Sequentialized Sampling Importance Resampling and Scalable IWAE

Chin-Wei Huang[†] **Aaron Courville**[†‖]
[†]Montreal Institute for Learning Algorithms (MILA) [‖]CIFAR Fellow
`chin-wei.huang@umontreal.ca` `courvila@iro.umontreal.ca`

Abstract

We propose a new sequential algorithm for Sampling Importance Resampling. The algorithm serves as a solution to expensive evaluation of importance weight, and can be interpreted as stochastically and iteratively refining the particles by correcting them towards the target distribution as pool size increases. We apply this algorithm to variational inference with Importance Weighted Lower Bound and propose a memory-scalable training procedure [1] that implicitly improves the variational proposal.

1 Sequentializing Sampling Importance Resampling

1.1 Sampling Importance Resampling

Given an unnormalized target distribution $\tilde{p}(x)$ and proposal distribution $q(x)$, the Sampling Importance Resampling (SIR) proceeds as follows:

1. draw x_i for $1 \leq i \leq n$ from $q(x)$
2. calculate the importance weight $w_i = \frac{\tilde{p}(x_i)}{q(x_i)}$
3. calculate the normalized importance weight $\bar{w}_i = \frac{w_i}{\sum_i w_i}$
4. draw index variable $y_j \sim mul(\bar{w}_1, ..., \bar{w}_n)$ for $1 \leq j \leq m$

The density of the set of resampled particles $x_{y_1}, ..., x_{y_m}$ should resemble the pdf of the target distribution, and the new samples will be approximately distributed by $p(x)$ (Bishop, 2007). On average, the samples can be improved by increasing the pool size n, and becomes corrected when $n \to \infty$. The procedure is visualized in Figure 1a.

1.2 SeqSIR

The above procedure can be combined with the idea of *reservoir sampling*, so that we need not evaluate all n samples at the same time, which will be an issue when n is large or when evaluation of a sample (i.e. computation of w_i) is expensive. The intuition is to keep a running sum of the importance weights while we evaluate the pool samples sequentially, and then decide to keep the old sample or replace it with the new one based on the ratio of the new sample's importance weight to the running sum. This is what we call *Sequentialized Sampling Importance Resampling* (SEQSIR), which is summarized in Algorithm 1. See Figure 1b for illustration.

Note that density and importance weight are computed on log scale to deal with numerical instability, and log-sum-exp operation (LSE) is used in place of addition to calculate the running sum of

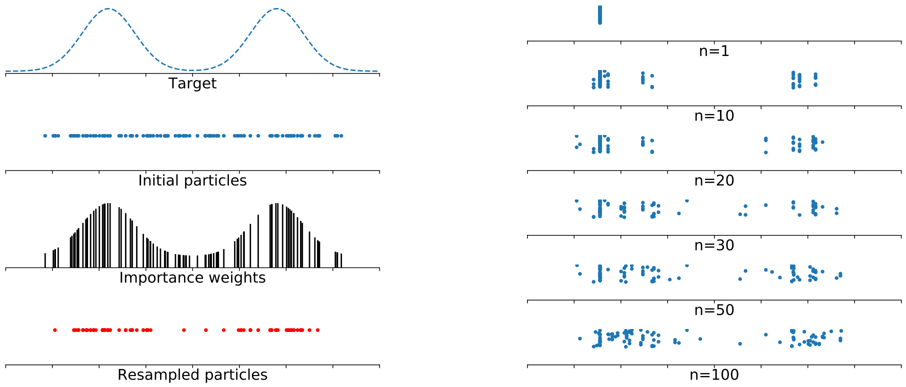
[1] See `https://github.com/CW-Huang/SeqIWAE` for implementation.

Second workshop on Bayesian Deep Learning (NIPS 2017), Long Beach, CA, USA.

Algorithm 1 Sequentialized Sampling Importance Resampling and Stochastic Iterative Refinement

```
procedure SEQSIR (
    logp, logq          ▷ unnormalized target density function and proposal density function
    ss                  ▷ $n$ samples to be evaluated
)
    A ← $-\infty$       ▷ initialize accumulated sum of importance weight on log scale
    s_old ← **0**       ▷ initialize sample
    n← len([s1,...,sn])
    for i=1,...,n do
        s_new = ss[i]
        A, s_old ← STOCHREFINE(logp, logq, A, s_old, s_new)
    return s_old

procedure STOCHREFINE (
    logp, logq          ▷ unnormalized target density function and proposal density function
    A                   ▷ accumulated sum of importance weight on log scale
    s_old, s_new        ▷ old and new samples
)
    w_new ← logp(s_new) - logq(s_new)
    A ← LSE(A, w_new)
    u ← unif(0,1)
    if w_new - A >= log u then return A, s_new
    else return A, s_old
```



(a) Sampling Importance Resampling: All the samples are evaluated in parallel and resampled according to the importance weights.

(b) Sequentialized Sampling Importance Resampling: All the samples start from the same location, and resampled when a new location is proposed.

Figure 1: Comparison of batch SIR and sequentialized SIR

importance weights. The algorithm can be easily made parallelizable by comparing multiple uniform samples as particles at the same time; see Figure 2 for an example of independent evaluations.

2 Estimating Gradient of Importance Weighted Lower Bound

When dealing with intractable integral in the marginal likelihood of models such as deep latent Gaussian models (Kingma and Welling, 2013; Rezende et al., 2014), one needs to resort to approximate inference. Variational methods are a family of algorithms that cast approximate inference as an optimization problem (Jordan et al., 1999): we maximize a lower bound (such as the *evidence lower bound*, ELBO) on the marginal likelihood:

$$\mathcal{L}_{ELBO} = \mathbf{E}_{q(z)} \left[\log \frac{p(x,z)}{q(z)} \right] \leq \log p(x) \qquad (1)$$

for some latent variable z, observed variable x, predefined joint distribution p and variational distribution q.

When combined with Monte Carlo sampling, one can derive a tighter lower bound than the traditional lower bound by drawing multiple samples (say, n) to evaluate the likelihood ratio, known as the *importance weighted lower bound* (Burda et al., 2015), or IWLB:

$$\mathcal{L}_{ELBO} \leq \mathcal{L}^n_{IWLB} = \mathbf{E}_{\{z_i\} \overset{\text{iid}}{\sim} q(z)} \left[\log \sum_{i=1}^{n} \frac{1}{n} \frac{p(x, z_i)}{q(z_i)} \right] < \log p(x) \tag{2}$$

for some $n \geq 1$. The first inequality becomes equality when $n = 1$, and the second gap can be closed by taking $n \to \infty$. This is an appealing property, as to make the training update less unbiased, one simply needs to draw more samples from the proposal. However, one problem with this method is that training algorithms scale linearly with the number of samples in time and memory [2]. Due to memory constraint in practice, training with multiple samples is usually achieved at the cost of smaller batch size. This increases the variance of estimating the gradient as we draw less samples from the data distribution, and precludes the possibility of training with even larger pool size to enjoy the asymptotic property of importance sampling. In this regard, one can modify the training procedure of maximizing IWLB by using SeqSIR, which we describe below.

2.1 SeqIWAE

Algorithm 2 Gradient estimate and stochastic update of IWAE

```
procedure SeqIWAE (
    PZ, PXZ, QZX                  ▷ functions of prior, likelihood and approximate posterior
    UPDATE                        ▷ update function of VAE
    n                             ▷ pool size
    x                             ▷ minibatch of training data
)
    function LOGP(e)
        return log PZ(e) + log PXZ(x,e)
    function LOGQ(e)
        return log QZX(e,x)
    epss = randn(n)
    eps = SeqSIR( LOGP, LOGQ, epss )
    UPDATE(x,eps)
```

In this section, we develop a sequentialized training algorithm for Importance Weighted Autoencoder (Burda et al., 2015) called SeqIWAE, which can also be applied to other variational inference problems that seek to maximize IWLB.

The reparameterization trick (Rezende et al., 2014; Kingma and Welling, 2013) allows one to separate noise from parameters of a sample drawn from a distribution. Take any Gaussian distribution as example. We can first draw samples ϵ from a standard Gaussian and then transform the sample by taking $z = \mu + \sigma \cdot \epsilon$. One can thus express a one-sample Monte Carlo estimate of the ELBO and estimate of the gradient of the ELBO as functions of the input data point and the parameter-free noise, i.e. $\mathcal{L}_{ELBO}(x, \epsilon)$ and $\nabla_\phi \mathcal{L}_{ELBO}(x, \epsilon)$. In Algorithm 2, we combine SeqSIR to estimate the gradient of $\mathcal{L}_{IWLB}$ by selecting from a pool of parameter-free samples based on their scores.

This procedure leaves the estimate of gradient unbiased. As shown by (Burda et al., 2015), the update rule is as follows:

$$\nabla_\phi \mathcal{L}^n_{IWLB} = \mathbf{E}_{\{\epsilon_i\}} \sum_{i=1}^{n} \tilde{w}_i \nabla_\phi \mathcal{L}_{ELBO}(x, \epsilon_i) \tag{3}$$

where $\tilde{w}_i = \frac{p(z)p(x|z_i)/q(z_i|x)}{\sum_i p(z)p(x|z_i)/q(z_i|x)}$ is the normalized importance weight and $z_i = g(\epsilon_i, x)$ is reparameterized sample through parameter-free noise. As the derivation below shows, Eq. 3 can be estimated

[2] Sublinearity in time complexity can be achieved via parallelizing evaluation of the importance weights.

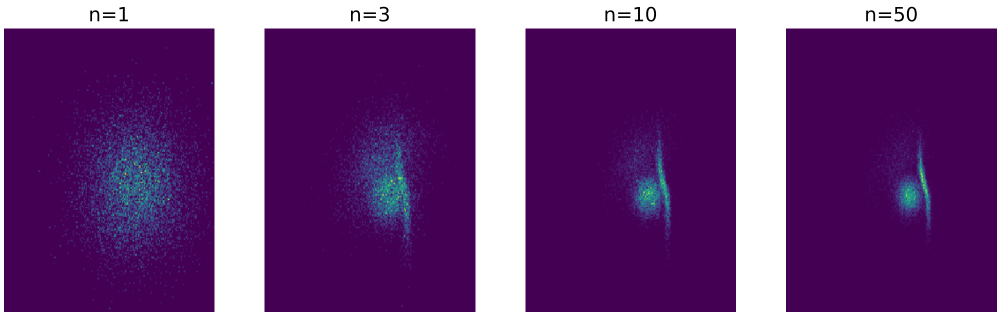


Figure 2: SeqSIR as stochastic iterative refinement (Bachman and Precup, 2015; Cremer et al., 2017): when the pool size increases, the samples drawn from the proposal distribution are corrected towards the true posterior

by resampling (originally suggested in Burda et al. (2015)), and thus by the stochastic refinement procedure introduced in the previous section.

$$\begin{aligned}\nabla_\phi \mathcal{L}^n_{IWLB} &= \mathbf{E}_{\{\epsilon_i\}, i \sim mul(\tilde{w}_i)} \nabla_\phi \mathcal{L}_{ELBO}(x, \epsilon_i) \\ &= \mathbf{E}_{\{\epsilon_i\}} \left[\frac{\sum_{i'=1}^{n-1} w_{i'}}{\sum_{j=1}^{n} w_j} \underbrace{\left[\sum_{i=1}^{n-1} \frac{w_i}{\sum_{i'=1}^{n-1} w_{i'}} \nabla_\phi \mathcal{L}_{ELBO}(x, \epsilon_i) \right]}_{\nabla_\phi \mathcal{L}^{n-1}_{IWLB}} + \frac{w_n}{\sum_{j=1}^{n} w_j} \left[\nabla_\phi \mathcal{L}_{ELBO}(x, \epsilon_n) \right] \right]\end{aligned} \tag{4}$$

As a result, the gradient of $\mathcal{L}^n_{IWLB}$ can be estimated by estimating $\nabla \mathcal{L}_{ELBO}$ using the sampled noise ϵ_j selected by taking $\epsilon_{j+1} = \epsilon_{j+1}$ if $u_{j+1} \geq \frac{w_j}{\sum_{j'=1}^{n} w_j}$, otherwise ϵ_j, where $u_{j+1} \sim unif(0, 1)$ for $j = 2, ..., n$. This corresponds to StochRefine in Algorithm 1. Each iteration in expectation improves the approximation as it corresponds to the estimate of the gradient of a tighter bound. It can also be viewed as the same lower bound as ELBO, but with the original proposal distribution corrected towards the true posterior as one draw more samples iteratively (Bachman and Precup, 2015; Cremer et al., 2017), as in Figure 2.

Furthermore, note that sublinearity in time complexity now becomes $\mathcal{O}(n)$ as evaluation of importance weights is sequential, and that memory goes from $\mathcal{O}(n)$ to $\mathcal{O}(1)$.

3 Conclusion

In this note, we developed a method to sequentially resample from a pool of particles based on their importance weights. Our algorithm finds application in approximate inference where evaluation of importance weight is expensive such as deep generative models (i.e. importance weighted autoencoders). Such a slight modification of the update rule is simple to plug in, and makes it possible for one to enjoy the asymptotic property of importance sampling by trading time for memory.

Acknowledgments

We thank Shawn Tan for helpful discussion and feedback.

References

Bachman, P. and Precup, D. (2015). Training Deep Generative Models: Variations on a Theme. *ArXiv e-prints*.

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. (2015). Importance weighted autoencoders. *CoRR*, abs/1509.00519.

Cremer, C., Morris, Q., and Duvenaud, D. (2017). Reinterpreting Importance-Weighted Autoencoders. *ArXiv e-prints*.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286. PMLR.