Improving Variational Autoencoders on Robustness, Regularization, and Task-Invariance

ロバスト性，正則化，タスク不変性に関する
変分オートエンコーダの改善

Hiroshi Takahashi

高橋 大志

Abstract

The variational autoencoder (VAE) is a powerful generative model that can learn data distributions by using latent variables and deep neural networks. The VAE can capture the high-dimensional and complicated data, can explicitly estimate the probability of a data point, and can be used for representation learning. Hence, the VAE can be applied to various fields such as image generation, text generation, speech synthesis, multi-modal analysis, graph mining, semi-supervised learning, continual learning, and out-of-distribution detection. However, the VAE has serious drawbacks and its real-world applications are work in progress. Our goal is to solve its drawbacks and make it practically applicable.

The VAE is composed of an encoder, decoder, and prior of the latent variable. The encoder models the conditional distribution of the latent variable given the data point. The decoder models the conditional distribution of the data point given the latent variable. The prior regularizes the encoder using the Kullback-Leibler (KL) divergence, which is modeled by a standard Gaussian distribution. The encoder and decoder are optimized by maximizing the evidence lower bound (ELBO), which is the tractable lower bound of the log-likelihood.

This dissertation addresses three serious drawbacks in real-world applications of the VAE: (1) instability of training, (2) over-regularization, and (3) poor performance with insufficient data points.

First, we focus on (1) the instability of training, which is caused by the the sensitivity of the decoder. For continuous data points, the decoder is modeled by the Gaussian distribution, where its mean and variance is estimated from an input data point by using deep neural networks. This Gaussian decoder is sensitive to the error between the data point and its estimated mean when its estimated variance is almost zero. With biased datasets, which are common in real-world applications,

this decoded variance often becomes almost zero, making the training unstable. We solve this instability problem by a Bayesian approach to the variance estimation: we set a prior for the variance of the Gaussian decoder, and marginalize it out analytically, which leads to proposing the Student-t VAE. We experimentally show that our Student-t VAE is robust and achieves high performance.

Next, we focus on (2) the over-regularization, which is caused by the too simple standard Gaussian prior. This leads to poor performance and prevents us from meeting the performance requirements of real-world applications. As a sophisticated prior, we introduce the aggregated posterior, the expectation of the decoder over the data distribution. Although this is known as the optimal prior for the VAE in terms of maximizing the ELBO, the KL divergence with this optimal prior cannot be calculated in a closed form. We can use the density ratio trick to estimate the KL divergence without modeling the aggregated posterior explicitly. However, since this KL divergence is high-dimensional density ratio, the density ratio trick does not work well. To solve this, we rewrite this KL divergence into the sum of the analytically calculable term and the low-dimensional density ratio, to which the density ratio is applied. We experimentally show that the VAE with our implicit optimal prior achieves high performance.

Finally, we focus on (3) the poor performance with insufficient data points, which are common in real-world applications. To overcome the lack of training data on the target task, the conditional VAE (CVAE) is widely used, which aims to share task-invariant knowledge with multiple tasks through the task-invariant latent variable. In the CVAE, the encoder models the conditional distribution of the latent variable given the data point and task, and is regularized by the task-invariant prior, which is modeled by the standard Gaussian distribution. Although this regularization encourages independence between the latent variable and task, the latent variable remains dependent on the task. We investigate why the CVAE cannot sufficiently reduce the task-dependency and show that the simple standard Gaussian prior is one of the causes. Based on this, we propose a theoretical optimal prior for reducing the task-dependency. We theoretically and experimentally show that our learned representation works well on the target tasks.

Acknowledgements

First, I would like to express my deepest gratitude to my supervisor Hisashi Kashima. He has provided continuous guidance and a lot of advice, which are essential in my Ph.D. I would like to express my gratitude to the dissertation committee members, Prof. Akihiro Yamamoto and Prof. Masatoshi Yoshikawa. Their advice is helpful in improving the quality of my thesis and determining the future direction of my work.

I would also like to express grateful thank to the collaborators of my studies. Tomoharu Iwata has taught me about research from the very beginning and always given me a lot of great advice. Yuki Yamanaka, Masanori Yamada, and Sekitoshi Kanai have always joined in my research discussions with me. Atsutoshi Kumagai has shared his experience and ideas for improving my research. Satoshi Yagi has taught me how to research in NTT.

Finally, I would like to thank my family for their devotion, continuous support, and encouragement over the years.

List of Publications

Publications Included in this Dissertation

International Conference

1. **(Chapter 3)**: Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, Satoshi Yagi.
   Student-t Variational Autoencoder for Robust Density Estimation.
   In *Proceedings of the 27-th International Conference on Artificial Intelligence (IJCAI)*, 2696–2702, 2018.

   https://www.ijcai.org/proceedings/2018/374

2. **(Chapter 4)**: Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, Satoshi Yagi.
   Variational Autoencoder with Implicit Optimal Priors.
   In *Proceedings of the 33-rd AAAI Conference on Artificial Intelligence (AAAI)*, 5066–5073, 2019.

   https://doi.org/10.1609/aaai.v33i01.33015066

3. **(Chapter 5)**: Hiroshi Takahashi, Tomoharu Iwata, Atsutoshi Kumagai, Sekitoshi Kanai, Masanori Yamada, Yuuki Yamanaka, Hisashi Kashima.
   Learning Optimal Priors for Task-Invariant Representations in Variational Autoencoders.
   In *Proceedings of the 28-th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 1739–1748, 2022.

https://dl.acm.org/doi/10.1145/3534678.3539291

Journals

1. **(Chapter 3)**: 高橋大志, 岩田具治, 山中友貴, 山田真徳, 八木哲志, 鹿島久嗣.
   "Student-t VAE によるロバスト確率密度推定".
   人工知能学会論文誌, 36(3), A-KA4_1, 2021.

https://doi.org/10.1527/tjsai.36-3_A-KA4

Publications Not Included in this Dissertation

International Conference

1. Sekitoshi Kanai, Masanori Yamada, Shin'ya Yamaguchi, Hiroshi Takahashi, Yasutoshi Ida.
   Constraining Logits by Bounded Function for Adversarial Robustness.
   In *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*, 1–8, 2021.
https://ieeexplore.ieee.org/abstract/document/9533777

2. Yuki Yamanaka, Tomoharu Iwata, Hiroshi Takahashi, Masanori Yamada, Sekitoshi Kanai.
   Autoencoding binary classifiers for supervised anomaly detection.
   In *Proceedings of the 2019 Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, 647–659, 2019.
https://link.springer.com/chapter/10.1007/978-3-030-29911-8_50

Preprints

1. Sekitoshi Kanai, Shin'ya Yamaguchi, Masanori Yamada, Hiroshi Takahashi, Yasutoshi Ida.

Switching One-Versus-the-Rest Loss to Increase the Margin of Logits for Adversarial Robustness.
arXiv:2207.10283, 2022.
https://arxiv.org/abs/2207.10283

2. Sekitoshi Kanai, Masanori Yamada, Hiroshi Takahashi, Yuki Yamanaka, Yasutoshi Id.
Smoothness Analysis of Loss Functions of Adversarial Training.
arXiv:2103.01400, 2021.
https://arxiv.org/abs/2103.01400

3. Masanori Yamada, Sekitoshi Kanai, Tomoharu Iwata, Tomokatsu Takahashi, Yuki Yamanaka, Hiroshi Takahashi, Atsutoshi Kumagai.
Adversarial Training Makes Weight Loss Landscape Sharper in Logistic Regression.
arXiv:2102.02950, 2021.
https://arxiv.org/abs/2102.02950

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Deep generative models, which try to estimate data distributions by using deep neural networks, play an important role in artificial intelligence. For the past decade, a lot of powerful approaches have been presented such as autoregressive models (Uria et al., 2016), variational autoencoders (VAE) (Kingma and Welling, 2014; Rezende et al., 2014), generative adversarial nets (GAN) (Goodfellow et al., 2014), flow-based models (Rezende and Mohamed, 2015; Papamakarios et al., 2021), and diffusion models (Sohl-Dickstein et al., 2015; Song and Ermon, 2019; Ho et al., 2020). Since they can capture the high-dimensional and complicated data, they can be used in various fields such as image generation (Gregor et al., 2015; van den Oord et al., 2017), text generation (Bowman et al., 2016; Yang et al., 2017), speech synthesis (Hsu et al., 2017; Akuzawa et al., 2018), multi-modal analysis (Khattar et al., 2019; Ivanovic et al., 2020), graph mining (Kipf and Welling, 2016; Li et al., 2020), semi-supervised learning (Ehsan Abbasnejad et al., 2017; Xu et al., 2017), continual learning (Achille et al., 2018; Egorov et al., 2021), and out-of-distribution detection (Xiao et al., 2020; Havtorn et al., 2021). However, they still have serious drawbacks and their real-world applications are work in progress. Our goal is to solve their drawbacks and make them practically applicable.

In this dissertation, we focus on the VAE for the following two reasons. First, the VAE can handle the probability of a data point explicitly, which enables us to apply it to various tasks easily. For example, in anomaly detection tasks, we can naturally model the normality of a data point by using the probability (An and Cho, 2015). Second, the VAE is also a powerful approach for representation learning: the latent

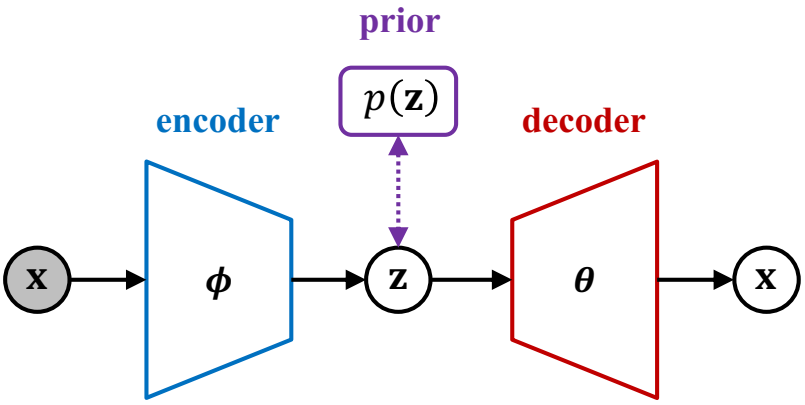


Figure 1.1: Diagram of VAE.

variables in the VAE improve the performance of downstream applications (Higgins et al., 2016).

The VAE is composed of the encoder, decoder, and prior of the latent variable (Kingma and Welling, 2014; Rezende et al., 2014). The encoder models the posterior of the latent variable given the data point. The decoder models the posterior of the data point given the latent variable. The prior regularizes the encoder using the Kullback-Leibler (KL) divergence, which is modeled by a standard Gaussian distribution. The encoder and decoder are optimized by maximizing the evidence lower bound (ELBO), which is the tractable lower bound of the log-likelihood. Figure 1.1 shows the diagram of the VAE. In chapter 2, we provide a more detailed review of the VAE.

In this dissertation, we try to solve following three serious drawbacks in real-world applications of the VAE:

1. instability of training for continuous data (Section 3),
2. over-regularization by prior (Section 4), and
3. poor performance with insufficient data (Section 5).

1.1 Instability of Training for Continuous Data

First, we focus on the instability of training for continuous data, which is caused by the sensitivity of the decoder. For continuous data points, the decoder is modeled by the Gaussian distribution, where its mean and variance is estimated from an input

data point by using deep neural networks. We call this type of VAE the Gaussian VAE. However, the training of the Gaussian VAE often becomes unstable. The reason is as follows: the training objective function of Gaussian VAE is sensitive to the error between the data point and its decoded mean when its decoded variance is almost zero, and hence, the objective function can give an extremely large value even with a small error. We call this problem *zero-variance problem*. This zero-variance problem often occurs with *biased data*, i.e. in which some clusters of data points have small variance. In the example of sensor data, some sensors show the same values most of the time, but sometimes show different values. Such biased features would be important for e.g. anomaly detection. In addition to sensor data, datasets such as network and media datasets often have this bias. Therefore, this problem is serious considering the application of the VAE to real-world applications.

Our purpose is to solve this instability by making the decoder robust to the error between the data point and its decoded mean. Intuitively, we can improve the robustness of the decoder by replacing the Gaussian distribution with the Student-t distribution, which is a heavy-tailed distribution (Lange et al., 1989). In chapter 3, we verify the validity of this intuition through introducing a Bayesian approach to the Gaussian decoder. At first, to prevent the decoded variance from being too small, we set a Gamma distribution as the conjugate prior for its inverse. Unfortunately, this prior has constant hyperparameters and lacks the flexibility in density estimation. To solve this, we next make these hyperparameters dependent on latent variables and marginalize the decoded variance out analytically. As a result, we obtain the Student-t distribution as the decoder distribution. We call this proposed method the Student-t VAE. As can be seen from this derivation, the Student-t VAE can avoid the zero-variance problem without lacking the flexibility, which leads to the stability of the training.

1.2 Over-Regularization by Prior

Next, we focus on the over-regularization. Recent research shows that the prior plays an important role in the VAE (Hoffman and Johnson, 2016). Although the standard Gaussian prior is usually used, this simple prior incurs over-regularization. This leads to poor performance and prevents us from meeting the performance requirements

of real-world applications. This over-regularization is also known as the posterior-collapse phenomenon (van den Oord et al., 2017). To improve the performance of the VAE, the aggregated posterior prior has been introduced, which is the expectation of the encoder over the data distribution (Hoffman and Johnson, 2016). The aggregated posterior is an optimal prior in terms of maximizing the training objective function of the VAE. However, KL divergence with the aggregated posterior cannot be calculated in a closed form, which prevents us from using this optimal prior. In previous work (Tomczak and Welling, 2018), the aggregated posterior is modeled by using the finite mixture of encoders for calculating the KL divergence in a closed form. Nevertheless, it has sensitive hyperparameters such as the number of mixture components, which are difficult to tune.

Our purpose is to solve this over-regularization without difficult hyperparameter tuning. In chapter 4, we propose the VAE with implicit optimal priors, where the aggregated posterior is used as the prior, but the KL divergence is directly estimated without modeling the aggregated posterior explicitly. This implicit modeling enables us to avoid the difficult hyperparameter tuning for the aggregated posterior model. We use the density ratio trick, which can estimate the density ratio between two distributions without modeling each distribution explicitly, since the KL divergence is the expectation of the density ratio between the encoder and aggregated posterior. Although the density ratio trick is powerful, it is experimentally shown to work poorly in high dimensions (Sugiyama et al., 2012; Rosca et al., 2018). Unfortunately, with high-dimensional datasets, the density ratio between the encoder and the aggregated posterior also becomes high-dimensional. To avoid the density ratio estimation in high dimensions, we rewrite the KL divergence with the aggregated posterior to the sum of two terms. The first term is the KL divergence between the encoder and the standard Gaussian prior, which can be calculated in a closed form. The other term is the low-dimensional density ratio between the aggregated posterior and the standard Gaussian distribution, to which the density ratio trick is applied.

1.3 Poor Performance with Insufficient Data

Finally, we focus on the poor performance with insufficient data points. Although the VAE is the powerful generative model, it cannot perform well with insufficient

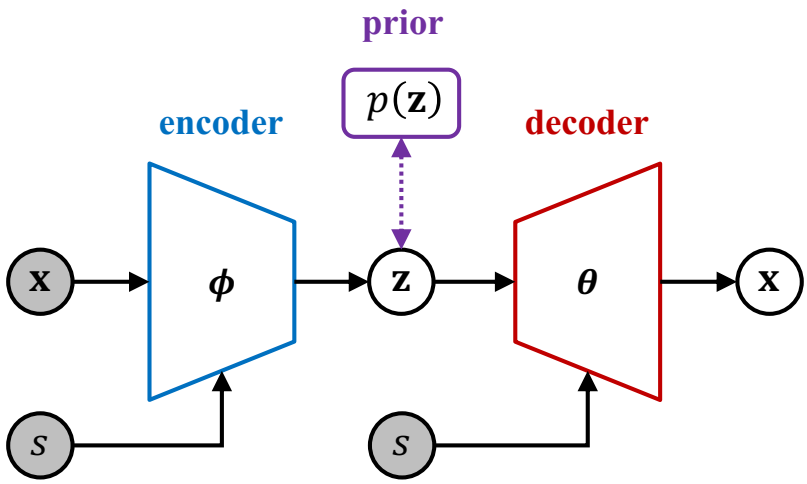


Figure 1.2: Diagram of CVAE.

data points since it requires many data points for training neural networks. This is a serious problem in real-world applications considering it is often impossible to prepare sufficient data points. For example, when detecting anomalies in a new factory, there is initially not enough data for training.

To overcome the lack of training data on the target task, multi-task learning (Caruana, 1997), which improves the performance on each task by sharing task-invariant knowledge with multiple tasks, has been widely researched. In the above example, we can obtain training data points from the other factories. To share task-invariant knowledge effectively, recent research has mainly focused on obtaining the task-invariant representations of data points and has shown promising results in many applications (Ghifary et al., 2015; Liu et al., 2017; Long et al., 2017). To apply the multi-task learning for the VAE, the conditional VAE (CVAE) (Kingma et al., 2014; Sohn et al., 2015) has been widely used, which aims to obtain the task-invariant variable in an unsupervised manner.

The CVAE is composed of the encoder, decoder, and prior of the latent variable. The encoder models the posterior of the latent variable given the data point and task. The decoder models the posterior of the data point given the latent variable and task. The prior regularizes the encoder using the Kullback-Leibler (KL) divergence and is modeled by a standard Gaussian distribution. The encoder and decoder are optimized by maximizing the evidence lower bound (ELBO), which is the tractable lower bound of the log-likelihood. Figure 1.2 shows the diagram of the CVAE. In

chapter 2, we provide a more detailed review of the CVAE. Since the encoder of all tasks is regularized by the same standard Gaussian prior, the CVAE is expected to reduce the dependency of the latent variable on the task. We call this the task-dependency.

However, the task-dependency of the CVAE remains in many cases (Louizos et al., 2016). In such cases, the encoder maps data points of a task and another task to different parts in the latent space, even if these data points represent the same concept (e.g., same class). This worsens the performance of downstream applications for target tasks. As an approach to reduce the task-dependency, the variational fair autoencoder (VFAE) (Louizos et al., 2016) was presented. The VFAE introduces the additional regularizer for the posteriors of the latent variables given tasks on the basis of maximum mean discrepancy (MMD) (Gretton et al., 2007) so that they become similar. Although the VFAE can reduce the task-dependency, its learned representation does not work well on the target tasks. This is because the MMD regularizer loosens the lower bound of the log-likelihood, preventing the VFAE from maximizing the log-likelihood sufficiently. As a result, the VFAE does not generalize well for the test data on the target tasks.

In chapter 5, we first reveal the cause of the remaining task-dependency in the CVAE. To measure the task-dependency, we introduce the mutual information (Cover and Thomas, 2012) between tasks and latent variables. We show that this mutual information is bounded above by the expectation of the KL divergence between the encoder and the prior, which is minimized in the CVAE training. Thus, the CVAE tries to minimize this mutual information via minimizing its upper bound. However, this is not a tight upper bound because the simple standard Gaussian prior loosens the upper bound, and the mutual information cannot be minimized sufficiently. Hence, the simple prior is one of the causes of the remaining task-dependency in the CVAE.

To solve this, we propose an optimal prior for reducing the task-dependency, which provides the tightest upper bound of the mutual information. We also show that this optimal prior presents a better lower bound of the log-likelihood than those of the CVAE and VFAE, which enables us to obtain better representations for the improved performance on the target tasks. Therefore, our approach can obtain better task-invariant latent representations than existing methods.

Chapter 2

Preliminaries

In this chapter, we provide a detailed review of the variational autoencoder (VAE) (Kingma and Welling, 2014; Rezende et al., 2014) and conditional VAE (Kingma et al., 2014; Sohn et al., 2015).

2.1 Variational Autoencoder

First, we review the VAE. The VAE is a probabilistic latent variable model that relates an observed variable vector $\mathbf{x}$ to a low-dimensional latent variable vector $\mathbf{z}$ by a conditional distribution. The VAE models the probability of a data point $\mathbf{x}$ by

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z}, \tag{2.1}$$

where $p_\theta(\mathbf{x}|\mathbf{z})$ is the conditional distribution of $\mathbf{x}$ given $\mathbf{z}$ and s, and $p(\mathbf{z})$ is a prior of $\mathbf{z}$. $p_\theta(\mathbf{x}|\mathbf{z})$ is modeled by a neural network with parameter θ, called the decoder. For example, if $\mathbf{x}$ is binary, the decoder is modeled by a Bernoulli distribution $\mathcal{B}(\mathbf{x}|\mu_\theta(\mathbf{z}))$, where $\mu_\theta(\mathbf{z})$ is neural networks with parameter θ. If $\mathbf{x}$ is continuous, the decoder is modeled by a Gaussian distribution $\mathcal{N}(\mathbf{x}|\mu_\theta(\mathbf{z}), \sigma^2_\theta(\mathbf{z}))$, where $\mu_\theta(\mathbf{z})$ and variance $\sigma^2_\theta(\mathbf{z})$ are neural networks with parameter θ. Figure 2.1(a) shows the graphical model of the decoder. $p(\mathbf{z})$ is modeled by a standard Gaussian distribution $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ (Kingma and Welling, 2014; Rezende et al., 2014).

The VAE is trained to maximize the ELBO, which is the tractable lower bound of the log-likelihood since the log-likelihood itself is hard to handle. The ELBO for

(a) Decoder (b) Encoder

Figure 2.1: Graphical models of the encoder and the decoder of the VAE. (a) The decoder estimates data point $\mathbf{x}$ from latent variable vector $\mathbf{z}$. (b) The encoder estimates $\mathbf{z}$ from $\mathbf{x}$.

each data point $\mathbf{x}$ is derived from Jensen's inequality as follows:

$$\ln p_\theta(\mathbf{x}) = \ln \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln \frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \equiv \mathcal{L}(\mathbf{x};\theta,\phi), \quad (2.2)$$

where $\mathbb{E}[\cdot]$ is the expectation, and $q_\phi(\mathbf{z}|\mathbf{x})$ is the posterior of $\mathbf{z}$ given $\mathbf{x}$. $q_\phi(\mathbf{z}|\mathbf{x})$ is modeled by a neural network with parameter ϕ, called the encoder. The encoder is usually modeled by a Gaussian distribution $\mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \sigma^2_\phi(\mathbf{x}))$, where $\mu_\phi(\mathbf{x})$ and $\sigma^2_\phi(\mathbf{x})$ are neural networks with parameter ϕ. Figure 2.1(b) shows the graphical model of the encoder. The ELBO (Eq. (2.2)) can be also written as

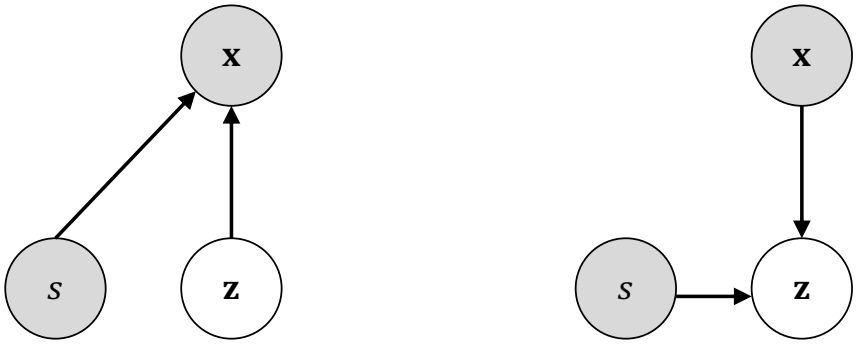
$$\mathcal{L}(\mathbf{x};\theta,\phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})), \quad (2.3)$$

where $D_{\mathrm{KL}}(P\|Q)$ is the Kullback Leibler (KL) divergence between P and Q. The second expectation term in Eq. (2.3) is called the reconstruction term, which is also known as the negative reconstruction error.

The parameters θ and ϕ are optimized by maximizing the ELBO as follows:

$$\max_{\theta,\phi} \int p_{\mathcal{D}}(\mathbf{x})\mathcal{L}(\mathbf{x};\theta,\phi)\mathrm{d}\mathbf{x}, \quad (2.4)$$

where $p_{\mathcal{D}}(\mathbf{x})$ is the data distribution.



(a) Decoder (b) Encoder

Figure 2.2: Graphical models of the encoder and the decoder of the CVAE. (a) The decoder estimates data point $\mathbf{x}$ from latent variable vector $\mathbf{z}$ and task index s. (b) The encoder estimates $\mathbf{z}$ from $\mathbf{x}$ and s.

2.2 Conditional Variational Autoencoder

Next, we review the CVAE. Suppose that we are given task index s for each data point $\mathbf{x}$. The CVAE models the conditional probability of $\mathbf{x}$ given s using latent variable vector $\mathbf{z}$ as follows:

$$p_\theta(\mathbf{x}|s) = \int p_\theta(\mathbf{x}|\mathbf{z}, s)p(\mathbf{z})\mathrm{d}\mathbf{z}, \tag{2.5}$$

where $p_\theta(\mathbf{x}|\mathbf{z}, s)$ is the conditional distribution of $\mathbf{x}$ given $\mathbf{z}$ and s, and $p(\mathbf{z})$ is a prior of $\mathbf{z}$. $p_\theta(\mathbf{x}|\mathbf{z}, s)$ is modeled by a neural network with parameter θ, called the decoder. Figure 2.2(a) shows the graphical model of the decoder. $p(\mathbf{z})$ is modeled by a standard Gaussian distribution $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ (Kingma et al., 2014; Sohn et al., 2015).

The CVAE is trained to maximize the ELBO, which is the tractable lower bound of the log-likelihood since the log-likelihood itself is hard to handle. The ELBO for each data point $\mathbf{x}$ is derived from Jensen's inequality as follows:

$$\ln p_\theta(\mathbf{x}|s) = \ln \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\frac{p_\theta(\mathbf{x}|\mathbf{z}, s)p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}, s)}\right] \tag{2.6}$$

$$\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln \frac{p_\theta(\mathbf{x}|\mathbf{z}, s)p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}, s)}\right] \tag{2.7}$$

$$\equiv \mathcal{L}(\mathbf{x}, s; \theta, \phi), \tag{2.8}$$

where $q_\phi(\mathbf{z}|\mathbf{x}, s)$ is the posterior of $\mathbf{z}$ given $\mathbf{x}$ and s. $q_\phi(\mathbf{z}|\mathbf{x}, s)$ is modeled by a neural network with parameter ϕ, called the encoder. Figure 2.2(b) shows the graphical model of the encoder. Eq. (2.8) can be rewritten as follows:

$$\mathcal{L}(\mathbf{x}, s; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)} \left[\ln p_\theta(\mathbf{x}|\mathbf{z}, s)\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|p(\mathbf{z})). \tag{2.9}$$

The first expectation term in Eq. (2.9) is known as the negative reconstruction error.

The parameters θ and ϕ are optimized by maximizing the ELBO as follows:

$$\max_{\theta,\phi} \mathbb{E}_{p_\mathcal{D}(\mathbf{x},s)} \left[\mathcal{L}(\mathbf{x}, s; \theta, \phi)\right], \tag{2.10}$$

where $p_\mathcal{D}(\mathbf{x}, s) = p_\mathcal{D}(\mathbf{x}|s)p_\mathcal{D}(s)$ is the joint distribution of the data point and task.

Chapter 3

Student-t Variational Autoencoder for Robust Density Estimation

3.1 Introduction

In this chapter, we focus on the instability of training for continuous data, described in Section 1.1. For continuous data, the Gaussian distribution is used as the decoder. We provide a detailed review of this Gaussian VAE in section 3.2. However, the training of the Gaussian VAE often becomes unstable. In section 3.3, we investigate this instability problem by applying the Gaussian VAE to some dataset, and reveal that the training objective function of Gaussian VAE becomes sensitive to the error between the data point and its decoded mean when its decoded variance is almost zero. In section 3.4, to solve this problem, we propose the Student-t VAE, which introduces Bayesian approach to the inference of the Gaussian decoder. We evaluate our Student-t VAE on various datasets in section 3.5, and discuss the relationship between our Student-t VAE and existing approaches in section 3.6. Finally, we summarize our contributions and discuss the future work in section 3.7.

3.2 Preliminaries

First, we again review the VAE (Kingma and Welling, 2014; Rezende et al., 2014), focusing on the case of continuous data points. The VAE estimates the probability

of data point $\mathbf{x}$ by using latent variable vector $\mathbf{z}$ as follows:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z}, \tag{3.1}$$

where $p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mu_\theta(\mathbf{z}), \sigma^2_\theta(\mathbf{z}))$ is the Gaussian decoder, and $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ is the standard Gaussian prior. We call this type of VAE the Gaussian VAE.

Given a dataset $\mathbf{X} = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right\}$, the ELBO for each data point $\mathbf{x}^{(i)}$ can be written as

$$\mathcal{L}(\mathbf{x}^{(i)}; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})\|p(\mathbf{z})), \tag{3.2}$$

where $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \sigma^2_\phi(\mathbf{x}))$ is the encoder. The expectation term in Eq. (3.2), known as the negative reconstruction error, can be approximated by using the reparameterization trick (Kingma and Welling, 2014) as follows:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\left[\ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z})\right] \simeq \frac{1}{L}\sum_{\ell=1}^{L} \ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}), \tag{3.3}$$

where $\mathbf{z}^{(i,\ell)} = \mu_\phi(\mathbf{x}^{(i)}) + \varepsilon^{(i,\ell)}\sigma_\phi(\mathbf{x}^{(i)})$, $\varepsilon^{(i,\ell)}$ is a sample drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$, and L is the sample size of the reparameterization trick. $L = 1$ is usually used (Kingma and Welling, 2014). The KL divergence term in Eq. (3.2) and its gradient can be calculated in a closed form since both distributions are Gaussian (Kingma and Welling, 2014). Hence, Eq. (3.2) is approximated as

$$\mathcal{L}(\mathbf{x}^{(i)}; \theta, \phi) \simeq \frac{1}{L}\sum_{\ell=1}^{L} \ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})\|p(\mathbf{z})) \tag{3.4}$$

$$\equiv \hat{\mathcal{L}}(\mathbf{x}^{(i)}; \theta, \phi), \tag{3.5}$$

and the parameters θ and ϕ are optimized by maximizing the following ELBO:

$$\frac{1}{N}\sum_{i=1}^{N} \hat{\mathcal{L}}(\mathbf{x}^{(i)}; \theta, \phi) \tag{3.6}$$

using stochastic gradient descent (SGD) (Duchi et al., 2011; Zeiler, 2012; Tieleman and Hinton, 2012; Kingma and Ba, 2015). In this chapter, we minimize the negative

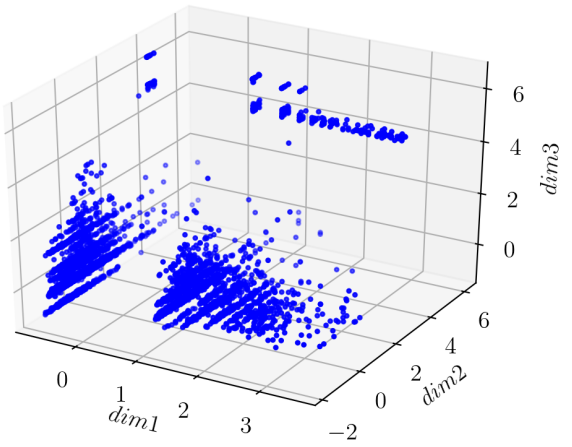


Figure 3.1: Visualization of the SMTP data.

of (3.6) instead of maximizing it.

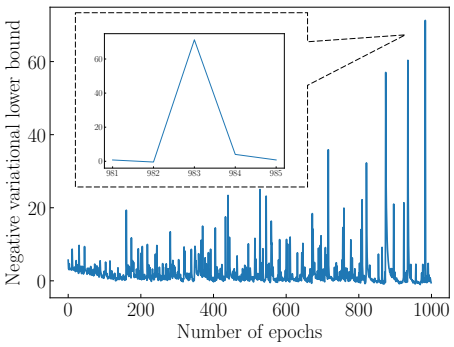
3.3 Instability of Training Gaussian VAE

To investigate the Gaussian VAE, we applied it to SMTP data[1], which is a subset of the KDD Cup 1999 data and provided by the scikit-learn community (Pedregosa et al., 2011). The KDD Cup 1999 data were generated using a closed network and hand-injected attacks for evaluating the performance of supervised network intrusion detection, and they have often been used also for unsupervised anomaly detection. The SMTP data consists of three-dimensional continuous data, and contains 95,156 data points. Figure 3.1 shows a visualization of this dataset. This dataset has some bias: the variance of some clusters of data points is small along the dimension directions.

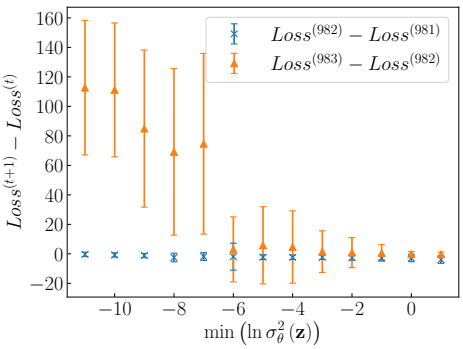
We trained the Gaussian VAE using this dataset by Adam (Kingma and Ba, 2015) with mini-batch size of 100. We used a two-dimensional latent variable vector $\mathbf{z}$, two-layer neural networks (500 hidden units per layer) as the encoder and the decoder, and a hyperbolic tangent as the activation function. The data were standardized with zero mean and unit variance. We used 10% of this dataset for training.

Figure 3.2(a) shows the mean training loss, which equals $-\sum_{i=1}^{N} \hat{\mathcal{L}}(\mathbf{x}^{(i)}; \theta, \phi)/N$. The training loss was very unstable. One reason for this is that the variance $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$

[1]This dataset is available at `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_kddcup99.html`



(a) Mean training loss.



(b) Relation of decoded variance to loss.

Figure 3.2: (a) Mean training loss for the SMTP data. The inset is an enlargement near the 983rd epoch. (b) Relation of the decoded variance to the difference in training losses. $Loss^{(t)}$ represents $-\hat{\mathcal{L}}(\mathbf{x}^{(i)}; \theta, \phi)$ for each data at the tth epoch, and $\min(\ln \sigma_\theta^2(\mathbf{z}))$ represents minimum of $\ln \sigma_\theta^2(\mathbf{z})$. We plotted the means and standard deviations of $Loss^{(t)} - Loss^{(t-1)}$, where $\min(\ln \sigma_\theta^2(\mathbf{z})) \in [c - 0.5, c + 0.5]$, for c in $\mathbb{Z}$. From 982nd to 983rd, training loss of data points with small $\ln \sigma_\theta^2(\mathbf{z})$ increased drastically.

in the decoder $\mathcal{N}(\mathbf{x}^{(i)}|\mu_\theta(\mathbf{z}^{(i,\ell)}), \sigma_\theta^2(\mathbf{z}^{(i,\ell)}))$ becomes almost zero, where $\mathbf{z}^{(i,\ell)}$ is sampled from the encoder $\mathcal{N}(\mathbf{z}^{(i,\ell)}|\mu_\phi(\mathbf{x}^{(i)}), \sigma_\phi^2(\mathbf{x}^{(i)}))$. For example, at the 983rd epoch, the training loss jumped up sharply. Figure 3.2(b) shows the relationship between the difference in the training losses and the variance $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$ at this epoch. The training loss of the data points with small variance $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$ increased drastically.

When the decoded variance $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$ is almost zero, the Gaussian decoder is sensitive to the error between the data point and its decoded mean: even if $\mathbf{x}^{(i)}$ differs only slightly from its decoded mean $\mu_\theta(\mathbf{z}^{(i,\ell)})$, the value of the first term of the training objective function (3.5):

$$\begin{aligned} \ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}) &= \ln \mathcal{N}(\mathbf{x}^{(i)}|\mu_\theta(\mathbf{z}^{(i,\ell)}), \sigma_\theta^2(\mathbf{z}^{(i,\ell)})) \\ &= \sum_d \left[-\frac{(\mathbf{x}_d^{(i)} - \mu_{\theta,d}(\mathbf{z}^{(i,\ell)}))^2}{2\sigma_{\theta,d}^2(\mathbf{z}^{(i,\ell)})} - \frac{1}{2} \ln 2\pi\sigma_{\theta,d}^2(\mathbf{z}^{(i,\ell)}) \right] \end{aligned} \tag{3.7}$$

changes drastically, where d is the dimension index of $\mathbf{x}^{(i)}$, $\mu_\theta(\mathbf{z}^{(i,\ell)})$, and $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$. This sensitivity makes the training of the Gaussian VAE unstable. We call this problem the zero-variance problem. This zero-variance problem often occurs with

biased data, since some data points are from a cluster with very small variance, and their decoded variance becomes much smaller as the training proceeds. Since the cause of this problem is the sensitivity of the objective function, changing the optimizer and tuning the hyperparameters of optimizer do not matter.

3.4 Student-t VAE

We would like to solve this zero-variance problem by making the decoder robust to the error between the data point and its decoded mean. Intuitively, we can solve this problem by using the Student-t distribution as the decoder, as described in section 1.1, In this section, we theoretically derive the Student-t decoder through a Bayesian approach to the inference of the Gaussian decoder.

We introduce a prior distribution for the variance of the Gaussian decoder. Let the precision parameter $\tau_\theta(\mathbf{z})$ be the inverse of the variance, $\tau_\theta(\mathbf{z}) = 1/\sigma_\theta^2(\mathbf{z})$. We use the conjugate prior for the precision parameter, which is the following Gamma distribution:

$$\mathrm{Gam}(\tau|a,b) = \frac{b^a \tau^{a-1} \exp(-b\tau)}{\Gamma(a)}, \tag{3.8}$$

where a is a shape parameter and b is a rate parameter. The domains of a and b are positive.

3.4.1 MAP Estimation

First, we present the maximum a posteriori (MAP) estimation as a way to solve the zero-variance problem. To simplify the calculation, we use $\mathrm{Gam}(\tau \mid 1, b)$ as the prior for $1/\sigma_\theta^2(\mathbf{z})$. Its log probability is given by

$$\ln \mathrm{Gam}(\tau_\theta(\mathbf{z}) \mid 1, b) = \ln b - b\tau_\theta(\mathbf{z}) \propto -\frac{b}{\sigma_\theta^2(\mathbf{z})}. \tag{3.9}$$

Accordingly, the objective function of MAP estimation for the VAE is:

$$\frac{1}{L}\sum_{l=1}^{L}\left\{\ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}) - \frac{b}{\sigma_\theta^2(\mathbf{z}^{(i,\ell)})}\right\} - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})\|p(\mathbf{z})). \tag{3.10}$$

We call this type of VAE the MAP VAE. This objective function can be viewed as a regularized version of the original Gaussian VAE objective function (3.5), where the small variance $\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$ is penalized by $-b/\sigma_\theta^2(\mathbf{z}^{(i,\ell)})$ with the regularization parameter b. This regularization parameter b can be tuned by cross-validation, which requires a heavy computational cost. In addition, the MAP VAE has a problem in that the prior distribution is assumed to be independent of latent variables $\mathbf{z}$, which leads to a lack of flexibility in density estimation.

3.4.2 Marginalization of the Variance Parameter

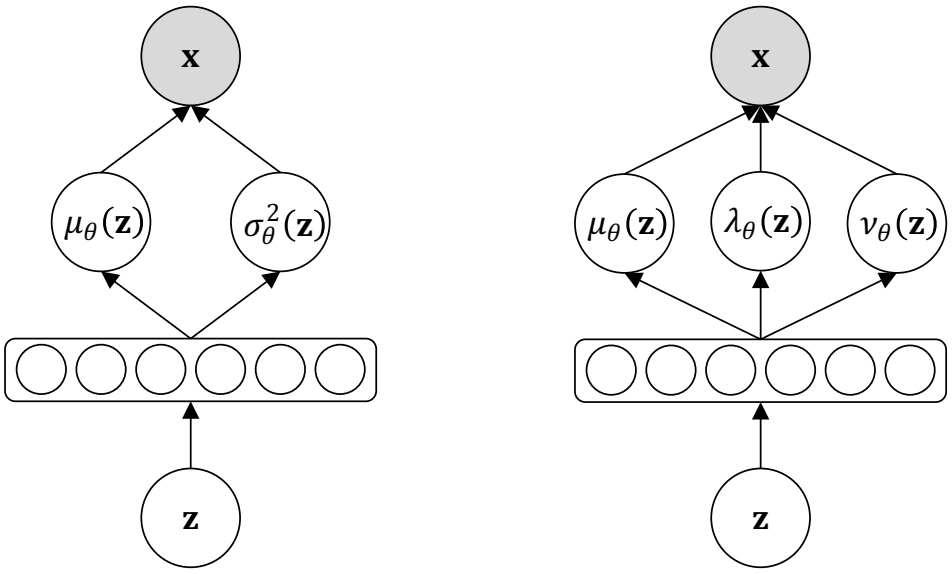
We propose a more flexible and computationally efficient approach by introducing a Gamma prior distribution that depends on latent variables, $\mathrm{Gam}(\tau|a(\mathbf{z}), b(\mathbf{z}))$, where $a(\mathbf{z})$ and $b(\mathbf{z})$ are the shape and rate parameters, respectively, which take the latent variable $\mathbf{z}$ as their inputs. By analytically integrating out the precision τ, the conditional distribution of the data point $\mathbf{x}$ given the latent variable $\mathbf{z}$ becomes a Student-t distribution as follows:

$$\begin{aligned} p_\theta(\mathbf{x}|\mathbf{z}) &= \int_0^\infty \mathcal{N}(\mathbf{x}|\mu_\theta(\mathbf{z}), \tau^{-1})\mathrm{Gam}(\tau|a(\mathbf{z}), b(\mathbf{z}))\mathrm{d}\tau \\ &= \frac{\Gamma(\frac{\nu_\theta(\mathbf{z})+1}{2})}{\Gamma(\frac{\nu_\theta(\mathbf{z})}{2})} \left(\frac{\lambda_\theta(\mathbf{z})}{\pi\nu_\theta(\mathbf{z})}\right)^{\frac{1}{2}} \left[1 + \frac{\lambda_\theta(\mathbf{z})(\mathbf{x}-\mu_\theta(\mathbf{z}))^2}{\nu_\theta(\mathbf{z})}\right]^{-\frac{\nu_\theta(\mathbf{z})+1}{2}} \\ &= \mathrm{St}(\mathbf{x}|\mu_\theta(\mathbf{z}), \lambda_\theta(\mathbf{z}), \nu_\theta(\mathbf{z})), \end{aligned} \tag{3.11}$$

where $\lambda_\theta(\mathbf{z}) = a(\mathbf{z})/b(\mathbf{z})$ is a precision[2] of a Student-t distribution, and $\nu_\theta(\mathbf{z}) = 2a(\mathbf{z})$ is a degree of freedom. We use neural networks to model the parameters of the Student-t distribution, $\lambda_\theta(\mathbf{z})$ and $\nu_\theta(\mathbf{z})$ as well as $\mu_\theta(\mathbf{z})$, and use them as the decoder. We call this type of VAE the Student-t VAE. The diagram of Student-t decoder is illustrated in Figure 3.3.

Figure 3.4 shows a plot of the Student-t distribution. The Student-t distribution is obtained by mixing an infinite number of Gaussians that have the same mean but different variances, and it has an important property: the tail heaviness. It is well-known that a heavy-tailed distribution is robust because the probability in the tail region is higher than that of a light-tailed distribution such as a Gaussian

[2]This parameter is not always equal to the inverse of the variance.



(a) Gaussian decoder. (b) Student-t decoder.

Figure 3.3: The diagram of decoders. The decoder network takes the latent variable z as input, and estimates the parameters of assumed distribution. (a) Gaussian decoder network estimates the mean $\mu_\theta(\mathbf{z})$ and variance $\sigma^2_\theta(\mathbf{z})$. (b) Student-$t$ decoder network estimates the mean $\mu_\theta(\mathbf{z})$, precision $\lambda_\theta(\mathbf{z})$, and degree of freedom $\nu_\theta(\mathbf{z})$.

distribution (Lange et al., 1989). Therefore, the Student-t decoder is robust to the error between the data point and its decoded mean, which makes the training of the Student-t VAE stable. Since the degree of tail heaviness can be adjusted by tuning ν, the Student-t VAE can set an appropriate robustness for each data point by estimating $\nu_\theta(\mathbf{z})$, which leads to obtaining better flexibility than the MAP VAE. Since there is no need to use the cross-validation for tuning the parameters such as the MAP VAE, the Student-t VAE requires only a lightweight computational cost.

3.5 Experiments

In this section, we evaluate the robustness of the training and the density estimation performance of the Student-t VAE.

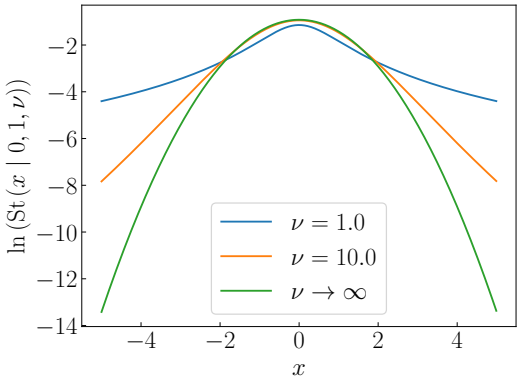


Figure 3.4: Plot of Student-t distribution $\mathrm{St}(x \mid 0, 1, \nu)$ in log scale for various values of ν. The Student-t distribution has a heavier tail compared with a Gaussian, and in the case of $\nu \to \infty$, $\mathrm{St}(x|\mu, \lambda, \nu)$ corresponds to a Gaussian $\mathcal{N}(x|\mu, \lambda^{-1})$.

	SMTP	Aloi	Thyroid	Cancer	Satellite
Data size	95,156	50,000	6,916	367	5,100
Dimension	3	27	21	30	36

Table 3.1: Number of data points and dimensions of five datasets.

3.5.1 Data

We used the following five datasets: SMTP, Aloi, Thyroid, Cancer and Satellite. The SMTP data is the same as the data used in Section 3.3, where we used 10% of this dataset for training. We also used 10% of this dataset for validation and the remaining 80% for test. The Aloi data is the Amsterdam library of object images (Geusebroek et al., 2005), and the Thyroid, Cancer and Satellite datasets were obtained from the UCI Machine Learning Repository (Lichman, 2013). We used the transformations of these datasets by (Goldstein and Uchida, 2016) [3]. We used 50% of the dataset for training, 10% for validation, and the remaining 40% for test. The total number of data points and the dimensions of the data points of the five datasets are listed in Table 5.1.

[3]These datasets are available at https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/OPQMVF

Dataset	Gaussian	MAP($b = 1$)	MAP($b = 0.001$)	Student-t
SMTP	-1.248 ± 0.404	-4.864 ± 0.020	-1.932 ± 0.404	**0.827 ± 0.105**
Aloi	45.418 ± 5.457	-38.210 ± 0.156	30.406 ± 0.383	**77.022 ± 0.539**
Thyroid	15.519 ± 4.422	-31.266 ± 0.159	18.037 ± 1.318	**69.543 ± 0.634**
Cancer	**-18.668 ± 3.448**	-45.895 ± 0.843	**-19.017 ± 3.273**	**-18.253 ± 2.629**
Satellite	**-1.852 ± 0.370**	-50.895 ± 0.238	**-1.899 ± 0.372**	**-1.811 ± 0.289**

Table 3.2: Comparison of test log-likelihoods. We highlighted the best result in bold, and we also highlighted the results in bold which are not statistically different from the best result according to a pair-wise t-test. We use 5% as p-value.

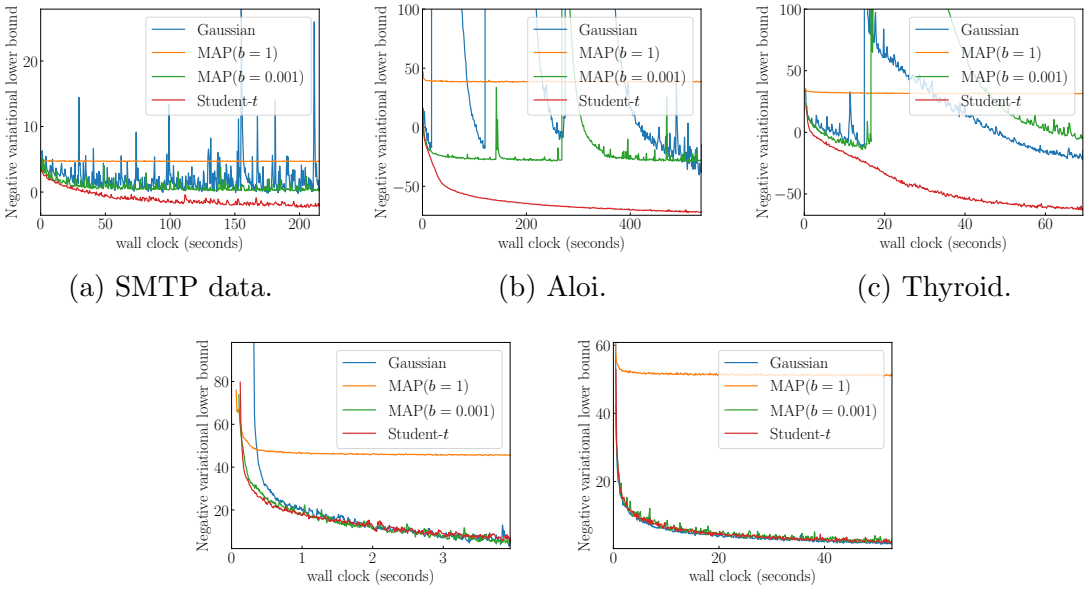
3.5.2 Setup

We used two-layer neural networks (500 hidden units per layer) as the encoder and the decoder, and a hyperbolic tangent as the activation function. We trained the VAE by using Adam (Kingma and Ba, 2015) with mini-batch size of 100. We set the sample size of the reparameterization trick to $L = 1$. The maximum number of epochs was 500, and we used early-stopping (Goodfellow et al., 2016) based on the validation data. We used a two-dimensional latent variable vector $\mathbf{z}$ with the SMTP data, and a 20-dimensional latent variable vector for the other datasets. For the evaluation, we calculated the log-likelihood of the test data by using the importance sampling (Burda et al., 2016). We set the sample size of the importance sampling to 100. We ran all experiments ten times each. The data was standardized with zero mean and unit variance. We used the following setup: CPU was Intel Xeon E5-2640 v4 2.40GHz, the memory size was 1 TB, and GPU was NVIDIA Tesla M40.

3.5.3 Results

Figures 3.5(a)–3.5(e) show the relationship between the mean training loss and wall clock (seconds), and Table 4.2 compares the test log-likelihoods of the Gaussian VAE, MAP VAE, and Student-t VAE.

First, we focused on the Gaussian VAE. With the SMTP, Aloi and Thyroid data, training of the Gaussian VAE was unstable, and the test log-likelihoods of the Gaussian VAE were worse than those of the Student-t VAE. On the other hand, with the Cancer and Satellite data, the training of the Gaussian VAE was stable, and the Gaussian VAE and the Student-t VAE achieved the comparable test log-



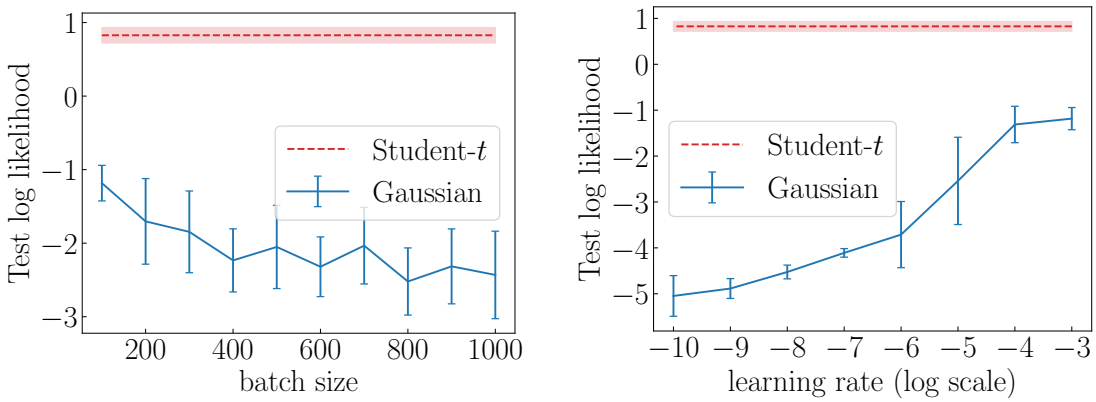
(a) SMTP data. (b) Aloi. (c) Thyroid.

(d) Cancer. (e) Satellite.

Figure 3.5: Mean training loss $(-\sum_{i=1}^{N} \hat{\mathcal{L}}(\mathbf{x}^{(i)}; \theta, \phi)/N)$ for each data set.

likelihoods. Figures 3.6(a) and 3.6(b) show the test log-likelihoods by the Gaussian VAE for different mini-batch sizes, and different learning rates for the SMTP data, respectively. Even though the mini-batch size and learning rate changed, the test log-likelihood with the Gaussian VAE was worse than that with the Student-t VAE. These results indicate that improving the robustness of the objective function is better than tuning the hyper parameters of the optimizer.

Second, we focused on the MAP VAE. When the regularization parameter was large, $b = 1$, the negative variational lower bound was stable but it did not decrease well. When the regularization parameter was small, $b = 0.001$ with the SMTP, Aloi and Thyroid data, the negative variational lower bound became unstable, and it behaved similarly to that of the Gaussian VAE. The test log-likelihoods of the MAP VAE were equal to or worse than the Gaussian VAE, because the constant parameter of the Gamma prior b was not flexible. Figure 3.7 shows the test log-likelihoods with different regularization parameter values b for the SMTP data. When $b \leq 0.001$, the test log-likelihood of the MAP VAE was not statistically different from the Gaussian VAE, and when $b \geq 0.01$, the larger b was, the worse the test log-likelihood of the MAP VAE became. These results indicate that setting an appropriate robustness



(a) different mini-batch sizes.

(b) different learning rates.

Figure 3.6: Test log-likelihoods by the Gaussian VAE for different mini-batch sizes and learning rates for the SMTP data. We plotted the test log-likelihoods of the Student-t VAE in Table 4.2 by dashed line for comparison. The semi-transparent area and error bar represent standard deviations.

for each data point is effective to avoid zero-variance problem without lacking the flexibility of density estimation.

Third, we focused on the Student-t VAE. With all of the datasets, the Student-t VAE reduced the training loss stably, and obtained the equal to or better density estimation performance than that of the Gaussian VAE and the MAP VAE. Since the Student-t VAE can set the appropriate robustness for each data point by estimating $\nu_\theta(\mathbf{z})$, it avoids zero-variance problem while improving the flexibility of the decoder, which makes the training stable and leads to obtaining good density estimation performance.

In addition, we evaluated the stability of training under the noisy environment. We added noisy samples which follows uniform distribution to SMTP data, and evaluated the test log-likelihoods. Figure 3.8 shows the relationship between the test log-likelihoods and the ratio of noisy samples to dataset. Whereas the training of Gaussian VAE becomes unstable as the ratio of noisy samples increases, the training of Student-t VAE continues to be stable. This result indicates that the Student-t VAE is useful under the noisy environments such as real-world applications.

These results indicate that the Student-t VAE is a good alternative to the Gaussian VAE: the training of the Student-t VAE is stable, which requires only a lightweight computational cost, and the density estimation performance of the Student-t VAE

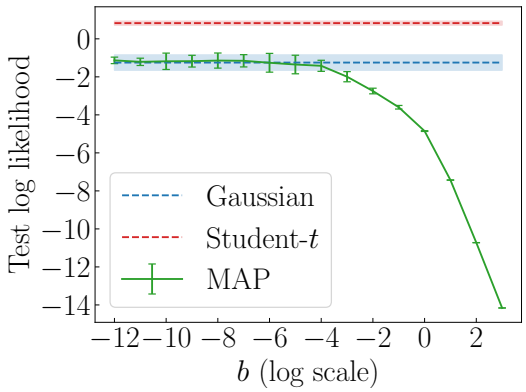


Figure 3.7: Relationship between the test log-likelihoods and regularization parameter b with the MAP VAE for SMTP data. We plotted the test log-likelihoods of the Gaussian VAE and Student-t VAE in Table 4.2 by dashed line for comparison. The semi-transparent area and error bar represent standard deviations.

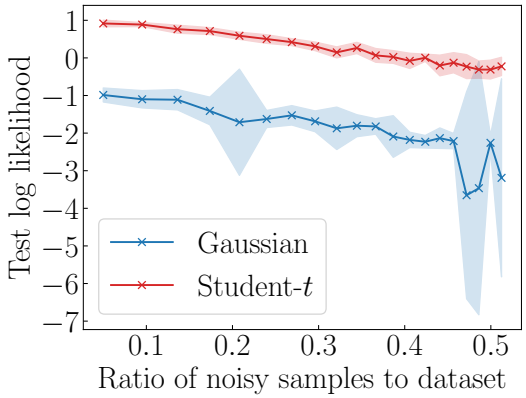


Figure 3.8: Comparison of the test log-likelihoods of the Gaussian VAE and those of Student-t VAE under noisy environment. The semi-transparent area and error bar represent standard deviations.

is equal to or better than that of the Gaussian VAE.

3.6 Related Work

Regarding related work on improving the stability of training the VAE, a number of methods have been proposed that reduce the variance of stochastic gradients (Johnson and Zhang, 2013; Wang et al., 2013; Kingma et al., 2015; Roeder et al., 2017; Miller et al., 2017). These methods focused on the stability of optimization methods, such as the reparameterization trick and the stochasticity due to data subsampling. The Student-t VAE focuses on the different part from these methods:

the robustness of the training objective function. This requires only a lightweight computational cost, and can be used together with these existing methods.

It is well-known that the Student-t distribution has robustness (Lange et al., 1989), and this robustness has been applied to a number of machine learning algorithms, such as stochastic neighbor embedding (Maaten and Hinton, 2008), Gaussian process (Jylänki et al., 2011), and Bayesian optimization (Martinez-Cantin et al., 2017). These algorithms used a Student-t distribution to reduce the influence of noise included in the observed data. The Student-t VAE uses a Student-t distribution for reducing the influence of the error between the data point and its decoded mean, which makes the training objective function of Student-t VAE robust.

3.7 Conclusion

We proposed the Student-t variational autoencoder: a robust multivariate density estimator based on the variational autoencoder (VAE). The training of the Gaussian VAE often becomes unstable. We investigated the cause of this instability, and revealed that the Gaussian decoder is sensitive to the error between the data point and its decoded mean when the decoded variance is almost zero.

In order to improve the robustness of the VAE, we introduced a Bayesian approach to the Gaussian decoder: we set a Gamma prior for the inverse of the decoded variance and marginalized it out analytically, which led to using the Student-t distribution as the decoder. Since the Student-t distribution is a heavy-tailed distribution, the Student-t decoder is robust to the error, which makes the training stable. We demonstrated that the robustness of the training and the high density estimation performance of the Student-t VAE in experiments using five datasets.

In the future, we will try to apply the Student-t VAE to real-world applications such as anomaly detection (Suh et al., 2016) and image generation (van den Oord et al., 2016).

Chapter 4

Variational Autoencoder with Implicit Optimal Priors

4.1 Introduction

In this chapter, we focus on the over-regularization by prior, described in Section 1.2. Although the standard Gaussian distribution is usually used for the prior, this simple prior incurs over-regularization. The aggregated posterior prior, the expectation of the encoder over the data distribution, has been introduced as the optimal prior in terms of the maximizing the ELBO (Hoffman and Johnson, 2016). Since the KL divergence with the aggregated posterior cannot be calculated in a closed form, the previous work (Tomczak and Welling, 2018) models this optimal prior by using the finite mixture of encoders, called the VampPrior. Nevertheless, it has sensitive hyperparameters that are difficult to tune. we review these priors in section 4.2,

In section 4.3, we directly approximate the KL divergence with the aggregated posterior. This enables us to avoid the difficult hyperparameter tuning for modeling the aggregated posterior. We can simply use the density ratio trick to approximation this KL divergence. However, since this KL divergence is high-dimensional density ratio, the density ratio trick does not work well. To avoid density ratio estimation in high dimension, we rewrite this KL divergence into the sum of following two terms:

- the KL divergence between the encoder and the standard Gaussian, and
- the density ratio between the aggregated posterior and the standard Gaussian.

The first term can be calculated in a closed form, and we can accurately appropriate second term since this is low-dimensional density ratio.

We discuss the relationship between our approach and existing approaches in section 4.4, and evaluate our approach on various datasets in section 4.5. Finally, we summarize our contributions in section 4.6.

4.2 Preliminaries

4.2.1 Variational Autoencoder

First, we again review the VAE (Kingma and Welling, 2014; Rezende et al., 2014), focusing on the prior $p_\lambda(\mathbf{z})$. The VAE estimates the data point $\mathbf{x}$ by using a low-dimensional latent variable vector $\mathbf{z}$ as follows:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z}, \tag{4.1}$$

where $p_\theta(\mathbf{x}|\mathbf{z})$ is the decoder.

The ELBO for each data point $\mathbf{x}$ can be written as

$$\mathcal{L}(\mathbf{x};\theta,\phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p_\theta(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p_\lambda(\mathbf{z})), \tag{4.2}$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ is the encoder. The expectation term in Eq. (4.2) is known as the negative reconstruction error, which we also call the the reconstruction term in this chapter.

The parameters of the encoder θ and decoder ϕ are optimized by maximizing the following ELBO:

$$\max_{\theta,\phi} \int p_{\mathcal{D}}(\mathbf{x})\mathcal{L}(\mathbf{x};\theta,\phi)\mathrm{d}\mathbf{x}, \tag{4.3}$$

where $p_{\mathcal{D}}(\mathbf{x})$ is the data distribution.

4.2.2 Aggregated Posterior

The training of VAE is maximizing the reconstruction term with regularization by KL divergence between the encoder and the prior $p_\lambda(\mathbf{z})$. $p_\lambda(\mathbf{z})$ is usually modeled by

a standard Gaussian distribution $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ (Kingma and Welling, 2014). However, this is not an optimal prior for the VAE. This simple prior incurs over-regularization, which is one of the causes of the poor performance (Hoffman and Johnson, 2016). This phenomenon is called the posterior-collapse (van den Oord et al., 2017).

The optimal prior that maximizes the objective function of VAE (Eq. (4.3)) can be derived analytically. The maximization of Eq. (4.3) with respect to the prior $p_\lambda(\mathbf{z})$ is written as follows:

$$\begin{aligned}
\arg\max_{p_\lambda(\mathbf{z})} \int p_{\mathcal{D}}(\mathbf{x})\mathcal{L}(\mathbf{x};\theta,\phi)\mathrm{d}\mathbf{x} &= \arg\max_{p_\lambda(\mathbf{z})} \int p_{\mathcal{D}}(\mathbf{x})\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p_\lambda(\mathbf{z})\right]\mathrm{d}\mathbf{x} \\
&= \arg\max_{p_\lambda(\mathbf{z})} \int \left\{ \int p_{\mathcal{D}}(\mathbf{x}) q_\phi(\mathbf{z}|\mathbf{x})\mathrm{d}\mathbf{x} \right\} \ln p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} \\
&= \arg\max_{p_\lambda(\mathbf{z})} -H\Big(\int p_{\mathcal{D}}(\mathbf{x}) q_\phi(\mathbf{z}|\mathbf{x})\mathrm{d}\mathbf{x}, p_\lambda(\mathbf{z}) \Big), \qquad (4.4)
\end{aligned}$$

where $-H(P, Q)$ is the negative cross entropy between P and Q. Since $-H(P, Q)$ takes a maximum value when P is equal to Q, the optimal prior $p^*_\lambda(\mathbf{z})$ that maximizes Eq. (4.3) is

$$p^*_\lambda(\mathbf{z}) = \int p_{\mathcal{D}}(\mathbf{x}) q_\phi(\mathbf{z}|\mathbf{x})\mathrm{d}\mathbf{x} \equiv q_\phi(\mathbf{z}). \qquad (4.5)$$

This distribution $q_\phi(\mathbf{z})$ is called the aggregated posterior [1].

When we use the standard Gaussian prior $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$, the KL divergence $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$ can be calculated in a closed form (Kingma and Welling, 2014). However, when we use the aggregated posterior $q_\phi(\mathbf{z})$ as the prior, the KL divergence

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|q_\phi(\mathbf{z})) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z})}\right] \qquad (4.6)$$

cannot be calculated in a closed form, which prevents us from using the aggregated posterior as the prior.

[1] Note that the aggregated *posterior* is NOT the product of the prior and the likelihood, which is the way the word *posterior* is usually used.

4.2.3 Previous work: VampPrior

In previous work, the aggregated posterior is modeled by using the finite mixture of encoders to calculate the KL divergence. Given a dataset $\mathbf{X} = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\right\}$, the aggregated posterior can be simply modeled by an empirical distribution:

$$q_\phi(\mathbf{z}) \simeq \frac{1}{N} \sum_{i=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}^{(i)}). \tag{4.7}$$

Nevertheless, this empirical distribution incurs over-fitting (Tomczak and Welling, 2018). Thus, the VampPrior (Tomczak and Welling, 2018) models the aggregated posterior by

$$q_\phi(\mathbf{z}) \simeq \frac{1}{K} \sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}^{(k)}), \tag{4.8}$$

where K is the number of mixtures, and $\mathbf{u}^{(k)}$ is the same dimensional vector as a data point. $\mathbf{u}$ is regarded as the pseudo input for the encoder, and is optimized during the training of the VAE through the stochastic gradient descent (SGD). If $K \ll N$, the VampPrior can avoid over-fitting (Tomczak and Welling, 2018). The KL divergence with the VampPrior can be calculated by the Monte Carlo approximation. The VAE with the VampPrior achieves better density estimation performance than the VAE with the standard Gaussian prior and the VAE with the Gaussian mixture prior (Dilokthanakul et al., 2016). However, this approach has a major drawback: it has sensitive hyperparameters such as the number of mixtures K, which are difficult to tune.

From the above discussion, the aggregated posterior seems to be difficult to model explicitly. In the next section, instead, we estimate the KL divergence with the aggregated posterior without modeling the aggregated posterior explicitly.

4.3 Proposed Method

In this section, we propose the approximation method of the KL divergence with the aggregated posterior, and describe the optimization procedure of our approach.

4.3.1 Estimating the KL Divergence

As shown in Eq. (4.6), the KL divergence with the aggregated posterior is the expectation of the logarithm of the density ratio $q_\phi(\mathbf{z}|\mathbf{x})/q_\phi(\mathbf{z})$. In this section, we introduce the density ratio trick (Sugiyama et al., 2012; Goodfellow et al., 2014), which can estimate the ratio of two distributions without modeling each distribution explicitly. Hence, there is no need to model the aggregated posterior explicitly. By using the density ratio trick, $q_\phi(\mathbf{z}|\mathbf{x})/q_\phi(\mathbf{z})$ can be estimated by using a probabilistic binary classifier $D(\mathbf{x}, \mathbf{z})$.

However, the density ratio trick has a serious drawback: it is experimentally shown to work poorly in high dimensions (Sugiyama et al., 2012; Rosca et al., 2018). Unfortunately, if $\mathbf{x}$ is high-dimensional, $q_\phi(\mathbf{z}|\mathbf{x})/q_\phi(\mathbf{z})$ also becomes a high-dimensional density ratio. The reason is as follows. Since the $q_\phi(\mathbf{z}|\mathbf{x})$ is a conditional distribution of $\mathbf{z}$ given $\mathbf{x}$, the density ratio trick has to use a probabilistic binary classifier $D(\mathbf{x}, \mathbf{z})$, which takes $\mathbf{x}$ and $\mathbf{z}$ jointly as an input. In fact, $D(\mathbf{x}, \mathbf{z})$ estimates the density ratio of joint distributions of $\mathbf{x}$ and $\mathbf{z}$, which is a high-dimensional density ratio with high-dimensional $\mathbf{x}$ (Mescheder et al., 2017).

To avoid the density ratio estimation in high dimensions, we rewrite the KL divergence $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|q_\phi(\mathbf{z}))$ as follows:

$$\begin{aligned} D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|q_\phi(\mathbf{z})) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z})}\right] \\ &= \int q_\phi(\mathbf{z}|\mathbf{x}) \ln \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} \mathrm{d}\mathbf{z} + \int q_\phi(\mathbf{z}|\mathbf{x}) \ln \frac{p(\mathbf{z})}{q_\phi(\mathbf{z})} \mathrm{d}\mathbf{z} \\ &= D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln \frac{q_\phi(\mathbf{z})}{p(\mathbf{z})}\right]. \end{aligned} \tag{4.9}$$

The first term in Eq. (4.9) is KL divergence between the encoder and standard Gaussian distribution, which can be calculated in a closed form. The second term is the expectation of the logarithm of the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$. We estimate $q_\phi(\mathbf{z})/p(\mathbf{z})$ with the density ratio trick. Since the latent variable vector $\mathbf{z}$ is low-dimensional, the density ratio trick works well.

We can estimate the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$ as follows. First, we prepare the samples from $q_\phi(\mathbf{z})$ and samples from $p(\mathbf{z})$. We can sample from $p(\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x})$ since these distributions are a Gaussian, and we can also sample from the aggregated

posterior $q_\phi(\mathbf{z})$ by using ancestral sampling: we choose a data point $\mathbf{x}$ from a dataset randomly and sample $\mathbf{z}$ from the encoder given this data point $\mathbf{x}$. Second, we label $y = 1$ to samples from $q_\phi(\mathbf{z})$ and $y = 0$ to samples from $p(\mathbf{z})$. Then, we define $p^*(\mathbf{z}|y)$ as follows:

$$p^*(\mathbf{z}|y) \equiv \begin{cases} q_\phi(\mathbf{z}) & (y = 1) \\ p(\mathbf{z}) & (y = 0) \end{cases}. \tag{4.10}$$

Third, we introduce a probabilistic binary classifier $D(\mathbf{z})$ that discriminates between the samples from $q_\phi(\mathbf{z})$ and samples from $p(\mathbf{z})$. If $D(\mathbf{z})$ can discriminate these samples perfectly, we can rewrite the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$ by using Bayes theorem and $D(\mathbf{z})$ as follows:

$$\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} = \frac{p^*(\mathbf{z}|y=1)}{p^*(\mathbf{z}|y=0)} = \frac{p^*(y=0)p^*(y=1|\mathbf{z})}{p^*(y=1)p^*(y=0|\mathbf{z})} = \frac{p^*(y=1|\mathbf{z})}{p^*(y=0|\mathbf{z})} \equiv \frac{D(\mathbf{z})}{1-D(\mathbf{z})}, \tag{4.11}$$

where $p^*(y = 0)$ equals $p^*(y = 1)$ since the number of samples is the same. We model $D(\mathbf{z})$ by $\sigma(T_\psi(\mathbf{z}))$, where $T_\psi(\mathbf{z})$ is a neural network with parameter ψ and input $\mathbf{z}$, and $\sigma(\cdot)$ is a sigmoid function. We train $T_\psi(\mathbf{z})$ to maximize the following objective function:

$$T^*(\mathbf{z}) = \max_\psi \mathbb{E}_{q_\phi(\mathbf{z})}\left[\ln(\sigma(T_\psi(\mathbf{z})))\right] + \mathbb{E}_{p(\mathbf{z})}\left[\ln(1-\sigma(T_\psi(\mathbf{z})))\right]. \tag{4.12}$$

By using $T^*(\mathbf{z})$, we can estimate the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$ as follows:

$$\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} = \frac{\sigma(T^*(\mathbf{z}))}{1-\sigma(T^*(\mathbf{z}))} \Leftrightarrow T^*(\mathbf{z}) = \ln\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})}. \tag{4.13}$$

Therefore, we can estimate the KL divergence between the encoder and aggregated posterior $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|q_\phi(\mathbf{z}))$ by

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|q_\phi(\mathbf{z})) = D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[T^*(\mathbf{z})\right]. \tag{4.14}$$

4.3.2 Optimization Procedure

From the above discussion, we obtain the training objective function of the VAE with our implicit optimal prior:

$$\max_{\theta,\phi} \int p_{\mathcal{D}}(\mathbf{x}) \left\{ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[\ln p_\theta(\mathbf{x}|\mathbf{z}) + T_\psi(\mathbf{z}) \right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})) \right\} \mathrm{d}\mathbf{x}, \tag{4.15}$$

where $T_\psi(\mathbf{z})$ maximizes the Eq. (4.12). Given a dataset $\mathbf{X} = \left\{ \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)} \right\}$, we optimize the Monte Carlo approximation of this objective:

$$\max_{\theta,\phi} \frac{1}{N} \sum_{i=1}^{N} \left\{ \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})} \left[\ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}) + T_\psi(\mathbf{z}) \right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) \| p(\mathbf{z})) \right\}, \tag{4.16}$$

and we approximate the expectation term by the reparameterization trick (Kingma and Welling, 2014):

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})} \left[\ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}) + T_\psi(\mathbf{z}) \right] \simeq \frac{1}{L} \sum_{\ell=1}^{L} \left\{ \ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}) + T_\psi(\mathbf{z}^{(i,\ell)}) \right\}, \tag{4.17}$$

where $\mathbf{z}^{(i,\ell)} = \mu_\phi(\mathbf{x}^{(i)}) + \varepsilon^{(i,\ell)} \odot \sigma_\phi(\mathbf{x}^{(i)})$, $\varepsilon^{(i,\ell)}$ is a sample drawn from $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$, $\odot$ is the element-wise product, and L is the sample size of the reparameterization trick. Then, the resulting objective function is

$$\max_{\theta,\phi} \frac{1}{N} \sum_{i=1}^{N} \left[\frac{1}{L} \sum_{\ell=1}^{L} \left\{ \ln p_\theta(\mathbf{x}^{(i)}|\mathbf{z}^{(i,\ell)}) + T_\psi(\mathbf{z}^{(i,\ell)}) \right\} - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) \| p(\mathbf{z})) \right]. \tag{4.18}$$

We optimize this model with stochastic gradient descent (SGD) (Duchi et al., 2011; Zeiler, 2012; Tieleman and Hinton, 2012; Kingma and Ba, 2015) by iterating a two-step procedure: we first update θ and ϕ to maximize Eq. (4.18) with fixed ψ and next update ψ to maximize the Monte Carlo approximation of Eq. (4.12) with fixed θ and ϕ, as follows:

$$\max_{\psi} \frac{1}{M} \sum_{i=1}^{M} \ln(\sigma(T_\psi(\mathbf{z}_1^{(i)}))) + \frac{1}{M} \sum_{j=1}^{M} \ln(1 - \sigma(T_\psi(\mathbf{z}_0^{(j)}))), \tag{4.19}$$

where $\mathbf{z}_1^{(i)}$ is a sample drawn from $q_\phi(\mathbf{z})$, $\mathbf{z}_0^{(j)}$ is a sample drawn from $p(\mathbf{z})$, and M

is the sampling size of Monte Carlo approximation. Note that we need to compute the gradient of $T_\psi(\mathbf{z})$ with respect to ϕ in the optimization of Eq. (4.18) since $T_\psi(\mathbf{z})$ models $\ln q_\phi(\mathbf{z})/p(\mathbf{z})$. However, when $T_\psi(\mathbf{z})$ equals $T^*(\mathbf{z})$, the expectation of this gradient becomes zero, as follows:

$$\begin{aligned}
\mathbb{E}_{p_\mathcal{D}(\mathbf{x})q_\phi(\mathbf{z}|\mathbf{x})}\left[\nabla_\phi T^*(\mathbf{z})\right] &= \mathbb{E}_{q_\phi(\mathbf{z})}\left[\nabla_\phi \ln q_\phi(\mathbf{z})\right] \\
&= \int q_\phi(\mathbf{z}) \frac{\nabla_\phi q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} \mathrm{d}\mathbf{z} \\
&= \nabla_\phi \int q_\phi(\mathbf{z}) \mathrm{d}\mathbf{z} \\
&= \nabla_\phi 1 \\
&= 0. \qquad (4.20)
\end{aligned}$$

Therefore, we ignore this gradient in the optimization [2]. We also note that $T_\psi(\mathbf{z})$ is likely to overfit to the log density ratio between the empirical aggregated posterior (Eq. (4.7)) and the standard Gaussian distribution. As mentioned in Section 4.2.3, this over-fitting also incurs over-fitting of the VAE (Tomczak and Welling, 2018). Therefore, we use the regularization techniques such as dropout (Srivastava et al., 2014) for $T_\psi(\mathbf{z})$, which prevents it from over-fitting. We train ψ more than θ and ϕ: if we update θ and ϕ for J_1 steps, we update ψ for J_2 steps, where J_2 is larger than J_1. Algorithm 5.1 shows the pseudo code of the optimization procedure of this model, where K is the minibatch size of SGD.

4.4 Related Work

For improving the performance of the VAE, numerous works have focused on the regularization effect of the KL divergence between the encoder and the prior. These works improve either the encoder or the prior.

First, we focus on the works about the prior. Although the optimal prior for the VAE is the aggregated posterior, the KL divergence with the aggregated posterior cannot be calculated in a closed form. As described in Section 4.2.3, the VampPrior (Tomczak and Welling, 2018) has been presented to solve this problem. However, it has sensitive hyperparameters such as the number of mixtures K. Since the Vamp-

[2] There is almost the same discussion in (Mescheder et al., 2017).

Algorithm 4.1 VAE with Implicit Optimal Priors

while not converged **do**
  for J_1 steps **do**
    Sample minibatch $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(K)}\right\}$ from $\mathbf{X}$
    Compute the gradients of Eq. (4.18) w.r.t. θ and ϕ
    Update θ and ϕ with their gradients
  for J_2 steps **do**
    Sample minibatch $\left\{\mathbf{z}_0^{(1)}, \ldots, \mathbf{z}_0^{(K)}\right\}$ from $p(\mathbf{z})$
    Sample minibatch $\left\{\mathbf{z}_1^{(1)}, \ldots, \mathbf{z}_1^{(K)}\right\}$ from $q_\phi(\mathbf{z})$
    Compute the gradient of Eq. (4.19) w.r.t. ψ
    Update ψ with its gradient

Prior requires a heavy computational cost, these hyperparameters are difficult to tune. In contrast to this, our approach can estimate the KL divergence more easily and robustly than the VampPrior since it does not need to model the aggregated posterior explicitly. In addition, since the computational cost of our approach is much more lightweight than that of VampPrior, the hyperparameters of our approach are easier to tune than those of VampPrior.

There are approaches on improving the prior other than the aggregated posterior. For example, non-parametric Bayesian distribution (Nalisnick and Smyth, 2017) and hyperspherical distribution (Davidson et al., 2018) are used for the prior. These approaches aim to obtain the useful and interpretable latent representation rather than improving the density estimation performance, which is opposite to our purpose. We should mention the disadvantage of our approach compared with these approaches. Since our prior is implicit, we cannot sample from our prior directly. Instead, we can sample from the aggregated posterior, which our implicit prior models, by using ancestral sampling. That is, when we sample from the prior, we need to prepare a data point.

Next, we focus on the works about the encoder. To improve the performance of the VAE, these works increase the flexibility of the encoder. The normalizing flow (Rezende and Mohamed, 2015; Kingma et al., 2016; Huang et al., 2018) is one of the main approaches, which applies a sequence of invertible transformations to the latent variable vector until a desired level of flexibility is attained. Our approach is

orthogonal to the normalizing flow and can be used together with it.

The similar approaches to ours are the adversarial variational Bayes (AVB) (Mescheder et al., 2017) and the adversarial autoencoders (AAE) (Makhzani et al., 2015; Tolstikhin et al., 2017). These approaches use the implicit encoder network, which takes as input a data point $\mathbf{x}$ and Gaussian random noise and produces a latent variable vector $\mathbf{z}$. Since the implicit encoder does not assume the distribution type, it can become a very flexible distribution. In these approaches, the standard Gaussian distribution is used for the prior. Although the KL divergence between the implicit encoder and the standard Gaussian prior $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$ cannot be calculated in a closed form, the AVB estimates this KL divergence by using the density ratio trick. However, this estimation does not work well with high-dimensional datasets since this KL divergence also becomes a high-dimensional density ratio (Rosca et al., 2018). Our approach can avoid this problem since we use the density ratio trick in a low dimension. The AAE is an expansion of the Autoencoder rather than the VAE. The AAE regularizes the aggregated posterior to be close to the standard Gaussian prior by minimizing the KL divergence $D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$. The AAE also uses the density ratio trick to estimate this KL divergence, and this works well since this KL divergence is a low-dimensional density ratio. However, the AAE cannot estimate the probability of a data point. Our approach is based on the VAE, and can estimate the probability of a data point.

4.5 Experiments

In this section, we experimentally evaluate the performance of our approach.

4.5.1 Data

We used five datasets: OneHot (Mescheder et al., 2017), MNIST (Salakhutdinov and Murray, 2008), OMNIGLOT (Burda et al., 2016), FreyFaces[3], and Histopathology (Tomczak and Welling, 2016). OneHot consists of only four-dimensional one hot vectors: $(1,0,0,0)^{\mathrm{T}}$, $(0,1,0,0)^{\mathrm{T}}$, $(0,0,1,0)^{\mathrm{T}}$, and $(0,0,0,1)^{\mathrm{T}}$. This simple dataset is useful for observing the posterior of the latent variable, which is used in (Mescheder et al., 2017). MNIST and OMNIGLOT are binary image datasets, and FreyFaces

[3]This dataset is available at `https://cs.nyu.edu/~roweis/data/frey_rawface.mat`

Table 4.1: Number and dimensions of datasets

	Dimension	Train size	Valid size	Test size
OneHot	4	1,000	100	1,000
MNIST	784	50,000	10,000	10,000
OMNIGLOT	784	23,000	1,345	8,070
FreyFaces	560	1,565	200	200
Histopathology	784	6,800	2,000	2,000

and Histopathology are grayscale image datasets. These image datasets are useful for measuring the density estimation performance, which are used in (Tomczak and Welling, 2018). The number and the dimensions of data points of the five datasets are listed in Table 5.1.

4.5.2 Setup

We compared our implicit optimal prior with standard Gaussian prior and VampPrior. We set the dimensions of the latent variable vector to 2 for OneHot, and 40 for other datasets. We used two-layer neural networks (500 hidden units per layer) for the encoder, the decoder, and the density ratio estimator. We used the gating mechanism (Dauphin et al., 2016) for the encoder and the decoder and used a hyperbolic tangent as the activation function for the density ratio estimator. We initialized the weights of these neural networks in accordance with the method in (Glorot and Bengio, 2010). We used a Gaussian distribution as the encoder. As the decoder, we used a Bernoulli distribution for OneHot, MNIST, and OMNIGLOT and used a Gaussian distribution for FreyFaces and Histopathology, means of which were constrained to the interval $[0, 1]$ by using a sigmoid function. We trained all methods by using Adam (Kingma and Ba, 2015) with a mini-batch size of 100 and learning rate in $[10^{-4}, 10^{-3}]$. We set the maximum number of epochs to 1,000 and used early-stopping (Goodfellow et al., 2016) on the basis of validation data. We set the sample size of the reparameterization trick to $L = 1$. In addition, we used warm-up (Bowman et al., 2015) for the first 100 epochs of Adam. For MNIST and OMNIGLOT, we used dynamic binarization (Salakhutdinov and Murray, 2008) during the training of VAE to avoid over-fitting. For image datasets, we calculated the

log-likelihood of the test data by using the importance sampling (Burda et al., 2016). We set the sample size of the importance sampling to 10. We ran all experiments eight times each.

With VampPrior, we set the number of mixtures K to 50 for OneHot, 500 for MNIST, FreyFaces, and Histopathology, and 1,000 for OMNIGLOT. In addition, for image datasets, we used a clipped relu function that equals $\min(\max(x, 0), 1)$ to scale the pseudo inputs in $[0, 1]$ since the range of data points of these datasets is $[0, 1]$ [4].

With our approach, we used dropout (Srivastava et al., 2014) in the training of the density ratio estimator since it is likely to over-fit. We set the keep probability of dropout to 50%. We updated the parameter of the density ratio estimator: ψ for 10 epochs during the updating of the parameters of VAE: θ and ϕ for one epoch. We set the sampling size of Monte Carlo approximation in Eq. (4.19) to $M = N$.

In addition, we compared our approach with adversarial variational Bayes (AVB) on OneHot. We set the dimension of the Gaussian random noise input of AVB to 10, and other settings are almost the same as those for our approach.

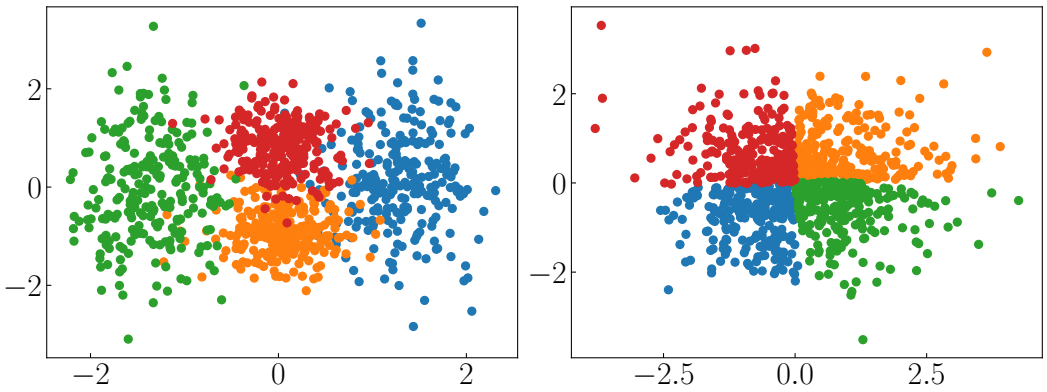
4.5.3 Results

Figures 4.1(a)–4.1(d) show the posteriors of latent variable of each approach on OneHot, and Figures 4.2(a)–4.2(d) show the evidence lower bound of each approach on OneHot.

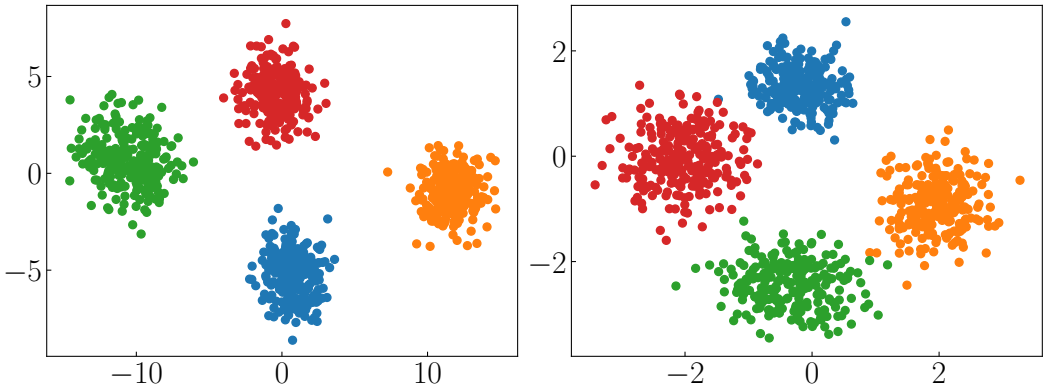
These results show the difference between these approaches. We can see that the evidence lower bound (ELBO) of the standard VAE (VAE with standard Gaussian prior) on OneHot was worse than the optimal log-likelihood on this dataset: $-\ln(4) \approx -1.386$. The over-regularization incurred by the standard Gaussian prior can be given as a reason. The posteriors were overlapped, and it became difficult to discriminate between samples from these posteriors. Hence, the decoder became confused when reconstructing. This caused the poor density estimation performance.

On the other hand, the ELBOs of AVB, VAE with VampPrior, and our approach are much closer to the optimal log-likelihood than the standard VAE. We note that the ELBOs of the AVB and our approach are the estimated values, and that these approaches may overestimate the ELBO on OneHot since the training data and validation data of OneHot are the same. First, we focus on the AVB. Although there

[4] We referred to https://github.com/jmtomczak/vae_vampprior



(a) Standard VAE. (b) AVB.

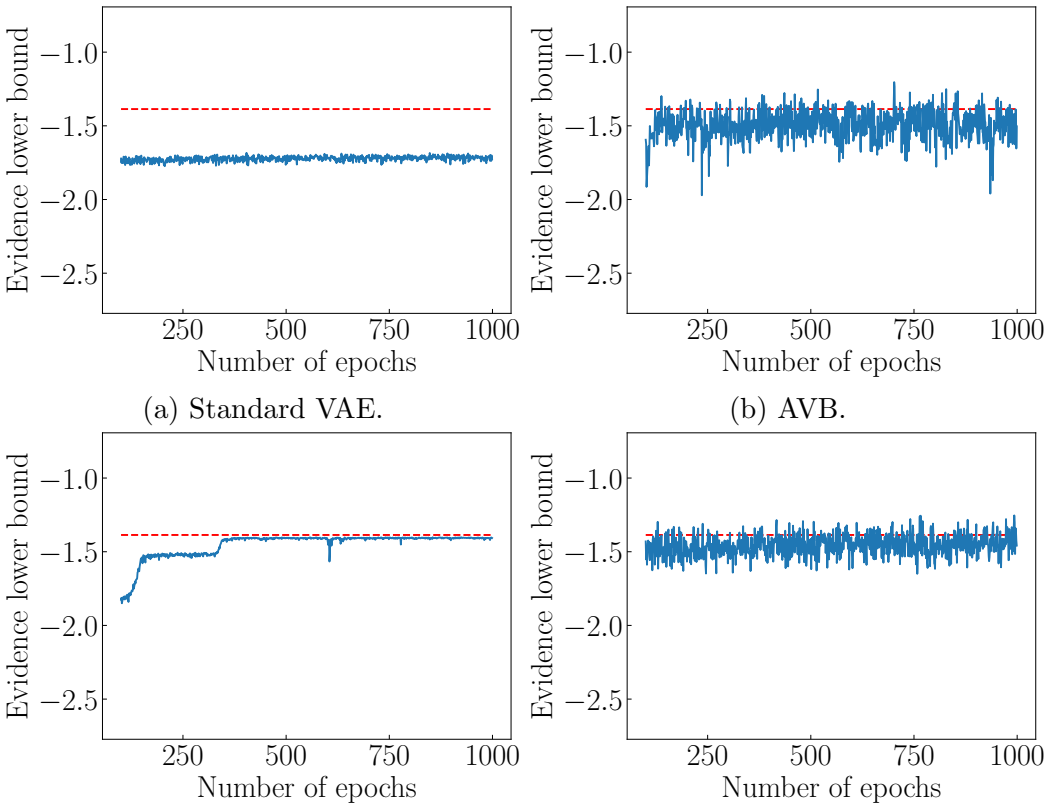


(c) VAE with VampPrior. (d) Proposed method.

Figure 4.1: Comparison of posteriors of latent variable on OneHot. We plotted samples drawn from $q_\phi(\mathbf{z}|\mathbf{x})$, where $\mathbf{x}$ is a one hot vector: $(1,0,0,0)^{\mathrm{T}}$, $(0,1,0,0)^{\mathrm{T}}$, $(0,0,1,0)^{\mathrm{T}}$, or $(0,0,0,1)^{\mathrm{T}}$. We used test data for this sampling. Samples in each color correspond to each latent representation of one hot vectors. (a) Standard VAE (VAE with standard Gaussian prior). (b) AVB. (c) VAE with VampPrior. (d) Proposed method.

is still the strong regularization by the standard Gaussian prior, the posteriors barely overlapped, and the data point was easy to reconstruct from the latent representation. The reason is that the implicit encoder network of AVB can learn complex posterior distributions. Next, we focus on the VAE with VampPrior and our approach. The VampPrior and our implicit optimal prior model the aggregated posterior that is the optimal prior for the VAE. These priors made the posteriors of these approaches different from each other, and the data point was easy to reconstruct from the latent representation.

Table 4.2 compares the test log-likelihoods on four image datasets. We used bold



(a) Standard VAE. (b) AVB.

(c) VAE with VampPrior. (d) Proposed method.

Figure 4.2: Comparison of the evidence lower bound (ELBO) with validation data on OneHot. We plotted the ELBO from 100 to 1,000 epochs since we used warm-up for the first 100 epochs. The optimal log-likelihood on this dataset is $-\ln(4) \approx -1.386$. We plotted this value by a dashed line for comparison. (a) Standard VAE (VAE with standard Gaussian prior). (b) AVB. (c) VAE with VampPrior. (d) Proposed method.

to highlight the best result and the results that are not statistically different from the best result according to a pair-wise t-test. We used 5% as the p-value. We did not compare with AVB since the estimated log marginal likelihood of AVB with high-dimensional datasets such as images is not accurate (Rosca et al., 2018).

First, we focus on the VampPrior. We can see that test log-likelihoods of VampPrior are better than those of standard VAE. However, we found two drawbacks with the VampPrior. One is that the pseudo inputs of VampPrior are difficult to optimize. For example, the pseudo inputs have an initial value dependence. Although the warm-up helps in solving this problem, it seems difficult to solve completely. The

Table 4.2: Comparison of test log-likelihoods on four image datasets.

	Standard VAE	VAE with VampPrior	Proposed method
MNIST	-85.84 ± 0.07	-83.90 ± 0.08	≈ **-83.21 ± 0.13**
OMNIGLOT	-111.39 ± 0.11	-110.53 ± 0.09	≈ **-108.48 ± 0.16**
FreyFaces	1382.53 ± 3.57	**1392.62 ± 6.25**	≈ **1396.27 ± 2.75**
Histopathology	1081.53 ± 0.70	1083.11 ± 2.10	≈ **1087.42 ± 0.60**

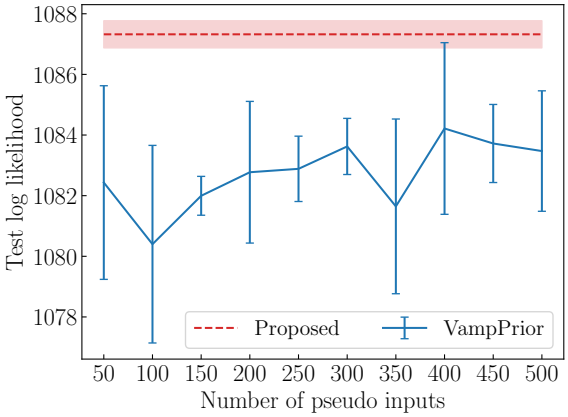


Figure 4.3: Relationship between the test log-likelihoods and number of pseudo inputs of VampPrior on Histopathology. We plotted the test log-likelihoods of our approach by a dashed line for comparison. The semi-transparent area and error bar represent standard deviations.

other is that the number of mixtures K is a sensitive hyperparameter. Figure 4.3 shows the test log-likelihoods with various K on Histopathology. The high standard deviation of the VampPrior indicates its high dependence of the pseudo input initial values. In addition, even though we choose the optimal K, the test log-likelihood of the VampPrior is worse than that of our approach.

Next, we focus on our approach. Our approach obtained the equal to or better density estimation performance than the VampPrior. Since our approach models the aggregated posterior implicitly, it can estimate the KL divergence more easily and robustly than the VampPrior. In addition, it has a much more lightweight computational cost than the VampPrior. In the training phase on MNIST, our approach was almost 2.83 times faster than the VampPrior. Therefore, although our approach has as many hyperparameters, like the neural architecture of the density ratio estimator, as the VampPrior, these hyperparameters are easier to tune than

those of the VampPrior.

These results indicate that our implicit optimal prior is a good alternative to the VampPrior: our implicit optimal prior can be optimized easily and robustly, and its density estimation performance is equal to or better than that of the VAE with the VampPrior.

4.6 Conclusion

In this chapter, we proposed the VAE with implicit optimal priors. Although the standard Gaussian distribution is usually used for the prior, this simple prior incurs over-regularization, which is one of the causes of poor performance. To solve this, the aggregated posterior has been introduced, which is optimal in terms of maximizing the training objective function of VAE. However, KL divergence between the encoder and the aggregated posterior cannot be calculated in a closed form, which prevents us from using this optimal prior. Even though explicit modeling of the aggregated posterior has been tried, this optimal prior is difficult to model explicitly.

With the proposed method, we introduced the density ratio trick for estimating this KL divergence directly. Since the density ratio trick can estimate the density ratio between two distributions without modeling each distribution explicitly, there is no need to model the aggregated posterior explicitly. Although the density ratio trick is useful, it does not work well in a high dimension. Unfortunately, the KL divergence between the encoder and the aggregated posterior is high-dimensional. Hence, we rewrite the KL divergence into the sum of two terms: the KL divergence between the encoder and the standard Gaussian distribution that can be calculated in a closed form, and the low-dimensional density ratio between the aggregated posterior and the standard Gaussian distribution, to which the density ratio trick is applied. We experimentally showed the high density estimation performance of the VAE with this implicit optimal prior.

Chapter 5

Learning Optimal Priors for Task-Invariant Representations in Variational Autoencoders

5.1 Introduction

In this chapter, we focus on the poor performance with insufficient data, described in Section 1.3. Although the VAE is powerful generative model, it cannot perform well with insufficient data points since it requires many data points for training neural networks. To overcome the lack of training data on the target task, we can apply multi-task learning (Caruana, 1997) to the VAE, where task-invariant latent variable shares task-invariant knowledge effectively. On the multiple tasks, the CVAE (Kingma et al., 2014; Sohn et al., 2015) has been widely used.

Although the CVAE is expected to reduce the dependency of the latent variable on the task, this task-dependency remains in many cases (Louizos et al., 2016). As an approach to reduce the task-dependency, the variational fair autoencoder (VFAE) (Louizos et al., 2016) was presented, which introduces the additional regularizer on the basis of maximum mean discrepancy (MMD) (Gretton et al., 2007). However, its learned representation does not work well on the target tasks. The reason is why the MMD regularizer loosens the ELBO, which prevents the VFAE from maximizing the log-likelihood sufficiently. We review the CVAE and VFAE in section 5.2.

In section 5.3, we first reveal the cause of the remaining task-dependency in the CVAE by using mutual information (Cover and Thomas, 2012) between tasks and latent variables. We show that this mutual information is bounded above by the expectation of the KL divergence between the encoder and the prior, Since this expectation term is minimized in the training, the CVAE also tries to minimize the mutual information. However, this expectation term is not a tight upper bound because the simple standard Gaussian prior loosens the upper bound. This prevents the CVAE from minimizing the mutual information sufficiently. Hence, the simple prior is one of the causes of the remaining task-dependency in the CVAE. To solve this, we propose an optimal prior for reducing the task-dependency. We also show that this optimal prior presents a better lower bound of the log-likelihood than those of the CVAE and VFAE. This enables us to obtain better representations for the improved performance on the target tasks.

Our contributions can be summarized as follows:

- We show why the simple standard Gaussian prior cannot sufficiently reduce the task-dependency in the CVAE.

- We propose a theoretical optimal prior for reducing the task-dependency.

- We show that this optimal prior presents a better lower bound of the log-likelihood than those of the CVAE and VFAE, which enables us to obtain better representations for the improved performance on the target tasks.

- We experimentally show that our approach obtains better task-invariant representations than existing approaches on various datasets, which improves the performances of various downstream applications such as density estimation and classification.

We discuss the relationship between our approach and existing approaches in section 5.4, and evaluate our approach on various datasets in section 5.5. Finally, we conclude this chapter in section 5.6.

5.2 Preliminaries

5.2.1 Conditional Variational Autoencoder

First, we again review the CVAE (Kingma et al., 2014; Sohn et al., 2015), focusing on the task-dependency. The CVAE models the conditional probability of the data point $\mathbf{x}$ given task index s using latent variable vector $\mathbf{z}$ as follows:

$$p_\theta(\mathbf{x}|s) = \int p_\theta(\mathbf{x}|\mathbf{z}, s)p(\mathbf{z})\mathrm{d}\mathbf{z}, \tag{5.1}$$

where $p_\theta(\mathbf{x}|\mathbf{z}, s)$ is the decoder, and $p(\mathbf{z})$ is the prior. $p(\mathbf{z})$ is modeled by a standard Gaussian distribution $\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ (Kingma et al., 2014; Sohn et al., 2015).

The ELBO of the CVAE for each data point $\mathbf{x}$ can be written as

$$\mathcal{L}(\mathbf{x}, s; \theta, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln p_\theta(\mathbf{x}|\mathbf{z}, s)\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|p(\mathbf{z})), \tag{5.2}$$

where $q_\phi(\mathbf{z}|\mathbf{x}, s)$ is the encoder. The expectation term in Eq. (5.2) is known as the negative reconstruction error.

The parameters θ and ϕ are optimized by maximizing the ELBO as follows:

$$\begin{aligned}\mathcal{F}_{\mathrm{CVAE}}(\theta, \phi) &= \mathbb{E}_{p_D(\mathbf{x},s)}\left[\mathcal{L}(\mathbf{x}, s; \theta, \phi)\right] &(5.3)\\ &= \mathbb{E}_{p_D(\mathbf{x},s)q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln p_\theta(\mathbf{x}|\mathbf{z}, s)\right] - \mathbb{E}_{p_D(\mathbf{x},s)}\left[D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|p(\mathbf{z}))\right], &(5.4)\end{aligned}$$

where $p_D(\mathbf{x}, s) = p_D(\mathbf{x}|s)p_D(s)$ is the joint distribution of the data point and task.

Since encoder $q_\phi(\mathbf{z}|\mathbf{x}, s)$ is regularized by task-invariant prior $p(\mathbf{z})$ through the KL divergence during the above optimization, the CVAE can reduce the dependency of $\mathbf{z}$ on s, called the task-dependency. Task-invariant $\mathbf{z}$ helps us to overcome the lack of training data points for each task since it helps in sharing the task-invariant knowledge with multiple tasks.

Although the CVAE can reduce the task-dependency to some extent, the task-dependency remains in many cases (Louizos et al., 2016). That is, the posteriors of $\mathbf{z}$ given s:

$$q_\phi(\mathbf{z}|s) = \int q_\phi(\mathbf{z}|\mathbf{x}, s)p_D(\mathbf{x}|s)\mathrm{d}\mathbf{x}, \tag{5.5}$$

differ whenever s differs.

5.2.2 Variational Fair Autoencoder

In previous work, the VFAE (Louizos et al., 2016) regularizes $q_\phi(\mathbf{z}|s)$ for each s so that they become similar by minimizing the MMD (Gretton et al., 2007), which is a distance on the space of probability measures. When s is binary, the training objective function of the VFAE is

$$\mathcal{F}_{\mathrm{VFAE}}(\theta, \phi) = \mathcal{F}_{\mathrm{CVAE}}(\theta, \phi) - \alpha \ell_{\mathrm{MMD}}(q_\phi(\mathbf{z}|s=0), q_\phi(\mathbf{z}|s=1)), \tag{5.6}$$

where $\ell_{\mathrm{MMD}}(P, Q)$ is the approximation of MMD between P and Q, and $\alpha > 0$ is a hyperparameter that is tuned according to the validation data. When there are $K \geq 3$ tasks, the VFAE minimizes the sum of the MMD between $q_\phi(\mathbf{z}|s=k)$ and its marginalized posterior $q_\phi(\mathbf{z})$ for each task k:

$$\mathcal{F}_{\mathrm{VFAE}}(\theta, \phi) = \mathcal{F}_{\mathrm{CVAE}}(\theta, \phi) - \alpha \sum_{k=1}^{K} \ell_{\mathrm{MMD}}(q_\phi(\mathbf{z}|s=k), q_\phi(\mathbf{z})), \tag{5.7}$$

where

$$q_\phi(\mathbf{z}) = \sum_{k=1}^{K} q_\phi(\mathbf{z}|s=k) p_D(s=k). \tag{5.8}$$

Although this MMD regularization can reduce the task-dependency, the learned representation does not work well on the target tasks for the following reason. Since the MMD provides a non-negative value, the following inequality holds:

$$\mathcal{F}_{\mathrm{CVAE}}(\theta, \phi) \geq \mathcal{F}_{\mathrm{VFAE}}(\theta, \phi). \tag{5.9}$$

That is, $\mathcal{F}_{\mathrm{VFAE}}(\theta, \phi)$ is the looser lower bound of the log-likelihood than $\mathcal{F}_{\mathrm{CVAE}}(\theta, \phi)$ and the VFAE cannot maximize the log-likelihood sufficiently. As a result, the VFAE does not generalize well for test data on the target tasks.

5.3 Proposed method

5.3.1 Analyzing Task-Dependency

Our aim is to obtain the task-invariant latent variable $\mathbf{z}$ without loosening the ELBO of the CVAE. To tackle this problem, we first investigate the cause of the dependency of $\mathbf{z}$ on the task s in the CVAE.

For the degree of this task-dependency, we introduce the mutual information (Cover and Thomas, 2012) between tasks and latent variables $I(S;Z)$, where S is a random variable for tasks and Z is that for latent variables. Assume that we have K tasks and the probability of the task k is given by $p_D(s=k)=\pi_k$, where $\pi_k \geq 0$ and $\sum_k \pi_k = 1$. Then, the mutual information $I(S;Z)$ can be written as follows:

$$I(S;Z) = \mathbb{E}_{q_\phi(\mathbf{z}|s)p_D(s)}\left[\ln \frac{q_\phi(\mathbf{z}|s)}{q_\phi(\mathbf{z})}\right] = \sum_{k=1}^{K} \pi_k D_{\mathrm{KL}}(q_\phi(\mathbf{z}|s=k)\|q_\phi(\mathbf{z})). \tag{5.10}$$

This mutual information measures the dependency of S and Z. $I(S;Z)$ is non-negative, providing a large value when Z depends on S and a small value when Z is almost independent of S. Note that $I(S;Z)$ gives zero if and only if Z is completely independent of S.

For this mutual information, we have the following theorem:

Theorem 5.1. $\mathcal{R}(\phi) \equiv \mathbb{E}_{p_D(\mathbf{x},s)}\left[D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x},s)\|p(\mathbf{z}))\right]$ is upper bound of $I(S;Z)$, which is minimized in $\mathcal{F}_{\mathrm{CVAE}}(\theta,\phi)$.

Proof. $\mathcal{R}(\phi)$ can be rewritten as follows:

$$\mathcal{R}(\phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)p_D(\mathbf{x},s)}\left[\ln \frac{q_\phi(\mathbf{z}|\mathbf{x},s)}{q_\phi(\mathbf{z})}\right] + D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})), \tag{5.11}$$

and the expectation term can be also rewritten as

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)p_D(\mathbf{x},s)}\left[\ln\frac{q_\phi(\mathbf{z}|\mathbf{x},s)}{q_\phi(\mathbf{z})}\right] = \sum_{k=1}^{K}\pi_k\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s=k)p_D(\mathbf{x}|s=k)}\left[\ln\frac{q_\phi(\mathbf{z}|\mathbf{x},s=k)}{q_\phi(\mathbf{z})}\right] \tag{5.12}$$

$$= \sum_{k=1}^{K}\pi_k\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s=k)p_D(\mathbf{x}|s=k)}\left[\ln\frac{q_\phi(\mathbf{z}|\mathbf{x},s=k)}{q_\phi(\mathbf{z}|s=k)}\right] + \sum_{k=1}^{K}\pi_k D_{\mathrm{KL}}(q_\phi(\mathbf{z}|s=k)\|q_\phi(\mathbf{z})) \tag{5.13}$$

$$= \sum_{k=1}^{K}\pi_k I(X^{(k)};Z^{(k)}) + I(S;Z), \tag{5.14}$$

where X is a random variable for data points, and

$$I(X^{(k)};Z^{(k)}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s=k)p_D(\mathbf{x}|s=k)}\left[\ln\frac{q_\phi(\mathbf{z}|\mathbf{x},s=k)}{q_\phi(\mathbf{z}|s=k)}\right] \tag{5.15}$$

is the mutual information between X and Z when $s=k$. Therefore, $\mathcal{R}(\phi)$ can be finally rewritten as

$$\mathcal{R}(\phi) = \sum_{k=1}^{K}\pi_k I(X^{(k)};Z^{(k)}) + I(S;Z) + D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})). \tag{5.16}$$

Since the KL divergence and mutual information give a non-negative value, $\mathcal{R}(\phi)$ is upper bound of the mutual information $I(S;Z)$. □

Figure 5.1 shows the relationship between the mutual information $I(S;Z)$ and the KL term $\mathcal{R}(\phi)$. This theorem shows that the CVAE tries to minimize the mutual information $I(S;Z)$ by minimizing its upper bound $\mathcal{R}(\phi)$. However, $\mathcal{R}(\phi)$ is not a tight upper bound of $I(S;Z)$ because the KL divergence between the marginalized posterior and simple standard Gaussian prior $D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ usually gives a large value. This prevents the CVAE from minimizing $I(S;Z)$ sufficiently. That is, the simple prior $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0},\mathbf{I})$ is one cause of the task-dependency.

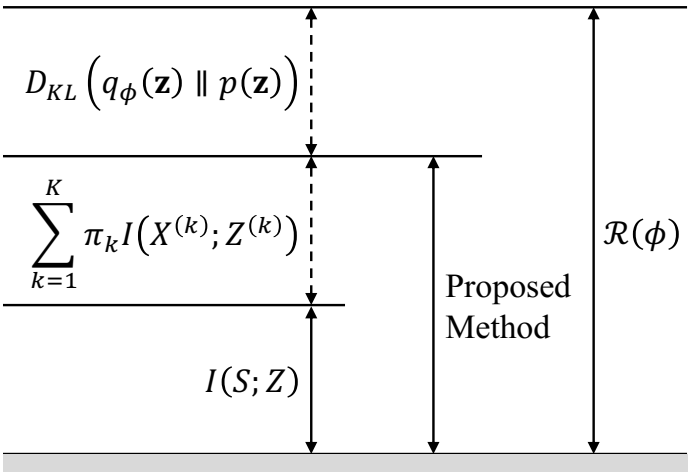


Figure 5.1: Relationship between the mutual information $I(S;Z)$, the KL term of the CVAE $\mathcal{R}(\phi)$ and our KL term in Eq. (5.17).

5.3.2 Optimal Prior for Task-Invariant Representations

When the prior $p(\mathbf{z})$ is equal to the marginalized posterior $q_\phi(\mathbf{z})$, $D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ gives zero, and $\mathcal{R}(\phi)$ becomes the tightest upper bound of $I(S;Z)$ since the other terms in $\mathcal{R}(\phi)$ do not depend on the prior. Therefore, $q_\phi(\mathbf{z})$ is the optimal prior in terms of reducing the task-dependency.

By replacing the standard Gaussian prior $p(\mathbf{z})$ in $\mathcal{F}_{\mathrm{CVAE}}(\theta,\phi)$ with the optimal prior $q_\phi(\mathbf{z})$, we obtain the ELBO with this optimal prior as follows:

$$\mathcal{F}_{\mathrm{Proposed}}(\theta,\phi) = \mathbb{E}_{p_D(\mathbf{x},s)q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln p_\theta(\mathbf{x}|\mathbf{z},s)\right] - \mathbb{E}_{p_D(\mathbf{x},s)}\left[D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x},s)\|q_\phi(\mathbf{z}))\right]. \tag{5.17}$$

For this objective function, we have the following theorem:

Theorem 5.2. $\mathcal{F}_{\mathrm{Proposed}}(\theta,\phi)$ (Eq. (5.17)) is always larger than or equal to $\mathcal{F}_{\mathrm{CVAE}}(\theta,\phi)$ (Eq. (5.4)) and $\mathcal{F}_{\mathrm{VFAE}}(\theta,\phi)$ (Eqs. (5.6) and (5.7)).

Proof. $\mathcal{F}_{\text{Proposed}}(\theta, \phi)$ can be rewritten as follows:

$$\begin{aligned}\mathcal{F}_{\text{Proposed}}(\theta, \phi) &= \mathbb{E}_{p_D(\mathbf{x},s)q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln p_\theta(\mathbf{x}|\mathbf{z}, s)\right] \\ &\quad - \mathbb{E}_{p_D(\mathbf{x},s)}\left[D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|p(\mathbf{z}))\right] \\ &\quad + D_{\text{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})) \qquad (5.18) \\ &= \mathcal{F}_{\text{CVAE}}(\theta, \phi) + D_{\text{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})). \qquad (5.19)\end{aligned}$$

Since the KL divergence gives a non-negative value, the following inequality holds:

$$\mathcal{F}_{\text{Proposed}}(\theta, \phi) \geq \mathcal{F}_{\text{CVAE}}(\theta, \phi) \geq \mathcal{F}_{\text{VFAE}}(\theta, \phi). \qquad (5.20)$$

□

This theorem shows that $\mathcal{F}_{\text{Proposed}}(\theta, \phi)$ is also a better lower bound of the log-likelihood than $\mathcal{F}_{\text{CVAE}}(\theta, \phi)$ and $\mathcal{F}_{\text{VFAE}}(\theta, \phi)$. This contributes to obtaining better representation for the improved performance on the target tasks.

From the above discussion, we can reduce the task-dependency and obtain better representations by using $q_\phi(\mathbf{z})$ as the prior.

5.3.3 Optimization with Density Ratio Trick

Although our aim is to optimize our objective function (Eq.(5.17)), it is difficult to approximate $D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|q_\phi(\mathbf{z}))$ because this contains a high-dimensional density ratio. Instead, we approximate and optimize Eq. (5.19), which is the deformation of Eq.(5.17). We can accurately estimate the KL divergence between $q_\phi(\mathbf{z})$ and $p(\mathbf{z})$:

$$D_{\text{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})) = \int q_\phi(\mathbf{z}) \ln \frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} \mathrm{d}\mathbf{z} \qquad (5.21)$$

with the density ratio trick (Sugiyama et al., 2012; Goodfellow et al., 2014; Takahashi et al., 2019), which can estimate the density ratio between two distributions using samples from both distributions.

To obtain $q_\phi(\mathbf{z})/p(\mathbf{z})$, we apply the density ratio trick as follows. Suppose that y is the binary label representing whether $\mathbf{z}$ comes from $q_\phi(\mathbf{z})$ $(y = 1)$ or $p(\mathbf{z})$ $(y = 0)$,

and $p^*(\mathbf{z}|y)$ is the following conditional distribution:

$$p^*(\mathbf{z}|y) \equiv \begin{cases} q_\phi(\mathbf{z}) & (y=1) \\ p(\mathbf{z}) & (y=0) \end{cases}. \tag{5.22}$$

According to the Bayes theorem, the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$ is rewritten as

$$\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} = \frac{p^*(\mathbf{z}|y=1)}{p^*(\mathbf{z}|y=0)} = \frac{p^*(y=1|\mathbf{z})}{p^*(y=0|\mathbf{z})}\frac{p^*(y=0)}{p^*(y=1)} = \frac{p^*(y=1|\mathbf{z})}{p^*(y=0|\mathbf{z})}, \tag{5.23}$$

where we assume $p^*(y=0) = p^*(y=1)$ because we can prepare the same number of samples from both distributions. Hence, if we have a probabilistic binary classifier $p^*(y|\mathbf{z})$ that outputs the probability of y with input $\mathbf{z}$, we can estimate the density ratio. We model $p^*(y=1|\mathbf{z})$ by $\sigma(T_\psi(\mathbf{z}))$, where $\sigma(\cdot)$ is a sigmoid function and $T_\psi(\mathbf{z})$ is a neural network with parameter ψ. To train $T_\psi(\mathbf{z})$, we solve the following binary classification problem:

$$\psi^* = \arg\max_\psi \mathbb{E}_{q_\phi(\mathbf{z})}\left[\ln(\sigma(T_\psi(\mathbf{z})))\right] + \mathbb{E}_{p(\mathbf{z})}\left[\ln(1-\sigma(T_\psi(\mathbf{z})))\right]. \tag{5.24}$$

With $T_{\psi^*}(\mathbf{z})$, the density ratio $q_\phi(\mathbf{z})/p(\mathbf{z})$ can be rewritten as

$$\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} = \frac{\sigma(T_{\psi^*}(\mathbf{z}))}{1-\sigma(T_{\psi^*}(\mathbf{z}))}, \tag{5.25}$$

and its logarithm can be rewritten as

$$\ln\frac{q_\phi(\mathbf{z})}{p(\mathbf{z})} = T_{\psi^*}(\mathbf{z}). \tag{5.26}$$

Therefore, $D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z}))$ can be approximated by $T_{\psi^*}(\mathbf{z})$ as follows:

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})) \simeq \mathbb{E}_{q_\phi(\mathbf{z})}\left[T_{\psi^*}(\mathbf{z})\right]. \tag{5.27}$$

With Eq. (5.27), $\mathcal{F}_{\text{Proposed}}(\theta, \phi)$ (Eq. (5.19)) can be approximated as

$$\begin{aligned}
\mathcal{F}_{\text{Proposed}}(\theta, \phi) &\simeq \mathcal{F}_{\text{CVAE}}(\theta, \phi) + \mathbb{E}_{q_\phi(\mathbf{z})}\left[T_{\psi^*}(\mathbf{z})\right] && (5.28)\\
&= \mathbb{E}_{p_D(\mathbf{x},s)}\left[\mathcal{L}(\mathbf{x}, s; \theta, \phi) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},s)}\left[T_{\psi^*}(\mathbf{z})\right]\right] && (5.29)\\
&= \mathbb{E}_{p_D(\mathbf{x},s)q_\phi(\mathbf{z}|\mathbf{x},s)}\left[\ln p_\theta(\mathbf{x}|\mathbf{z}, s) + T_{\psi^*}(\mathbf{z})\right]\\
&\quad - \mathbb{E}_{p_D(\mathbf{x},s)}\left[D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s)\|p(\mathbf{z}))\right]. && (5.30)
\end{aligned}$$

Given a dataset $\mathcal{D} = \left\{(\mathbf{x}^{(1)}, s^{(1)}), \ldots, (\mathbf{x}^{(N)}, s^{(N)})\right\}$, we can approximate Eq. (5.30) by the Monte Carlo approximation and the reparameterization trick (Kingma and Welling, 2014) as follows:

$$\sum_{n=1}^{N}\sum_{\ell=1}^{L}\left\{\ln p_\theta(\mathbf{x}^{(n)}|\mathbf{z}^{(n,\ell)}, s^{(n)}) + T_\psi(\mathbf{z}^{(n,\ell)})\right\} - \frac{1}{N}\sum_{n=1}^{N} D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}^{(n)}, s^{(n)})\|p(\mathbf{z})), \tag{5.31}$$

where L is the sample size of the reparameterization trick, and $\mathbf{z}^{(n,\ell)}$ is a sample drawn from $q_\phi(\mathbf{z}|\mathbf{x}^{(n)}, s^{(n)})$ by the reparameterization trick. Similarly, we approximate Eq. (5.24) as follows:

$$\frac{1}{N'}\sum_{n'=1}^{N'}\left\{\ln(\sigma(T_\psi(\mathbf{z}_1^{(n')}))) + \ln(1 - \sigma(T_\psi(\mathbf{z}_0^{(n')})))\right\}, \tag{5.32}$$

where $\mathbf{z}_0^{(n')}$ is a sample drawn from $p(\mathbf{z})$, $\mathbf{z}_1^{(n')}$ is a sample drawn from $q_\phi(\mathbf{z})$ by the ancestral sampling[1], and N' is the sample size of the Monte Carlo approximation. We set N' to be the same as the dataset size of all tasks N. We alternately optimize Eq. (5.31) and Eq. (5.32) like generative adversarial networks (Goodfellow et al., 2014) by the stochastic gradient descent (SGD) such as Adam (Kingma and Ba, 2015). Note that we fix ψ when optimizing Eq. (5.31) and fix θ and ϕ when optimizing Eq. (5.32). Algorithm 5.1 shows pseudo code of our approach, where M and M' are the sizes of mini-batch, and I_1 and I_2 are numbers of iteration for SGD.

[1]We choose $(\mathbf{x}^{(n')}, s^{(n')})$ from a dataset randomly and sample $\mathbf{z}$ from $q_\phi(\mathbf{z}|\mathbf{x}^{(n')}, s^{(n')})$ by the reparameterization trick.

Algorithm 5.1 Training procedure of our approach.

Input: Dataset $\mathcal{D}$, mini-batch sizes M, M', numbers of iteration I_1, I_2 and size of reparameterization trick L

Output: Parameters of CVAE θ, ϕ and density ratio estimator ψ

while not converged **do**
  for I_1 steps **do** ▷ Training CVAE with our prior
    Sample mini-batch $\left\{(\mathbf{x}^{(m)}, s^{(m)})\right\}_{m=1}^{M}$ from dataset $\mathcal{D}$
    Sample $\left\{\mathbf{z}^{(m,\ell)}\right\}_{m=1,\ell=1}^{M,L}$ from $q_\phi(\mathbf{z}|\mathbf{x}^{(m)}, s^{(m)})$
    Compute the gradients of Eq. (5.31) w.r.t. θ and ϕ
    Update θ and ϕ with their gradients
  for I_2 steps **do** ▷ Training density ratio estimator
    Sample mini-batch $\left\{\mathbf{z}_0^{(m')}, \mathbf{z}_1^{(m')}\right\}$ from $p(\mathbf{z})$ and $q_\phi(\mathbf{z})$
    Compute the gradient of Eq. (5.32) w.r.t. ψ
    Update ψ with its gradient

5.4 Related work

5.4.1 Transfer and Multi-Task Learning

Numerous approaches have been presented to overcome the lack of training data for the VAE. They can be roughly divided into transfer learning and multi-task learning. Transfer learning aims to improve the performance on the target task by transferring the knowledge of the source task to the target task, and many approaches based on this have recently been presented (Brunner et al., 2018; Cífka et al., 2021). In contrast, our approach is based on multi-task learning. Although transfer learning approaches can only improve the performance on the target task, our approach can improve the performance on multiple tasks simultaneously by sharing the task-invariant knowledge through the task-invariant latent variable. For training the VAE on multiples tasks, the CVAE (Kingma et al., 2014; Sohn et al., 2015) and its variants such as the VFAE (Louizos et al., 2016) have been used, which aims to obtain the task-invariant representations. However, the task-dependency often remains in the CVAE, and the VFAE does not work well because its training objective function is not a tight lower bound of the log-likelihood. Our approach can reduce the task-dependency and provide better lower bound of the log-likelihoods, which enables us to obtain better representations for the improved performance.

5.4.2 Learning Priors of VAE

Numerous approaches focus on the VAE prior. For example, the non-parametric (Nalisnick and Smyth, 2017; Casale et al., 2018) or hyper-spherical (Davidson et al., 2018) distribution is used as the prior to obtain the interpretable representations. Among them, the aggregated posterior (Hoffman and Johnson, 2016), which improves the density estimation performance, has been widely researched. Since this prior is hard to handle, many approaches try to approximate this prior. These approaches can be divided into two categories: explicit modeling and implicit approximation. The former approximates this prior explicitly by using generative models such as the mixture of the encoder (Tomczak and Welling, 2018) or another VAE (Klushyn et al., 2019). The latter approximates this prior implicitly using density ratio estimation (Takahashi et al., 2019; Aneja et al., 2021). Our prior (Eq. (5.8)) corresponds to the multi-task extension of the aggregated posterior, and we approximate it implicitly. We theoretically show that our prior is optimal in terms of reducing task-dependency.

5.4.3 Introducing Task-Specific Representations

Recent approaches introduce the task-specific latent variable into the CVAE architecture (Ilse et al., 2020) for disentangling the task-invariant and task-specific representations. With the task-specific latent variable, the conditional probability of $\mathbf{x}$ given s is modeled as

$$\int p_\theta(\mathbf{x}|\mathbf{z}, \boldsymbol{\zeta}) p(\mathbf{z}) p_\lambda(\boldsymbol{\zeta}|s) \mathrm{d}\mathbf{z}\mathrm{d}\boldsymbol{\zeta}, \tag{5.33}$$

where $p_\theta(\mathbf{x}|\mathbf{z}, \boldsymbol{\zeta})$ is the decoder distribution, $\mathbf{z}$ is the task-invariant latent variable, $\boldsymbol{\zeta}$ is the task-specific latent variable, $p(\mathbf{z})$ is the task-invariant prior, and $p_\lambda(\boldsymbol{\zeta}|s)$ is the task-specific prior. $p_\lambda(\boldsymbol{\zeta}|s)$ is modeled by a neural network with parameter λ.

Its training objective function for each data point is given as

$$\mathbb{E}_{q_\phi(\mathbf{z},\boldsymbol{\zeta}|\mathbf{x},s)} \left[\ln p_\theta(\mathbf{x}|\mathbf{z}, \boldsymbol{\zeta})\right] - \beta_1 D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, s) \| p(\mathbf{z})) - \beta_2 D_{\mathrm{KL}}(q_\phi(\boldsymbol{\zeta}|\mathbf{x}, s) \| p_\lambda(\boldsymbol{\zeta}|s)), \tag{5.34}$$

where

$$q_\phi(\mathbf{z}, \boldsymbol{\zeta}|\mathbf{x}, s) = q_\phi(\mathbf{z}|\mathbf{x}, s)q_\phi(\boldsymbol{\zeta}|\mathbf{x}, s) \tag{5.35}$$

is the encoder distribution, and β_1 and β_2 are hyperparameters that are equal to or larger than one. Especially when $\beta_1 = \beta_2 = 1$, this becomes the ELBO. Since latent variables $\mathbf{z}$ and $\boldsymbol{\zeta}$ are regularized by task-invariant and task specific priors, respectively, it is expected that the task-invariant components of data points are encoded to $\mathbf{z}$ and their task-specific components are encoded to $\boldsymbol{\zeta}$. In addition, recent approaches (Ilse et al., 2020) introduce an additional classifier $q_\varphi(s|\boldsymbol{\zeta})$ to make $\boldsymbol{\zeta}$ dependent on s. $q_\varphi(s|\boldsymbol{\zeta})$ is modeled by a neural network with parameter φ. We call this model the β-CVAE.

However, this requires carefully tuning hyperparameters β_1 and β_2 to disentangle task-invariant and task-specific representations (Ilse et al., 2020). This incurs heavy computational costs. Moreover, it has been reported to work well even without the task-specific latent variable (Ilse et al., 2020). In Section 5.5, we compare our approach with the β-CVAE. Note that our approach can be used with such approaches for learning the prior of task-invariant latent variable.

5.5 Experiments

In this section, we qualitatively and quantitatively evaluate our approach on multi-task datasets.

5.5.1 Evaluation Metrics

For qualitative evaluation, we visualize the latent variable by using t-distributed Stochastic Neighbor Embedding (Maaten and Hinton, 2008) to reduce the dimensions of the latent variable to 2. For quantitative evaluation, we used following three metrics: density estimation performance, number of active units, and downstream classification accuracy.

Density Estimation Performance

First, we calculated the log-likelihood of test data using the importance sampling (Burda et al., 2016) to evaluate the generalization performance of the generative model. Given test data $(\mathbf{x}^*, s^*)$, its conditional log-likelihood under our approach is approximated as follows:

$$\ln p_\theta(\mathbf{x}^*|s^*) = \ln \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^*,s^*)} \left[\frac{p_\theta(\mathbf{x}^*|\mathbf{z}, s^*) q_\phi(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}^*, s^*)} \right] \tag{5.36}$$

$$\simeq \ln \frac{1}{J} \sum_{j=1}^{J} \frac{p_\theta(\mathbf{x}^*|\mathbf{z}^{(j)}, s^*) p(\mathbf{z}^{(j)})}{q_\phi(\mathbf{z}^{(j)}|\mathbf{x}^*, s^*)} \exp(T_\psi(\mathbf{z}^{(j)})), \tag{5.37}$$

where J is the sample size of the importance sampling and $\mathbf{z}^{(j)}$ is a sample drawn from $q_\phi(\mathbf{z}|\mathbf{x}^*, s^*)$. We set J to 10.

Downstream Classification Accuracy

Next, we calculated the downstream classification accuracy, which measures the effect of task-invariant representations on classification problem. Suppose that the target task has a small amount of labeled data whereas source tasks have a large amount of unlabeled data. We first obtain task-invariant representations from target and source tasks using an unsupervised estimator, after which we train an additional classifier with the representations of target data and these labels. Using the estimator and classifier, we calculated the classification accuracy for test data of the target task. We used k-nearest neighbor classifier as an additional classifier, and set the number of neighbors to three.

Number of Active Units

Finally, we calculated the number of active units (Burda et al., 2016), which measures the activity of each dimension of latent variables. This metric is frequently used to evaluate the latent variable of the VAE. The d-th dimension of the latent variable: z_d is active if

$$\mathbb{V}_{p_D(\mathbf{x},s)} \left[\mathbb{E}_{q_\phi(z_d|\mathbf{x},s)} \left[z_d \right] \right] > \delta, \tag{5.38}$$

Table 5.1: Numbers and dimensions of data points of datasets.

	Dimension	Train size	Valid size	Test size
USPS	784	6,438	1,000	1,860
MNIST	784	10,000	10,000	10,000
SynthDigits	1,024	10,000	10,000	9,553
SVHN	1,024	10,000	10,000	26,032
Frey	560	1,565	200	200
Olivetti	560	150	100	150
UMist	560	300	75	200

where $\mathbb{V}(\cdot)$ computes the variance, and δ is a user-defined threshold. We set δ to 0.01 (Burda et al., 2016). If z_d contains useful information about the data, it becomes active since it changes as the data point changes. Hence, the higher the number of active dimensions, the better the learned representation.

5.5.2 Data

We used two handwritten digits (USPS (Hull, 1994) and MNIST (LeCun et al., 1998)), two house number digits (SynthDigits (Ganin and Lempitsky, 2015) and SVHN (Netzer et al., 2011)), and three face datasets (Frey, Olivetti, and UMist)[2]. We binarized USPS and MNIST and resized USPS to have the same resolution as MNIST. We converted SVHN and SynthDigits into gray-scale images. We resized face datasets to have the same resolution. The numbers and dimensions of data points of the datasets are listed in Table 5.1.

On digit datasets, we conducted two-task experiments, in which the source task has a large amount of data whereas the target task has a small amount of data. We evaluated all metrics on test data of the target task. We used all data points of the training and validation of the source task but used only 100 training data points of the target task. The pairs of source and target tasks are as follows: USPS→MNIST, MNIST→USPS, SynthDigits→SVHN, and SVHN→SynthDigits.

On face datasets, we conducted the three-task experiment, in which evaluation on each task was performed simultaneously using a single estimator. We evaluated the density estimation performance and the number of active units by the mean score

[2]These datasets are available at `https://cs.nyu.edu/~roweis/data.html`

for test data of each task.

5.5.3 Setup

We compared our approach with the VAE, CVAE, β-CVAE, and VFAE. We set the dimensions of the latent variable vector to 40 for all datasets. We used two-hidden-layer neural networks for the encoder and the decoder of all approaches, the prior and the additional classifier of the β-CVAE, and the density ratio estimator of our approach. We set the number of units of a hidden layer to 500 and used a hyperbolic tangent as the activation function. For the encoder, we used a Gaussian distribution. For the decoder, we used a Bernoulli distribution for USPS and MNIST and a Gaussian distribution for SynthDigits, SVHN, and face datasets.

We trained all methods using Adam (Kingma and Ba, 2015) optimizer with a mini-batch size of 100. We set the learning rate to 10^{-3} for the density ratio estimator and to 10^{-4} for the other neural networks. We set the maximum number of epochs to 1000 for digit datasets and to 2000 for face datasets. We used early-stopping (Goodfellow et al., 2016) on the basis of validation data. We set the sample size of the reparameterization trick to $L = 1$. We updated the parameter of the density ratio estimator: ψ for 10 epochs during the updating of the parameters of the encoder and decoder: θ and ϕ for one epoch. To avoid over-fitting, we used dropout (Goodfellow et al., 2016) for the last hidden layer of the aforementioned neural networks. We set the dropout rate to 50% for the density ratio estimator and to 10% for the other neural networks. For the VAE, we ignored the task index in the training. For the VFAE, we tuned the hyperparameter α in Eq. (5.6) from $[0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]$ in accordance with the validation data. For the β-CVAE, we set the dimension of task-specific latent variable to 40, and set the hyperparameters $\beta_1 = \beta_2 = 1$ since this is most advantageous for density estimation.

We ran all experiments eight times while changing the random seeds. We provide the details of the set up in the appendix. The machine specifications used in the experiments are as follows: CPU is Intel Xeon E5-2699A v4 2.40GHz, the memory size is 500GB, and GPU is NVIDIA Tesla P100.

5.5.4 Justification of Hyperparameters

Numbers of Layers and Units of Neural Networks The VFAE (Louizos et al., 2016) used one-hidden-layer neural networks for the encoder and decoder, and set the maximum number of hidden units to 500. However, we experimentally confirmed that deeper and wider neural networks achieve better performance. Therefore, we used two-hidden-layer neural networks, and set the number of all hidden units to 500.

Dimension of Latent Variable We set the dimensions of the latent variable vector to 40 for all datasets. We confirmed that tuning the dimension of the latent variable has almost no effect on the performance when the dimension is high enough.

Dropout Rate We used dropout to avoid over-fitting, which works better than other regularization techniques such as weight decay (Krogh and Hertz, 1992). We also set the dropout rate to 10% for the encoder and decoder and to 50% for the density ratio estimator. We set the high dropout rate for the density ratio estimator because it is likely to over-fit.

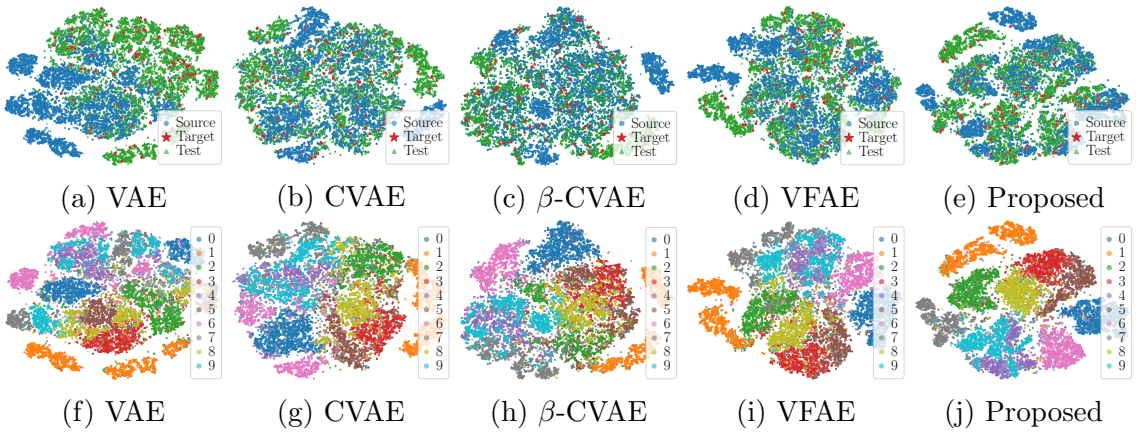
5.5.5 Results

Qualitative Results

We first provide an intuitive explanation with visualization. Figure 5.2 visualizes the latent variables of each approach on USPS→MNIST. Figures 5.2(a)–5.2(e) are color-coded by the source training data, target training data, and target test data. Figures 5.2(f)–5.2(j) are color-coded by digits, which are unknown during training. They show the task-dependency of each approach intuitively.

First, we focus on the VAE (Figures 5.2(a) and 5.2(f)). With some digits such as 0, 6, and 9, the latent variables of the source task were farther from those of the target task. That is, the task-dependency remained in the VAE.

Second, we focus on the CVAE (Figures 5.2(b) and 5.2(g)), β-CVAE (Figures 5.2(c) and 5.2(h)), and VFAE (Figures 5.2(d) and 5.2(i)). Compared with the VAE, the latent variables of the source task were closer to those of the target task when the latent variables corresponded to the same digit. That is, the task-dependency was



(a) VAE (b) CVAE (c) β-CVAE (d) VFAE (e) Proposed

(f) VAE (g) CVAE (h) β-CVAE (i) VFAE (j) Proposed

Figure 5.2: (a-e) Visualization of the latent variables on USPS→MNIST. Samples in each color correspond to each digit. (f-j) Visualization of the latent variables on USPS→MNIST. The color indicates that the node is in source training data, target training data, or target test data.

reduced to some extent by using the CVAE architecture. There was no significant difference between the CVAE and β-CVAE in this visualization. For example, the β-CVAE reduced the task-dependency for digit 6 but did not reduce that for digit 2, which is opposite to the result of the CVAE. This shows the difficulty of disentangling task-specific and task-independent representations. The VFAE reduced the task-dependency more than the CVAE (e.g., the digit 6) and the β-CVAE (e.g., the digit 2) in accordance with the MMD regularizer. However, the latent variables that corresponded to different digits heavily overlapped. In this case, it is difficult for the decoder to reconstruct the data points from the latent variables, which causes the large reconstruct error. This leads to the poor performance of density estimation and downstream classification.

Third, we focus on our approach (Figures 5.2(e) and 5.2(j)). Our approach reduced the task-dependency similar to the VFAE. In addition, the latent variables that corresponded to different digits rarely overlapped. Hence, the data points were easy to reconstruct from the latent variables, which reduced the reconstruction error. This leads to the improved performances of density estimation and downstream classification.

Table 5.2: Comparison of density estimation performance. We use bold to highlight the best result and the results that are not statistically different from the best result according to a pair-wise t-test. We used 5% as the p-value.

	VAE	CVAE	β-CVAE	VFAE	Proposed
USPS→MNIST	-163.23 ± 2.15	-152.32 ± 1.64	-168.95 ± 2.20	-151.86 ± 2.12	**-149.08 ± 0.86**
MNIST→USPS	-235.23 ± 1.54	**-211.18 ± 0.55**	-268.21 ± 42.53	**-210.93 ± 0.50**	**-212.11 ± 1.48**
SynthDigits→SVHN	1146.04 ± 35.65	1397.36 ± 10.89	1314.95 ± 60.26	1395.64 ± 11.61	**1430.27 ± 11.44**
SVHN→SynthDigits	760.66 ± 8.85	814.63 ± 10.09	**938.83 ± 13.84**	820.55 ± 12.21	855.51 ± 11.41
Face Datasets	895.41 ± 2.98	902.99 ± 3.69	855.24 ± 48.60	823.08 ± 7.39	**913.08 ± 5.05**

Table 5.3: Comparison of downstream classification accuracy. We use bold to highlight the best result and the results that are not statistically different from the best result according to a pair-wise t-test. We used 5% as the p-value.

	VAE	CVAE	β-CVAE	VFAE	Proposed
USPS→MNIST	0.52 ± 2.15	0.53 ± 0.02	0.54 ± 0.01	0.53 ± 0.01	**0.68 ± 0.01**
MNIST→USPS	0.64 ± 0.01	0.67 ± 0.01	0.65 ± 0.02	0.67 ± 0.01	**0.74 ± 0.02**
SynthDigits→SVHN	0.20 ± 0.00	**0.21 ± 0.00**	0.19 ± 0.01	0.20 ± 0.00	0.19 ± 0.00
SVHN→SynthDigits	0.25 ± 0.01	0.25 ± 0.00	0.17 ± 0.01	0.25 ± 0.00	**0.26 ± 0.00**

Quantitative Results

We next give quantitative evaluation by metrics described in Section 5.5.1. Tables 5.2, 5.3, and 5.4 compare the density estimation performances, downstream classification accuracies, and numbers of active units, respectively.

First, we focus on the VAE. The density estimation performances of the VAE were worse than those of other approaches, and the downstream classification accuracies of the VAE were equal to or worse than those of other approaches. This indicates that the CVAE architecture works well for obtaining task-invariant representation to some extent. Interestingly, the numbers of active units of the VAE are equal to or larger than those of other approaches, except for the face datasets. This is likely because the VAE tries to encode task-specific components of all tasks to latent variables, which requires larger representation capacity than the CVAE architecture.

Next, we focus on the VFAE and CVAE. The density estimation performances of the VFAE were almost the same as those of the CVAE in digit datasets but worse than those of the CVAE in face datasets. The downstream classification accuracies of the VFAE were almost the same as or worse than those of the CVAE. These results indicate that the MMD regularizer of the VFAE does not improve the quality of representation. The numbers of active units indicate the similar tendency, except for

Table 5.4: Comparison of number of active units. We use bold to highlight the best result and the results that are not statistically different from the best result according to a pair-wise t-test. We used 5% as the p-value.

	VAE	CVAE	β-CVAE	VFAE	Proposed
USPS→MNIST	**8.00 ± 2.29**	7.38 ± 0.48	7.75 ± 1.30	8.00 ± 0.87	**9.00 ± 0.00**
MNIST→USPS	**10.25 ± 3.31**	9.00 ± 0.00	7.75 ± 0.43	8.88 ± 0.33	**14.00 ± 4.36**
SynthDigits→SVHN	**40.00 ± 0.00**	**39.88 ± 0.33**	14.62 ± 3.04	**40.00 ± 0.00**	**40.00 ± 0.00**
SVHN→SynthDigits	**39.88 ± 0.33**	**39.62 ± 0.48**	11.12 ± 4.91	**39.50 ± 0.87**	**40.00 ± 0.00**
Face Datasets	17.38 ± 4.85	22.38 ± 10.00	9.62 ± 1.11	**40.00 ± 0.00**	37.25 ± 2.49

the face datasets. The reason the number of active units of the VFAE on face datasets is larger than that of the CVAE may be as follows. As the MMD regularization on more than two tasks (Eq. (5.7)) is likely to be too strong, the VFAE tries to maximize its ELBO using latent variables to the fullest extent.

Next, we focus on the β-CVAE. On SVHN→SynthDigits, the density estimation performance was better than other approaches. However, the density estimation performances were worse than other approaches on the other datasets. Moreover, the downstream classification performances were almost the same as those of the other approaches except in the case of our approach. These results indicate that task-specific latent variable does not always work well. Furthermore, the numbers of active units were smaller than those of other approaches. This is because the active units are distributed to the task-invariant and task-specific latent variables.

Finally, we focus on our approach. Our approach achieved equal to or better density estimation performance than other approaches except the SVHN→SynthDigits, and achieved equal to or better downstream classification accuracies than the other approaches. In addition, the numbers of active units of our approach were larger than those of the other approaches except in the case of face datasets, and slightly worse than that of VFAE on face datasets. These results indicate that our approach can obtain good task-invariant latent variables in many situations.

5.6 Conclusion

In this chapter, we explain why the standard Gaussian prior cannot sufficiently reduce the task-dependency in the CVAE, and propose a theoretical optimal prior for reducing the task-dependency. We also show that this optimal prior presents a

better lower bound of the log-likelihood than that of the CVAE and VFAE, which enable us to obtain better representation for the improved performance. In addition, we demonstrate that our approach can obtain better task-invariant representations than existing approaches, which leads to the improved performances of density estimation and downstream classification on various datasets. In the future, we will try to extend our approach to deal with more difficult problem settings such as meta learning (Vilalta and Drissi, 2002) or test-time adaptation (Wang et al., 2021).

Chapter 6

Conclusion and Future Direction

6.1 Conclusion

This dissertation solved following three serious drawbacks in real-world applications of the VAE:

1. instability of training for continuous data (Section 3),
2. over-regularization by prior (Section 4), and
3. poor performance with insufficient data (Section 5).

We summarize our contributions as follows.

Instability of Training for Continuous Data For continuous data points, the decoder of the VAE is modeled by the Gaussian, which we call the Gaussian VAE. However, the training of the Gaussian VAE often becomes unstable. We investigated why the training becomes unstable, and revealed that the Gaussian decoder is sensitive to the error between the data point and its decoded mean when the decoded variance is almost zero. To solve this, we introduced a Bayesian approach to the Gaussian decoder: we set a Gamma prior for the inverse of the decoded variance and marginalized it out analytically, which led to proposing our Student-t VAE. Since the Student-t distribution is a heavy-tailed distribution, the Student-t decoder is robust to the error, which makes the training stable. We experimentally showed the robustness of the training and the high density estimation performance of our Student-t VAE.

Over-Regularization by Prior The prior in the VAE is usually modeled by the standard Gaussian distribution, but this simple prior incurs over-regularization, which is one of the causes of poor performance. As a sophisticated prior, the aggregated posterior has been introduced, which is optimal in terms of maximizing the ELBO of VAE. However, since the KL divergence with the aggregated posterior cannot be calculated in a closed form, we cannot this optimal prior directly. Although the VampPrior has been presented to model the aggregated posterior explicitly, it has sensitive hyperparameters that are difficult to tune.

With the proposed method, we directly estimated the KL divergence with the aggregated posterior by using the density ratio trick. This enabled us to avoid modeling the aggregated posterior explicitly. That is, there is no need to model the aggregated posterior explicitly. However, the density ratio trick does not work well for the high-dimensional density ratio such as the KL divergence with the aggregated posterior. To solve this, we rewrote the KL divergence into the sum of two terms: the KL divergence between the encoder and the standard Gaussian distribution, and the low-dimensional density ratio between the aggregated posterior and the standard Gaussian distribution. The first term can be calculated in a closed form, and we can accurately approximate the second term since it is low-dimensional density ratio. We experimentally showed that the VAE with our implicit optimal prior achieved better performance than existing approaches.

Poor Performance with Insufficient Data Although the VAE is powerful, its performance becomes poor with insufficient data points since it requires many data points for training neural networks. To overcome the lack of training data on the target task, we focused on training the VAE on multiple tasks. This enables us to obtain the task-invariant latent variable that shares task-invariant knowledge effectively. For this purpose, the CVAE has been widely used. The CVAE tries to minimize the dependency of the latent variable on the task through the minimizing the KL divergence the encoder and the standard Gaussian prior. However, this task-dependency remains in many cases.

To solve this, we investigated why the CVAE cannot reduce the task-dependency

sufficiently, and revealed that the standard Gaussian prior is one of the causes. Based on this investigation, we proposed a theoretical optimal prior for reducing the task-dependency, and showed that this optimal prior can provide a better lower bound of the log-likelihood than that of the CVAE. This enabled us to obtain better representation for the improved performance. We experimentally showed that our approach obtained better task-invariant representations and achieved better performances than existing approaches.

Through the above contributions, we have made the VAE somewhat practically applicable: our VAE can be applied to biased data, perform well, and handle the case that the target task has insufficient data points. We have applied our VAE to several anomaly detection applications such as factory monitoring and autonomous driving.

6.2 Future Direction

Finally, we discuss the future directions. Our goal is to solve the drawbacks of the VAE and make the VAE practically applicable. Although this goal has been achieved to some extent with the above contributions, some serious drawbacks still remain in the VAE.

One of the serious drawbacks is that the VAE does not work well in the case where the data distribution is not stationary, i.e., the data distribution changes over time. For example, in anomaly detection for autonomous driving, sensor values change gradually as each component degrades over time. Like this, non-stationary real-world datasets are common and need to be addressed. Another serious drawback is that deep generative models including the VAE are actually not able to detect the out-of-distribution samples sufficiently (Nalisnick et al., 2018). This is very serious problem for anomaly detection, one of the key applications of VAE.

We will try to solve these critical problems and make VAE truly practically applicable.

Bibliography

A. Achille, T. Eccles, L. Matthey, C. Burgess, N. Watters, A. Lerchner, and I. Higgins. Life-long disentangled representation learning with cross-domain latent homologies. *Advances in Neural Information Processing Systems*, 31, 2018.

K. Akuzawa, Y. Iwasawa, and Y. Matsuo. Expressive speech synthesis via modeling expressions with variational autoencoder. *Proc. Interspeech 2018*, pages 3067–3071, 2018.

J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1):1–18, 2015.

J. Aneja, A. Schwing, J. Kautz, and A. Vahdat. A contrastive learning approach for training variational autoencoder priors. *Advances in Neural Information Processing Systems*, 34, 2021.

S. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, 2016.

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer. MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer. In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 747–754. ISMIR, 2018.

Y. Burda, R. B. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. In *ICLR (Poster)*, 2016.

R. Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

F. P. Casale, A. Dalca, L. Saglietti, J. Listgarten, and N. Fusi. Gaussian process prior variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

O. Cífka, A. Ozerov, U. Şimşekli, and G. Richard. Self-supervised VQ-VAE for one-shot music style transfer. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 96–100. IEEE, 2021.

T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083*, 2016.

T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. Hyperspherical variational auto-encoders. *34th Conference on Uncertainty in Artificial Intelligence (UAI-18)*, 2018.

N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

E. Egorov, A. Kuzina, and E. Burnaev. Boovae: Boosting approach for continual learning of vae. *Advances in Neural Information Processing Systems*, 34, 2021.

M. Ehsan Abbasnejad, A. Dick, and A. van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5888–5897, 2017.

Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015.

J.-M. Geusebroek, G. J. Burghouts, and A. W. Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pages 2551–2559, 2015.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

M. Goldstein and S. Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173, 2016.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. DRAW: A recurrent neural network for image generation. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1462–1471, 2015.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2007.

J. D. Havtorn, J. Frellsen, S. Hauberg, and L. Maaløe. Hierarchical vaes know what they don't know. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 4117–4128, 2021.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. 2016.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

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

W.-N. Hsu, Y. Zhang, and J. Glass. Learning latent representations for speech generation and transformation. *Proc. Interspeech 2017*, pages 1273–1277, 2017.

C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville. Neural autoregressive flows. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2078–2087, 2018.

J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

M. Ilse, J. M. Tomczak, C. Louizos, and M. Welling. DIVA: Domain invariant variational autoencoders. In *Medical Imaging with Deep Learning*, pages 322–348. PMLR, 2020.

B. Ivanovic, K. Leung, E. Schmerling, and M. Pavone. Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *IEEE Robotics and Automation Letters*, 6(2):295–302, 2020.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

P. Jylänki, J. Vanhatalo, and A. Vehtari. Robust Gaussian process regression with a Student-t likelihood. *Journal of Machine Learning Research*, 12(Nov):3227–3257, 2011.

D. Khattar, J. S. Goud, M. Gupta, and V. Varma. MVAE: Multimodal variational autoencoder for fake news detection. In *The world wide web conference*, pages 2915–2921, 2019.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *2nd International Conference on Learning Representations*, 2014.

D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

T. N. Kipf and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

A. Klushyn, N. Chen, R. Kurle, B. Cseke, and P. van der Smagt. Learning hierarchical priors in vaes. *Advances in neural information processing systems*, 32, 2019.

A. Krogh and J. A. Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

K. L. Lange, R. J. Little, and J. M. Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

J. Li, J. Yu, J. Li, H. Zhang, K. Zhao, Y. Rong, H. Cheng, and J. Huang. Dirichlet graph variational autoencoder. *Advances in Neural Information Processing Systems*, 33, 2020.

M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

P. Liu, X. Qiu, and X. Huang. Adversarial multi-task learning for text classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, 2017.

M. Long, Z. Cao, J. Wang, and S. Y. Philip. Learning multiple tasks with multilinear relationship networks. In *Advances in neural information processing systems*, pages 1594–1603, 2017.

C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *International Conference on Learning Representations (ICLR)*, 2016.

L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

R. Martinez-Cantin, M. McCourt, and K. Tee. Robust Bayesian optimization with Student-t likelihood. *arXiv preprint arXiv:1707.05729*, 2017.

L. Mescheder, S. Nowozin, and A. Geiger. Adversarial Variational Bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pages 2391–2400, 2017.

A. Miller, N. Foti, A. D'Amour, and R. P. Adams. Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems 30*, pages 3711–3721, 2017.

E. Nalisnick and P. Smyth. Stick-breaking variational autoencoders. In *International Conference on Learning Representations (ICLR)*, 2017.

E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? In *International Conference on Learning Representations*, 2018.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1278–1286, 2014.

G. Roeder, Y. Wu, and D. K. Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, pages 6928–6937, 2017.

M. Rosca, B. Lakshminarayanan, and S. Mohamed. Distribution matching in variational inference. *arXiv preprint arXiv:1802.06847*, 2018.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879. ACM, 2008.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

M. Sugiyama, T. Suzuki, and T. Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

S. Suh, D. H. Chae, H.-G. Kang, and S. Choi. Echo-state conditional variational autoencoder for anomaly detection. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 1015–1022. IEEE, 2016.

H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi. Variational autoencoder with implicit optimal priors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5066–5073, 2019.

T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.

J. M. Tomczak and M. Welling. Improving variational auto-encoders using householder flow. *arXiv preprint arXiv:1611.09630*, 2016.

J. M. Tomczak and M. Welling. VAE with a VampPrior. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pages 1214–1223, 2018.

B. Uria, M.-A. Côté, K. Gregor, I. Murray, and H. Larochelle. Neural autoregressive distribution estimation. *The Journal of Machine Learning Research*, 17(1):7184–7220, 2016.

A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with PixelCNN decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798, 2016.

A. van den Oord, O. Vinyals, and k. kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6309–6318, 2017.

R. Vilalta and Y. Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95, 2002.

C. Wang, X. Chen, A. J. Smola, and E. P. Xing. Variance reduction for stochastic gradient optimization. In *Advances in Neural Information Processing Systems*, pages 181–189, 2013.

D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

Z. Xiao, Q. Yan, and Y. Amit. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. *Advances in Neural Information Processing Systems*, 33, 2020.

W. Xu, H. Sun, C. Deng, and Y. Tan. Variational autoencoder for semi-supervised text classification. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick. Improved variational autoencoders for text modeling using dilated convolutions. In *International Conference on Machine Learning*, pages 3881–3890, 2017.

M. D. Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.