better samples, we argue that it is not necessarily true. A model could achieve a low F-PPL by simply generating identical (or nearly-identical) high-probability sequences, like those observed from the VAE, MoG-VAE, and Vamp-VAE in Table 5. This reflects how an overly-simplified or restrictive assumption in the prior can lead to less diversity in samples.

Indeed, we find that models with very low F-PPL values often have very high R-PPL values. A lower R-PPL indicates the distribution of generated samples matches the distribution of the training data. From Table 6 we observe that adding M_{IS} is beneficial as it leads to a lower R-PPL. FlowPrior has the best R-PPL, and shows the capability of capturing characteristics of the target distribution that are not captured by simpler priors.

Generation Diversity. To identify which model has richer usage of latent variables, we use self-BLEU to measure the diversity of a set of samples. We observe significant improvements in FlowPrior in Table 6, which implies a diverse latent representation and a better utilization of the latent variable.

7 Related Work

When considering the parameterized family of VAE models, expressive latent components (i.e., posterior and prior) have been widely studied in computer vision (Dinh et al., 2015, 2016; Kingma and Dhariwal, 2018). However, multimodal priors have been seldom applied to language, with some exceptions (Serban et al., 2017; He et al., 2018; Ziegler and Rush, 2019; Ma et al., 2019; Lee et al., 2020).

Chen et al. (2017) use autoregressive flow for the prior and posterior and experiment with images. Ziegler and Rush (2019) propose several *autoregressive* NF architectures and characterize performance on character-level language modeling. Ma et al. (2019) design priors using the Glow architecture to improve the performance of non-autoregressive neural machine translation. Lee et al. (2020) empirically characterize the performance of NF and simple Gaussian priors in token-level latent variable models, and observe that flexible priors yield higher log-likelihoods but not better BLEU scores on machine translation tasks.

Our work differs from that of Ziegler and Rush (2019) and Chen et al. (2017) as we are using a non-autoregressive flow-based architecture for the prior, while they are using autoregressive NF. Also, we focus on models with a single latent variable for an entire sentence, while similar prior work has focused on token-level latent variables (Ziegler and Rush, 2019; Ma et al., 2019; Lee et al., 2020).

Several others have employed NF for flexible modeling in NLP. Setiawan et al. (2020) present a variational translation model that uses NF in the approximate posterior while keeping the prior as Gaussian. Wang and Wang (2019) apply NF to a variational Wasserstein autoencoder in order to make the posterior more flexible. Jin et al. (2019) use transformed distributions via NF to model the emission density, which improves parsing performance as compared to Gaussian baselines.

8 Conclusion

We proposed a method, FlowPrior, that uses normalizing flow to define the prior in a sentence VAE and adds the importance-sampled marginal likelihood (M_{IS}) as a second term to the standard VAE objective. Our empirical results show FlowPrior yields a substantial improvement in language modeling and generation tasks as compared to prior work. Adding M_{IS} improves performance for other models as well, especially in settings when the prior parameters are being learned.

Acknowledgments

We would like to thank Sam Wiseman, Qingming Tang, and Mingda Chen for helpful discussions, and the anonymous reviewers for their comments that improved this paper.

References

Johannes Ballé, Valero Laparra, and Eero P Simoncelli. 2016. Density modeling of images using a generalized normalization transformation. In *International Conference on Learning Representations*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders. In *International Conference on Learning Representations*.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018. Variational sequential labelers for semi-supervised learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 215–226, Brussels, Belgium. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, Florence, Italy. Association for Computational Linguistics.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2017. Variational lossy autoencoder. In *International Conference on Learning Representations*.

Chris Cremer, Quaid Morris, and David Duvenaud. 2017. Reinterpreting importance-weighted autoencoders. In *International Conference on Learning Representations (Workshop Track)*.

Laurent Dinh, David Krueger, and Yoshua Bengio. 2015. Nice: Non-linear independent components estimation. In *International Conference on Learning Representations*.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using Real NVP. In *International Conference on Learning Representations*.

Le Fang, Chunyuan Li, Jianfeng Gao, Wen Dong, and Changyou Chen. 2019. Implicit deep latent variable models for text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3946–3956, Hong Kong, China. Association for Computational Linguistics.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 240–250, Minneapolis, Minnesota. Association for Computational Linguistics.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. 2015. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Brussels, Belgium. Association for Computational Linguistics.

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

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. 2019. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*.

Matthew D Hoffman and Matthew J Johnson. 2016. ELBO surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, volume 1, page 2.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

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

Lifeng Jin, Finale Doshi-Velez, Timothy Miller, Lane Schwartz, and William Schuler. 2019. Unsupervised learning of PCFGs with normalizing flow. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2442–2452, Florence, Italy. Association for Computational Linguistics.

Yoon Kim, Sam Wiseman, Andrew C Miller, David Sontag, and Alexander M Rush. 2018. Semi-amortized variational autoencoders. In *Proceedings of the 35th International Conference on Machine Learning*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *International Conference on Learning Representations*.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improved variational inference with inverse autoregressive flow. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4743–4751. Curran Associates, Inc.

Jason Lee, Dustin Tran, Orhan Firat, and Kyunghyun Cho. 2020. On the discrepancy between density estimation and sequence generation. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 84–94, Online. Association for Computational Linguistics.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3603–3614, Hong Kong, China. Association for Computational Linguistics.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Online. Association for Computational Linguistics.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4282–4292, Hong Kong, China. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Tom Pelsmaeker and Wilker Aziz. 2020. Effective estimation of deep generative language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7220–7236, Online. Association for Computational Linguistics.

Tom Rainforth, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. 2018. Tighter variational bounds are not necessarily better. In *International Conference on Machine Learning*, pages 4277–4285. PMLR.

Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. 2019. Preventing posterior collapse with delta-VAEs. In *International Conference on Learning Representations*.

Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *International Conference on Learning Representations*.

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2017. A hybrid convolutional variational autoencoder for text generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 627–637, Copenhagen, Denmark. Association for Computational Linguistics.

Iulian Vlad Serban, Alexander Ororbia II, Joelle Pineau, and Aaron Courville. 2017. Piecewise latent variables for neural variational text processing. In *Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing*, pages 52–62, Copenhagen, Denmark. Association for Computational Linguistics.

Hendra Setiawan, Matthias Sperber, Udhyakumar Nallasamy, and Matthias Paulik. 2020. Variational neural machine translation with normalizing flows. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7771–7777, Online. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Jakub M Tomczak and Max Welling. 2018. VAE with a VampPrior. In *Proceedings of AISTATS*.

Prince Zizhuang Wang and William Yang Wang. 2019. Riemannian normalizing flow on variational Wasserstein autoencoder for text modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 284–294, Minneapolis, Minnesota. Association for Computational Linguistics.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3881–3890. JMLR. org.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, Austin, Texas. Association for Computational Linguistics.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. 2017. InfoVAE: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*.

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

Zachary M Ziegler and Alexander M Rush. 2019. Latent normalizing flows for discrete sequences. In *International Conference on Learning Representations*.

Appendix

A Normalizing Flow

Change of Variable Formula. We start with a base density $p_\epsilon(\epsilon)$, and define a *bijection* function $f : \epsilon \rightarrow Z$, that maps from $\epsilon \sim p_\epsilon$ to the target $z \sim p_Z$. According to the *change of variable* formula:

$$\log p_Z(z) = \log p_\epsilon(f^{-1}(z)) + \log \left| \det \left(\frac{\partial f^{-1}(z)}{\partial z} \right) \right|$$

where $\frac{\partial f^{-1}(z)}{\partial z}$ is the Jacobian of f at z.

Normalizing Flows. A normalizing flow is a sequence of *invertible*, *deterministic* transformations. By repeatedly applying the rule for change of variables, the base density is transformed into a more complex one. Networks parameterized using NF can be trained through exact maximum log-likelihood computation. Exact sampling is performed by drawing sample from the base distribution and performing the chain of transformations. Our work uses NF because it allows a flexible functional form, and it is capable of capturing data complexity and performing exact likelihood computation and sampling.

Real NVP. Computing the Jacobian of functions with high dimension and the determinants of large matrices (i.e., the two main computation in NF) are very expensive. Prior work has addressed this challenge by introducing efficient transformations (Dinh et al., 2015, 2016; Germain et al., 2015; Kingma et al., 2016; Kingma and Dhariwal, 2018; Ho et al., 2019).

Our flow-based prior is based on *real-valued non-volume preserving* (real NVP; Dinh et al., 2016) which is efficient in both training and sampling. The main building block of real NVP transformation is the *affine coupling layer*.

An affine coupling layer is a bijective transformation $f_i : z_{i-1} \rightarrow z_i$ that follows the equations:

$$z_i^{(1:d)} = z_{i-1}^{(1:d)}$$
$$z_i^{(d+1:D)} = z_i^{(d+1:D)} \odot \exp(s(z_{i-1}^{(1:d)})) + t(z_{i-1}^{(1:d)})$$

where D is the dimensionality, $z_i^{(1:d)}$ stands for the first d dimensions of z_i ($d < D$); s and t denote the functions for *scale* and *translation* operations that map from $\mathbb{R}^d \rightarrow \mathbb{R}^{D-d}$; and $\odot$ denotes element-wise product.

The Jacobian determinant and inverse of the affine coupling layer are easy to compute. The transformation is flexible because its computation of the Jacobian determinant and inverse do not require any operation with the functions s and t, so these two functions could be designed to be arbitrarily complex.

B Datasets

The statistic of our dataset is in Table 7. For Yelp and SNLI, we follow Li et al. (2019) and create the dataset with downsampling. For Yahoo, we truncate sentences to length 100 due to computational constraints.

	# Train	# Dev	# Test	Avg L	Max L	# Vocab
PTB	42,068	3,370	3,761	21	82	10,002
Yahoo	100,000	10,000	10,000	68	100	19,982
Yelp	100,000	10,000	10,000	9	15	9,389
SNLI	100,000	10,000	10,000	12	82	19,978

Table 7: Statistics of the datasets. # Train/Dev/Test is the number of train/dev/test instances. Avg L and Max L are the average and maximum length of the sequences in the training sets. # Vocab is the size of the vocabulary including ⟨unk⟩, ⟨sos⟩, ⟨eos⟩, and ⟨pad⟩.

	PTB	Yahoo	SNLI	Yelp
Word Embedding	256	512	128	128
Encoder Hidden States	256	1024	512	512
Decoder Hidden States	256	1024	512	512

Table 8: The size of word embeddings and hidden states in VAE models used in this paper, which are adopted from prior work.

C Training Details

We use a batch size of 32 and train using SGD without momentum. The optimizer is initialized with learning rate 1 or 0.5, and the learning rate is decayed by 1/2 if the dev loss is not improved in two consecutive epochs. The training stops early after 5 learning rate decay operations. We use a linear annealing schedule that increases the weight from 0 to 1 in the first 10 or 20 epoch for the weight of both KL and $\mathrm{M_{IS}}$ term if they are in the training objective. When training with the combined objective, we start adding $\mathrm{M_{IS}}$ after training ELBO objective 10 epochs. For each model variation, we experiment with 5 different random seeds and report the median numbers in the paper.

D Hyperparameter Settings

Across all the experiments for our implemented baselines (i.e., standard VAE, MoG-VAE, VampVAE) and our proposed model FlowPrior, we follow prior work (Kim et al., 2018; He et al., 2019; Li et al., 2019) and use a single-layer LSTM encoder and decoder with a 32-dimensional latent variable.[9] We follow the prior work (He et al., 2019; Li et al., 2019) and set the embedding dimension as in Table 8. We set a dropout rate of 0.5 to both the

[9]For other baselines, we use their open source implementations.

Model	PPL(↓)	KL	AU(↑)	MI(↑)
		PTB		
VAE	101.4 / 101.6	0.00 / 4.46	0 / 32	0.0 / 0.1
VAE+M_{IS}	96.9 / 95.8	1.57 / 6.34	24 / 32	0.6 / 1.5
MoG-VAE	98.2 / 96.8	0.00 / 2.35	0 / 32	0.00 / 0.68
Vamp-VAE	98.3 / 97.3	0.00 / 2.31	0 / 32	0.00 / 0.72
FlowPrior	94.7 / 93.6	3.28 / 7.21	2 / 31	2.3 / 2.8
		Yahoo		
VAE	65.8 / 64.6	0.00 / 4.88	0 / 32	0.0 / 0.9
VAE+M_{IS}	63.9 / 61.7	2.72 / 13.31	1 / 32	2.0 / 1.7
MoG-VAE	64.6 / 67.3	0.00 / 1.83	0 / 32	0.0 / 0.6
Vamp-VAE	74.8 / 75.9	0.01 / 1.24	0 / 32	0.0 / 0.6
FlowPrior	62.5 / 68.3	1.43 / 10.99	4 / 25	1.6 / 0.6
		Yelp		
VAE	35.1 / 37.5	0.00 / 3.59	0 / 32	0.0 / 1.0
VAE+M_{IS}	33.2 / 39.6	2.91 / 4.16	28 / 32	0.9 / 2.2
MoG-VAE	35.2 / 39.8	0.01 / 1.81	0 / 32	0.0 / 0.6
Vamp-VAE	35.0 / 39.4	0.00 / 1.78	0 / 32	0.0 / 0.6
FlowPrior	31.8 / 39.0	4.15 / 10.13	2 / 32	2.5 / 2.6
		SNLI		
VAE	26.0 / 30.5	0.00 / 1.84	0 / 32	0.0 / 0.9
VAE+M_{IS}	25.3 / 17.8	1.23 / 15.48	23 / 32	0.5 / 2.0
MoG-VAE	28.1 / 27.5	0.44 / 2.28	1 / 32	0.4 / 0.7
Vamp-VAE	26.0 / 29.3	0.00 / 5.11	0 / 32	0.0 / 0.8
FlowPrior	22.4 / 26.2	3.83 / 7.59	3 / 32	1.0 / 3.2

Table 9: Results when comparing standard KL and FB KL for several models. The left part in each cell shows training with standard KL and the right part shows using FB KL instead.

input embeddings and the output embeddings before the softmax layer in the decoder. All the parameters are initialized with a uniform distribution U(-0.01,0.01). For both MoG and Vamp-VAE we use 100 components/pseudo-inputs in the prior. For real NVP, we use 10 affine coupling layers with 3-layer MLP networks for the parameterized scale and translation operations with the dimensionality of 32. We follow Dinh et al. (2016) to compose the affine coupling layers in an alternative pattern and add batch normalization (Ballé et al., 2016) between adjacent affine coupling layers. For models trained with FB KL, we set the target rate as 2, 4, or 8.

E Additional Results with Free Bits KL

Using the Free Bits method can help achieve a consistently better AU and higher KL as shown in the overall results in the main text. We report additional empirical comparisons to focus on measuring the impact of FB for three models in Table 9.

Though adding FB yields higher AU and MI, it is not always true that it leads to a better test PPL and reconstruction. This phenomenon has been pointed out by Razavi et al. (2019) that adding FB makes the objective non-smooth which can lead to optimization difficulties. Possible solutions could be changing to a better training procedure. Li et al. (2019) remedy this issue by combining pretraining with FB, namely using a pretrained autoencoder to initialize the inference network before starting training the VAE networks. This suggests that it may be necessary to pretrain the inference network and decoder to unilaterally benefit from FB.

Table 10 considers standard VAEs and FlowPrior when comparing the use of standard KL to FB KL. Using FB KL does not lead to a higher BLEU score in FlowPrior, though FB does improve BLEU when combined with standard VAE and VAE+M_{IS}.

	Yelp		**SNLI**	
	KL	FB KL	KL	FB KL
VAE	0.26	0.49	1.66	2.18
VAE+M_{IS}	0.65	0.71	1.91	3.57
FlowPrior	1.50	0.74	4.87	3.64

Table 10: Test set reconstruction BLEU scores.

	Yelp			**SNLI**		
	F-PPL	R-PPL	SB	F-PPL	R-PPL	SB
VAE	4	30248	96	4	51127	100
VAE+M_{IS}	5	10818	30	4	19047	73
MoG-VAE	4	30413	100	3	45979	77
MoG-VAE+M_{IS}	4	33624	100	6	5257	26
Vamp-VAE	4	32504	100	4	56050	100
Vamp-VAE+M_{IS}	7	5280	10	5	8420	29
FlowPrior	209	1677	3	42	5725	13
VAE+FB	7	7517	29	4	22536	42
VAE+M_{IS}+FB	8	5713	13	4	24204	48
FlowPrior+FB	8	5179	9	15	4876	11

Table 11: Forward PPL (F-PPL), Reverse PPL (R-PPL), and Self-BLEU (SB) of greedy-decoded prior samples.

F Additional Results with FB and M_{IS}

Table 11 shows the impact of M_{IS} and FB on F-PPL, R-PPL, and self-BLEU with greedy generation from prior samples.

G Reconstruction Results with Sampling

Tables 12-13 show the reconstruction performance with standard sampling and nucleus sampling with $p = 0.9$ (Holtzman et al., 2020). We observe the trends are consistent with the results that use greedy decoding.

	Yelp		SNLI	
	ELBO	ELBO+M_{IS}	ELBO	ELBO+M_{IS}
Standard	0.07	0.15	0.43	0.57
MoG	0.08	0.00	0.66	2.36
Vamp	0.06	0.32	0.55	0.76
Real NVP	0.28	0.60	0.97	1.91
	KL	FB KL	KL	FB KL
VAE	0.07	0.13	0.43	0.74
VAE+M_{IS}	0.15	0.28	0.57	0.95
MoG	0.08	0.08	0.66	0.54
Vamp	0.06	0.10	0.55	0.78
FlowPrior	0.60	0.34	1.91	0.91

Table 12: Test set reconstruction BLEU scores using standard sampling in decoding.

	Yelp		SNLI	
	ELBO	ELBO+M_{IS}	ELBO	ELBO+M_{IS}
Standard	0.08	0.20	0.56	0.72
MoG	0.09	0.06	0.80	2.66
Vamp	0.08	0.44	0.71	1.01
Real NVP	0.30	0.72	1.31	2.40
	KL	FB KL	KL	FB KL
VAE	0.08	0.17	0.56	1.03
VAE+M_{IS}	0.20	0.36	0.72	1.22
MoG	0.09	0.16	0.80	0.82
Vamp	0.08	0.13	0.71	1.07
FlowPrior	0.72	0.40	2.40	1.23

Table 13: Test set reconstruction BLEU scores using nucleus sampling in decoding.

H Interpolation with Sampling

Table 14 shows more examples of interpolation-based generation with greedy decoding. We show results with sampling methods for decoding in Tables 15 and 16. The results with greedy decoding provide a lower-variance way to interpret the learned latent space. The additional results with sampling methods provide a richer picture as they also capture the randomness in the relationship between the latent variable and the text. This is especially helpful when we observe repetition in neighboring samples with greedy decoding, as we see with MoG-VAE and Vamp-VAE in Table 14. Even with sampling, FlowPrior shows a smoother semantic evolution in the latent space than MoG-VAE and Vamp-VAE, at least in terms of aspects of the subjects of the generated sentences.

I Sampling from Priors

Table 17 shows more greedy generations from prior samples. We observe substantial improvements in term of generation diversity in FlowPrior and Flow-

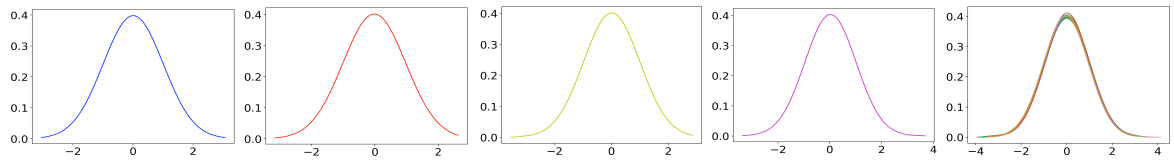
Mog-VAE
The man is wearing a black shirt .
The man is wearing a black shirt .
The man is wearing a black shirt .
A man is standing in front of a building .
A man is standing in front of a building .
A man is standing in front of a building .
A man is standing in front of a building .
A man is standing in front of a building .
A man is standing in front of a building .
A man is standing in front of a building .
Vamp-VAE
Three people are sitting on a bench .
People are walking down the street .
People are walking down the street .
People are walking down the street .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
Man in a blue shirt and jeans is sitting on a bench .
FlowPrior
The dog is running through the snow .
Two young boys are playing in the snow .
There is a man in a blue shirt and a woman in a black shirt and black pants .
Three people are sitting on a bench .
two men are standing on a bench
A girl is sitting on a bench .
A young girl is sitting on a bench .
A young man is sitting on a bench .
A woman in a black shirt is sitting on a bench .
A woman is sitting on a bench .

Table 14: Interpolation between two prior samples with greedy decoding. Dataset used is SNLI. In each cell, the first sentence and the last sentence correspond to the two sampled latent codes, and between are linearly interpolated samples.

Prior + FB. While standard VAE always yields identical samples because the latent variable is ignored, using FB KL yields better sample diversity as it encourages more information encoded into latent variable during training. This can be easily observed by comparing the samples from standard VAE with those from VAE+FB, comparing MoG-VAE with MoG-VAE + M_{IS}, comparing Vamp-VAE with Vamp-VAE + M_{IS}.

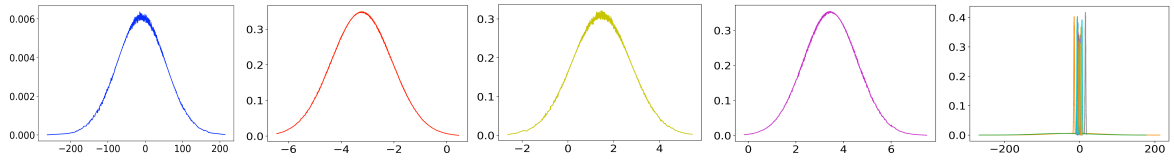
J More Visualizations of Real NVP Prior and FlowPrior

Visualizations of each dimension alone and all dimensions together are in Figures 2 and 3.



(a) dim 1 (b) dim 2 (c) dim 3 (d) dim 4 (e) all 32 dims

Figure 2: Visualization of dimensions of learned prior when using real NVP on SNLI dataset. Plots from left to right are first dimension alone, second dimension alone, third dimension alone, fourth dimension alone, and all 32 dimensions together.



(a) dim 1 (b) dim 2 (c) dim 3 (d) dim 4 (e) all 32 dims

Figure 3: Visualization of dimensions of learned prior when using real NVP with M_{IS} (i.e., FlowPrior) on SNLI dataset. Plots from left to right are first dimension alone, second dimension alone, third dimension alone, fourth dimension alone, and all 32 dimensions together.

K Reproducibility

We train our models on 1080 TI and 2080 TI GPUs. The number of parameters in each model is listed in Table 18. We report the average runtime for each approach on the PTB dataset in Table 19. We report the validation performance in Table 20.

The datasets are downloaded via https://github.com/jxhe/vae-lagging-encoder/blob/master/prepare_data.py, https://nlp.stanford.edu/projects/snli/, https://github.com/fangleai/Implicit-LVM/tree/master/lang_model_ptb/data, and https://github.com/fangleai/Implicit-LVM/tree/master/lang_model_yelp/data.

The links to open source implementation of the other baselines follow: Cyc-VAE (Fu et al., 2019) : github.com/snakeztc/NeuralDialog-CVAE; Lag-VAE (He et al., 2019) : github.com/jxhe/vae-lagging-encoder; Pre-VAE+FB (Li et al., 2019) : github.com/bohanli/vae-pretraining-encoder.

MoG-VAE
A girl is laughing at the beach .
The dog is walking in the water .
Man in khaki jacket painting an elephant .
The man is breakdancing .
People are outside on a sunny day .
Some people are playing in the snow .
Two men are working in a lab .
The boy is at the beach .
Five soccer players playing soccer .
Men stand on a pier .
Vamp-VAE
A person is riding a bicycle at a parade .
A brown and white dog with a brown collar is climbing over its hind legs while lying on the floor , talking and fabric in the grass .
A little girl wearing a yellow shirt looks at a fountain while a man is kneeling next to her and a child .
Two men play dominoes .
Young lady in blue dress waits at a bus .
A woman carrying a small child looking through a window . Three men are fighting with swords .
The little boy is riding his scooter down the paved road .
A man wearing a yellow suit eats a hotdog on a wooden table .
A group of friends are smiling.
FlowPrior
three dogs are in the water
3 people walking down the street with their hands in the air .
Man getting a picture
Three people are on the beach .
There are two dogs standing near each other .
Two men in white uniforms are cleaning on a mess of an escalator .
Two girls are getting some exercise together .
Two women working in a restaurant outdoors .
The children are riding .
The child is standing on the sidewalk in front of an apartment building .

Table 15: Interpolation between two prior samples with nucleus sampling for decoding. Dataset used is SNLI. In each cell, the first sentence and the last sentence correspond to the two sampled latent codes, and between are linearly interpolated samples.

MoG-VAE
A girl is attending a birthday of popcorn .
The dog is walking in the green snow .
Man lady on the beach .
The man is breakdancing .
People are outside on a sunny day .
Some people are rooting in an outdoor resaurant .
Two men both face out for directions on a street .
Big hikers .
Five soccer players playing soccer after fifty finish in a field .
Men stand on a doorstep .
Vamp-VAE
Two farmers share a drink , while one looks at a woolen .
A couple in black and white with a long blue scarf are standing in a store wearing a yellow hat .
A three people are riding a white bike through the desert along a road .
A tribesman near a playground .
People all gathered in the street looking at something on a woman .
A couple of angel making corn on a mountainside in a city .
A man holding a sign can and the young female .
The man playing a guitar concert festival .
The shirtless woman and woman performing on sand .
dog is outside .
FlowPrior
A dog is jumping through the air to catch a Frisbee in the air .
A brown and white dog chewing on a red disc .
A crowd of people are blowing in a brown down balloon .
A woman is pushing her cart .
A person is skiing through a snowy mountain .
A woman carrying a small child is playing with her friend on a busy street .
A man with a black shirt and brown long-sleeve shirt is standing near a graffiti that has come poles off around two .
A man , dressed with purple and black stands in bottoms while bandannas disbelief .
A man is looking at shoulder on a rack .
A man is about to fall .

Table 16: Interpolation between two prior samples with standard sampling for decoding. Dataset used is SNLI. In each cell, the first sentence and the last sentence correspond to the two sampled latent codes, and between are linearly interpolated samples.

VAE	**VAE + FB**
A man is sitting on a bench .	Two men are playing basketball .
A man is sitting on a bench .	A man is playing a guitar .
A man is sitting on a bench .	The man is wearing a blue shirt .
A man is sitting on a bench .	The man is wearing a blue shirt .
A man is sitting on a bench .	Two men are playing basketball .
MoG-VAE	**MoG-VAE + $\mathbf{M}_{\mathrm{IS}}$**
The man is wearing a black shirt .	An older gentleman in a white shirt is walking in a parking lot .
A man is standing in front of a building .	A dog is running .
A man is standing in front of a building .	A woman is walking in a field .
A man is standing in front of a building .	A young girl in a red shirt is playing with a toy .
A man is standing in front of a building .	An older gentleman in a white shirt and white pants is standing on a ladder with a large ladder on his right hand
Vamp-VAE	**Vamp-VAE + $\mathbf{M}_{\mathrm{IS}}$**
A man is playing a guitar .	Women in a white dress and a man in a black shirt are standing in front of a microphone .
A man is playing a guitar .	Man in a blue shirt and jeans is sitting on a bench .
A man is playing a guitar .	The man is wearing a black shirt .
A man is playing a guitar .	People are walking down the street .
A man is playing a guitar .	Two men are playing a game of chess .
FlowPrior	**FlowPrior + FB**
Man in a blue shirt and blue jeans is sitting on a rock with a hammer .	Children are standing in the middle of a building with a man in a blue shirt and black pants .
Two young boys are playing in the snow .	A man is standing in the middle of a large building .
A dog is running through the snow .	The man is wearing a black shirt .
Two men are standing on a boat .	Two men are playing basketball .
A young man is sitting on a bench .	Girl in a blue shirt and black jacket standing on a bench .

Table 17: Greedy generation from prior samples.

	VAE	MoG-VAE	Vamp-VAE	FlowPrior
PTB	8917074	8922174	8922174	9023314
Yahoo	55319136	55325536	55325536	55425376
Yelp	10260973	10267373	10367213	10263073
SNLI	18403914	18406814	18406814	18510154

Table 18: Number of parameters in each model.

	VAE	MoG-VAE	Vamp-VAE	FlowPrior
PTB	4550	5100	7000	8550
Yahoo	40110	49115	87253	92526
Yelp	1640	2652	3510	5620
SNLI	1242	2190	4081	9720

Table 19: Average runtime of each approach (s).

	VAE	MoG-VAE	Vamp-VAE	FlowPrior
PTB	101.50	101.28	101.27	100.14
Yahoo	332.30	332.12	344.06	330.01
Yelp	35.03	35.05	35.05	32.10
SNLI	41.83	41.83	41.86	41.17

Table 20: Corresponding validation performance (NLL on validation set) for each reported test result.