Bottleneck Conditional Density Estimators

Rui Shu[*] Stanford University

Hung H. Bui Adobe Research

Mohammad Ghavamzadeh Adobe Research

Abstract

We propose a neural network framework for high-dimensional conditional density estimation. The Bottleneck Conditional Density Estimator (BCDE) is a variant of the conditional variational autoencoder (CVAE) that employs layer(s) of stochastic variables as the bottleneck between the high-dimensional input x and target y. The key to effectively train BCDEs is the hybrid blending of the conditional generative model with a joint generative model that leverages unlabeled data to regularize the conditional model. Using multiple stochastic layers as the bottleneck also allows us to impose an independence prior (x and y are independent *a priori*) to prevent overfitting when we only have access to a small number of labeled data. We show that the BCDE significantly outperforms the CVAE in MNIST quadrant prediction benchmarks in the fully supervised case and establishes new benchmarks in the semi-supervised case.

1 Introduction

Conditional density estimation (CDE) refers to the problem of estimating a conditional density $p(y \mid x)$ for the input vector x and target vector y. In CDE, y is typically continuous or high-dimensional, as opposed to a discrete class label, and we wish to estimate the full conditional density, as opposed to its conditional mean or mode.

Classical non-parametric conditional density estimators typically rely on local Euclidean distance in the original input and target space, and thus, quickly become ineffective in high-dimensions [2]. For high-dimensional CDE tasks, conditional variational autoencoders (CVAEs) have been proposed [14] and applied to a variety of problems, such as MNIST quadrant prediction, segmentation [14], attribute-based image generation [16], and machine translation [17]. However, CVAEs suffer from two major statistical deficiencies: **1)** they inherently do not learn the distribution of the input x and **2)** are unable to leverage unlabeled data. For many conditional density estimation tasks, the acquisition of labeled points is costly, motivating the need for semi-supervised CDE. While variational methods [6, 11] have been applied to semi-supervised classification (where y is simply a class label) [7, 9], semi-supervised CDE (where y is high-dimensional) remains an open problem.

We focus on a set of deep conditional generative models, which we call them *bottleneck conditional density estimators* (BCDEs). In BCDEs, the input x only influences the target y via layers of bottleneck stochastic variables $z = \{z_i\}$ in the conditional generative path. The BCDE naturally has a joint generative sibling model, where the bottleneck z generates x and y independently, a key feature that enables semi-supervised learning. Following [8], we propose a hybrid training framework that regularizes the conditionally-trained BCDE parameters toward the jointly-trained BCDE parameters. To reduce overfitting in label-sparse settings, we additionally regularize the jointly-trained BCDE parameters toward the prior belief that x and y are independent.

Using our BCDE hybrid training framework, we show that **1)** BCDE significantly outperforms CVAE in fully supervised CDE, **2)** hybrid training is important for good conditional density estimation in

[*]Work was done at Adobe Research

both supervised and semi-supervised cases, and **3)** the independence prior is important when labeled data are sparse. Finally, we **4)** establish new benchmarks for semi-supervised CDE.

2 Methods

CVAE. In [14], the authors introduced the conditional version of variational autoencoders by considering a directed probabilistic model, where the latent variable z and data distribution of y are both conditioned on the input x. The conditional generative path is

$$z \sim p_\theta(z \mid x) = \mathcal{N}(z \mid \mu_\theta(x), \text{diag}(\sigma^2_\theta(x))), \tag{1}$$
$$y \sim p_\theta(y \mid x, z) = f(y; x, z, \theta), \tag{2}$$

where θ denotes the parameters of the neural networks $\mu_\theta(\cdot)$, $\sigma^2_\theta(\cdot)$, and the decoder $f(\cdot)$. The conditional generative model is trained using the auto-encoding variational Bayes (AEVB) algorithm [6, 11] by introducing the recognition model $q_\phi(z \mid x, y) = \mathcal{N}\Big(z \mid \mu_\phi(x, y), \text{diag}\big(\sigma^2_\phi(x, y)\big)\Big)$ and maximizing the amortized-variational lower-bound of the conditional log-likelihood.

BCDE. For the bottleneck conditional density estimator, we enforce the behavior that $X \perp Y \mid Z$. Consequently, z serves as a bottleneck in the generative path from x to y. Conditional training of BCDE (Fig. 1d) maximizes the variational lower-bound $\mathcal{C}(\cdot)$ as

$$\ln p_\theta(y \mid x) \geq \mathcal{C}(\theta, \phi; x, y) = \mathbb{E}_{q_\phi(z|x,y)}\left[\ln \frac{p_\theta(z \mid x) p_\theta(y \mid z)}{q_\phi(z \mid x, y)}\right]. \tag{3}$$

The BCDE is a conditional generative model (Fig. 1d). However, due to the bottleneck, it has a joint generative sibling (Figs. 1a to 1c), where the bottleneck variables z point to x and y. Thus, the variational lower-bound of the joint model is

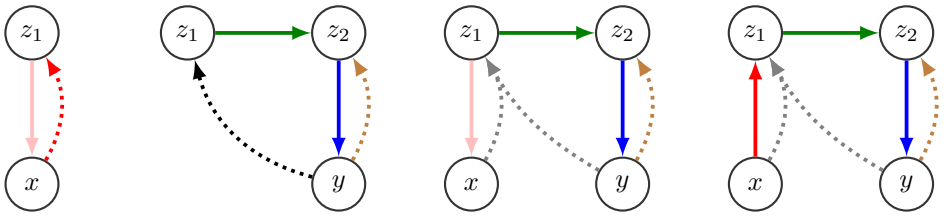
$$\ln p_{\theta'}(x, y) \geq \mathcal{J}_{xy}(\theta', \phi'; x, y) = \mathbb{E}_{q_{\phi'}(z|x,y)}\left[\ln \frac{p(z) p_{\theta'}(x \mid z) p_{\theta'}(y \mid z)}{q_{\phi'}(z \mid x, y)}\right]. \tag{4}$$

We use $\{\theta', \phi'\}$ to indicate the parameters of the joint model and reserve $\{\theta, \phi\}$ for the BCDE parameters. For samples where x or y is unobserved, we will need to compute the variational lower-bound for the marginal likelihoods. If x were to directly influence y, marginalizing out the unobserved x is potentially intractable. By exploiting the bottleneck, the conditional independence of x and y given z guarantees a tractable variational lower-bound for the marginal likelihoods

$$\ln p_{\theta'}(x) \geq \mathcal{J}_x(\theta', \phi'; x) = \mathbb{E}_{q_{\phi'}(z|x)}\left[\ln \frac{p(z) p_{\theta'}(x \mid z)}{q_{\phi'}(z \mid x)}\right], \tag{5}$$

$$\ln p_{\theta'}(y) \geq \mathcal{J}_y(\theta', \phi'; y) = \mathbb{E}_{q_{\phi'}(z|x)}\left[\ln \frac{p(z) p_{\theta'}(y \mid z)}{q_{\phi'}(z \mid y)}\right]. \tag{6}$$

Unlike the conditionally-trained BCDE, the jointly-trained BCDE imposes the constraint that the inference of z given x is sensitive to the distribution of x, since z takes on the additional task of reconstructing x. Therefore, we hypothesize that regularizing the conditionally-trained BCDE with the jointly-trained BCDE provides the following benefits: **1**) learning the distribution of x makes the



(a) Joint: (x_u) (b) Joint: (y_u) (c) Joint: (x_l, y_l) (d) Conditional: (x_l, y_l)

Figure 1: The joint and conditional components of a 2-layer BCDE. Dotted lines represent recognition models. Colors indicate the parameters that are strictly tied within the joint model and softly tied between the joint and conditional models.

inference of z given x robust to perturbations in the input, **2**) regularizing the conditional toward the joint encourages z to rely on representations that jointly encode x and y. In the semi-supervised regime, joint-regularization confers the additional benefit of influencing BCDE with unlabeled points.

We now describe our approach for regularizing the conditionally-trained BCDE with the jointly-trained BCDE. For notational simplicity, we denote the overall conditional and joint training objectives respectively as

$$\mathcal{C}(\theta, \phi) = \sum_{x,y \in L} \mathcal{C}(\theta, \phi; x, y), \tag{7}$$

$$\mathcal{J}(\theta', \phi') = \sum_{x,y \in L} \mathcal{J}_{xy}(\theta', \phi'; x, y) + \sum_{x \in U_x} \mathcal{J}_x(\theta', \phi'; x) + \sum_{y \in U_y} \mathcal{J}_y(\theta', \phi'; y), \tag{8}$$

where L is a dataset of paired (x, y) samples, and U_x and U_y are data sets of unpaired samples. To regularize the conditionally-trained BCDE, we introduce a hybrid training objective that softly ties the conditional to the joint,

$$\mathcal{H}(\theta, \phi, \theta', \phi') = \mathcal{C}(\theta, \phi) + \mathcal{J}(\theta', \phi') - \Lambda(\theta, \phi, \theta', \phi') - I(\theta'), \tag{9}$$

where Λ regularizes the conditionally-trained (θ, ϕ) toward the joint generatively-trained (θ', ϕ'), and I encourages independence between x and y by regularizing $p_{\theta'}(z_2 \mid z_1)$ toward the unit Gaussian. Details about how $\{\theta, \phi\}$ is tied to $\{\theta', \phi'\}$ is shown in Figure 1. It is important to note that $q(z_1|x)$ (Fig. 1a) is tied to $p(z_1|x)$ (Fig. 1d).

Note that Λ and I regularization can be achieved in a variety of ways. In our experiments, we implement Λ regularization by initializing the conditional model with the joint model parameters and performing early-stopping, and implement I regularization by initializing the joint model with auto-encoding parameters. In Figure 1c, this is done by severing the link between z_1 and z_2, and independently auto-encoding x with z_1 and y with z_2. Thus, independence prior requires $k \geq 2$ stochastic layers of z.

When $k \geq 2$, we build our recognition model using *top-down* inference [4]. Using 2-layer BCDE as an example, top-down inference allows us to exploit the conditional independence $Z_2 \perp X \mid Z_1$,

$$q(z_2, z_1 \mid x, y) = q(z_1 \mid x, y) q(z_2 \mid z_1, y), \tag{10}$$

$$q(z_2 \mid z_1, y) \propto p(z_2 \mid z_1) \ell(z_2 \mid y), \tag{11}$$

where $\ell(z_2 \mid y)$ is the normalized likelihood of z_2 given y. Thus, the distribution parameters for $q(z_2 \mid z_1, y)$ can be computed by combining $p(z_2 \mid z_1)$ and $\ell(z_2 \mid y)$ in the natural parameter space according to [4]. This not only simplifies the training of k-layer BCDE, the involvement of $p(z_2 \mid z_1)$ in the computation of $q(z_2 \mid z_1, y)$ also confers an interesting behavior: when the independence prior is applied to $p(z_2 \mid z_1)$, it also *directly* regularizes the variational approximation $q(z_2 \mid z_1, y)$.

3 Results

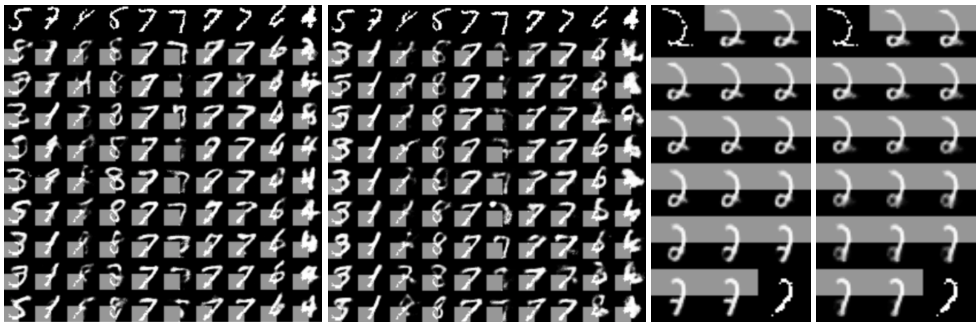
We chose the MNIST quadrant task as our benchmark task [13]. The number of unpaired x and y training samples is always kept at $50,000$, but number of labeled training data is reduced to $5,000$ for the semi-supervised task. The labeled data were randomly sub-sampled such that it was distributed evenly across classes. The MNIST digits were statically binarized by sampling the Bernoulli distribution [12]. The validation set is always kept at one-fifth of the size of the labeled training set. The test set is kept at $10,000$ as usual.

We compare hybridly-trained BCDE against CVAE as well as variants of Eq. (9): conditional training ($\mathcal{C}$ only), naïve pre-training (without J_{xy} and I), hybrid (without I), and hybrid + independence (full Eq. (9)). We optimized the various training objectives with *Adam* [5], using a learning rate of 0.001. Training was terminated using early-stopping on a validation set. We use multi-layered perceptrons (MLPs) for all neural networks in BCDE. All MLPs are batch-normalized [3] and parameterized with two hidden layers of 500 rectified linear units. All stochastic layers have 50 hidden units. The models were implemented in Python using Theano [15] and Keras [1].

Our results in the MNIST quadrant prediction tasks (Table 1) demonstrate that BCDE obtains state-of-the-art performance in supervised MNIST quadrant prediction (full 50k labels), and performs well

Models	Supervised 50k			Semi-sup. 5k labeled pairs		
	1-quad	2-quad	3-quad	1-quad	2-quad	3-quad
2-layer VAE (y only)	67.11	49.71	25.03	67.11	49.71	25.03
CVAE [14]	63.91	44.73	20.95	-	-	-
1-layer BCDE (conditional only)	64.58	45.26	20.91	77.28	54.17	24.83
2-layer BCDE (conditional only)	62.45	43.83	20.52	72.17	50.61	23.48
1-layer BCDE (naïve pre-training)	63.95	44.59	20.65	70.03	49.34	22.99
2-layer BCDE (naïve pre-training)	62.21	43.67	20.45	67.49	47.77	22.71
1-layer BCDE (hybrid)	63.64	44.34	20.43	69.64	49.35	23.21
2-layer BCDE (hybrid)	**61.79**	**43.14**	**19.93**	67.74	47.39	21.93
2-layer BCDE (hybrid+independence)	62.14	43.46	20.03	**66.72**	**46.81**	**21.63**

Table 1: We report the test set negative conditional log-likelihood scores for the MNIST quadrant prediction task [13]. In addition to the baselines (CVAE), we perform an ablation study of the BCDE objective Eq. (9) and consider its variants. Note that the 2-layer BCDE generally outperforms 1-layer BCDE due to having a more expressive variational family [4, 10].



(a) Hybrid (b) Conditional (c) Hybrid (d) Conditional

Figure 2: Comparison of conditional image generation for conditional versus hybrid 2-layer BCDE on the semi-supervised 1-quadrant task (2a-2b). Row 1 shows the original images. Rows 2-10 show nine attempts by each model to sample y according to x (the bottom-left quadrant, indicated in gray). Hybrid training yields better-looking samples and a higher-entropy conditional generative model (as can be seen from columns 1 and 2). Comparison of interpolation of y for hybrid versus conditional 2-layer BCDE on the fully-supervised top-down task (2c-2d). The top-region x shown in gray is used to predict the bottom-region y shown in black. Interpolation was performed in the space of z_1 and the resulting y was generated. The top-down task was chosen for easier visual comparison. To facilitate comparison, the interpolation of x was also created and shown in gray.

in the new semi-supervised task (with only 5k labels). The ablation study shows the importance of hybrid training, and of the independence prior in the semi-supervised case. Visual analysis reveals that hybrid-training produces a higher-entropy and more accurate generator (Figs. 2a-2b). Since hybrid training regularizes BCDE toward the joint model, which learns z_1 as a manifold of x, it is possible to interpolate between two different x's in the space of z_1 for the hybridly-trained BCDE and project the resulting y (Fig. 2c). The conditionally-trained BCDE, which does not explicitly learn a manifold of x, fails to generate a meaningful interpolation (Fig. 2d).

4 Conclusion

We have described a hybrid-training framework that regularizes a conditionally-trained neural variational conditional density estimator with a generatively-trained joint density estimator. When the number of labeled training data is small, we provide an additional regularizer that imposes an independence prior. When the independence prior is used in combination with top-down inference, we directly regularize both the generative and inference models. Our approach thus extends high-

dimensional conditional density estimation to the semi-supervised learning regime. Future work includes applying the proposed technique to other high-dimensional CDE tasks.

References

[1] C. François. Keras. `https://github.com/fchollet/keras`, 2016.

[2] M. P. Holmes, A. G. Gray, and C. L. Isbell. Fast Nonparametric Conditional Density Estimation. *ArXiv e-prints*, June 2012.

[3] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ArXiv e-prints*, Feb. 2015.

[4] C. Kaae Sønderby, T. Raiko, L. Maaløe, S. Kaae Sønderby, and O. Winther. Ladder Variational Autoencoders. *ArXiv e-prints*, 1602.02282, Feb. 2016.

[5] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *ArXiv e-prints*, Dec. 2014.

[6] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *ArXiv e-prints*, 1312.6114, Dec. 2013.

[7] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-Supervised Learning with Deep Generative Models. *ArXiv e-prints*, 1406.5298, June 2014.

[8] J. Lasserre, C. Bishop, and T. Minka. Principled hybrids of generative and discriminative models. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[9] L. Maaløe, C. Kaae Sønderby, S. Kaae Sønderby, and O. Winther. Auxiliary Deep Generative Models. *ArXiv e-prints*, 1602.05473, Feb. 2016.

[10] R. Ranganath, D. Tran, and D. M. Blei. Hierarchical Variational Models. *ArXiv e-prints*, 1511.02386, Nov. 2015.

[11] D. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *ArXiv e-prints*, 1401.4082, Jan. 2014.

[12] R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. *International Conference on Machine Learning*, 2008.

[13] K. Sohn, W. Shang, and L. H. Improved multimodal deep learning with variation of information. *Neural Information Processing Systems*, 2014.

[14] K. Sohn, X. Yan, and H. Lee. Learning structured output representation using deep conditional generative models. *Neural Information Processing Systems*, 2015.

[15] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, 1605.02688, May 2016.

[16] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2Image: Conditional Image Generation from Visual Attributes. *ArXiv e-prints*, 1512.00570, Dec. 2015.

[17] B. Zhang, D. Xiong, J. Su, H. Duan, and M. Zhang. Variational Neural Machine Translation. *ArXiv e-prints*, 1605.07869, May 2016.