uc3m

Neural Networks

Batch Normalization

Pablo Martínez Olmos, pamartin@ing.uc3m.es

José Carlos Aradillas, jaradill@ing.uc3m.es

Ignacio Peis, ipeis@pa.uc3m.es

Internal shift

- Training Deep Neural Networks is **complicated** by the fact that **the distribution of each layer's inputs changes during training**, as the parameters of the previous layers change

- We refer to this phenomenon as **internal covariate shift**

- **This slows down the training** by requiring lower learning rates, and **makes it notoriously hard to train models with saturating nonlinearities**
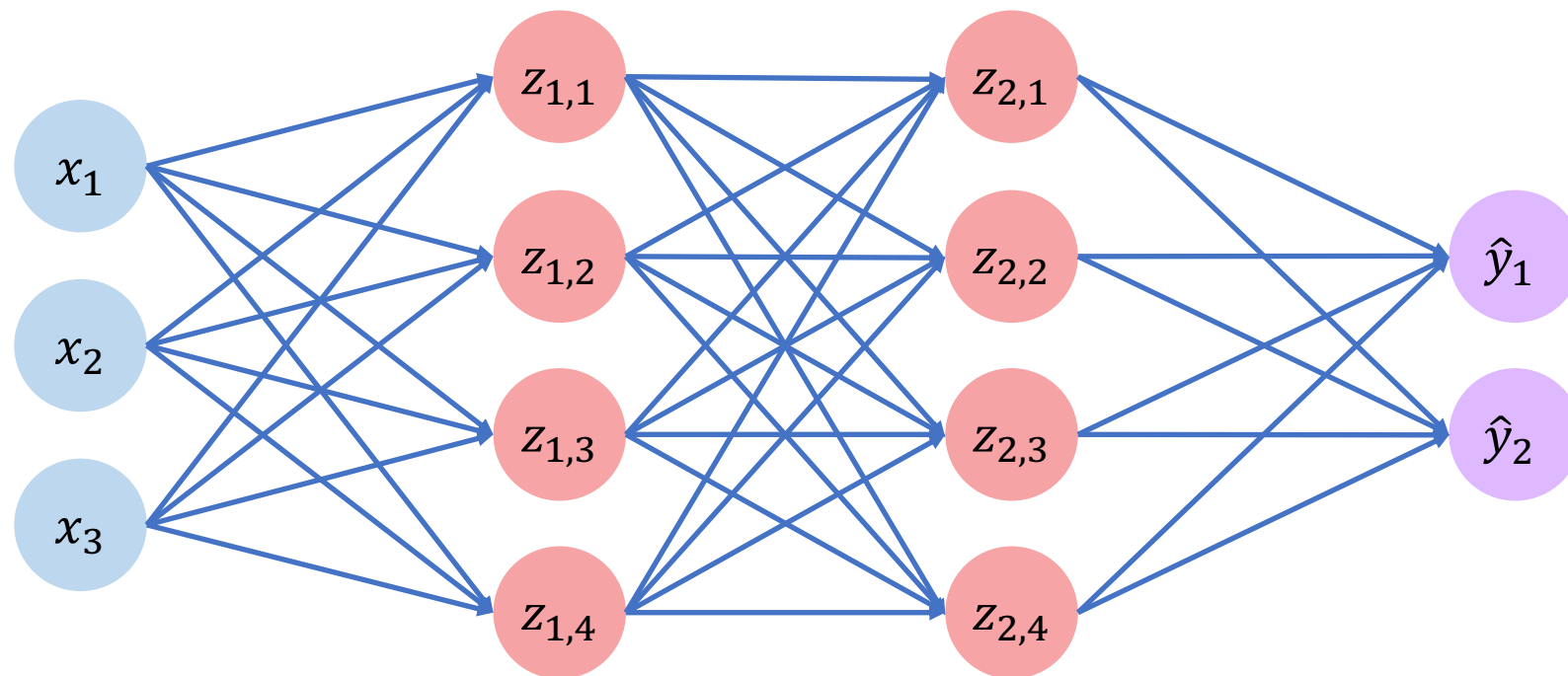
Batch Normalization

- By **fixing the distribution of the layer inputs** as the training progresses, we expect to improve the training speed

- Batch Normalization performs input normalization in a way that is **differentiable** and **does not require the analysis of the entire training set** after every parameter update

- **Normalize each scalar feature independently**

- Scale the normalized input **with trainable parameters**

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift
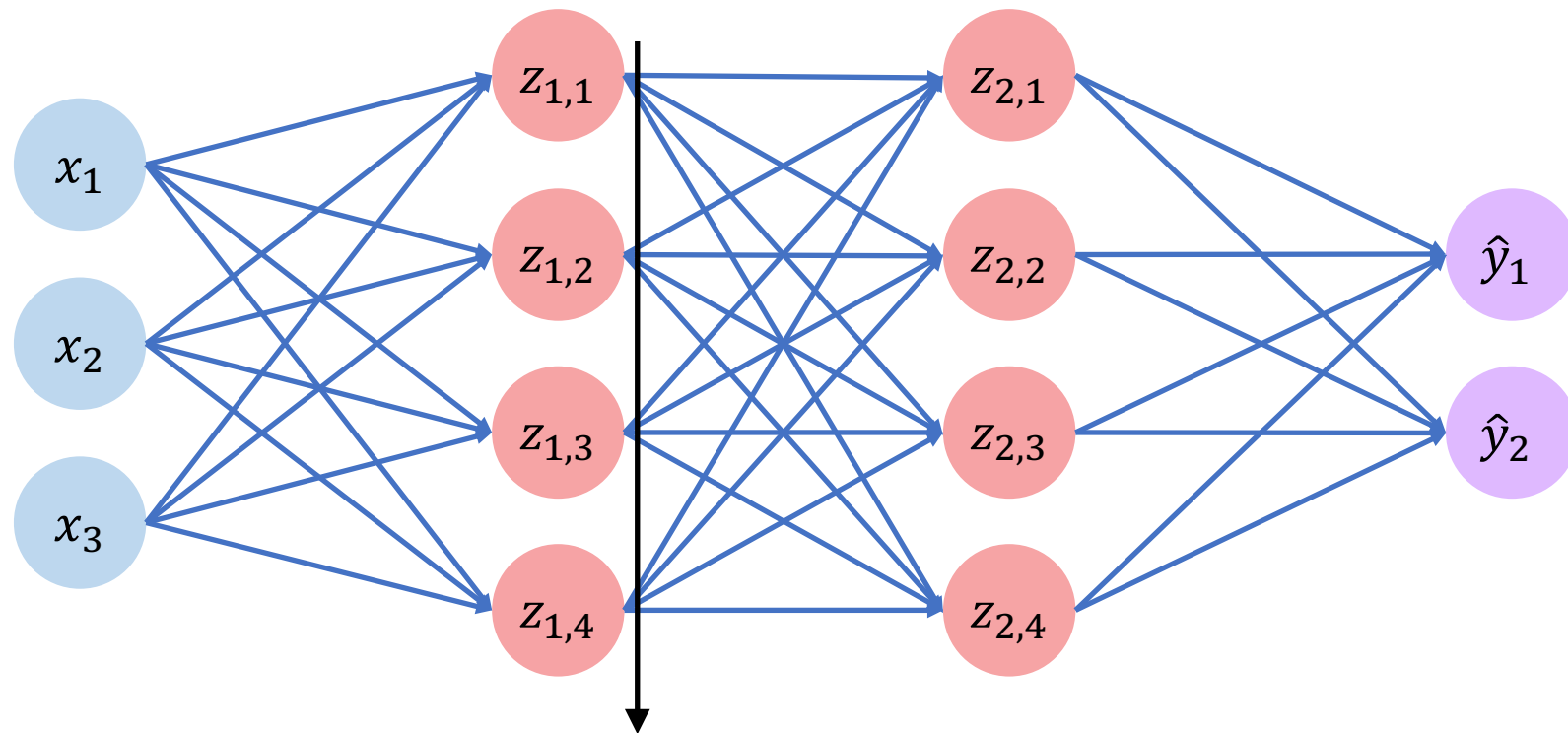

Sergey Ioffe   SIOFFE@GOOGLE.COM
Christian Szegedy   SZEGEDY@GOOGLE.COM
Google, 1600 Amphitheatre Pkwy, Mountain View, CA 94043
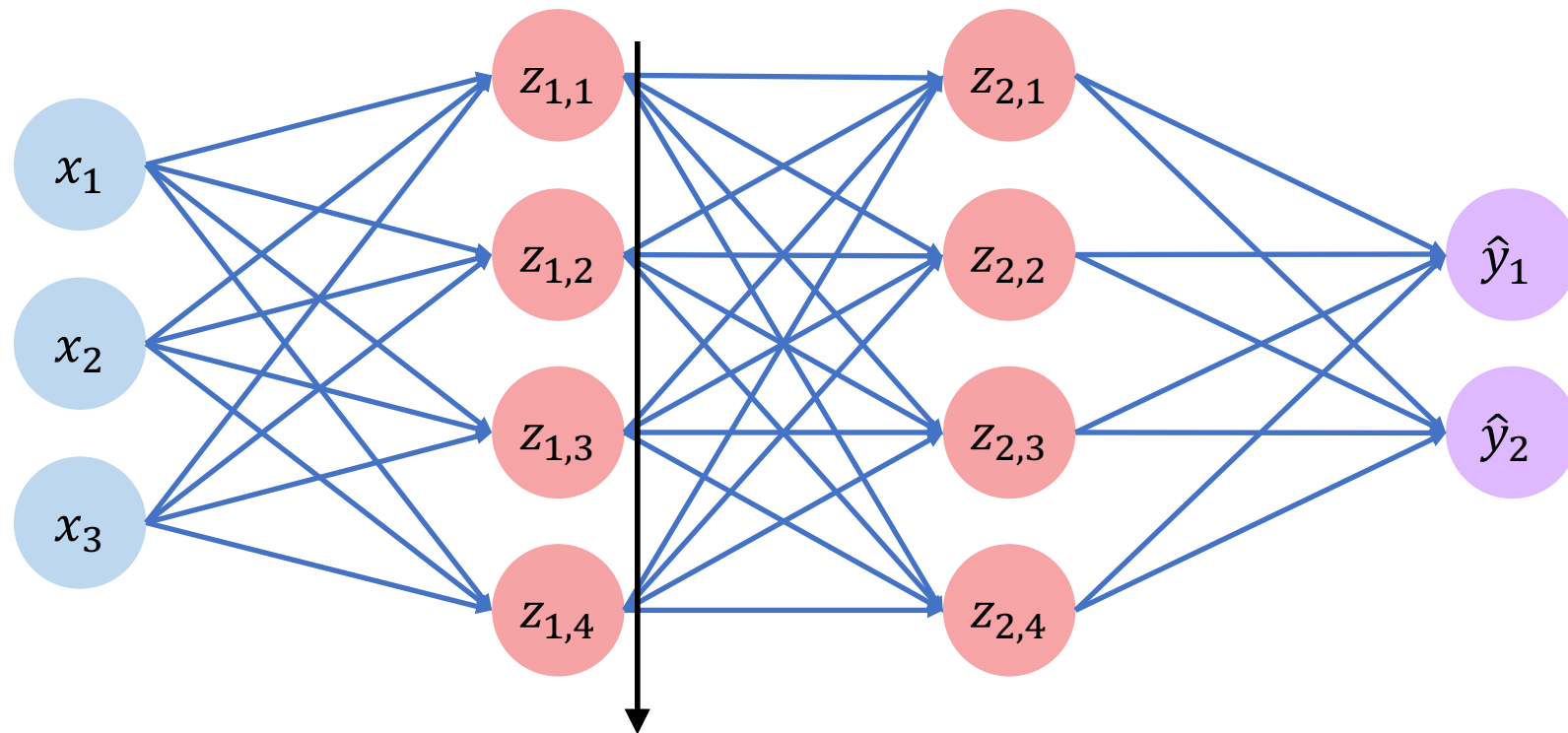
Batch Normalization
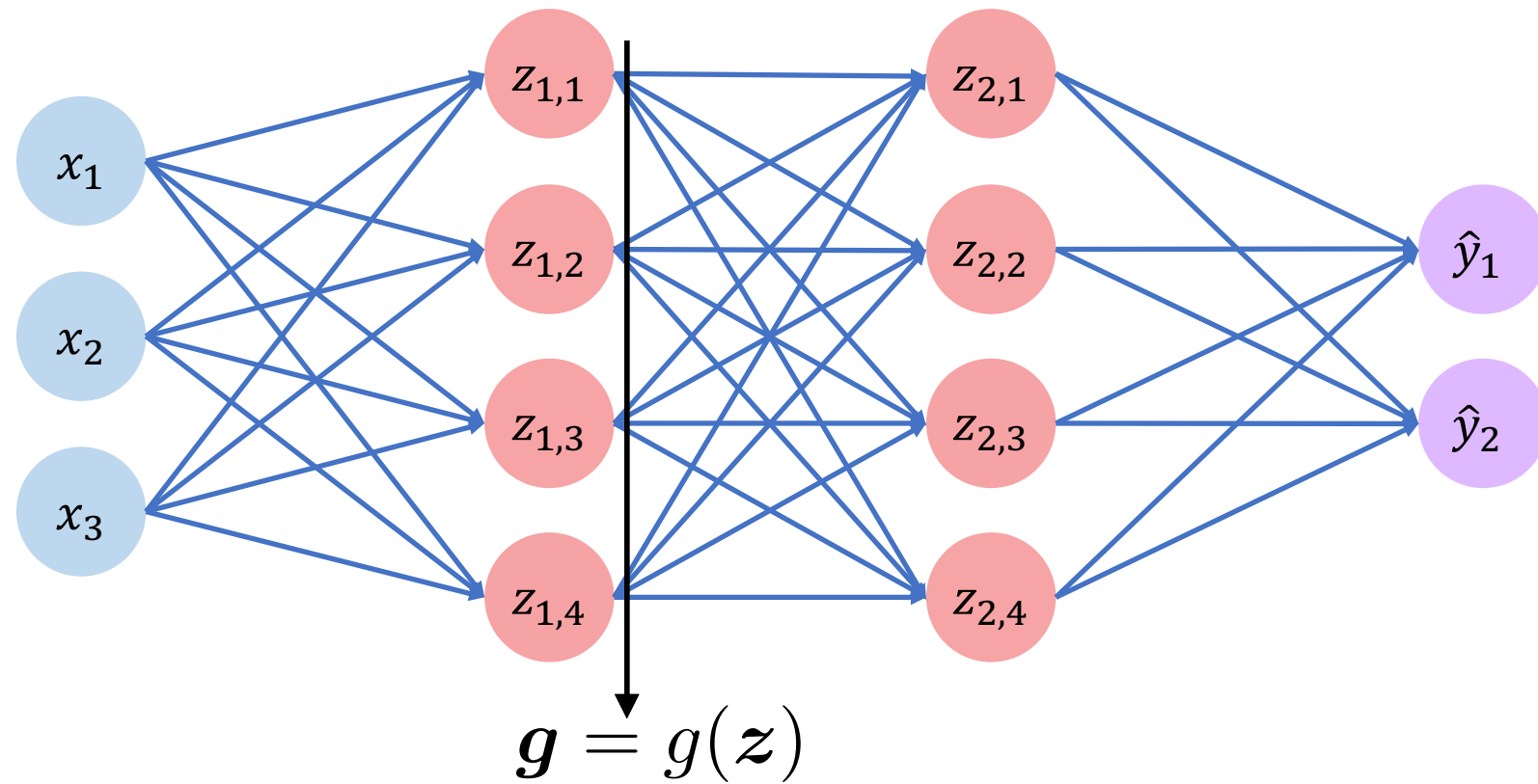
- Mini-batch of size M

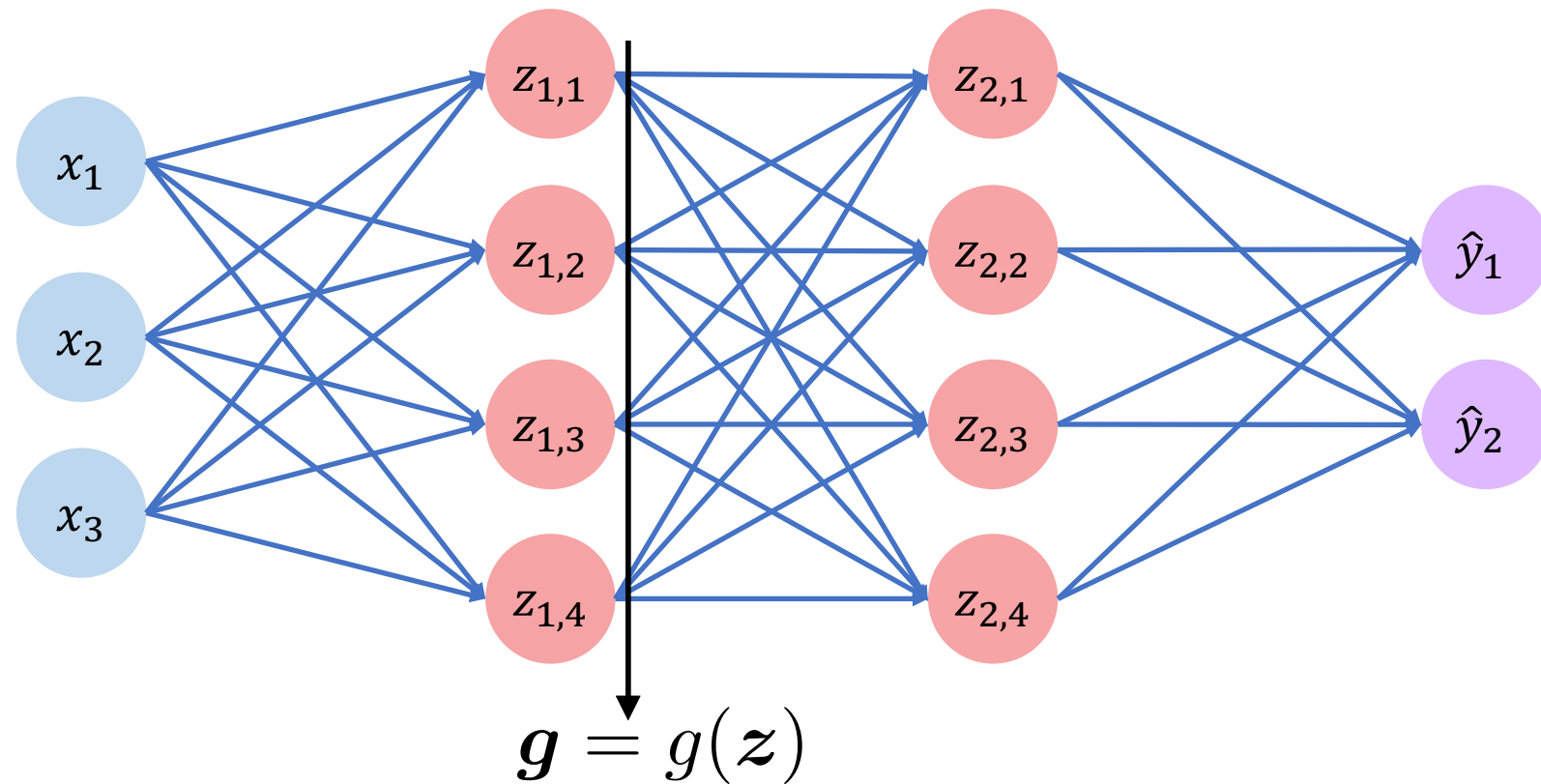# Batch Normalization

- Mini-batch of size M

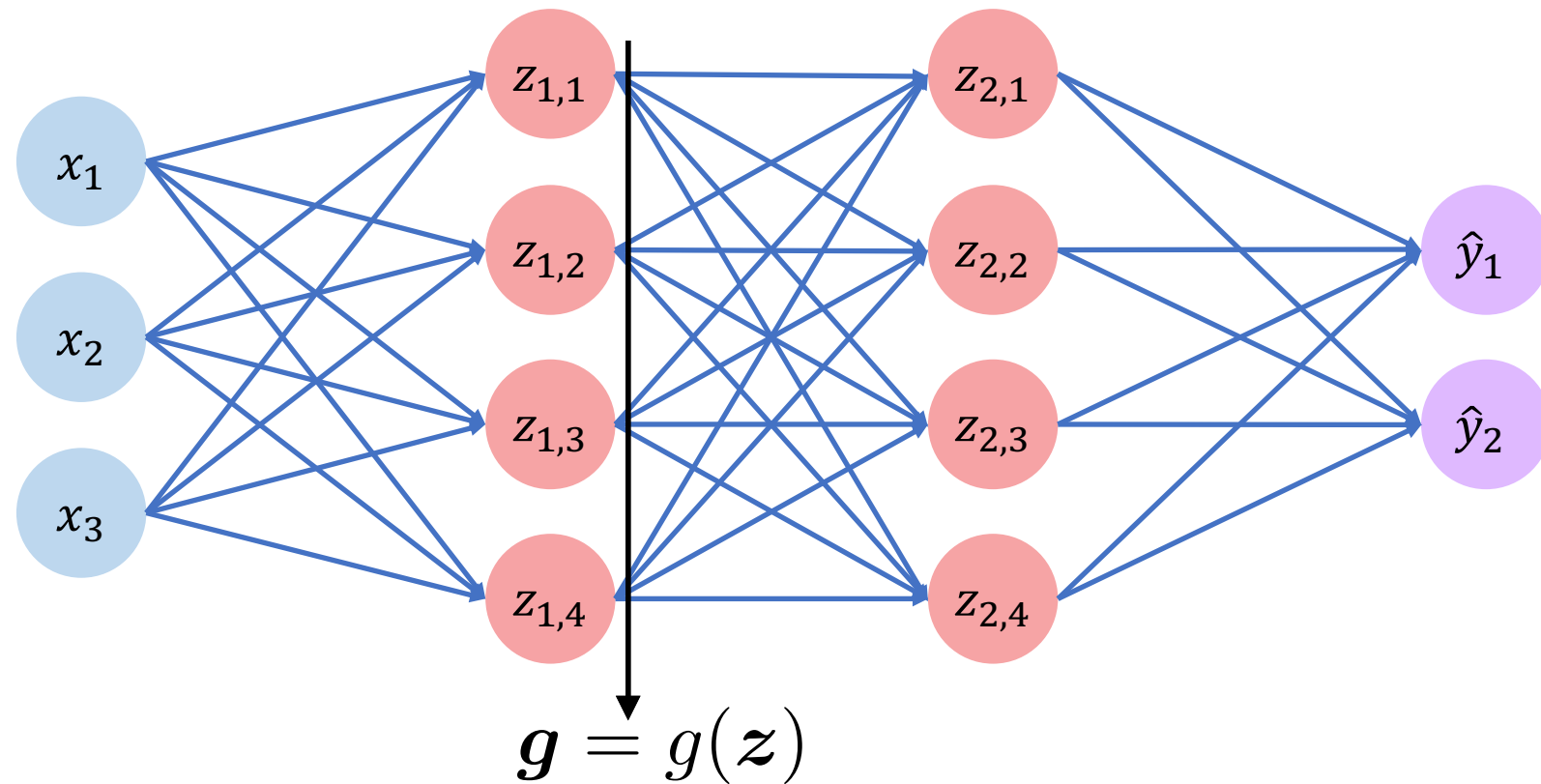# Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{z})$$

Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{z})$$

Mini-batch mean

$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{z})$$

Mini-batch mean
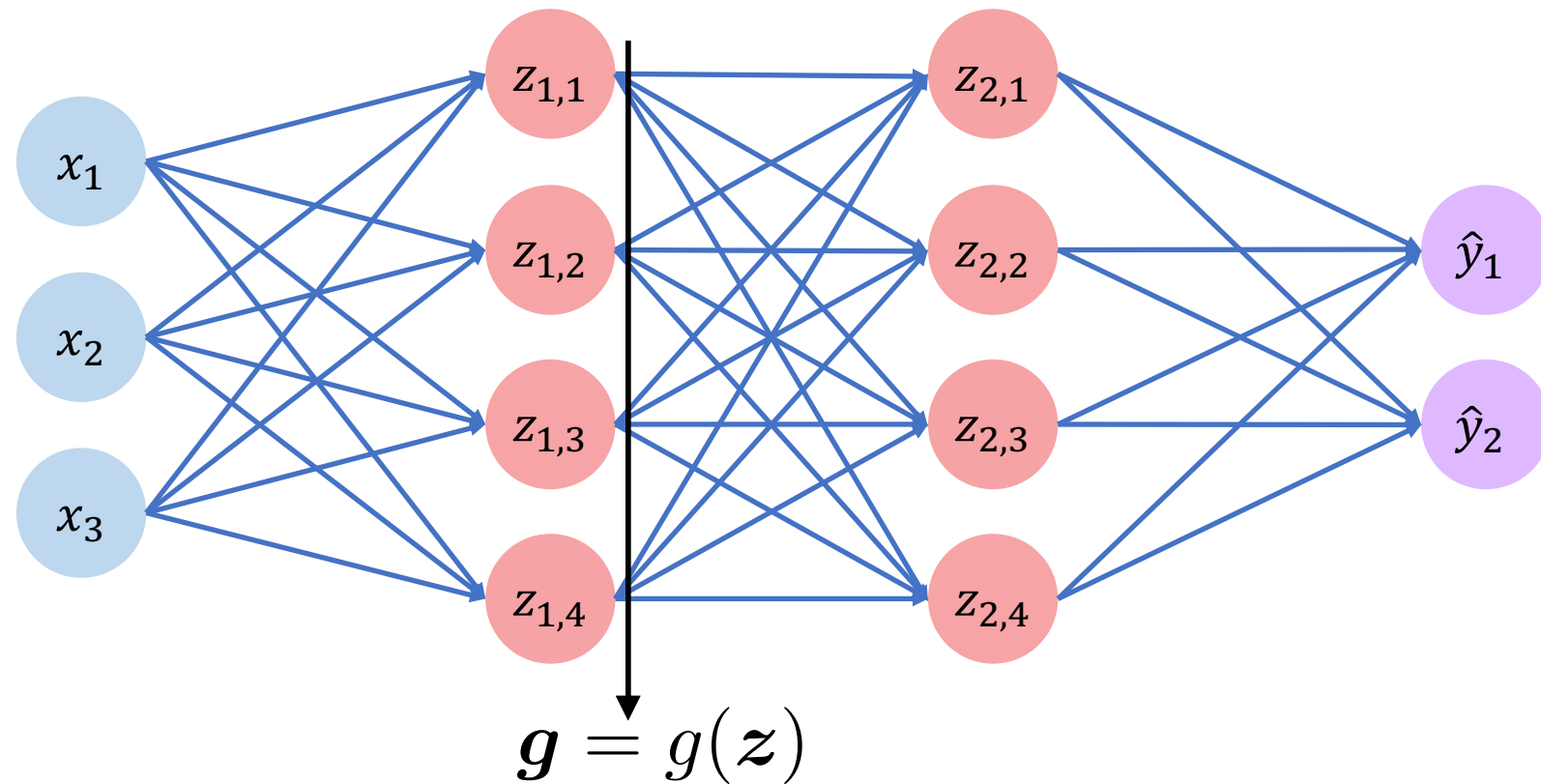
$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Mini-batch variance

$$\sigma_j^2 \leftarrow \frac{1}{M} \sum_{i=1}^{M} (z_j^{(i)} - \mu_j)^2$$

Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{z})$$

Mini-batch mean

$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Mini-batch variance

$$\sigma_j^2 \leftarrow \frac{1}{M} \sum_{i=1}^{M} (z_j^{(i)} - \mu_j)^2$$

Normalize

$$\hat{z}_j^{(i)} = \frac{z_j - \mu_j}{\sqrt{\sigma_j^2}}$$

Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{z})$$

Mini-batch mean

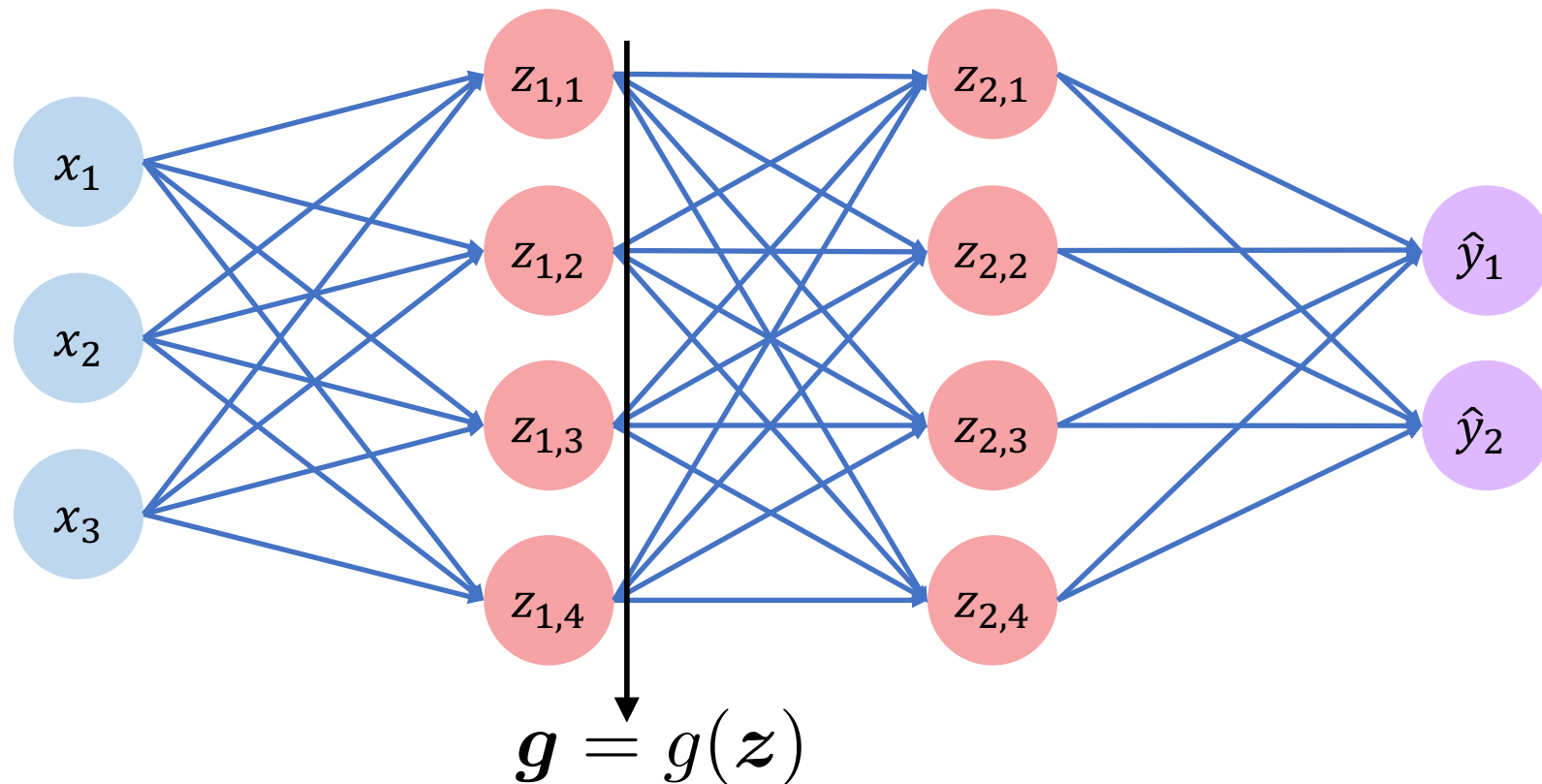$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Mini-batch variance

$$\sigma_j^2 \leftarrow \frac{1}{M} \sum_{i=1}^{M} (z_j^{(i)} - \mu_j)^2$$
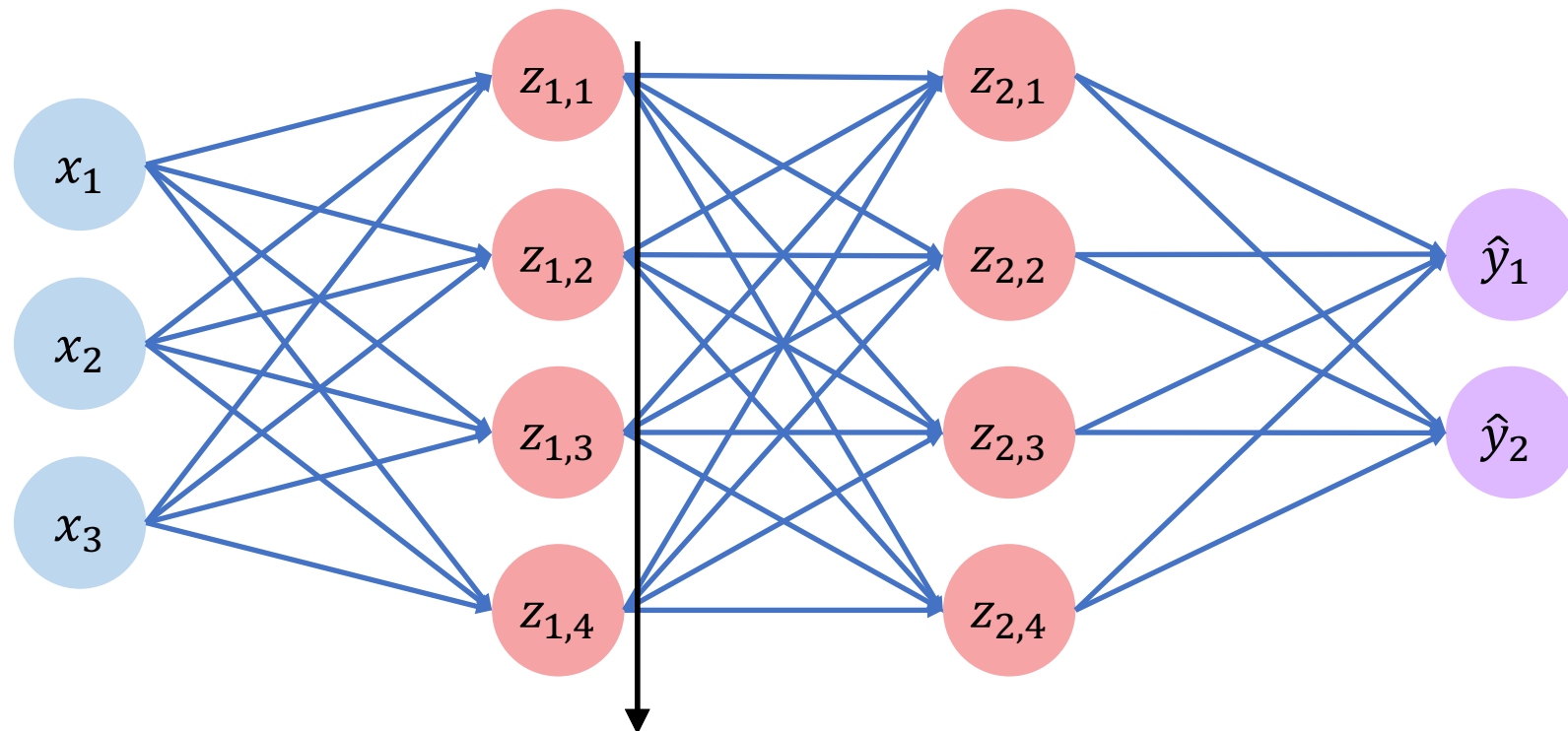
Normalize

$$\hat{z}_j^{(i)} = \frac{z_j - \mu_j}{\sqrt{\sigma_j^2}}$$

Scale and shift

$$a_j^{(i)} = \gamma \hat{z}_j^{(i)} + \beta$$

Batch Normalization

- Mini-batch of size M

Mini-batch mean

$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Mini-batch variance

$$\sigma_j^2 \leftarrow \frac{1}{M} \sum_{i=1}^{M} (z_j^{(i)} - \mu_j)^2$$
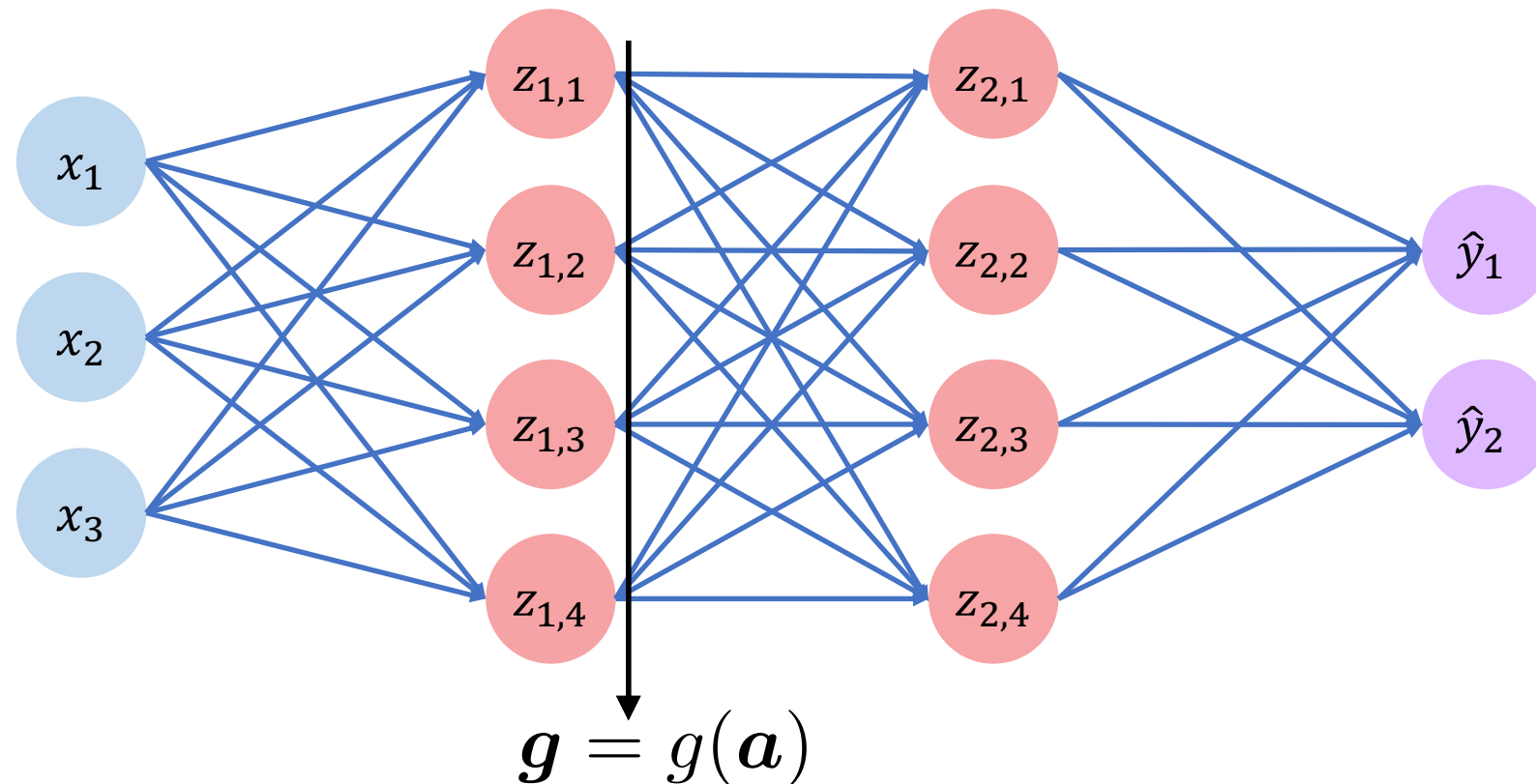
Normalize

$$\hat{z}_j^{(i)} = \frac{z_j - \mu_j}{\sqrt{\sigma_j^2}}$$

Scale and shift

$$a_j^{(i)} = \gamma \hat{z}_j^{(i)} + \beta$$

Batch Normalization

- Mini-batch of size M

$$\boldsymbol{g} = g(\boldsymbol{a})$$

Input to the non-linear activation

Mini-batch mean

$$\mu_j \leftarrow \frac{1}{M} \sum_{i=1}^{M} z_j^{(i)}$$

Mini-batch variance

$$\sigma_j^2 \leftarrow \frac{1}{M} \sum_{i=1}^{M} (z_j^{(i)} - \mu_j)^2$$

Normalize

$$\hat{z}_j^{(i)} = \frac{z_j - \mu_j}{\sqrt{\sigma_j^2}}$$

Scale and shift

$$a_j^{(i)} = \gamma \hat{z}_j^{(i)} + \beta$$

Batch Normalization

- During evaluation, we normalize according to the **whole validation/test datasets**. γ, β parameters are fixed!

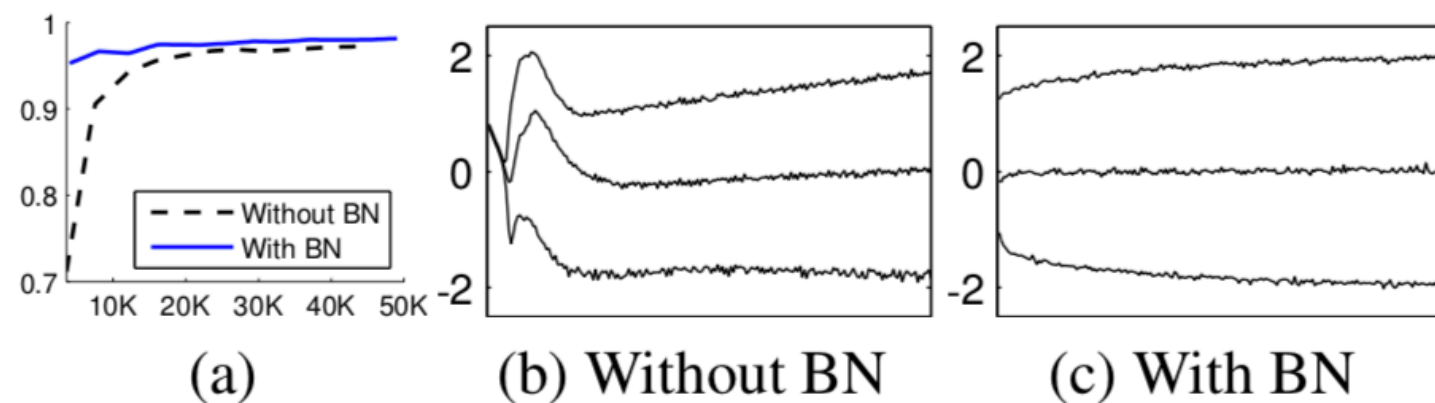- For CNNs, we jointly **normalise all pixels in the same feature map**

Figure 1. (a) *The test accuracy of the MNIST network trained with and without Batch Normalization, vs. the number of training steps. Batch Normalization helps the network train faster and achieve higher accuracy. (b, c) The evolution of input distributions to a typical sigmoid, over the course of training, shown as* $\{15, 50, 85\}$*th percentiles. Batch Normalization makes the distribution more stable and reduces the internal covariate shift.*