

Deep Learning - Foundations and Concepts

Chapter 18. Normalizing Flows

nonlineark@github

April 18, 2025

Outline

1 Coupling Flows

Normalizing flows

- We define a distribution $p_z(z)$ over a latent variable z .
- And we use a deep neural network architecture to define an invertible function $x = f(z; w)$ that transforms the latent space into the data space.
- We can ensure that the overall function is invertible if we make each layer of the network invertible:
 - $x = f^A(f^B(f^C(z)))$.
 - $z = g^C(g^B(g^A(x)))$.

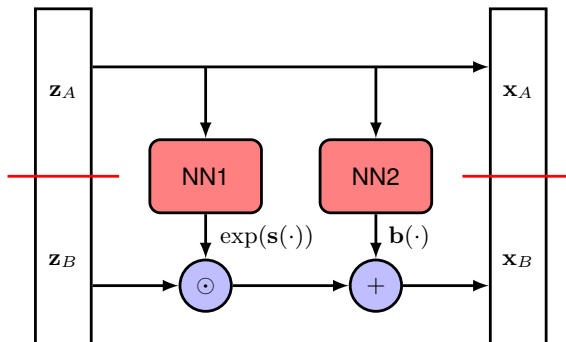
Normalizing flows

This approach to modelling a flexible distribution is called a normalizing flow because:

- The transformation of a probability distribution through a sequence of mappings is somewhat analogous to the flow of a fluid.
- The effect of the inverse mapping is to transform the complex data distribution into a normalized form, typically a Gaussian distribution.

Real NVP

Figure: A single layer of the real NVP normalizing flow model



Real NVP

Real-valued non-volume-preserving (real NVP) partitions the latent variable $z \in \mathbb{R}^D$ and the output vector $x \in \mathbb{R}^D$ into two parts: $z = (z_A, z_B)$ and $x = (x_A, x_B)$, where $z_A, x_A \in \mathbb{R}^d$. The transformation function is given by:

$$x_A = z_A$$

$$x_B = \exp(s(z_A; w)) \odot z_B + b(z_A; w)$$

where $s(z_A; w)$ and $b(z_A; w)$ are the real-valued outputs of neural networks, and \odot denotes element-wise multiplication of the two vectors.

Real NVP

The overall transformation is easily invertible:

$$z_A = x_A$$

$$z_B = \exp(-s(x_A; w)) \odot (x_B - b(x_A; w))$$

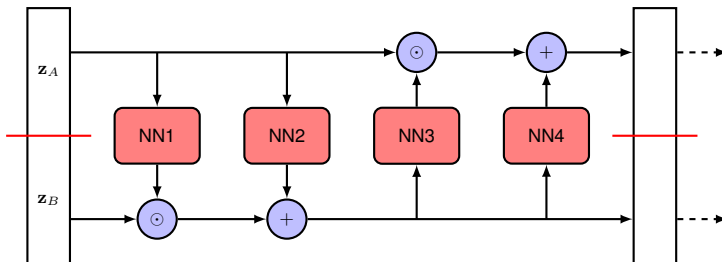
The Jacobian matrix of this inverse mapping is given by:

$$J = \begin{pmatrix} I_d & 0 \\ \frac{\partial z_B}{\partial x_A} & \text{diag}(\exp(-s(x_A; w))) \end{pmatrix}$$

We see that the Jacobian matrix is a lower triangular matrix, and its determinant is simply given by the product of the elements of $\exp(-s(x_A; w))$.

Real NVP

Figure: A double-layer structure where the roles of z_A and z_B are reversed in the second layer



Coupling flows

The real NVP model belongs to a broad class of normalizing flows called coupling flows:

$$x_A = z_A$$

$$x_B = h(z_B, g(z_A; w))$$

where:

- $h(z_B, g)$ is a function of z_B that is efficiently invertible for any given value of g and is called the coupling function.
- The function $g(z_A; w)$ is called a conditioner and is typically represented by a neural network.