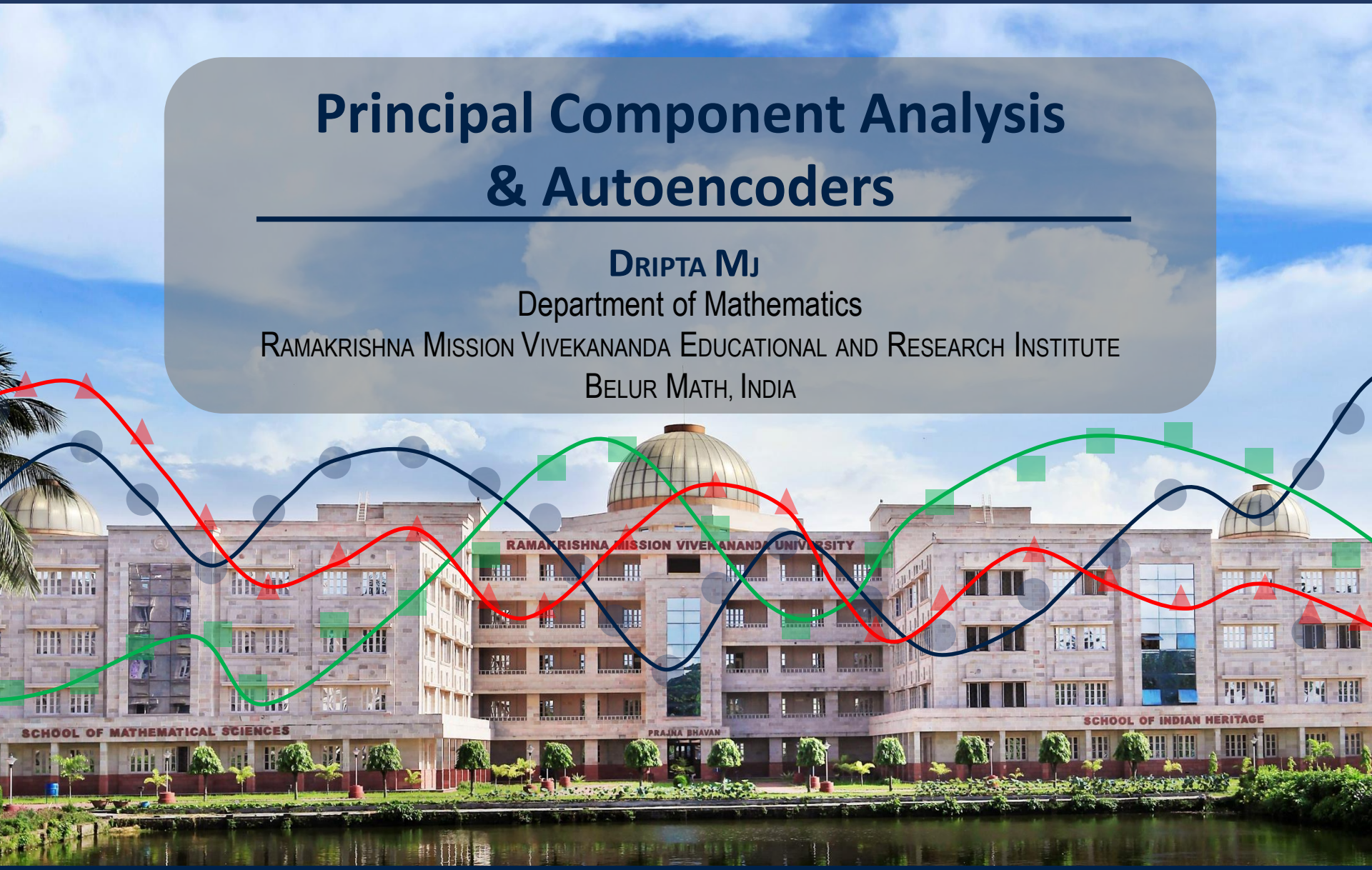
Principal Component Analysis & Autoencoders

Dripta Mj
Department of Mathematics
Ramakrishna Mission Vivekananda Educational and Research Institute
Belur Math, India



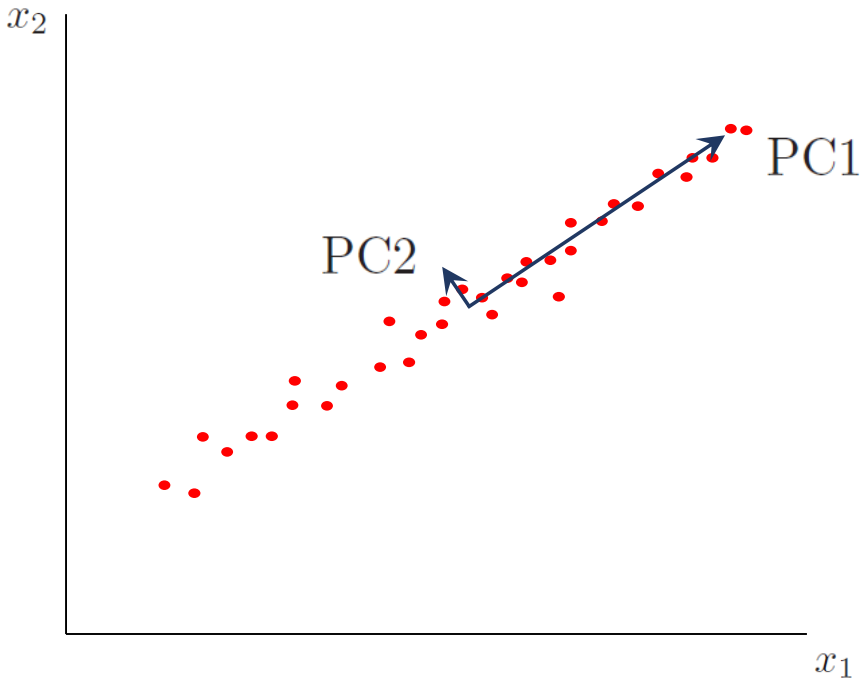
Dimensionality reduction

- Dimensionality reduction is the process of reducing the number of features of a dataset.
- Types: Feature selection, Feature extraction.
- Feature selection: Selects a subset of features.
  - Removes irrelevant features from the dataset.
- Feature extraction: Selects a few combinations of input features that capture most of the variations of the data.
  - Creates new features (through transformation) using existing ones.

Introduction to PCA

- Widely used method for dimensionality reduction.
- Original dataset – large number of interrelated input variables.
- Consider dataset: $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\}$, where $\mathbf{x}^{(n)}$is a D dimensional variable.
- Goal: Represent the data in a lower dimension Q $(< D)$.
  - Transform the data to a new uncorrelated set of variables – the principal components.
  - Extraction of the most informative Q linear combinations which explains the data.
  - This is the projection of the data in D dimensions onto a lower-dimensional subspace.
- Orthogonal projection of data onto a lower dimensional (linear) space, such that the variance of the projected data is maximized.

Principal components



- PC1: Direction of most variation
- PC2: Direction of second most variation orthogonal to PC1

Dataset

- Consider dataset: $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\}$, where $\mathbf{x}^{(n)}$is a D dimensional variable.

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & . & . & x_1^{(N)} \\ x_2^{(1)} & x_2^{(2)} & . & . & x_2^{(N)} \\ . & . & . & . & . \\ . & . & . & . & . \\ x_D^{(1)} & . & . & . & x_D^{(N)} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ . \\ . \\ \mathbf{x}_D \end{bmatrix}$$

- Want a lower-dimensional $(Q < D)$ representation of the data:

$$\mathbf{Z} = \begin{bmatrix} z_1^{(1)} & z_1^{(2)} & . & . & z_1^{(N)} \\ . & . & . & . & . \\ z_Q^{(1)} & . & . & . & z_Q^{(N)} \end{bmatrix}$$

Variance

- Consider a vector $\mathbf{x} = [x_1, x_2, ..., x_N]$ having a mean value of 0.

- The variance of the vector $\mathbf{x}$ can be computed as

$$\sigma_{\mathbf{x}}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - 0)(x_i - 0)$$

$$= \frac{1}{N-1}\mathbf{x}\mathbf{x}^{\mathrm{T}}$$

Covariance

- Now consider two vectors: $\mathbf{x} = [x_1, x_2, ..., x_N]$ and $\mathbf{z} = [z_1, z_2, ..., z_N]$, both having mean 0.
- The covariance between vectors $\mathbf{x}$ and $\mathbf{z}$ can be computed as

$$\sigma^2_{\mathbf{xz}} = \frac{1}{N-1}\mathbf{x}\mathbf{z}^{\mathrm{T}}$$

  - Covariance measures the correlation between variables.
- If $\sigma^2_{\mathbf{xz}} \approx 0$ then $\mathbf{x}$ and $\mathbf{z}$ are almost uncorrelated.

Covariance matrix

- Assume data is centered.
- The covariance matrix $\mathbf{S}$ can be obtained as:

$$\mathbf{S} = \frac{1}{N-1}\mathbf{X}\mathbf{X}^{\mathrm{T}}.$$

- Can write the covariance matrix as

$$\mathbf{S} = \begin{bmatrix} \sigma^2_{\mathbf{x}_1} & \sigma^2_{\mathbf{x}_1\mathbf{x}_2} & \cdot & \cdot & \sigma^2_{\mathbf{x}_1\mathbf{x}_D} \\ \sigma^2_{\mathbf{x}_2\mathbf{x}_1} & \sigma^2_{\mathbf{x}_2} & \cdot & \cdot & \sigma^2_{\mathbf{x}_1\mathbf{x}_D} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \sigma^2_{\mathbf{x}_D\mathbf{x}_1} & \cdot & \cdot & \cdot & \sigma^2_{\mathbf{x}_D} \end{bmatrix}$$

- The i-th diagonal term corresponds to the variance in the i-th dimension of the problem.
- The off-diagonal terms are the covariances.
- Small off-diagonal term indicates that the variables are almost uncorrelated.
- $\mathbf{S}$ is symmetric.

Covariance matrix

- Want to transform the covariance matrix $\mathbf{S}$ to $\mathbf{S_Z}$ that has the following form:

$$\mathbf{S_Z} = \begin{bmatrix} \sigma^2_{\mathbf{Z}_1} & 0 & \cdot & \cdot & 0 \\ 0 & \sigma^2_{\mathbf{Z}_2} & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \sigma^2_{\mathbf{Z}_D} \end{bmatrix}$$

- The transformed matrix $\mathbf{S_Z}$ has no correlation between the different dimensions.
- Can order the variances such that: $\sigma^2_{\mathbf{Z}_1} \geq \sigma^2_{\mathbf{Z}_2} \geq .... \geq \sigma^2_{\mathbf{Z}_D}$.
- So $\sigma^2_{\mathbf{Z}_1}$ is the largest variance, and the dimension corresponding to it is known as the first principal component.
- Similarly $\sigma^2_{\mathbf{Z}_2}$ is the variance of the second principal component.

Eigenvalue decomposition

- Eigenvalue decomposition of the covariance matrix $\mathbf{S}$:

$$\mathbf{S} = \mathbf{V}\Lambda\mathbf{V}^{-1}$$

  where Λ is a diagonal matrix, $\mathbf{V}$ is a matrix of eigenvectors of $\mathbf{S}$ with columns corresponding to right eigenvectors of $\mathbf{S}$.

- The diagonal elements of Λ are the eigenvalues of $\mathbf{S}$ for the corresponding eigenvectors.

- Since $\mathbf{S}$ is symmetric, the eigenvalues are real and the eigenvectors are orthogonal to each other.

- The eigenvectors can be made orthonormal by taking $\mathbf{V}\mathbf{V}^{\mathrm{T}} = \mathbf{I}$.

Linear transformation

- Consider the following linear transformation of the original data $\mathbf{X}$ into $\mathbf{Z}$:

$$\mathbf{Z} = \mathbf{V}^{\mathrm{T}}\mathbf{X}$$

$$\begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ \mathbf{z}^{(1)} & \mathbf{z}^{(2)} & \cdots & \mathbf{z}^{(N)} \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_D \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \uparrow & \uparrow & \cdots & \uparrow \\ \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \cdots & \mathbf{x}^{(N)} \\ \downarrow & \downarrow & \cdots & \downarrow \end{bmatrix}$$

Linear transformation

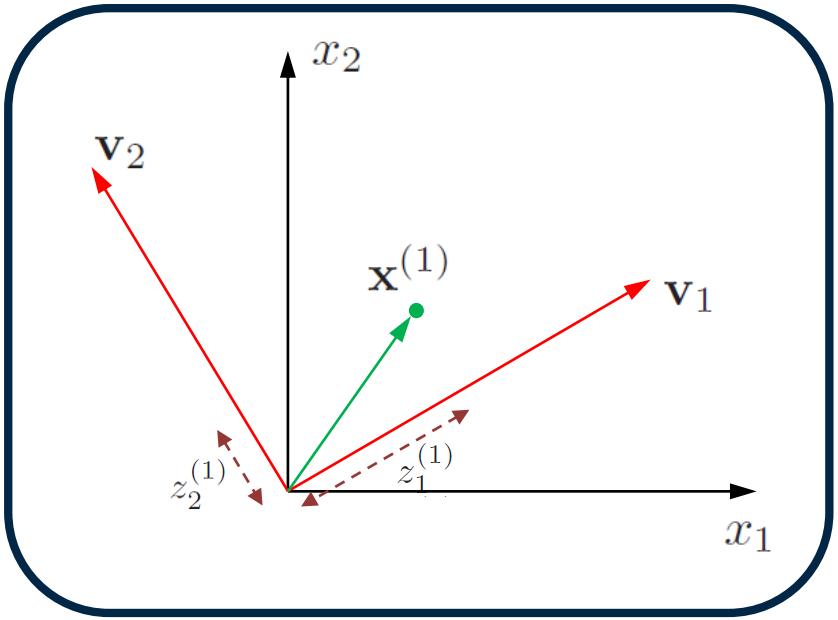
- Consider the following linear transformation of the original data $\mathbf{X}$ into $\mathbf{Z}$:

$$\mathbf{Z} = \mathbf{V}^{\mathrm{T}}\mathbf{X}$$

- Consider a 2D example where the transformation is applied to a single data point $\mathbf{x}^{(1)}$

$$\begin{bmatrix} \\ \mathbf{z}^{(1)} \\ \\ \end{bmatrix} = \begin{bmatrix} \longleftarrow & \mathbf{v}_1 & \longrightarrow \\ \longleftarrow & \mathbf{v}_2 & \longrightarrow \end{bmatrix} \begin{bmatrix} \\ \mathbf{x}^{(1)} \\ \\ \end{bmatrix}$$

$$\begin{bmatrix} z_1^{(1)} \\ z_2^{(1)} \end{bmatrix}$$



Linear transformation

- Consider the following linear transformation of the original data $\mathbf{X}$ into $\mathbf{Z}$:

$$\mathbf{Z} = \mathbf{V}^{\mathrm{T}}\mathbf{X}$$

- The covariance of $\mathbf{Z}$ can be obtained as:

$$\begin{aligned}
\mathbf{S}_{\mathbf{Z}} &= \frac{1}{N-1}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} \\
&= \frac{1}{N-1}(\mathbf{V}^{\mathrm{T}}\mathbf{X})(\mathbf{V}^{\mathrm{T}}\mathbf{X})^{\mathrm{T}} \\
&= \frac{1}{N-1}(\mathbf{V}^{\mathrm{T}}\mathbf{X})(\mathbf{X}^{\mathrm{T}}\mathbf{V}) \\
&= \frac{1}{N-1}\mathbf{V}^{\mathrm{T}}(\mathbf{X}\mathbf{X}^{\mathrm{T}})\mathbf{V} \\
&= \mathbf{V}^{\mathrm{T}}\Big(\frac{1}{N-1}\mathbf{X}\mathbf{X}^{\mathrm{T}}\Big)\mathbf{V} \\
&= \mathbf{V}^{\mathrm{T}}\mathbf{S}\mathbf{V}
\end{aligned}$$

Covariance matrix

- Consider the following linear transformation of the original data $\mathbf{X}$ into $\mathbf{Z}$:

$$\mathbf{Z} = \mathbf{V}^{\mathrm{T}}\mathbf{X}$$

- The covariance of $\mathbf{Z}$ can be obtained as:

$$\begin{aligned}\mathbf{S_Z} &= \mathbf{V}^{\mathrm{T}}\mathbf{V}\Lambda\mathbf{V}^{-1}\mathbf{V} \\ &= (\mathbf{V}^{\mathrm{T}}\mathbf{V})\Lambda(\mathbf{V}^{\mathrm{T}}\mathbf{V}) \qquad (\mathbf{V}^{-1} = \mathbf{V}^{\mathrm{T}} \text{ as } \mathbf{V}\mathbf{V}^{\mathrm{T}} = \mathbf{I}\ ) \\ &= \Lambda\end{aligned}$$

- The covariance matrix $\mathbf{S_Z}$ is diagonal as Λ is diagonal.

Diagonal covariance matrix

- So we have

$$\mathbf{S_Z} = \Lambda = \begin{bmatrix} \sigma^2_{\mathbf{Z}_1} & 0 & \cdot & \cdot & 0 \\ 0 & \sigma^2_{\mathbf{Z}_2} & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \sigma^2_{\mathbf{Z}_D} \end{bmatrix}$$

- The diagonal terms of $\mathbf{S_Z}$ correspond to variances along the dimensions of the transformed vector space.
- Note, the diagonal matrix Λ comprise the eigenvalues of $\mathbf{S}$.
- The variances along the projected dimensions (eigenvectors of $\mathbf{S}$) are the corresponding eigenvalues of $\mathbf{S}$.

Percentage of variance

- The percentage of variance explained by the jth principal component:

$$PV_j = \frac{\lambda_j}{\sum_{i=1}^{D} \lambda_i} \times 100$$

- The percentage of variance accounted for by the first Q principal components is given by:

$$PV = \frac{\sum_{i=1}^{Q} \lambda_i}{\sum_{i=1}^{D} \lambda_i} \times 100$$

AUTOENCODERS

Introduction

SUPERVISED LEARNING

NETWORK for Feature Discovery

z_1 z_2 ... z_H

NETWORK for Classification

DOG/CAT

SELF-SUPERVISED LEARNING

NETWORK for Feature Discovery

z_1 z_2 ... z_H

NETWORK

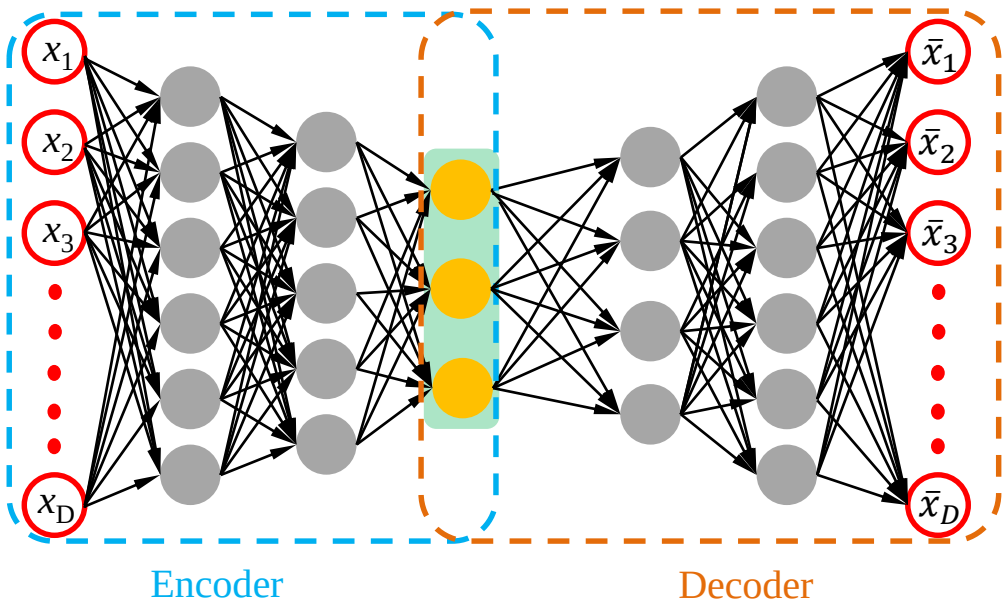
Introduction

- Neural network models that attempts to yield outputs same as that of the inputs.
- Why do we do this?
  - Want to learn the most important characteristics of the input data.
- Network comprise two components:
  - Encoder: Takes input $\mathbf{x}$ and generates a hidden representation

$$\mathbf{h} = \mathbf{f}(\mathbf{w}_e\mathbf{x} + \mathbf{w}_{0,e})$$

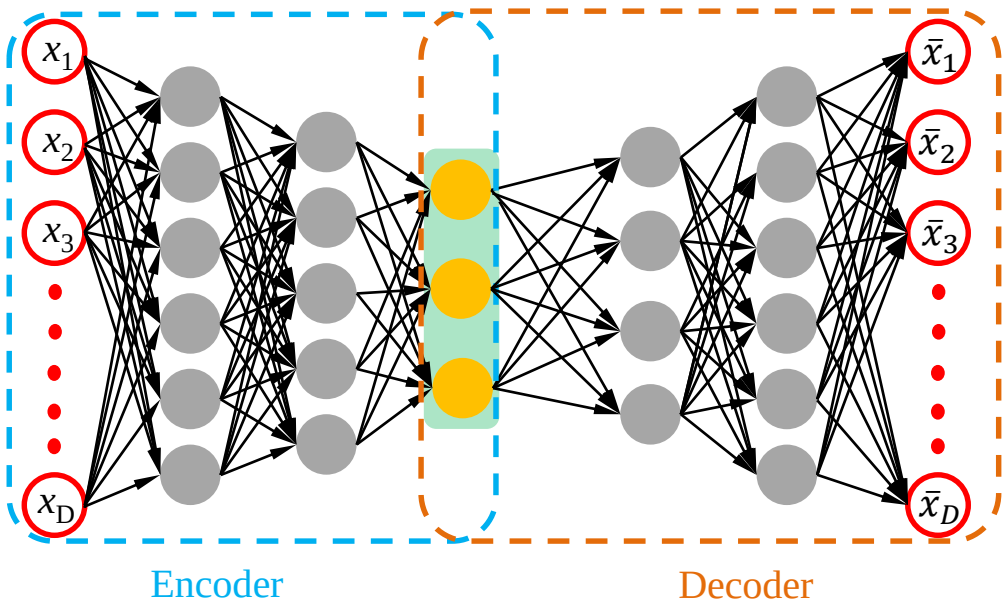
  - Decoder: Takes input $\mathbf{h}$ and outputs $\overline{\mathbf{x}} = g(\mathbf{w}_d\mathbf{h} + \mathbf{w}_{0,d})$, where $\overline{\mathbf{x}}$ is the reconstruction of $\mathbf{x}$.
- State-of-the-art autoencoders use stochastic mappings $p_{encoder}(\mathbf{h}|\mathbf{x})$ and $p_{decoder}(\mathbf{x}|\mathbf{h})$.

Undercomplete Autoencoder



- Such a representation compels the autoencoder to capture the most important features of the data.
- Loss function $L(\mathbf{x}, \overline{\mathbf{x}})$ penalizes $\overline{\mathbf{x}}$ if it is dissimilar to $\mathbf{x}$.
- The autoencoder learns the same subspace as PCA when:
  - the decoder is linear
  - the loss function is mean squared error

Undercomplete Autoencoder

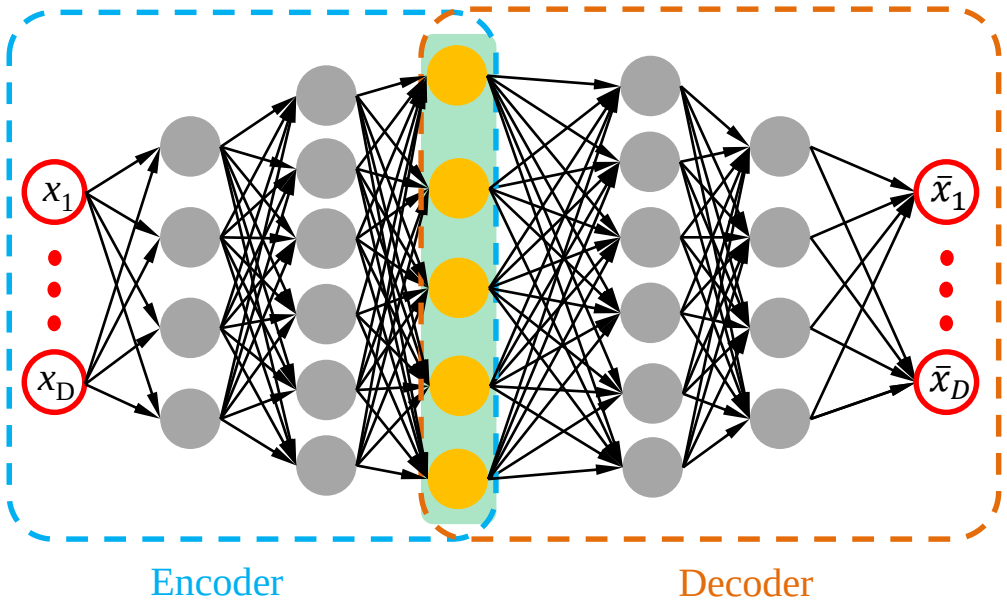


- Loss function (for real values) is (often) taken to be

$$L(\mathbf{x}, \overline{\mathbf{x}}) = \left|\left|\mathbf{x} - \overline{\mathbf{x}}\right|\right|^2$$

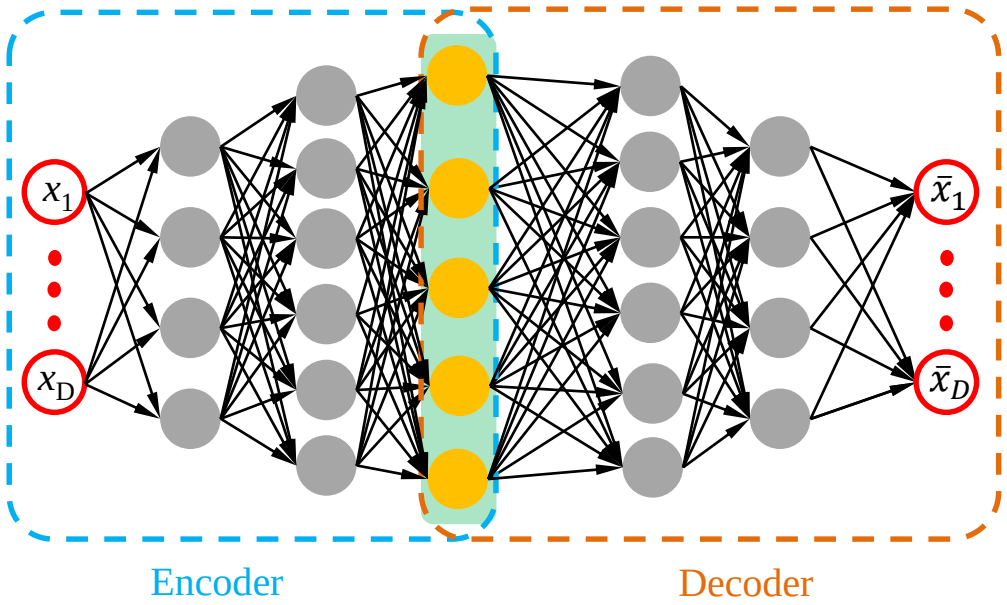
  - Also referred to as the reconstruction error.

Overcomplete Autoencoder



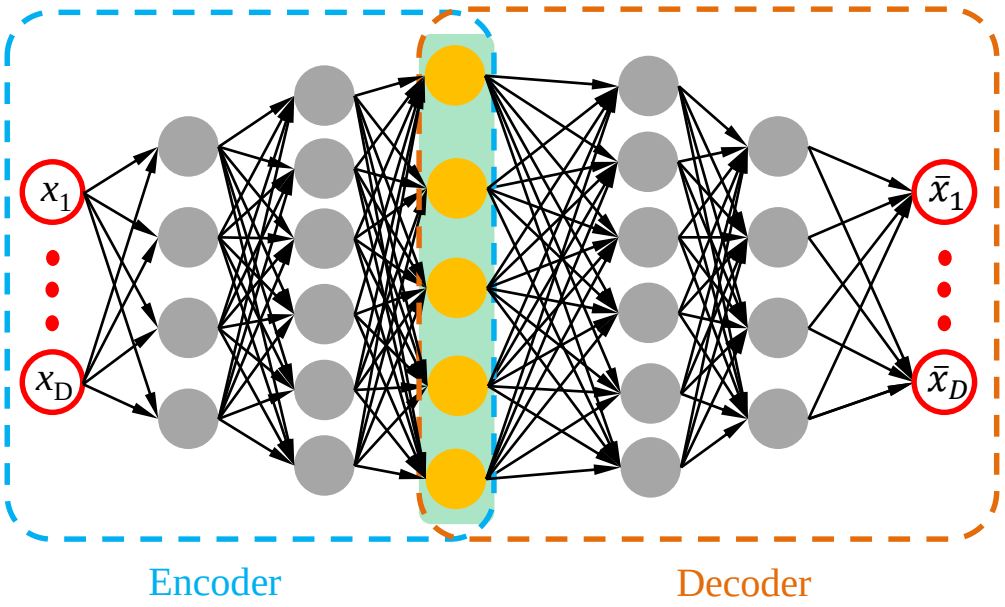
- Dimension of the hidden representation $\mathbf{h}$ is greater than equal to that of the input $\mathbf{x}$.
- May lead to trivial encoding where $\mathbf{x}$ is copied into $\mathbf{h}$, and then decoder copies $\mathbf{h}$ to $\overline{\mathbf{x}}$. Such mappings can be learnt using simple linear encoder and decoder.
  - Do not learn anything useful about the training data.

Regularized Autoencoder



- Overcomplete encoders are prone to overfitting due to the use of large number of parameters.
  - The model can just copy $\mathbf{x}$ to $\mathbf{h}$ and then $\mathbf{h}$ to $\overline{\mathbf{x}}$.
  - This can lead to poor generalization.
- Overfitting can also occur in case of undercomplete autoencoders.
  - For example, when there is a lot of redundancy in the input data, then the reduced dimension of the hidden representation may not be sufficient to remove all the redundancies.

Regularized Autoencoder

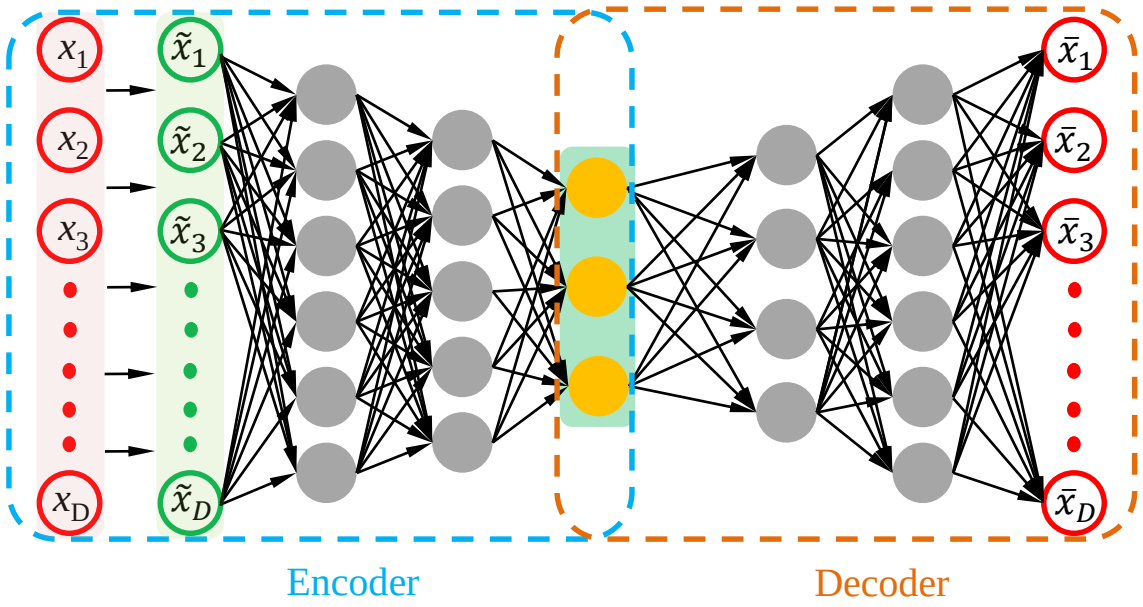


- Need regularization to overcome the issue.
- L_2 regularization objective:

$$L(\mathbf{x}, \overline{\mathbf{x}}) + \lambda ||\mathbf{w}||^2$$

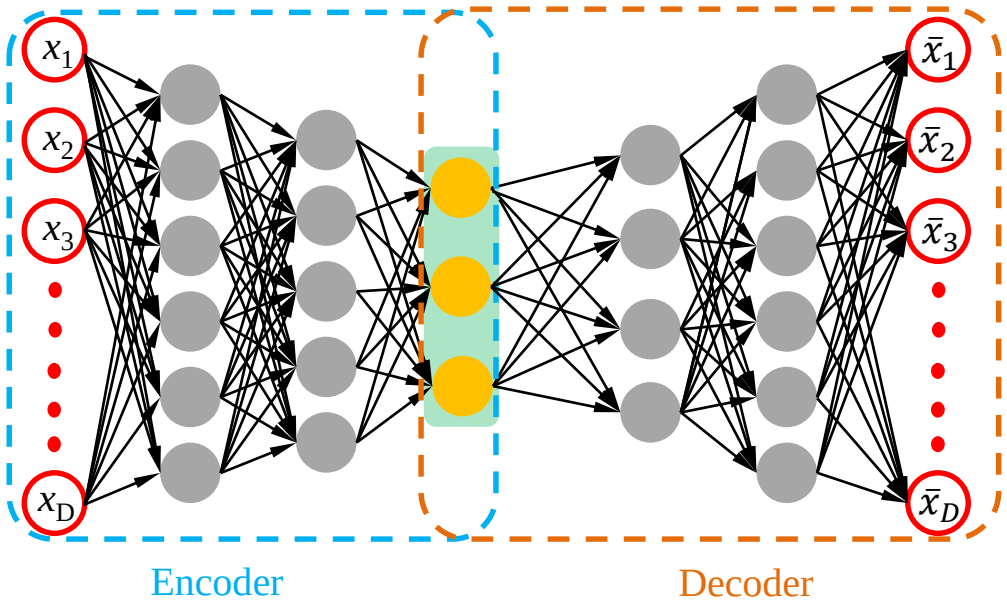
- Tie the weights of encoder and decoder: $\mathbf{w}_e^{\mathrm{T}} = \mathbf{w}_d^{\mathrm{T}}$.

Denoising Autoencoder



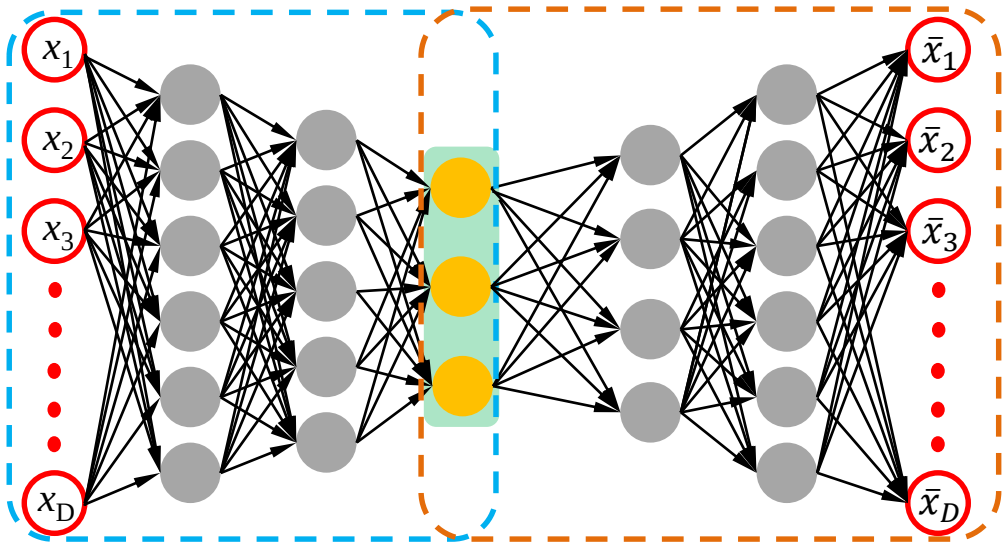
- Input $\mathbf{x}$ corrupted by some form of noise.
- Denoising autoencoders need to undo the corruption that has been done to the input.
  - Therefore the autoencoder cannot simply copy $\mathbf{x}$ to $\mathbf{h}$ and then $\mathbf{h}$ to $\overline{\mathbf{x}}$.
- The objective is still to reconstruct the original input $\mathbf{x}$. The loss function is $L(\mathbf{x}, \overline{\mathbf{x}})$.
  - The model is forced to capture the important characteristics of the data.

Sparse Autoencoder



- In sparse autoencoders, the hidden neurons are constrained such that they remain mostly "inactive".
  - Here "inactive" means that the outputs of the neurons will be 0 for sigmoid activation.
- By imposing this constraint an attempt is made to ensure that whenever a neuron is "active" then that neuron is capturing some really important characteristic/pattern in the data.

Sparse Autoencoder



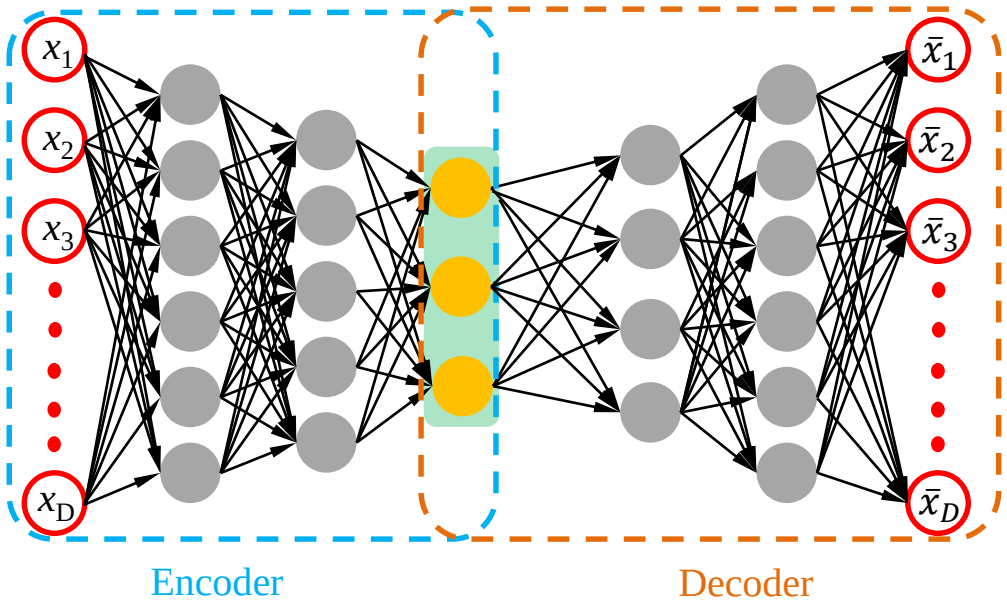
- The average value of activation of the kth neuron using the N training examples can be written as

$$\overline{\mu}_k = \frac{1}{N} \sum_{n=1}^{N} h_k(\mathbf{x}^{(n)})$$

- The kth neuron is sparse if the value of $\overline{\mu}_k$ is close to zero.
- Sparse autoencoders use a sparsity penalty $\Omega(\mathbf{h})$ on the hidden representation $\mathbf{h}$ in addition to the reconstruction error, and so the optimization objective becomes

$$L(\mathbf{x}, \overline{\mathbf{x}}) + \Omega(\mathbf{h})$$

Sparse Autoencoder



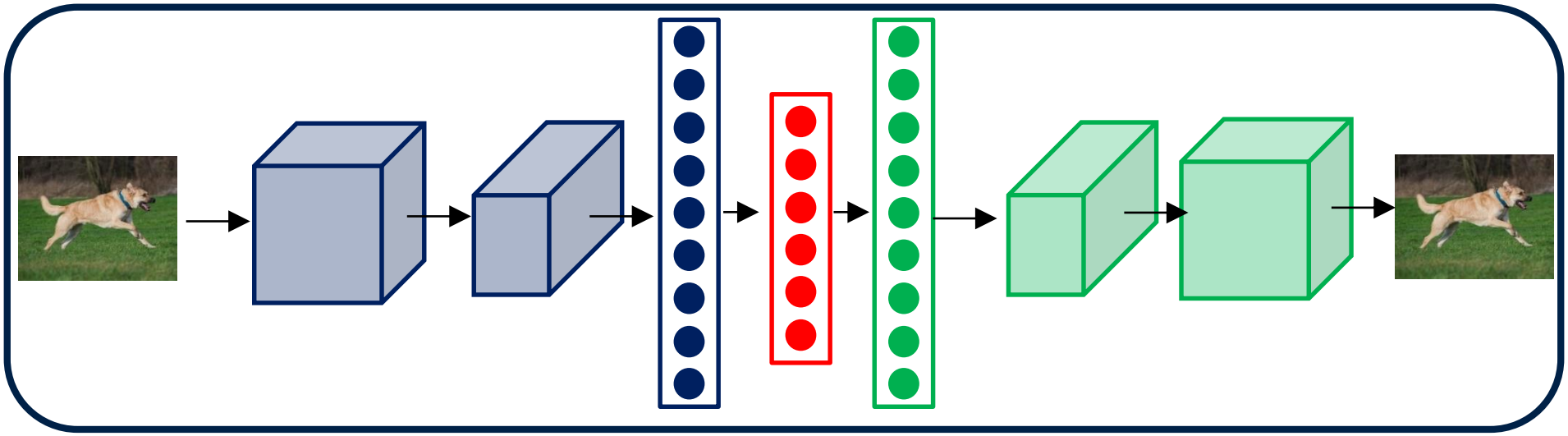
- The penalty term can take the following form:

$$\Omega(\mathbf{h}) = \sum_{k=1}^{K} \mu \log\left(\frac{\mu}{\overline{\mu}_k}\right) + (1-\mu)\log\left(\frac{1-\mu}{1-\overline{\mu}_k}\right)$$

where μ is the sparsity parameter whose value is close to 0.

– $\Omega(\mathbf{h})$ is minimum when $\overline{\mu}_k = \mu$.

Convolutional Autoencoder



- Convolutional Neural Networks are well suited for image datasets.
  - Why not use convolutional layers in autoencoders when the inputs are images
- The learnt hidden features can further be used for other purposes e.g. classification
  - The learnt representation can be used as initialization for other models.
  - Can be very useful for problems with lot of unlabelled data.