Report on "Improving Variational Inference using Autoregressive Flows" by Kingma et al

Ronak Dedhiya Dinesh (ronakdedhiya@iisc.ac.in; SR No: 06-18-05-19-52-21-1-20116)
Prashanth N Bhat(prashanthbn@iisc.ac.in; SR No: 04-03-06-19-52-21-1-20082)

September 28, 2024

1 Introduction

The main idea of the paper is to generate more powerful posterior distributions closer to true posterior compared to the basic isotropic Gaussian by applying series of invertible transformation. A simple MLP can be implemented as invertible transform with easy computation but requires a long chain of transformations to capture high-dimensional dependencies. On the other hand, autoregressive neural network can approximate complex distributions but are sequential and computationally heavy. The trick is to use the inverse autoregressive transform which can actually be parallized. As it turns out applying series of inverse autoregressive transform aids in building a more flexible posterior distribution.

2 Variational Auto Encoder

In vanilla variational autoencoder, the generative network (decoder) and posterior network (encoder) are related to each other through a set of diagnonal Gaussian variables usually labelled as z. The posterior network is an estimate of the true posterior, unfortunately our fully factorized diagnonal gaussian can't model every distribution. In particular, the fact that they're fully factored limit the ability to match the true posterior we're trying to model. The desirable properties with factored gaussian is a) computationally efficient to compute and differentiate the posterior. b) it's easy to sample at every minibatch. To replace it with certain complex distribution which can represent the true posterior, it is required to obtain the above deseired properties.

3 Normalizing flows

Introduced by Rezende et al [2], At its core, it is application of series of invertible transformations to our fully factored posterior distribution to make it into something more flexible that can match the posterior distribution. It's called normalizing flow because the density "flows" through each transform. The transformation f that yields these two properties of "easy determinant of the jacobian" and "easy inverse", allows to have a much capacity as needed in order to learn complex distribution [1].

$$\begin{aligned} &z_0 \sim q_\theta(z_0|x) \\ &z_k = f_\theta^K * f_\theta^{K-1} * \cdots * f_\theta^1(z_0) = f_k(z_{k-1}, x) \end{aligned} \tag{1}$$

So why does this help at all? The normalizing flow can be thought of as a sequence of expansions or contractions on the initial density, allowing for things such as multi-modal distributions (something we can't do with a basic Gaussian). Additionally, it allows us to have complex relationships between the variables instead of the independence we assume with our diagonal Gaussians.

Rezende et al proposed a simple invertible transformation which can be viewed as a MLP with bottleneck hidden layer with single unit. Since information goes through the single bottleneck, a long chain of transformations is required to capture high-dimensional dependencies.

4 Gaussian Autoregressive models

In order to find a type of normalizing flow that scales well to high-dimensional space, Gaussian Autoregressive models are considered which can be defined as:

If $y = \{y_i\}_{i=1}^D$ is a random vector, each variable is dependent only on previously indexed variable i.e. $y_i = f_i(y_{0:i-1})$

$$\begin{aligned} &y^0 = \mu^0 + \sigma^0 * z^0 \\ &y^i = \mu_\theta^i(y^{(1:i-1)}) + \sigma_\theta^i(y_\theta^{(1:i-1)}) * z^i \\ &z^i = \mathcal{N}(0,1)\forall i \end{aligned} \tag{2}$$

where $\mu_\theta^i(y^{(1:i-1)})$ and $\sigma_\theta^i(y_\theta^{(1:i-1)}) * z^i$ are general approximators parameterized by θ. In our case, we can transform any vector, not just a noise vector. This is shown on the left hand side of Figure 1 where we take a x vector and transform it to a y vector. You'll notice that we have to perform O(D) sequential computations, thus making it too slow for our intended purpose of use in a normalizing flow. But what about the inverse transform? The inverse transform can actually be parallelized as shown in below equation and the right hand side of the Figure 1.

$$x_i = \frac{y_i - \mu_i(y_{0:i-1})}{\sigma_i(y_{0:i-1}} \tag{3}$$

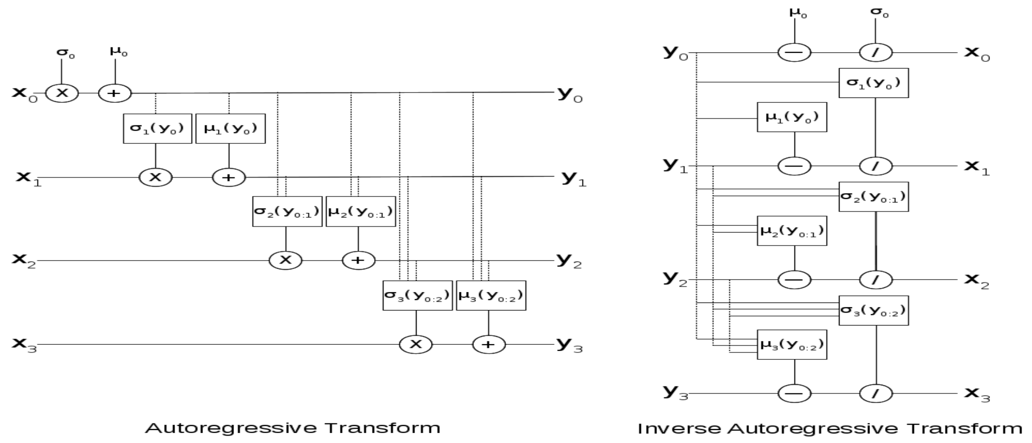

Figure 1: Gaussian version of Autoregressive Transform and Inverse Autoregressive Transforms

5 Inverse Autoregressive Flows

The transformations defined by the Inverse Gaussian Autoregressive models have a lower triangular Jacobian matrix. It means that the log-determinant of the transformation is simply minus of the sum of log standard deviations used in the autoregressive model:

$$log(det(\left|\frac{dz}{dy}\right|)) = -\sum_{i=1}^{D} log\sigma_\theta^i(y) \tag{4}$$

Adding an inverse autoregressive flow (IAF) to a variational autoencoder is as simple as adding a bunch of IAF transforms after the latent variables z. A context c is additionally generated, it avoids model to get to Nans while training IAF.

The steps of Inverse Autoregressive Flow are:

1. Map $x \rightarrow \mu_\theta^0, \sigma_\theta^0, c$ where c is an optional context variable
2. $z_0 = \mu_\theta^0 + \sigma_\theta^0 \cdot \epsilon^0$, where $\epsilon^0 \sim \mathcal{N}(0, I)$
3. For $i = 1 \ldots K$
    (a) $z_i = (z_{i-1} - \mu_\theta^i(z_{i-1}, c))/\sigma_\theta^i(z_{i-1}, c)$ where $(\mu_\theta^i, \sigma_\theta^i)$ are differently parameterized autoregressive model.
4. The final iterate z_k can have a flexible distribution to mimic the posterior

The model is trained with modified architecture for MNIST and CIFAR Dataset and found to produce best log likelihood scores.

6 Conclusion

Inverse autoregressive flow (IAF) is a new type of normalizing flow that scales well to high-dimensional latent space. The paper shows some creativity to use the inverse of an autoregressive flow in addition to the whole concept of a normalizing flow , benefitting from both the techniques.The combination of model flexibility, parallelizability across dimensions, and simple log-determinant,make this transformation interesting for use as a normalizing flow over high-dimensional latent space.

References

[1] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[2] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows.