Bayesian Semi-supervised Learning with Deep Generative Models

Jonathan Gordon
Department of Engineering
Cambridge University
jg801@cam.ac.uk

José Miguel Hernández-Lobato
Department of Engineering
Cambridge University
jmh233@cam.ac.uk

Abstract

Neural network based generative models with discriminative components are a powerful approach for semi-supervised learning. However, these techniques a) cannot account for model uncertainty in the estimation of the model's discriminative component and b) lack flexibility to capture complex stochastic patterns in the label generation process. To avoid these problems, we first propose to use a discriminative component with stochastic inputs for increased noise flexibility. We show how an efficient Gibbs sampling procedure can marginalize the stochastic inputs when inferring missing labels in this model. Following this, we extend the discriminative component to be fully Bayesian and produce estimates of uncertainty in its parameter values. This opens the door for semi-supervised Bayesian active learning.

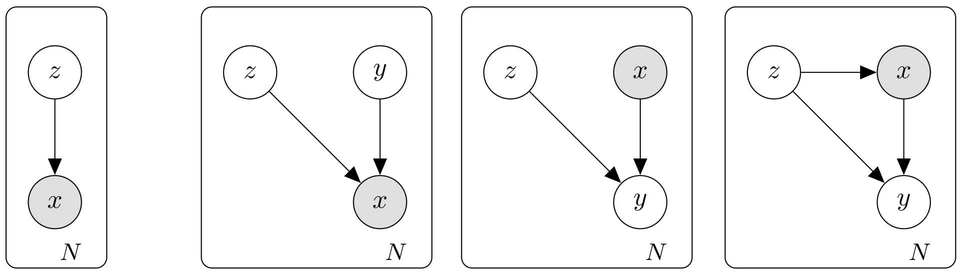
1 Introduction

Deep generative models (DGMs) can model complex high dimensional data via the use of latent variables. Recently, advances in variational training procedures such as stochastic backpropagation [17] and the reparametrization trick [9] have made training these models feasible and efficient. DGMs are particularly powerful when neural networks are used to parameterize generative distributions and inference networks, leading to the Variational Autoencoder (VAE, Figure 1a) [17, 9].

The ability to efficiently train such models has led to a plethora of interesting extensions, increasing the flexibility of posterior approximations, expressiveness, and capabilities of the models [2, 19, 12, 16]. An important extension to standard VAEs is for semi-supervised learning [10, 12], which incorporates labels into the generative model of the inputs (Figure 1b), extending the VAE to semi-supervised learning tasks. In this setting, the labels are treated as latent variables that influence the generative process for the inputs and a recognition network is then used as a discriminative component to infer missing labels.

One major drawback of these previous approaches for semi-supervised learning with VAEs is that after training, the generative model is discarded and the recognition network is the only element used as a discriminative component [10, 12]. This is unsatisfactory from a modeling perspective as the recognition network is just a tool for performing approximate inference, and cannot be used to quantify model uncertainty.

Closely related to deep generative models are Bayesian neural networks (BNNs) [14]. BNNs extend standard neural networks and explicitly model the uncertainty in the learned weights increasing robustness and opening the door to tasks requiring uncertainty such as active learning [5, 6] and Bayesian optimization [18]. Blundell et al. [1] extend ideas from stochastic variational inference [9, 15] to an efficient inference procedure for BNNs. Further, Depeweg et al. [3] show how learning can be extended to general α-divergence minimization [7, 13] and provide empirical evidence of the



(a) Standard VAE (b) Conditional VAE (c) BNN - stochastic inputs (d) Proposed model

Figure 1: Graphical model depiction for VAE and BNN based models and proposed approach.

benefit of introducing stochastic inputs (Figure 1c). However, in these works the stochastic inputs are of low (typically one) dimension.

We extend these ideas and develop a deep generative model with a discriminative component given by a BNN with stochastic inputs to accommodate semi-supervised learning. Our motivation is that after training, this BNN can be used to infer missing labels, rather than using the inference networks. This allows us to fully quantify any modeling uncertainty in the predictions of our discriminative component. We introduce two recognition networks and demonstrate how they can be used for training and prediction, as well as for posterior inference of high-dimensional stochastic inputs at prediction time. Our goal is to use the proposed method for semi-supervised Bayesian active learning.

2 Related Work

DGMs have recently shown to be very effective in semi-supervised learning tasks [10, 12], achieving state-of-the-art performance on a number of benchmarks. Our model is most similar to the work detailed by Kingma et al. [10]. However, since our discriminative component is a Bayesian neural network with random inputs, we use a slightly different inference network architecture.

Similarly, Bayesian deep learning has recently been shown to be highly effective in active learning regimes [6, 5]. In contrast to these works, the proposed model can perform semi-supervised and active learning simultaneously, which may lead to significant improvements. Another difference is that while Gal et al. [5] use dropout as a proxy for Bayesian inference [4] and Hernández-Lobato and Adams [6] use a technique called probabilistic backpropagation, we propose leveraging variational inference to explicitly model the weight uncertainty [1].

The proposed model builds on ideas from both DGMs and Bayesian deep networks to suggest a principled method for simultaneous semi-supervised and active learning.

3 Deep Generative Model of Labels

We propose extending the model developed by Depeweg et al. [3] (as in Fig. 1d) and including an inference network for z, allowing scalability of the latent dimension. Further, we propose inference procedures to allow this model to be used for semi-supervised learning, similarly to Kingma et al. [10], Maaløe et al. [11].

There are a few motivations for our approach: (i) it builds on the idea of VAEs, but does so in a manner that results in an explicit probabilistic model for the labels, (ii) it extends BNNs with stochastic inputs to include inference networks for the stochastic inputs, allowing generalizing these to high dimensional variables, and (iii) it naturally accommodates semi-supervised and active learning with the generative model. The generative model can be described as:

$$p(z) = \mathcal{N}(z; \mathbf{0}, \boldsymbol{I}) \ , \tag{1}$$
$$p_\theta(x|z) = \mathcal{N}(x; \mu_x, \nu_x) \ , \tag{2}$$
$$p_\theta(y|z, x) = \text{Cat}(y; \pi_y) \ , \tag{3}$$

where we parameterize the distributions of x, y with deep neural networks:

$$\mu_x = NN_x(z, \theta) \, , \qquad \log \nu_x = NN_x(z, \theta) \, , \tag{4}$$
$$\pi_y = NN_y(z, x, \theta) \, , \tag{5}$$

where NN_x and NN_y have weights $\mathcal{W}_x$ and $\mathcal{W}_y$ respectively, and $\theta = \{\mathcal{W}_x, \mathcal{W}_y\}$.

3.1 Variational Training of the Model

We propose a variational approach for training the model with both labeled and unlabeled data. In this section we are interested in point estimates of θ and Bayesian inference for z. We first derive the variational lower bound. In the semi-supervised setting, there are two lower bounds (ELBOs), for the labeled and unlabeled case.

3.1.1 Labeled Data ELBO

Following recent advances in variational inference [9, 17], we introduce inference networks $q_\phi(z|x, y)$ to approximate the intractable posterior distribution. Taking expectations w.r.t. q of the log likelihoods we have:

$$\log p_\theta(x, y) \geq \mathbb{E}_{q_{\phi(z|x,y)}} [\log p_\theta(x|z) + \log p_\theta(y|x, z)] - \mathrm{D}_{\mathrm{KL}} (q_\phi(z|x, y) || p(z)) = \mathcal{L}^l(x, y; \theta, \phi) \, , \tag{6}$$

where $q_\phi(z|x, y)$ is a recognition network parameterized by ϕ. The lower-bound contains a term for the likelihood associated with the pair of variables x and y, and a regularization term for the inference network. We can approximate expectations w.r.t. $q_\phi(z|x, y)$ via Monte Carlo estimation:

$$\mathcal{L}^l(x, y; \theta, \phi) \approx \frac{1}{L} \sum_{l=1}^{L} \left[\log p_\theta(x, y|z^l) - \ \log q_\phi(z^l|x) + \log p(z^l) \right] \ , \tag{7}$$

with $z^l \sim q_\phi(z|x, y)$, and for Gaussian recognition networks the KL term can be evaluated analytically.

3.1.2 Unlabeled Data ELBO

We can follow a similar approach to derive the lower bound for the unlabeled case. In this setting, we have:

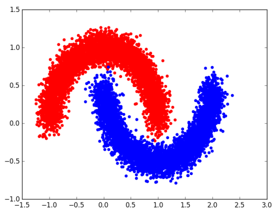
$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(y|x)} \left[\mathcal{L}^l(x, y; \theta, \phi) \right] + \mathcal{H} \left[q_\phi(y|x) \right] \mathcal{L}^u(x; \theta, \phi) \, , \tag{8}$$

where we have used the decomposition $q_\phi(z, y|x) = q_\phi(y|x) q_\phi(z|x, y)$, and $\mathcal{H}[\cdot]$ computes the entropy of a probability distribution. Thus, we have the two recognition networks $q_\phi(z|x, y)$ and $q_\phi(y|x)$. This form of the lower bound has data fit terms for both x and y in the generative model, as well as regularization terms for both recognition networks $q_\phi(z|x, y)$ and $q_\phi(y|x)$.

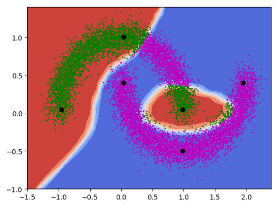
The recognition network $q_\phi(z|x, y)$ is shared by both the unlabeled and labeled objectives. The recognition network $q_\phi(y|x)$ is unique to the unlabeled data. Following the work in [10, 12], we add a weighted term to the final objective function to ensure that $q_\phi(y|x)$ is trained on all data such that

$$\mathcal{L}(\theta, \phi) = \sum_{(x,y) \sim \tilde{p}_l} \mathcal{L}^l(x, y; \theta, \phi) + \sum_{x \sim \tilde{p}_u} \mathcal{L}^u(x; \theta, \phi) + \alpha \mathbb{E}_{(x,y) \sim \tilde{p}_l} \left[\log q_\phi(y|x) \right] \ , \tag{9}$$

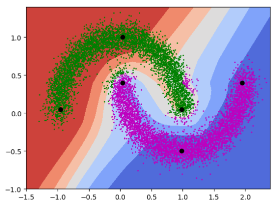
where α is a small positive constant which is initialized in a similar way as in [10], $\tilde{p}_l$ is the empirical distribution of labeled points and $\tilde{p}_u$ is the empirical distribution of unlabeled points.



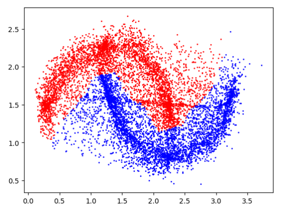
(a)



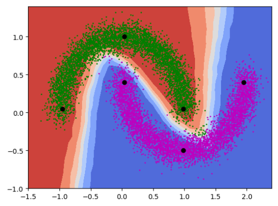
(b)



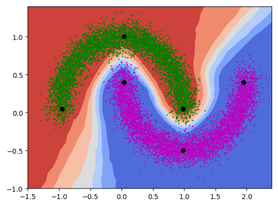
(c)



(d)



(e)



(f)

Figure 2: Preliminary experiments carried with the model. (a) Complete set of labeled data. (b) Contour plots learned by a standard DNN using only six labeled labeled data. (c) Contour plots learned by M2 [10] using only the depicted points with labels (the rest unsupervised). (d) Samples from the generative model after training. (e) Contour plots learned by the model using only the depicted points with labels (the rest unsupervised). (f) Contour plots learned with Bayesian training of the model.

3.2 Discrete Outputs

Optimizing Eq. (9) is straightforward when y is continuous as we can approximate expectations by sampling from $q_\phi(z|x, y)$ and $q_\phi(y|x)$ and using stochastic backpropagation and the reparameterization trick [9, 17] to yield differentiable estimators.

Despite recent efforts [8], reparameterization for discrete variables is as yet not a well-understood process. In the case where y is a discrete variable, that is, $y \in \{1, ..., K\}$, optimization of $\mathcal{L}^u$ requires approximating expectations w.r.t. $q_\phi(y|x)$. Rather than using Monte-Carlo approximations for this, we propose to directly compute the expectation by summing over the possible values of y:

$$\mathbb{E}_{q_\phi(y|x)}\left[f(y, z, x)\right] = \sum_{y \in \mathcal{Y}} q_\phi(y|x) f(x, y, z)\,. \tag{10}$$

Substituting this for the relevant terms in Eq. (8) yields:

$$\mathbb{E}_{q_\phi(y|x)}\left[\log p_\theta(x, y|z)\right] = \sum_{y \in \mathcal{Y}} q_\phi(y|x) \mathbb{E}_{q_\phi(z|x,y)}\left[\log p_\theta(x, y|z)\right] \tag{11}$$

and

$$\mathbb{E}_{q_\phi(y|x)}\left[\mathrm{D}_{\mathrm{KL}}(q_\phi(z|x, y)||p(z))\right] = \sum_{y \in \mathcal{Y}} q_\phi(y|x) \mathrm{D}_{\mathrm{KL}}\left(q_\phi(z|x, y)||p(z)\right)\,. \tag{12}$$

Taking explicit expectations w.r.t. $q_\phi(y|x)$ rather than Monte-Carlo approximations allows us to extend the training procedure to cases where y is discrete.

3.3 Introducing Model Uncertainty

A major advantage of this approach is that it enables us to express model uncertainty in the discriminative component $NN_y(z, x, \theta)$ by computing a posterior distribution on the weights W_y. For this, we consider the following prior and likelihood functions:

$$p_\theta(W_y) = \mathcal{N}(W_y; 0, \mathbf{I}) , \tag{13}$$

$$p_\theta(y|x, z, W_y) = \text{Cat}(y; \pi_y) , \tag{14}$$

where $\pi_y = NN_y(z, x, W_y)$ is parameterized by a neural network with weights W_y. Assuming a single labeled data point, the posterior distribution for the latent variables is:

$$p_\theta(W_y, z|x, y) = \frac{p_\theta(x, y|z, W_y)p(z)p(W_y)}{p(x, y)} . \tag{15}$$

This posterior distribution is intractable. Following the work by Blundell et al. [1], Depeweg et al. [3], we introduce an approximate posterior distribution for W_y given by $q_\phi(W_y) = \mathcal{N}(W_y; \mu_w, \sigma_w^2)$ with σ_w^2 being a diagonal covariance matrix. Note that we are assuming here that W_y is *a posteriori* independent of z. We also assume *a posteriori* independence between W_y and y in the unlabeled case. Re-deriving the lower bound for the case of one single labeled data point yields:

$$\begin{aligned}\mathcal{L}(x, y; \theta, \phi) = \mathbb{E}_{q_\phi(z,|x,y)q_\phi(W_y)} &\left[\log p_\theta(x, y|z, W_y)\right] - \\ &\text{D}_{\text{KL}}\left(q_\phi(z|x, y)\|p(z)\right) - \\ &\text{D}_{\text{KL}}\left(q_\phi(W_y)\|p(W_y)\right) .\end{aligned} \tag{16}$$

The corresponding derviation for the unlabeled lower bound can be obtained from Eq. (8) in a similar manner. We follow the work presented by Blundell et al. [1], and optimize the objective functions applying reparameterization to the weights W_y as well as z.

3.4 Prediction with the Model

To approximate $p(y_\star|x_\star)$ for a new example $x_\star$, we have to integrate $p_\theta(y_\star|x_\star, z, W_y)$ with respect to the posterior distribution on z and W_y. For this, W_y is sampled from $q_\phi(W_y)$, while z is sampled from the recognition network $q_\phi(z|x_\star, y_\star)$. Since this recognition network requires $y_\star$, we use a Gibbs sampling procedure, drawing the first sample of $y_\star$ from the recognition network $q_\phi(y_\star|x_\star)$. In particular,

$$\begin{aligned} y_\star^{(0)} &\sim q_\phi(y_\star|x_\star) , \\ &\left.\begin{aligned} W_y^{(\tau)} &\sim q_\phi(W_y) , \\ z^{(\tau)} &\sim q_\phi(z|x_\star, y_\star^{(\tau-1)}) , \\ y_\star^{(\tau)} &\sim p_\theta(y_\star|x_\star, z^{(\tau)}, W_y^{(\tau)}) , \end{aligned}\right\} \text{for } \tau = 1, \ldots, T. \end{aligned}$$

with the final prediction being an average over the samples. Using $q_\phi(y_\star|x_\star)$ to initialize the procedure increases efficiency and negates the need for a burn in period. In our experiments $T = 10$ produced good results.

4 Preliminary Results

In this section we detail preliminary results achieved by the proposed model. We experiment with toy data similar to that used by Maaløe et al. [12]. The data consists of 1e4 training and test samples, generated from a deterministic function with additive Gaussian noise (Figure 2a). A small set is selected as the labeled data, and the rest are unlabeled. We compare performance to that of a feed forward neural network.

All neural networks have two hidden layers with 128 neurons, and $z \in \mathbb{R}^5$. We set $\alpha = 0.1 * N_l$, where N_l is the number of labeled data points, and use RELU activations for all hidden layers.

The model converges on 100% accuracy in all cases for $N_l > 10$. When examining training curves for different values of N_l (not shown due to space constraints), we see that as the labeled set is larger, training converges faster and to better lower bounds.

Figures 2b, 2e, 2f show, respectively, the predictive probabilities learned by a deep neural network (DNN) which ignores the unlabeled data, the proposed approach for Semi-Supervised Learning using

Table 1: Test accuracy and log-likelihood for each method.

	DNN	SSLPE	SSLAPD
LOG-LIKELIHOOD	-1.20	-0.07	-0.01
ACCURACY	83.6	99.2	99.7

a Point Estimate for W_y (SSLPE) and the proposed approach for Semi-Supervised Learning using an Approximate Posterior Distribution for W_y (SSLAPD). We use $N_l = 6$ in all cases: three labeled examples selected from each class, in a similar manner to Maaløe et al. [12]. Table 1 shows the average test log-likelihood and predictive accuracy obtained by each method. DNN overfits the labeled data and achieves low predictive accuracy and test log-likelihood. SSLPE is able to leverage the unlabeled data to learn a smoother decision boundary that aligns with the data distribution, achieving much better predictive performance than DNN. Finally, SSLAPD makes predictions similar to those of SSLPE but with higher uncertainty in regions far away from the training data. Overall, SSLAPD is the best performing method. Finally, Figure 2d shows samples generated by SSLAPD, indicating this method has learned a good generative process and is able to synthesize compelling examples.

5 Discussion and Future Work

DGMs have been successfully applied to semi-supervised learning tasks, though by discarding the model and using only an inference network for predicting labels. This approach does not allow to account for model uncertainty.

In contrast, the proposed approach uses a Bayesian neural network for label prediction. In addition to being more satisfying from a modeling perspective, this opens the door to joint semi-supervised and active learning by accounting for model uncertainty.

Our experiments show that the proposed approach is promissing and able to produce wider confidence bands far away from the training data than alternative methods that ignore parameter uncertainty. Further experiments with alternative datasets such as MNIST are required. Future work includes developing methods for acquiring new samples from a pool set, performing joint semisupervised and active learning, and comparing with recent benchmark methods.

References

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[2] Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[3] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. *arXiv preprint arXiv:1605.07127*, 2016.

[4] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *arXiv preprint arXiv:1506.02142*, 2, 2015.

[5] Y. Gal, R. Islam, and Z. Ghahramani. Deep bayesian active learning with image data. *arXiv preprint arXiv:1703.02910*, 2017.

[6] J. M. Hernández-Lobato and R. P. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *ICML*, pages 1861–1869, 2015.

[7] J. M. Hernández-Lobato, Y. Li, M. Rowland, D. Hernández-Lobato, T. D. Bui, and R. E. Turner. Black-box α-divergence minimization. 2016.

[8] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with Gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[9] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[10] 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.

[11] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Improving semi-supervised learning with auxiliary deep generative models. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2015.

[12] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

[13] T. M. Moerland, J. Broekens, and C. M. Jonker. Learning multimodal transition dynamics for model-based reinforcement learning. *arXiv preprint arXiv:1705.00470*, 2017.

[14] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[15] R. Ranganath, S. Gerrish, and D. M. Blei. Black box variational inference. In *AISTATS*, pages 814–822, 2014.

[16] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[17] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[18] J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, M. M. A. Patwary, M. Prabhat, and R. P. Adams. Scalable bayesian optimization using deep neural networks. In *ICML*, pages 2171–2180, 2015.

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