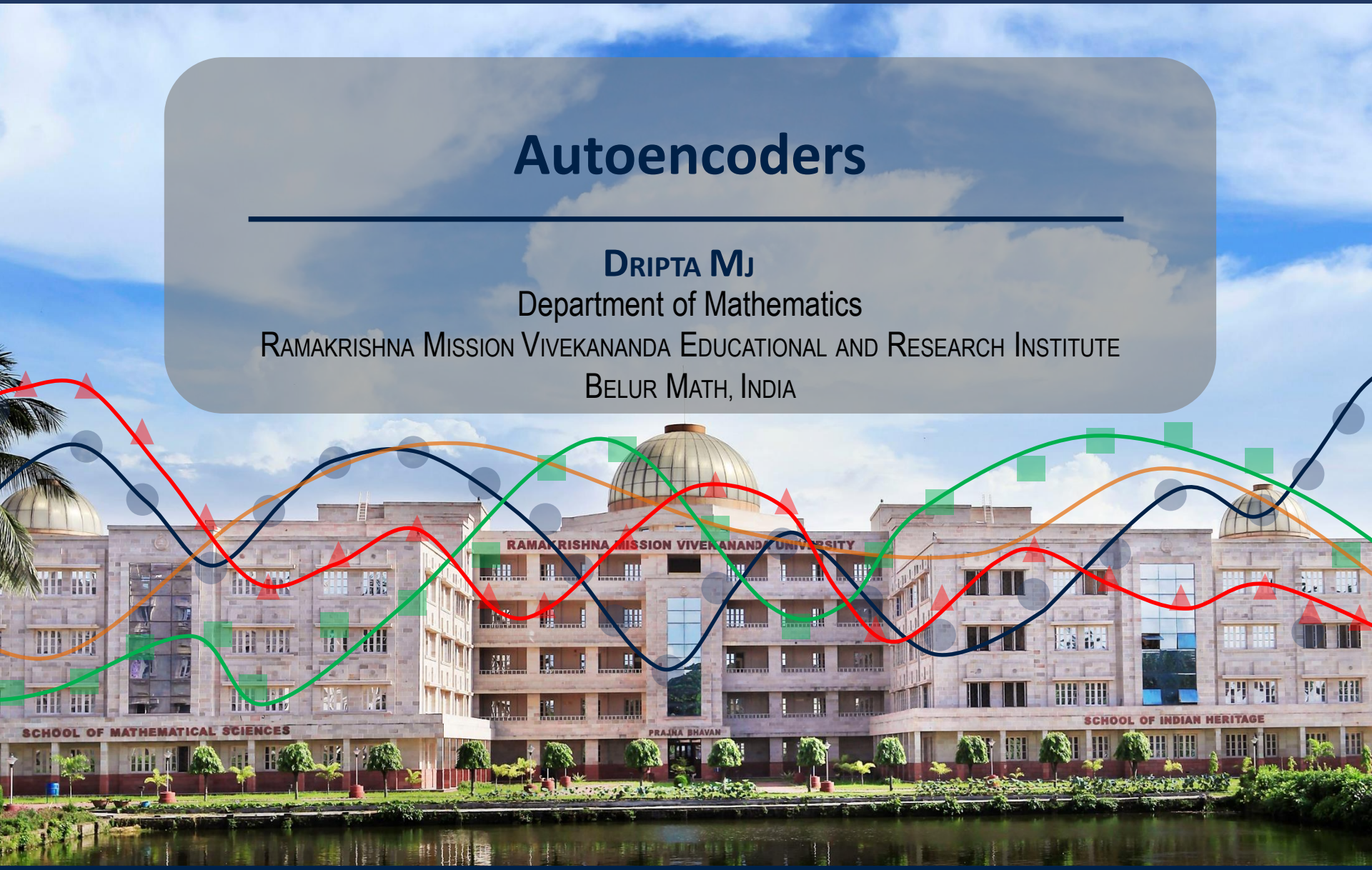
Autoencoders

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



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

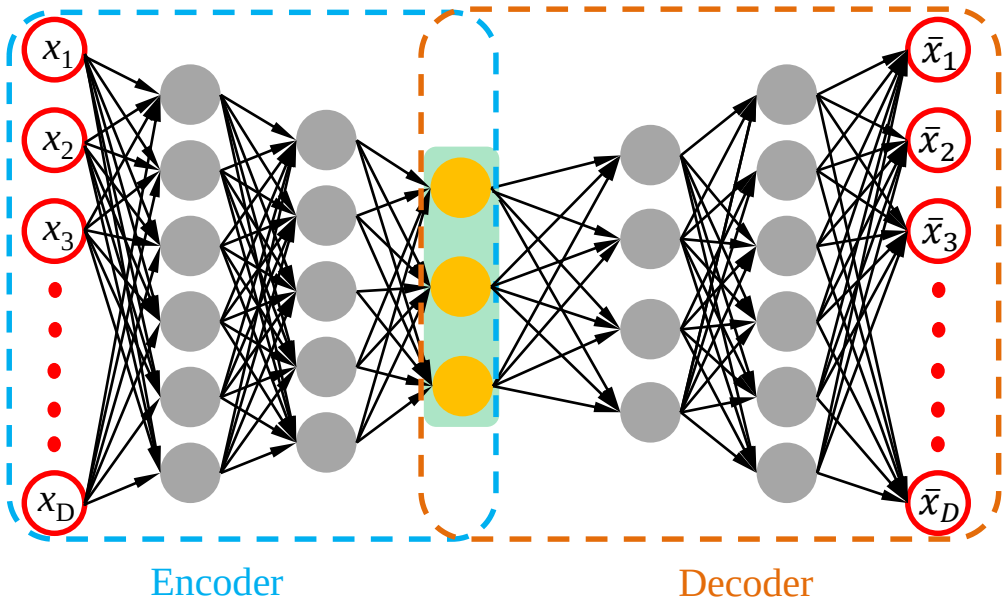
Introduction

- 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 $p_{decoder}(\mathbf{x}|\mathbf{h})$.

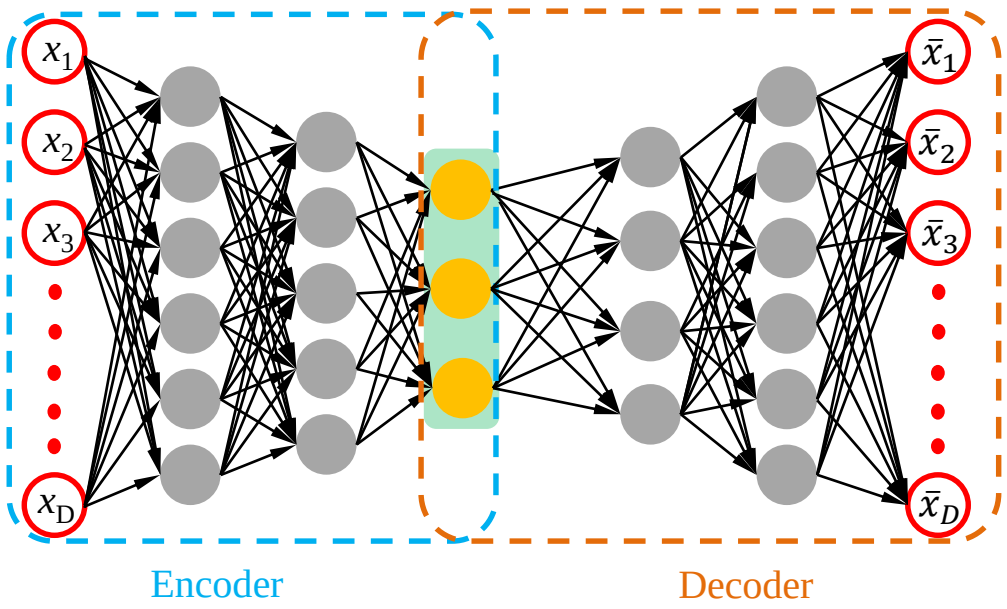
Undercomplete Autoencoder



- 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

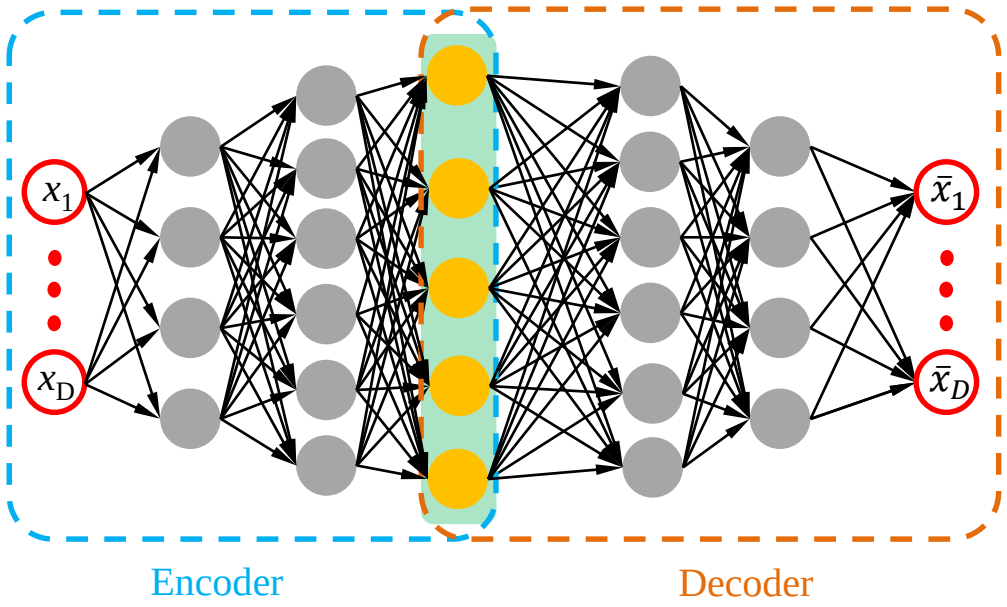


- Loss function (for real values) is (often) taken to be

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

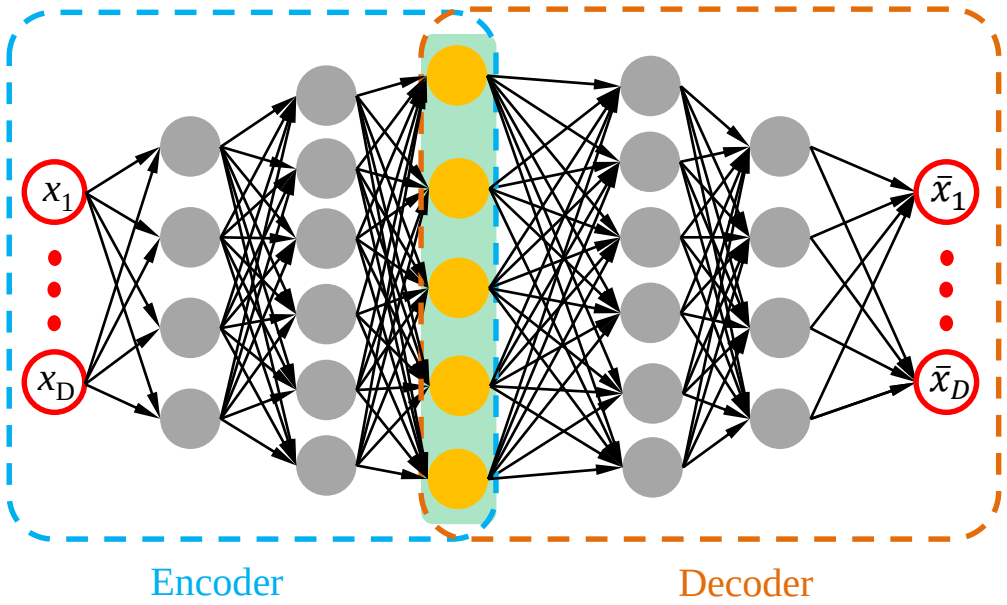
  - Also referred to as the reconstruction error.

Overcomplete Autoencoder



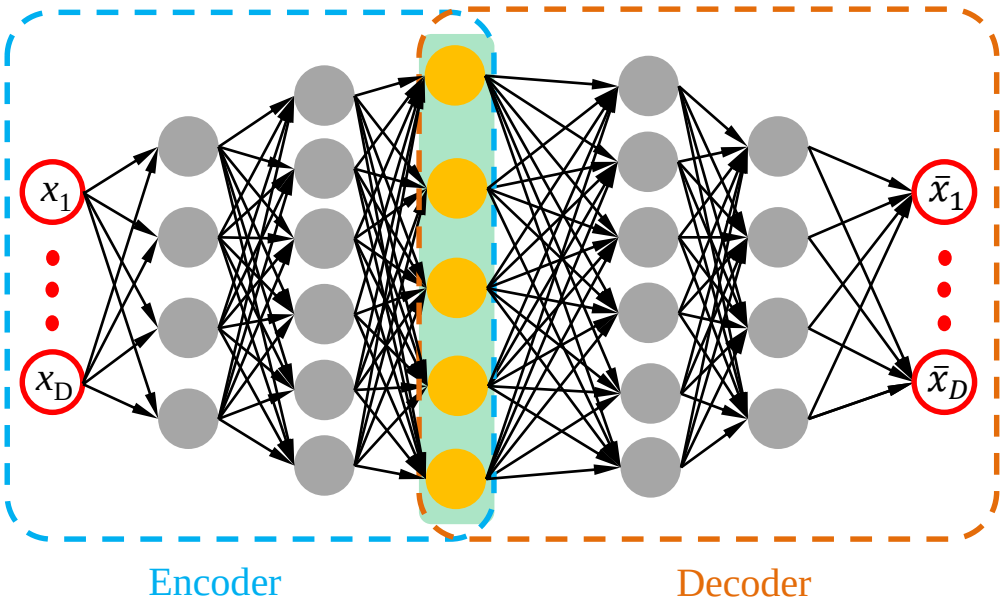
- Dimension of the hidden representation $\mathbf{h}$ is greater than equal to that of the input $\mathbf{x}$.
- May lead to trivial encoding where $\mathbf{x}$ is copied into $\mathbf{h}$, and then decoder copies $\mathbf{h}$ to $\overline{\mathbf{x}}$. Such mappings can be learnt using simple linear 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

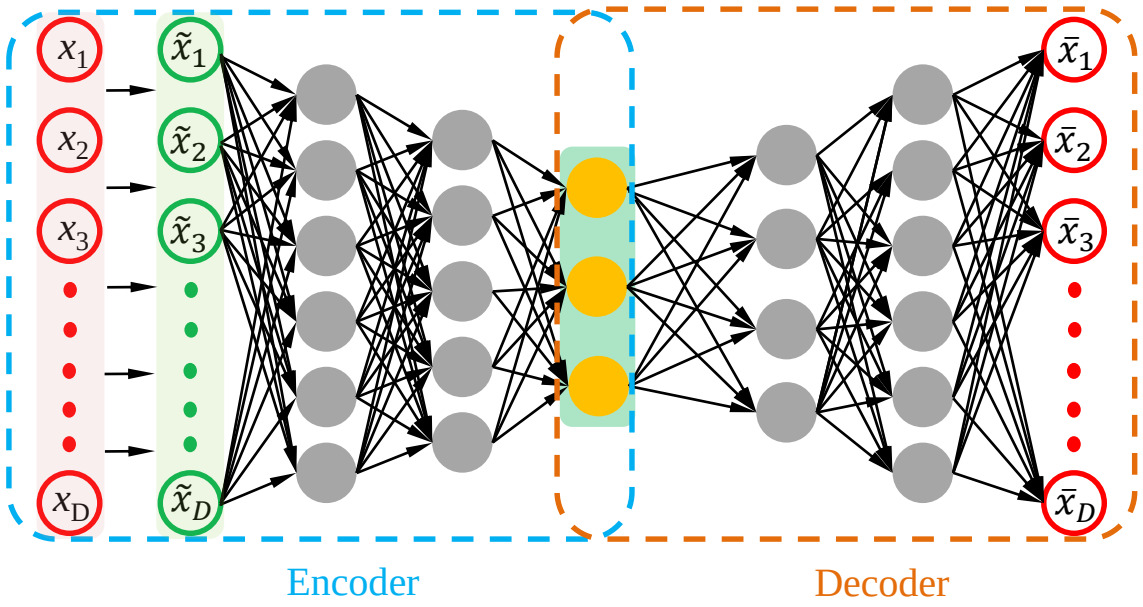


- Need regularization to overcome the issue.
- L_2 regularization objective:

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

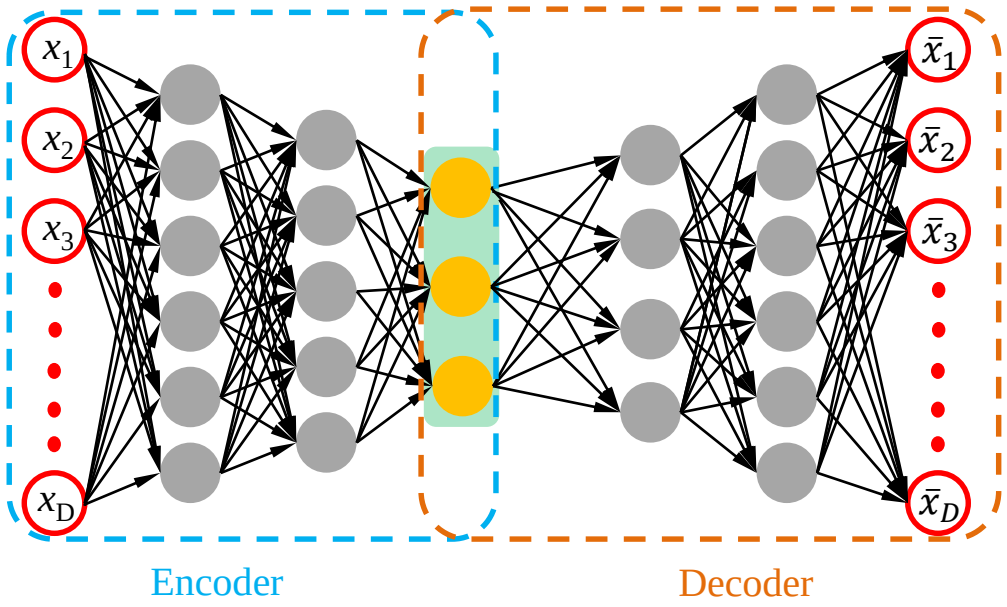
- Tie the weights of encoder and decoder: $\mathbf{w}_e = \mathbf{w}_d^{\mathrm{T}}$.

Denoising Autoencoder



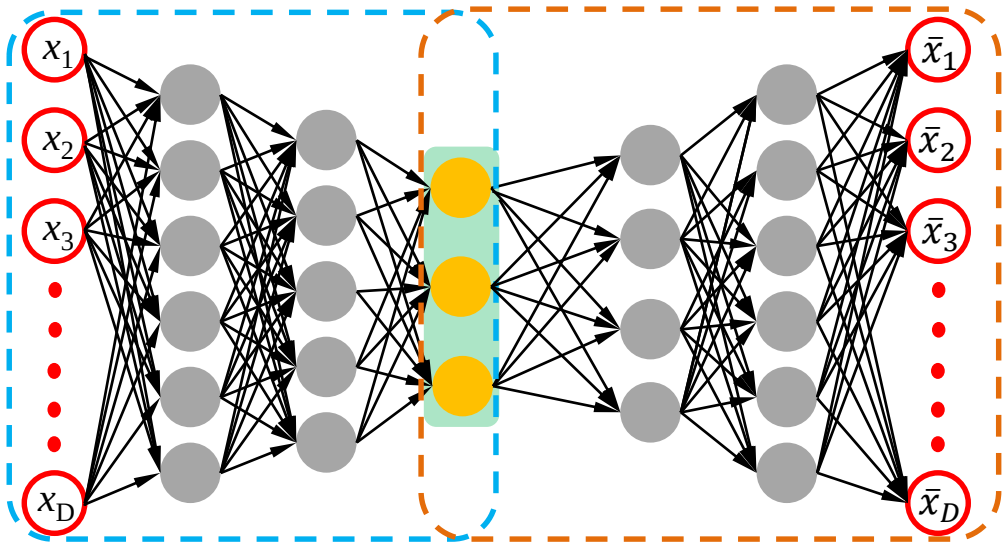
- Input $\mathbf{x}$ corrupted by some form of noise.
- Denoising autoencoders need to undo the corruption that has been done to the input.
  - Therefore the autoencoder cannot simply copy $\mathbf{x}$ to $\mathbf{h}$ and then $\mathbf{h}$ to $\overline{\mathbf{x}}$.
- The objective is still to reconstruct 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



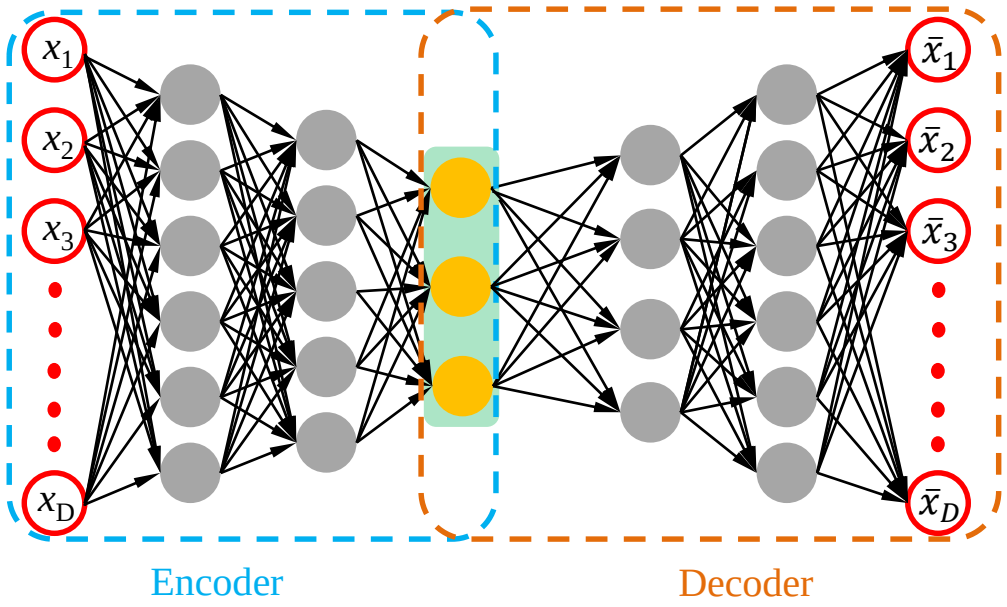
- The average value of activation of the kth neuron using the N training examples can be written as

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

- 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



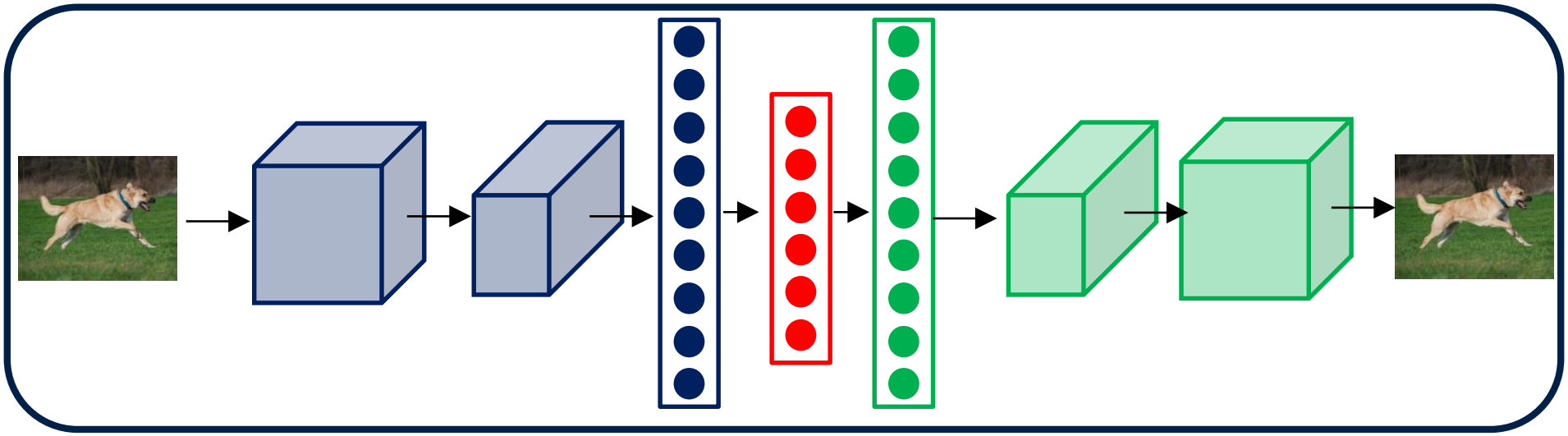
- The penalty term can take the following form:

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

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

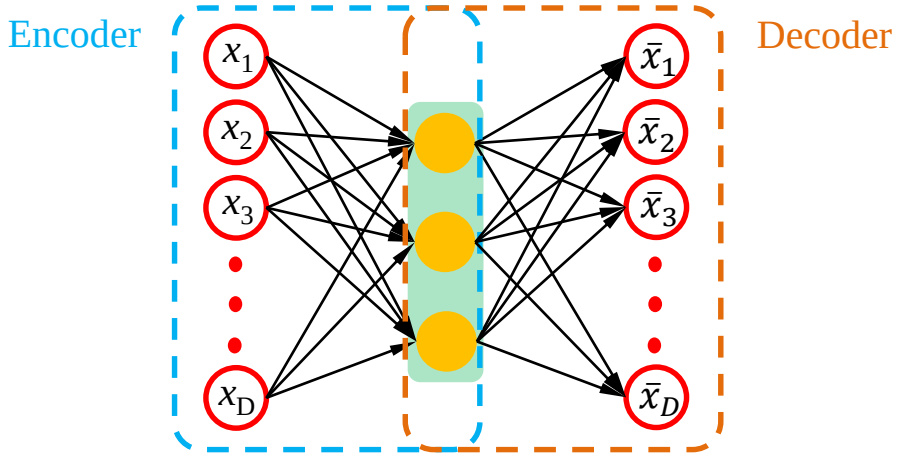
– $\Omega(\mathbf{h})$ is minimum when $\overline{\mu}_k = \mu$.

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.

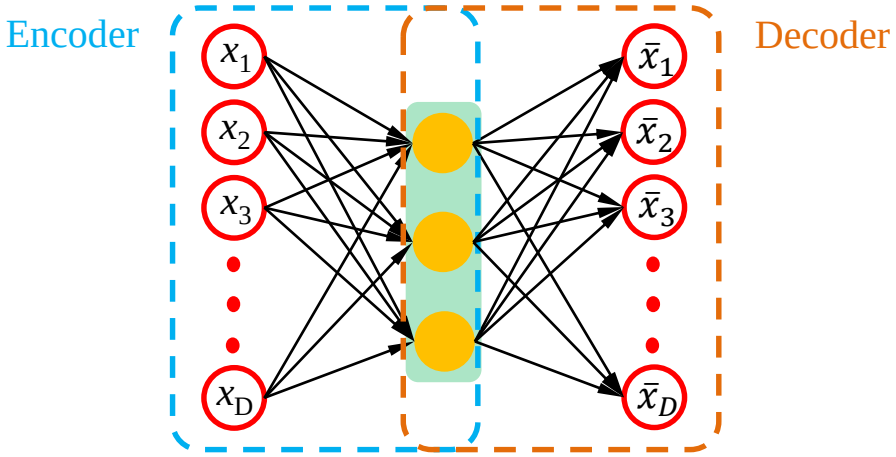
Contractive Autoencoder



- Suppose there are K units in the hidden layer and the dimension of the input is D.
- Jacobian matrix:

$$\frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_D} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots & \frac{\partial h_2}{\partial x_D} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial h_K}{\partial x_1} & \frac{\partial h_K}{\partial x_2} & \cdots & \frac{\partial h_K}{\partial x_D} \end{bmatrix}$$

Contractive Autoencoder



- The (k, d) element of the Jacobian matrix indicates the change in the output of the k-th unit of the layer w.r.t. to small change in the d-th input.
- Contractive autoencoders introduces a regularizer

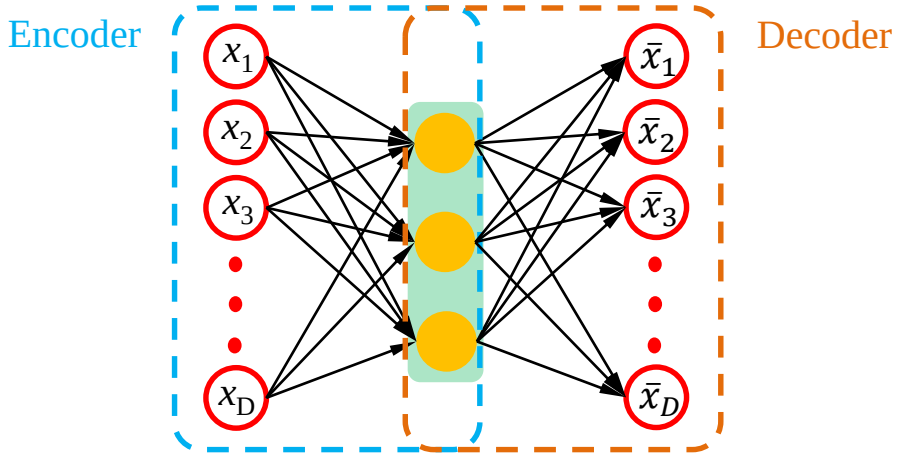
$$\Omega(\mathbf{h}) = \lambda \left|\left| \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right|\right|_F^2$$

where

$$\left|\left| \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right|\right|_F^2 = \sum_{k=1}^{K} \sum_{d=1}^{D} \left(\frac{\partial h_k}{\partial x_d} \right)^2$$

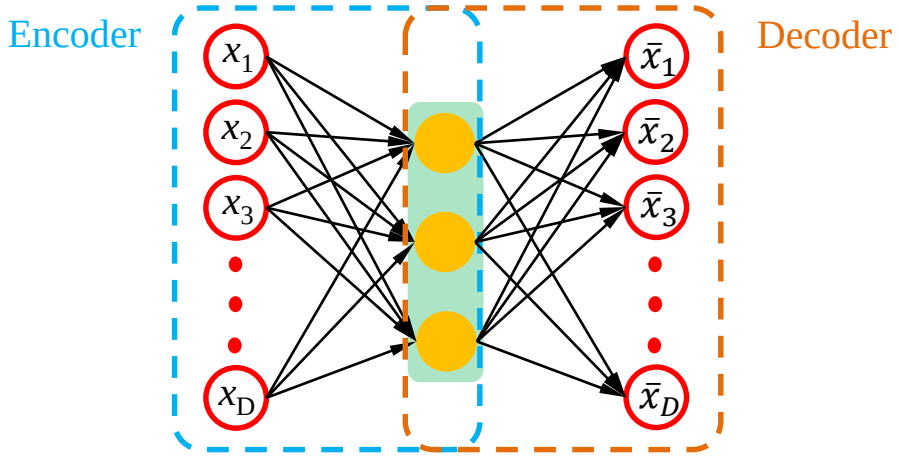
is the square of the Frobenius norm of the Jacobian matrix.

Contractive Autoencoder



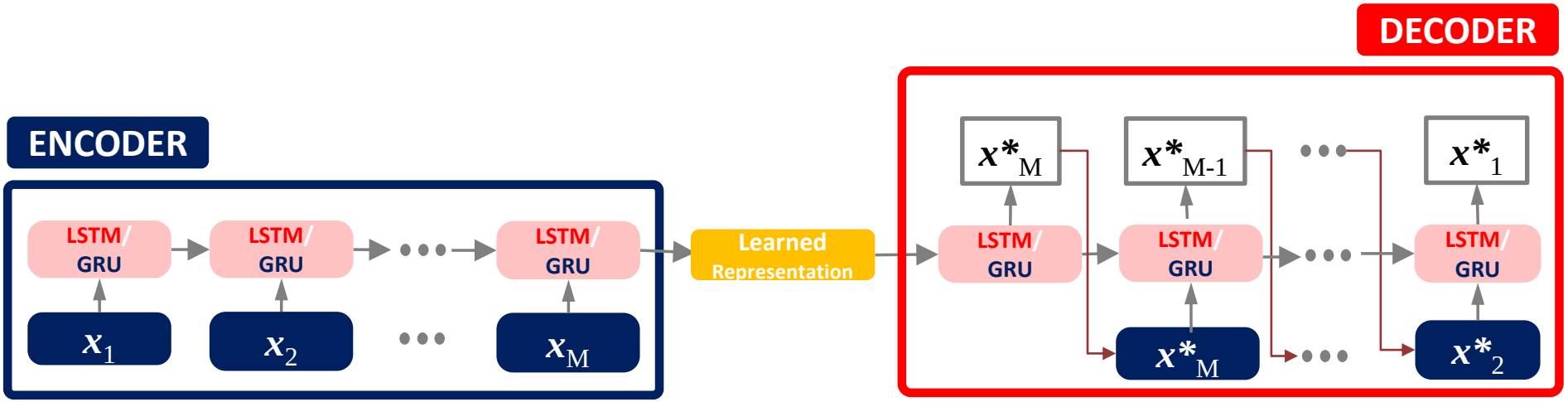
- Small value of $\frac{\partial h_k}{\partial x_d}$ indicates that the k-th unit is not much sensitive to the d-th input.
- Learning in contractive autoencoders involves balancing two competing forces: reconstruction error and contractive penalty.
- By minimizing the reconstruction error the model tries to learn the important variations in the data.
  - This tries to encourage the model to learn an identity function.
- The contractive penalty encourages learning features that are constant w.r.t. $\mathbf{x}$.
  - It tries to force the model to not capture variations in the data.

Contractive Autoencoder



- Implication: Capture only the very important variations.
  - A few entries in the Jacabian may have significant value.

LSTM Autoencoder

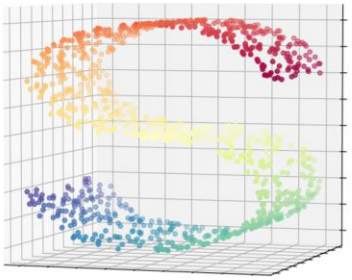


- Goal: Map an input sequence to a fixed length representation.
- Autoencoder Model: The encoder model takes the input sequence and the decoder model predicts the target sequence.
- The target sequence can be taken in reverse order
  - It is easier for the model in the beginning of the training process by looking at the low range correlations.
- Future Predictor Model: Based on the input sequence, predict the future (just after the input sequence).
- **Composite Model**: Autoencoder Model + Future Predictor Model

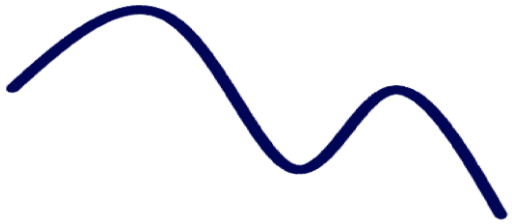
Paper: Srivastava *et. al.* "Unsupervised Learning of Video Representations using LSTMs", ICML 2015

Manifold

- A manifold is a **topological space** that locally resembles Euclidean space near each point.

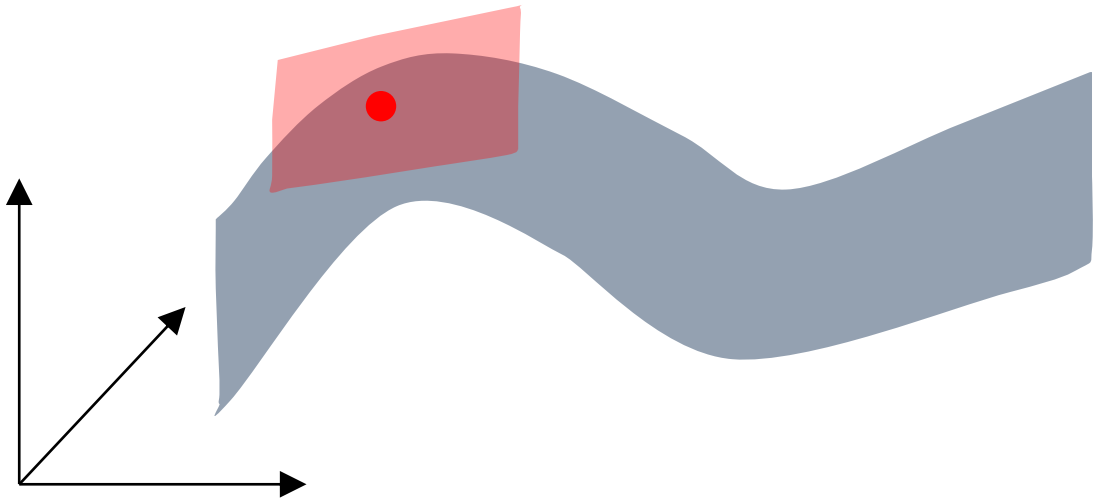


- Example of 1-D manifold: Curve.



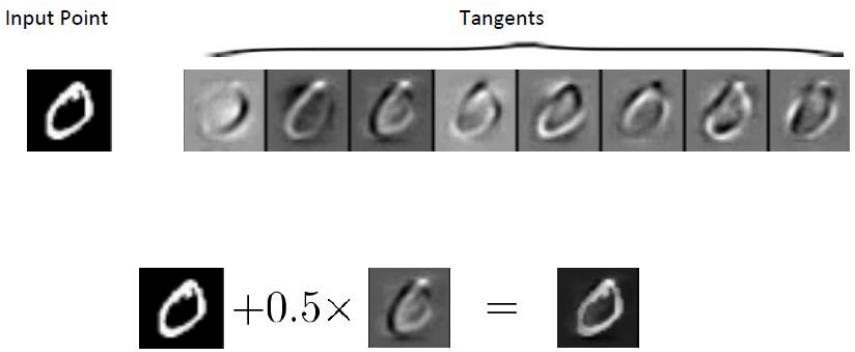
- Example of 2-D manifold: Surface of sphere, torus.

Tangent space



- Tangents specify how inputs are changing while staying on the manifold.
- As we move on the manifold, the tangent of the manifold changes.
  - For a linear manifold the tangent space is same everywhere.
- If the dimension of the manifold is d, then at any point the tangent plane is given by d basis vectors.
  - These vectors span the local directions of variation allowed on the manifold.

Tangent space



- For a trained CAE, the SVD of the Jacobian provides an ordered orthonormal basis of the most sensitive directions.
  - It has been found that the singular value spectrum is sharply decreasing, indicating a small number (relatively) of significantly sensitive directions.
  - The leading singular vectors form a basis for the tangent plane of the estimated manifold.

Source: Bengio *et. al.* "Representation Learning: A Review and New Perspectives", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2013