Improving Semi-Supervised Learning with Auxiliary Deep Generative Models

Lars Maaløe[1]
larsma@dtu.dk

Casper Kaae Sønderby[2]
casper.sonderby@bio.ku.dk

Søren Kaae Sønderby[2]
soren.sonderby@bio.ku.dk

Ole Winther[1,2]
olwi@dtu.dk

[1]Department of Applied Mathematics and Computer Science, Technical University of Denmark
[2]Bioinformatics Centre, Department of Biology, University of Copenhagen

Abstract

Deep generative models based upon continuous variational distributions parameterized by deep networks give state-of-the-art performance. In this paper we propose a framework for extending the latent representation with extra auxiliary variables in order to make the variational distribution more expressive for semi-supervised learning. By utilizing the stochasticity of the auxiliary variable we demonstrate how to train discriminative classifiers resulting in state-of-the-art performance within semi-supervised learning exemplified by an 0.96% error on MNIST using 100 labeled data points. Furthermore we observe empirically that using auxiliary variables increases convergence speed suggesting that less expressive variational distributions, not only lead to looser bounds but also slower model training.

1 Introduction

Deep neural networks have recently showed impressive performance on a wide range of tasks. These improvements have been achieved by better algorithms, faster computers and increased availability of large labeled datasets in many areas, e.g. image recognition (Deng et al., 2009). In many practical situations it is relatively inexpensive to acquire data but expensive to label it. This makes semi-supervised learning attractive. There exist many approaches to performing semi-supervised learning, e.g. *transductive SVM* (TSVM) (Joachims, 1999), *EM* methods (Nigam et al., 2000), graph-based methods (Blum and Chawla, 2001; Zhu et al., 2003; Pitelis et al., 2014) and deep auto-encoders (Rifai et al., 2011; Ranzato and Szummer, 2008; Weston et al., 2008).

Recently several different (deep) models have significantly improved the performance on semi-supervised classification tasks. Kingma et al. (2014) introduced a deep generative model for semi-supervised learning (DGM) by augmenting the *auto-encoding variational Bayes* (AEVB) model (Kingma, 2013; Rezende et al., 2014) algorithm with labeled units. Miyato et al. (2015) introduced an improved semi-supervised learner by applying adversarial training to deep networks. Finally Rasmus et al. (2015) introduced a generalization of the ladder network (Valpola, 2014) that has the ability to learn a latent classification variable.

In this article we introduce the *auxiliary deep generative model* (ADGM) and apply it to semi-supervised learning. In the ADGM, the variational encoder model has an extra set of stochastic variables compared to the generative decoder model. These extra so-called auxiliary variables makes the variational model more flexible and thus able to improve the variational lower bound (Agakov and Barber, 2004). The auxiliary variable and the input data is fed into a variational auto-encoder and a discriminative classifier. Empirically we show that the ADGM, (i) obtain state-of-the-art results on semi-supervised classification, (ii) is trainable end-to-end without the need for any pre-training, (iii) have good convergence properties and (iv) that its stochastic auxiliary variable is essential for good discriminative classification.

2 Methods

Kingma et al. (2014) introduced a probabilistic approach to semi-supervised learning by stacking a generative feature extractor (called M1) and a generative semi-supervised model (M2) into a stacked generative semi-supervised model (M1+M2). M1 is a variational auto-encoder, where the generative model is defined as $p_\theta(z)p_\theta(x|z)$ (*decoder* with parameters θ), with the variational approximation being $q_\phi(z|x)$ (*encoder* with parameters ϕ), as replacement for the intractable posterior $p_\theta(z|x)$ (Kingma, 2013). M2 includes labels y in the generative model: $p_\theta(x|y,z)p_\theta(z)p_\theta(y)$, decoder $q_\phi(z|y,x)$ and a discriminative classifier, $q_\phi(y|x)$. The generative model p combining M1 and M2 called M1+M2 is (cf. fig. 1a):

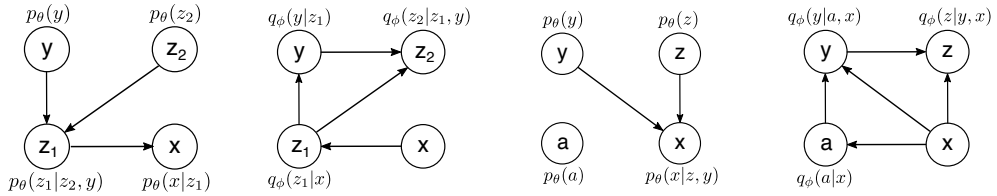
$$\text{M1} : p_\theta(x|z_1) = f(x; z_1, \theta), \tag{1}$$

$$\text{M2} : p(y) = \text{Cat}(y|\pi); \quad p(z_2) = \mathcal{N}(z_2|0, \text{I}); \quad p_\theta(z_1|y, z_2) = f(z_1; y, z_2, \theta)\ , \tag{2}$$

where $f(\cdot)$ are the decoders and $\text{Cat}(\cdot)$ is the multinomial distribution. The corresponding inference model Q is (cf. fig. 1b):

$$\text{M1} : \quad q_\phi(z_1|x) = \mathcal{N}(z_1|\mu_\phi(x), \text{diag}(\sigma^2_\phi(x))), \tag{3}$$

$$\text{M2} : \quad q_\phi(z_2|y, z_1) = \mathcal{N}(z_2|\mu_\phi(y, z_1), \text{diag}(\sigma^2_\phi(y, z_1))); \quad q_\phi(y|z_1) = \text{Cat}(y|\pi_\phi(z_1)). \tag{4}$$



(a) M1+M2 P model. (b) M1+M2 Q model. (c) ADGM P model. (d) ADGM Q model.

Figure 1: Probabilistic graphical models of M1+M2 and ADGM.

Since latent variables z_2 and y in M2 are marginally independent, the class specific information can be modeled through y and remaining information through z_2.

However, although both M2 and M1+M2 should be powerful generative models for semi-supervised learning direct application of these models *failed to deliver good results* in benchmarks. Kingma et al. (2014) reported $11.97\%(\pm 1.71)$ classification error for MNIST with 100 labeled examples and no results were reported for M1+M2. Instead Kingma et al. (2014) trained M1 to get latent features z_1 that was then used as input for training M2.

In this contribution we propose an alternative formulation with two sets of stochastic variables that converges end-to-end and achieves state-of-the-art performance. The ADGM has the generative model p defined as $p_\theta(a)p_\theta(y)p_\theta(z)p_\theta(x|y,z)$ (cf. fig. 1c), where a, y, z are the auxiliary variable, class label, and latent features, respectively. Learning the posterior distribution is intractable, thus we define the approximation as $q_\phi(a|x)q_\phi(z|y,x)$ and a classifier $q_\phi(y|a,x)$ (cf. fig. 1d). The distributions of the generative model p are

$$\begin{aligned} &p_\theta(x|z,y) = f(x; z, y, \theta); \quad &&p(z) = \mathcal{N}(z|0, \text{I}); \\ &p(y) = \text{Cat}(y|\pi); &&p(a) = \mathcal{N}(a|0, \text{I}). \end{aligned} \tag{5}$$

And for the corresponding inference model q

$$\begin{aligned} q_\phi(a|x) &= \mathcal{N}(a|\mu_\phi(x), \text{diag}(\sigma^2_\phi(x))), \\ q_\phi(z|y,x) &= \mathcal{N}(z|\mu_\phi(y,x), \text{diag}(\sigma^2_\phi(y,x))), \\ q_\phi(y|a,x) &= \text{Cat}(y|\pi_\phi(a,x))\ . \end{aligned} \tag{6}$$

The key point of the ADGM is that the auxiliary unit a introduces a class specific latent distribution between x and y allowing a more expressive discriminative distribution $q(y|a,x)$. Further the stochasticity of a maps each input into a distribution $q(a|x)$ used for the discriminative classifier, which is richer than a deterministic dependency between x and y (Agakov and Barber, 2004). Note that it is possible to let a be conditioned on x, y, z in the generative model, but we found that this did not improve the performance. The ADGM use multi-layered perceptrons (MLP) to model $q_\phi(a|x)$, $q_\phi(z|y,x)$, $q_\phi(y|a,x)$ and $p_\theta(x|z,y)$.

For Gaussian distributions we apply the *reparameterization* trick, introduced in (Kingma, 2013) to backpropagate the error signal through the latent variables. We approximate the expectations by drawing unbiased Monte Carlo (MC) estimates $\tilde{a} \sim q_\phi(a|x)$ and $\tilde{z} \sim q_\phi(z|y,x)$.

Variational Lower Bound

We optimize the model by maximizing the lower bound on the likelihood. The variational lower bound on the marginal likelihood for a single *labeled data point* is

$$\begin{aligned}\log p_\theta(x,y) &\geq \mathbb{E}_{q_\phi(a,z|x,y)}[\log p_\theta(a)p_\theta(y)p_\theta(z)p_\theta(x|y,z)] \\ &\quad - \mathit{KL}[q_\phi(a|x)q_\phi(z|y,x)||p_\theta(z)p_\theta(a)p_\theta(y)] \\ &= \mathbb{E}_{q_\phi(a,z|x,y)}[\log[p_\theta(a)p_\theta(y)p_\theta(z)p_\theta(x|y,z)] \\ &\quad - \log[q_\phi(a|x)q_\phi(z|y,x)]] = -\mathcal{L}(x,y)\ .\end{aligned} \tag{7}$$

For unlabeled data we further marginalize y:

$$\log p_\theta(x) \geq \sum_y q_\phi(y|a,x)(-\mathcal{L}(x,y)) + \mathcal{H}(q_\phi(y|a,x)) = -\mathcal{U}(x), \tag{8}$$

where $\mathcal{H}(\cdot)$ is the entropy. The discriminative classifier $q_\phi(y|a,x)$ (6) is included in the objective $-\mathcal{U}(x,y)$, but not in $-\mathcal{L}(x,y)$. Since the classification loss is not part of the labeled data x_l, y_l lower bound we introduce:

$$-\mathcal{L}_l(x_l,y_l) = -\mathcal{L}(x_l,y_l) - \alpha \cdot \mathbb{E}_{q_\phi(a,z|x_l,y_l)}[-\log q_\phi(y_l|a,x_l)], \tag{9}$$

where α is a weight between generative and discriminative learning. The variational lower bound for labeled x_l, y_l and unlabeled data x_u is

$$\mathcal{J} = \sum_{(x_l,y_l)} \mathcal{L}_l(x_l,y_l) + \sum_{(x_u)} \mathcal{U}(x_u)\ . \tag{10}$$

3 Results

Table 1 shows that the ADGM outperforms all previously proposed models on the MNIST dataset. The model convergence to around 1.5% is fast, and from that point the convergence speed declines (cf. Fig. 2a). In Fig. 2b we visualize 10 Gaussian distributed random samples conditioned on each class y.

	100 labels
AtlasRBF (Pitelis et al., 2014)	8.10% ($\pm$0.95)
Deep Generative Model (M1+M2) (Kingma et al., 2014)	3.33% ($\pm$0.14)
Virtual Adversarial (Miyato et al., 2015)	2.12%
Ladder (Rasmus et al., 2015)	1.06% ($\pm$0.37)
Auxiliary Deep Generative Model (1 MC)	2.25% ($\pm$ 0.08)
Auxiliary Deep Generative Model (10 MC)	**0.96% ($\pm$ 0.02)**

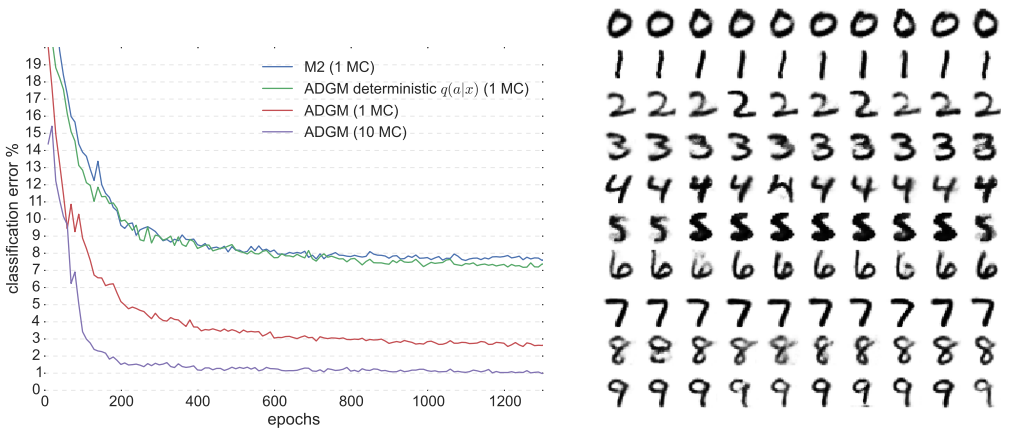
Table 1: Semi-supervised benchmarks on MNIST for 100 randomly labeled data points. The ADGM was trained by performing 1 and 10 Monte Carlo samples.

Fig. 3 shows the information contribution from the auxiliary units a and the latent units z. Like in Burda et al. (2015), the number of *contributing*/activated units in z is quite low ~ 20. The number of contributing auxiliary units a, on the other hand, is much larger. We speculate that this is due to the upweighting of the discriminative classification in the lower bound. Fig. 2a shows how the ADGM outperforms a similarly optimized M2 model and an ADGM where the auxiliary unit is deterministic. We found that convergence of the M2 model was highly unstable. The result in Fig. 2a is the far best achieved.

Implentation details

The ADGM is implemented in Python using Theano (Bastien et al., 2012) and Lasagne (Dieleman et al., 2015) libraries[1]. For training, we have used the *Adam* (Kingma and Ba, 2014) optimization framework with a learning rate of 3e-4 and exponential decay rate for the 1st and 2nd moment at 0.9 and 0.999 respectively. The learning rate was annealed by .75 every 200 epochs. α was defined

[1]Implementation will be made available in an extension framework to the Lasagne library named Parmesan on `https://github.com/larsmaaloee/auxiliary-deep-generative-models`.



(a) (b)

Figure 2: (a) 100 MC classification errors for MNIST test set on Kingma et al. (2014) M2 trained with 1 MC, ADGM with deterministic auxiliary units trained with 1 MC, ADGM trained with 1 MC and ADGM trained with 10 MC. All models were trained with the same hyperparameters. (b) 100 Gaussian distributed random samples drawn from a 100-dimensional z with y fixed to each class.

as $0.05 \cdot N$, where N is the number of unlabeled data points. We parametrized the MLPs with either one or two layers of 500 hidden units using rectified linear units. All stochastic layers used 100 linear hidden units to parameterize μ and $log(\sigma^2)$. The weights and biases are initialized using Glorot and Bengio (2010) scheme. We used 1 or 10 MC samples for training and 100 MC samples for evaluation.

We used MNIST as benchmark. In order to make a fair comparison with the ladder network, we have combined the training set of 50000 examples with the validation set of 10000 examples. The test set of 10000 remained as is. We used a batch size of 200 with the first half of the batch always being the 100 labeled samples. The labeled data are chosen randomly, but distributed evenly across classes. Before each epoch the normalized MNIST images were binarized by sampling Bernoulli distributions.

All experiments were carried out on GeForce GTX TITAN X GPUs. With 10 Monte Carlo samples the runtime was ∼64 s/epoch. The training converged to around 1.5% classification error in 200 epochs corresponding to 5 hours. The number of samples influences runtime and stability of training. With 1 Monte Carlo sample an epoch takes ∼16 s/epoch but in terms of predictive performance increasing the number of samples helped (cf. Fig. 2a).

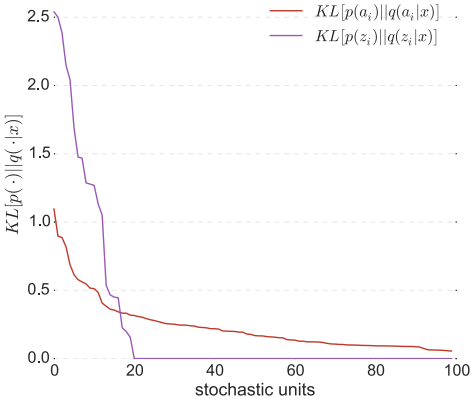


Figure 3: Number of active stochastic units for an ADGM trained on MNIST 100 labels. We compute $KL\left[p(a_i)||q(a_i|x)\right]$ and $KL\left[p(z_i)||q(z_i|x)\right]$ for each stochastic unit. A number close to zero indicates that $q(\cdot|x) \approx p(\cdot)$.

4 Conclusion

We have shown that making the discriminative distribution more flexible by introducing extra auxiliary variables gives state-of-the-art performance on the 100 labeled examples MNIST benchmark. We are in the progress of extending this to other semi-supervised scenarios. It is also of interest to extend this approach to both fully unsupervised and supervised generative settings. Currently we are combing the proposed framework with the new tighter bound by Burda et al. (2015), where a tight bound on $p(y, x)$ may be used directly for classification through $p(y|x) \propto p(y, x)$.

Acknowledgements

We thank Durk P. Kingma for helpful discussions. This research was supported by the Novo Nordisk Foundation and NVIDIA Corporation with the donation of TITAN X and Tesla K40 GPUs.

References

Agakov, F. and Barber, D. (2004). An Auxiliary Variational Method. In Pal, N., Kasabov, N., Mudi, R., Pal, S., and Parui, S., editors, *Neural Information Processing*, volume 3316 of *Lecture Notes in Computer Science*, pages 561–566. Springer Berlin Heidelberg.

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I. J., Bergeron, A., Bouchard, N., and Bengio, Y. (2012). Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Blum, A. and Chawla, S. (2001). Learning from Labeled and Unlabeled Data Using Graph Mincuts. In *Proceedings of the 18th International Conference on Machine Learning*, ICML '01, pages 19–26, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Burda, Y., Grosse, R., and Salakhutdinov, R. (2015). Importance Weighted Autoencoders. *arXiv preprint arXiv:1509.00519*.

Deng, J., Dong, W., Socher, R., jia Li, L., Li, K., and Fei-fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR09*.

Dieleman, S., Schlüter, J., Raffel, C., Olson, E., and Sønderby, S. K. (2015). Lasagne: First release.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics*.

Joachims, T. (1999). Transductive Inference for Text Classification Using Support Vector Machines. In *Proceedings of the 16th International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kingma, D. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., Rezende, D. J., Mohamed, S., and Welling, M. (2014). Semi-Supervised Learning with Deep Generative Models. *arXiv preprint arXiv:1406.5298*.

Kingma, Diederik P; Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*.

Miyato, T., Maeda, S.-i., Koyama, M., Nakae, K., and Ishii, S. (2015). Distributional Smoothing with Virtual Adversarial Training. *arXiv preprint arXiv:1507.00677*.

Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. (2000). Text Classification from Labeled and Unlabeled Documents Using EM. *Machince Learning*, 39(2-3):103–134.

Pitelis, N., Russell, C., and Agapito, L. (2014). Semi-supervised Learning Using an Unsupervised Atlas. In Calders, T., Esposito, F., Hllermeier, E., and Meo, R., editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8725 of *Lecture Notes in Computer Science*, pages 565–580. Springer Berlin Heidelberg.

Ranzato, M. A. and Szummer, M. (2008). Semi-supervised Learning of Compact Document Representations with Deep Networks. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 792–799, New York, NY, USA. ACM.

Rasmus, A., Valpola, H., Honkala, M., Berglund, M., and Raiko, T. (2015). Semi-Supervised Learning with Ladder Network. *arXiv preprint arXiv:1507.02672*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *arXiv preprint arXiv:1401.4082*.

Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., and Muller, X. (2011). The Manifold Tangent Classifier. In *NIPS'2011*.

Valpola, H. (2014). From neural pca to deep unsupervised learning. *arXiv preprint arXiv:1411.7783*.

Weston, J., Ratle, F., and Collobert, R. (2008). Deep learning via semi-supervised embedding. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1168–1175, New York, NY, USA. ACM.

Zhu, X., Ghahramani, Z., and Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919.