Approximate Gradient Descent for Training Implicit Generative Models

Yingzhen Li
University of Cambridge
yl494@cam.ac.uk

Abstract

This abstract presents our first attempt at applying gradient approximation methods to training implicit generative models. Concretely, the recently proposed Stein gradient estimator is utilised to approximate the gradient of the KL divergence from the (implicit) generator distribution to the data distribution. We empirically demonstrate that the proposed approach learns faster than directly minimising the maximum mean discrepancy.

1 Introduction

Implicit generative models are defined by a stochastic procedure that allows for direct generation of samples, but not for the evaluation of model probabilities. There is a recent interest in a specific type of algorithms to train implicit models, generative adversarial networks (GANs) (Goodfellow et al., 2014), which has been shown to be one of the most successful approaches to image and text generation (Radford et al., 2016; Yu et al., 2017; Arjovsky et al., 2017; Berthelot et al., 2017). In a nutshell, many of these GAN approaches follow the "approximate-then-optimise" procedure, i.e. they first approximate the model distribution or optimisation objective function using discriminators, and then use those approximations to learn the generator's parameters. However, for any finite number of data points there exists an infinite number of functions, with arbitrarily diverse gradients, that can approximate perfectly the objective function, and thus optimising such approximations can lead to unstable training and poor results. Therefore the approximated objective function needs to be well regularised in order to avoid over-fitting, which also explains the recent success of GAN stabilisation approaches such as the Lipschitz constraint in Wasserstein GAN (Arjovsky et al., 2017).

In this abstract we explore an alternative route, i.e. "optimise-then-approximate". Precisely, we propose training the generative model by minimising the Kullback-Leibler (KL) divergence from it to the data distribution using approximate gradient updates. This approximation employs the recently proposed *Stein gradient estimator* (Li and Turner, 2017) which is based on kernel methods. Therefore we also compare with directly minimising the kernel maximum mean discrepancy (MMD) (Gretton et al., 2012; Li et al., 2015; Dziugaite et al., 2015). Initial experiments on MNIST data demonstrate that our approach learns faster than direct MMD minimisation.

2 Approximating KL gradients

Given a dataset $\mathcal{D}$ containing i.i.d. samples we would like to learn a probabilistic model $q(\boldsymbol{x})$ for the underlying data distribution $p(\boldsymbol{x})$. In the case of implicit generative models, $q(\boldsymbol{x})$ is defined by a generative process: $\boldsymbol{x} \sim q(\boldsymbol{x}) \Leftrightarrow \boldsymbol{z} \sim \pi(\boldsymbol{z}), \boldsymbol{x} = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z})$. Here $\boldsymbol{f}$ is often a deep neural network parametrised by $\boldsymbol{\theta}$, and we assume $\boldsymbol{f}$ is differentiable w.r.t. $\boldsymbol{\theta}$. Now consider learning $\boldsymbol{\theta}$ by minimising the exclusive KL-divergence $\min_{\boldsymbol{\theta}} \mathrm{KL}[q||p]$. Using the reparameterisation trick (Kingma and Welling, 2014; Rezende et al., 2014), the gradient of the objective w.r.t. $\boldsymbol{\theta}$ is

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathrm{KL}[q||p] &= -\nabla_{\boldsymbol{\theta}}\mathbb{H}[q(\boldsymbol{x})] - \mathbb{E}_{\pi(\boldsymbol{z})}\left[\nabla_{\boldsymbol{f}}\log p(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}))^{\mathrm{T}}\nabla_{\boldsymbol{\theta}}\boldsymbol{f}\right] \\ &= \mathbb{E}_{\pi(\boldsymbol{z})}\left[\left[\nabla_{\boldsymbol{f}}\log q(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z})) - \nabla_{\boldsymbol{f}}\log p(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}))\right]^{\mathrm{T}}\nabla_{\boldsymbol{\theta}}\boldsymbol{f}(\boldsymbol{z})\right]. \\ &\approx \frac{1}{K}\sum_{k=1}^{K}\left[\nabla_{\boldsymbol{x}^k}\log q(\boldsymbol{x}^k) - \nabla_{\boldsymbol{x}^k}\log p(\boldsymbol{x}^k)\right]^{\mathrm{T}}\nabla_{\boldsymbol{\theta}}\boldsymbol{x}^k, \quad \boldsymbol{x}^k = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{z}^k), \boldsymbol{z}^k \sim \pi(\boldsymbol{z})\end{aligned} \tag{1}$$

As we typically assume the tractability of $\nabla_{\boldsymbol{\phi}}\boldsymbol{f}$, it remains to approximate both $\nabla_{\boldsymbol{x}}\log q(\boldsymbol{x})$ and $\nabla_{\boldsymbol{x}}\log p(\boldsymbol{x})$. Li and Turner (2017) has shown that these two gradients can be approximately computed using samples from the data and the generator. More precisely, we first define $\mathcal{K}(\boldsymbol{x}, \boldsymbol{y})$ as a translation invariant kernel that satisfies the boundary condition (Liu et al., 2016; Li and Turner, 2017). Such kernels include many common choices like the RBF kernel. Also assume we sample K instances, $\mathbf{Y} = \{\boldsymbol{y}^1, ..., \boldsymbol{y}^K\} \sim p$ and $\mathbf{X} = \{\boldsymbol{x}^1, ..., \boldsymbol{x}^K\} \sim q$, respectively. Then by denoting $\mathbf{K}_{\mathbf{XX}}$ as the matrix with entries $\mathcal{K}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ and similarly for $\mathbf{K}_{\mathbf{YY}}$ and $\mathbf{K}_{\boldsymbol{x}\mathbf{Y}}$, the approximate gradients are computed as

$$\begin{aligned}\hat{\mathbf{G}}_V^{\text{Stein}}(q) &:= -(\mathbf{K}_{\mathbf{XX}} + \eta\mathbf{I})^{-1}\langle\nabla, \mathbf{K}_{\mathbf{XX}}\rangle \approx \left(\nabla_{\boldsymbol{x}^1}\log q(\boldsymbol{x}^1), \cdots, \nabla_{\boldsymbol{x}^K}\log q(\boldsymbol{x}^K)\right)^{\mathrm{T}}, \\ \hat{\mathbf{G}}_V^{\text{Stein}}(p) &:= -(\mathbf{K}_{\mathbf{YY}} + \eta\mathbf{I})^{-1}\langle\nabla, \mathbf{K}_{\mathbf{YY}}\rangle \approx \left(\nabla_{\boldsymbol{y}^1}\log p(\boldsymbol{y}^1), \cdots, \nabla_{\boldsymbol{y}^K}\log p(\boldsymbol{y}^K)\right)^{\mathrm{T}},\end{aligned} \tag{2}$$

$$\begin{aligned}\forall\boldsymbol{x} \sim q, \quad \nabla_{\boldsymbol{x}}\log p(\boldsymbol{x})^{\mathrm{T}} \approx &- \left(\mathbf{K}_{\boldsymbol{x}\boldsymbol{x}} + \eta - \mathbf{K}_{\boldsymbol{x}\mathbf{Y}}(\mathbf{K}_{\mathbf{YY}} + \eta\mathbf{I})^{-1}\mathbf{K}_{\mathbf{Y}\boldsymbol{x}}\right)^{-1} \\ &\left(\mathbf{K}_{\boldsymbol{x}\mathbf{Y}}\hat{\mathbf{G}}_V^{\text{Stein}}(p) - \left(\mathbf{K}_{\boldsymbol{x}\mathbf{Y}}(\mathbf{K}_{\mathbf{YY}} + \eta\mathbf{I})^{-1} + \mathbf{1}^{\mathrm{T}}\right)\nabla_{\boldsymbol{x}}\mathcal{K}(\cdot, \boldsymbol{x})\right),\end{aligned} \tag{3}$$

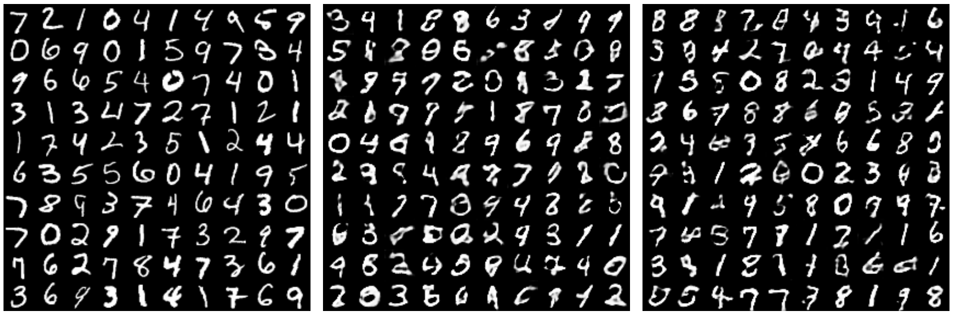
with $\langle\nabla, \mathbf{K}_{\mathbf{XX}}\rangle_{ij} = \sum_{k=1}^{K}\nabla_{x_j^k}\mathcal{K}(\boldsymbol{x}^i, \boldsymbol{x}^k)$. Plugging-in the above approximation to the last line of equation (1) returns the approximate KL-gradient which is then fed to any optimiser in use, e.g. Adam (Kingma and Ba, 2015). The choice the kernel $\mathcal{K}(\boldsymbol{x}, \boldsymbol{y})$ is crucial to the accuracy of the estimator. Li et al. (2015) used a mixture of RBF kernels with different bandwidths. We follow this strategy but instead use a mixture of IMQ kernels with different bandwidths, i.e. $\mathcal{K}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{n=1}^{N}\left[1 + \frac{1}{\sigma_n^2}||\boldsymbol{x} - \boldsymbol{y}||^2\right]^{-\frac{1}{2}}$. A key advantage of the Stein approach is that the derived gradient estimator is ubiquitous and therefore it is directly applicable to the mixture kernel. By contrast, another existing kernel-based technique, score matching gradient estimator (Hyvärinen, 2005; Sasaki et al., 2014; Strathmann et al., 2015), requires tedious derivations repeatedly even for the simplest cases such as the RBF kernel, let alone the mixture kernel version.[1] Also Li and Turner (2017) showed that the plug-in estimator using kernel density estimation (KDE) under-performs. Therefore these two kernel-based gradient estimators are not considered in this abstract.

3 Initial Results

We present an initial experiment on learning implicit generative models using the (continuous) MNIST data as a proof of concept. The generator follows the architecture design of DCGAN (Radford et al., 2016), and the latent variable $\boldsymbol{z}$ is of 50 dimensions. We use learning rate 0.00005 and Adam optimiser (Kingma and Ba, 2015). The mixture kernel contains $N = 4$ component, in which the (square of the) bandwidths are computed with the median trick on samples $\mathbf{X} \cup \mathbf{Y}$ then scaled up by $[0.5, 1.0, 2.0, 4.0]$, respectively. Figure 1 visualises the samples from the implicit generative model trained using direct MMD (V-statistic) minimisation and approximate KL gradient descent method, respectively. Here we use batch-size 100 in all the experiments to achieve a good speed-accuracy trade-off, although sample quality can be further improved by increasing the mini-batch size.

We further consider quantitative evaluations of the trained models. During training, 500 images are sampled from the model for every 50 epochs to compute the quantitative metrics. We compute their nearest neighbours in the training set using l_1 distance, and obtain a probability vector $\mathbf{p}$ by averaging over these neighbour images' label vectors. We also train an MLP classifier that achieves

[1] Li and Turner (2017) also presented a parametric version of the Stein gradient estimator that is derived in similar spirit as the score matching estimator, therefore for the same reason we did not test its performance here.



(a) test data samples (b) minimising MMD V-statistic (c) approximate KL SGD

Figure 1: Visualisation of MNIST samples

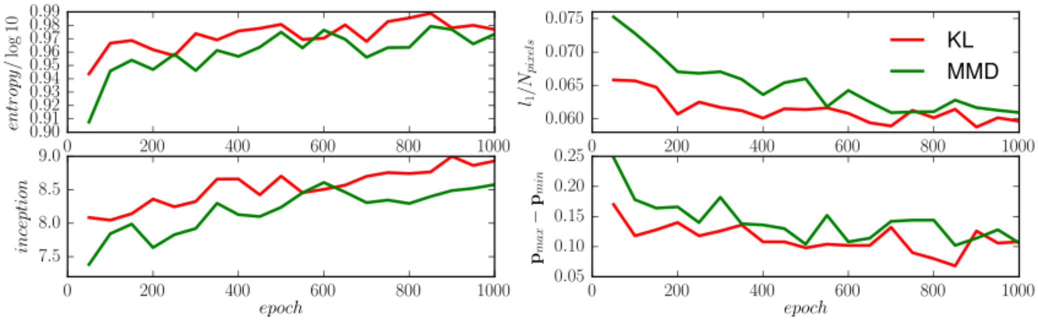


Figure 2: Quantitative evaluation on the trained models. The higher the better for the LHS panels and the other way around for the RHS ones. Training speed: 6.84s/epoch for approx. KL SGD and 10.77s/epoch for MMD minimisation. See main text for details.

98.16% accuracy, and compute on the samples the inception score (Salimans et al., 2016). The image samples from the test data achieve 9.82 inception score using the trained classifier (maximum value 10). Figure 2 shows these quantitative measures, and it is clear that our approach learns faster and better than direct MMD minimisation.[2]

4 Discussions on Kernel Learning

We have successfully demonstrated the feasibility of the proposed approximate KL gradient descent training using the MNIST experiment. However our initial experiments on natural image datasets such as cifar-10 show that both methods work poorly when using the same mixture IMQ kernel as in the MNIST case. Therefore, like many kernel-based machine learning algorithms, the selection of the kernel is key to the performance for both methods. In this regard, Sutherland et al. (2016) selects the hyper-parameters of the kernel by maximising the test power of a kernel two-sample test using MMD as the test statistic. This approach further improves the generated image quality on MNIST but has never been tested on natural image datasets. Another direction for kernel selection/learning is to combine commonly used kernels with a deep neural network. Wilson et al. (2016) introduced "deep kernels" by applying a base kernel (e.g. RBF) to feature vectors obtained by a deep neural network transformation. The very recently proposed MMD-GAN approach (Li et al., 2017) introduced an adversarial loss to learn the parameters of the neural network. We expect that the adversarial training idea can be applied to the proposed algorithm with minimal adjustments.

[2] Experiments are timed on an NVIDIA GeForce GTX TITAN X GPU. We found that applying automatic differentiation to the MMD V-statistic can be slower than manually computing $\nabla_{\boldsymbol{x}}\text{MMD}_{\text{V-stats}}(p, q)$ then applying the chain rule to obtain gradients for $\boldsymbol{\theta}$. In the latter case we expect the MMD optimisation to be about the same speed as the proposed approximate KL gradient descent approach.

Acknowledgements

I thank the reviewers' comments and the Schlumberger Foundation FFTF fellowship.

References

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Berthelot, D., Schumm, T., and Metz, L. (2017). Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*.

Dziugaite, G. K., Roy, D. M., and Ghahramani, Z. (2015). Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *NIPS*.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709.

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

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Póczos, B. (2017). Mmd gan: Towards deeper understanding of moment matching network. *arXiv preprint arXiv:1705.08584*.

Li, Y., Swersky, K., and Zemel, R. (2015). Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1718–1727.

Li, Y. and Turner, R. E. (2017). Gradient estimators for implicit models. *arXiv preprint arXiv:1705.07107*.

Liu, Q., Lee, J. D., and Jordan, M. I. (2016). A kernelized stein discrepancy for goodness-of-fit tests and model evaluation. In *ICML*.

Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of The 30th International Conference on Machine Learning (ICML)*.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *NIPS*.

Sasaki, H., Hyvärinen, A., and Sugiyama, M. (2014). Clustering via mode seeking by direct estimation of the gradient of a log-density. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 19–34. Springer.

Strathmann, H., Sejdinovic, D., Livingstone, S., Szabo, Z., and Gretton, A. (2015). Gradient-free hamiltonian monte carlo with efficient kernel exponential families. In *Advances in Neural Information Processing Systems*, pages 955–963.

Sutherland, D. J., Tung, H.-Y., Strathmann, H., De, S., Ramdas, A., Smola, A., and Gretton, A. (2016). Generative models and model criticism via optimized maximum mean discrepancy. *arXiv preprint arXiv:1611.04488*.

Wilson, A. G., Hu, Z., Salakhutdinov, R., and Xing, E. P. (2016). Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378.

Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). Seqgan: sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.