Supplementary: Learning Multi-layer Latent Variable Model with Short Run Inference Dynamics

Anonymous ECCV submission

Paper ID 6025

1 Appendix

1.1 Model Specification

For the multi-layer generator model, we have $z = (z_l, l = 1, \ldots, L)$ for which layer L is the top layer, and layer 1 is the bottom layer close to x. For simplicity, let $x = z_0$. Then, $p_\theta(z) = p_\theta(z_L) \prod_{l=0}^{L-1} p_\theta(z_l \mid z_{l+1})$. In our case, we have $z_L \sim \mathcal{N}(0, I)$, $[z_l|z_{l+1}] \sim \mathcal{N}(\mu_l(d_l(p_l(z_{l+1}))), \sigma_l^2(d_l(p_l(z_{l+1})))),\ l = 0, ..., L-1$. where $\mu_l()$ and $\sigma_l^2()$ are the mean vector and the diagonal variance-covariance matrix of z_l respectively, and they are functions of $d_l(p_l(z_{l+1}))$ where d_l are deterministic layers and p_l are projection layer to preserve dimensionality. d_l is defined as two subsequent $conv2d$ layers with $GeLU$ [3] activation functions and skip connection. p_l is a linear layer with subsequent $transpose\_conv2d$. μ_l and σ_l are a pair of $conv2d$ and $linear$ layers to project to dimensionality of z_l. Then, $z_l = \mu_l(d_l(p_l(z_{l+1}))) + \sigma_l(d_l(p_l(z_{l+1}))) \otimes \epsilon_l$ where $\epsilon_l \sim \mathcal{N}(0, I_{d_l})$. The final deterministic block o_0 is a $transpose\_conv2d$ layer projecting to the desired dimensionality of x. The range of x is bounded by $tanh()$.

Table 1 illustrates a specification with $L = 3$ latent layers, latent dimensions $d_3 = 32$, $d_2 = 64$, $d_1 = 128$ for z_3, z_2, z_1, respectively, and $n_f = 64$ channels.

l	operation	dimensions
3	$z_3 \sim N(0, I_{d_3})$	$[n, d_3, 1, 1]$
2	$z_{3,p} = p_2(z_3)$	$[n, n_f, 16, 16]$
2	$z_{3,d} = d_2(z_{3,p})$	$[n, n_f, 16, 16]$
2	$z_2 = \mu_2(z_{3,d}) + \sigma_2(z_{3,d}) \otimes \epsilon_2$	$[n, d_2, 1, 1]$
1	$z_{2,p} = p_1(z_2)$	$[n, n_f, 16, 16]$
1	$z_{2,d} = d_1(z_{2,p}) + z_{3,d}$	$[n, n_f, 16, 16]$
1	$z_1 = \mu_1(z_{2,d}) + \sigma_1(z_{2,d}) \otimes \epsilon_1$	$[n, d_1, 1, 1]$
0	$z_{1,p} = p_0(z_1)$	$[n, n_f, 16, 16]$
0	$z_{1,d} = d_0(z_{1,p}) + z_{2,d}$	$[n, n_f, 16, 16]$
0	$x = tanh(o_0(z_{1,d}))$	$[n, 3, 32, 32]$

Table 1: Specification of multi-layer generator model with $L = 3$ layers, latent dimensions $d_3 = 32$, $d_2 = 64$, $d_1 = 128$ for z_3, z_2, z_1, respectively, and $n_f = 64$ channels.

1.2 Training of Baselines

For ladder variational autoencoder [7], the generator model is defined in Table 1. The training follows the one outlined in [7]. We train the model with $T = 3 \times 10^5$ parameter updates optimized by Adam [5].

For GLO [1] and ABP [2], our model in Table 1 was reduced to a single-layer variational autoencoder.

For GLO, we used a re-implementation[1] in PyTorch. As outlined in [1], after training the model, the inferred latent vectors, z, were used to fit a multivariate Gaussian distribution from which z was drawn for sampling. The hyperparameters are as follows: $code\_dim = 128$, $n\_pca = 64 * 64 * 3 * 2$, $loss = l2$.

For ABP, 40 steps of persistent Markov Chains were used. The hyperparameters are as follows: 40 MCMC steps, Langevin discretization step size of 0.3, $\sigma = 0.3$, Adam [5] optimizer.

For Glow [6], the model was trained using the official code[2] with our datasets and the evaluation was performed with our implementation of the Frchet Inception Distance (FID) [4] with Inception v3 classifier [8] on $40,000$ generated example. The hyperparameters are as follows: $dal = 0$, $n\_batch\_train = 64$, $optimizer = adamax$, $n\_levels = 3$, $width = 512$, $depth = 16$, $n\_bits\_x = 8$, $learntop = False$, $flow\_coupling = 0$.

[1] https://github.com/tneumann/minimal_glo
[2] https://github.com/openai/glow

References

1. Bojanowski, P., Joulin, A., Lopez-Paz, D., Szlam, A.: Optimizing the latent space of generative networks. In: Dy, J.G., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. Proceedings of Machine Learning Research, vol. 80, pp. 599–608. PMLR (2018), http://proceedings.mlr.press/v80/bojanowski18a.html
2. Han, T., Lu, Y., Zhu, S., Wu, Y.N.: Alternating back-propagation for generator network. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. pp. 1976–1984 (2017), http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14784
3. Hendrycks, D., Gimpel, K.: Bridging nonlinearities and stochastic regularizers with gaussian error linear units. CoRR **abs/1606.08415** (2016), http://arxiv.org/abs/1606.08415
4. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA. pp. 6626–6637 (2017), http://papers.nips.cc/paper/7240-gans-trained-by-a-two-time-scale-update-rule-converge-to-a-local-nash-equilibrium
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980
6. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada. pp. 10236–10245 (2018), http://papers.nips.cc/paper/8224-glow-generative-flow-with-invertible-1x1-convolutions
7. Sønderby, C.K., Raiko, T., Maaløe, L., Sønderby, S.K., Winther, O.: Ladder variational autoencoders. In: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain. pp. 3738–3746 (2016), http://papers.nips.cc/paper/6275-ladder-variational-autoencoders
8. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 2818–2826 (2016). https://doi.org/10.1109/CVPR.2016.308, https://doi.org/10.1109/CVPR.2016.308