THE UNIVERSITY OF BRITISH COLUMBIA

# Topics in AI (CPSC 532S):
# Multimodal Learning with Vision, Language and Sound

## Lecture 10: Unsupervised Learning, Autoencoders

# **Unsupervised** Learning

We have access to $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N\}$ but not $\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \cdots, \mathbf{y}_N\}$

*slide from Louis-Philippe Morency

# **Unsupervised** Learning

We have access to $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N\}$ but not $\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \cdots, \mathbf{y}_N\}$

Why would we want to tackle such a task:

1. Extracting interesting information from data
   - Clustering
   - Discovering interesting trend
   - Data compression
2. Learn better representations

*slide from Louis-Philippe Morency

# **Unsupervised** Representation Learning

Force our **representations** to better model input distribution

— Not just extracting features for classification

— Asking the model to be good at representing the data and not overfitting to a particular task (we get this with ImageNet, but maybe we can do better)

— Potentially allowing for better generalization

Use for **initialization of supervised task**, especially when we have a lot of unlabeled data and much less labeled examples

*slide from Louis-Philippe Morency

# **Restricted** Boltzmann Machines (in one slide)

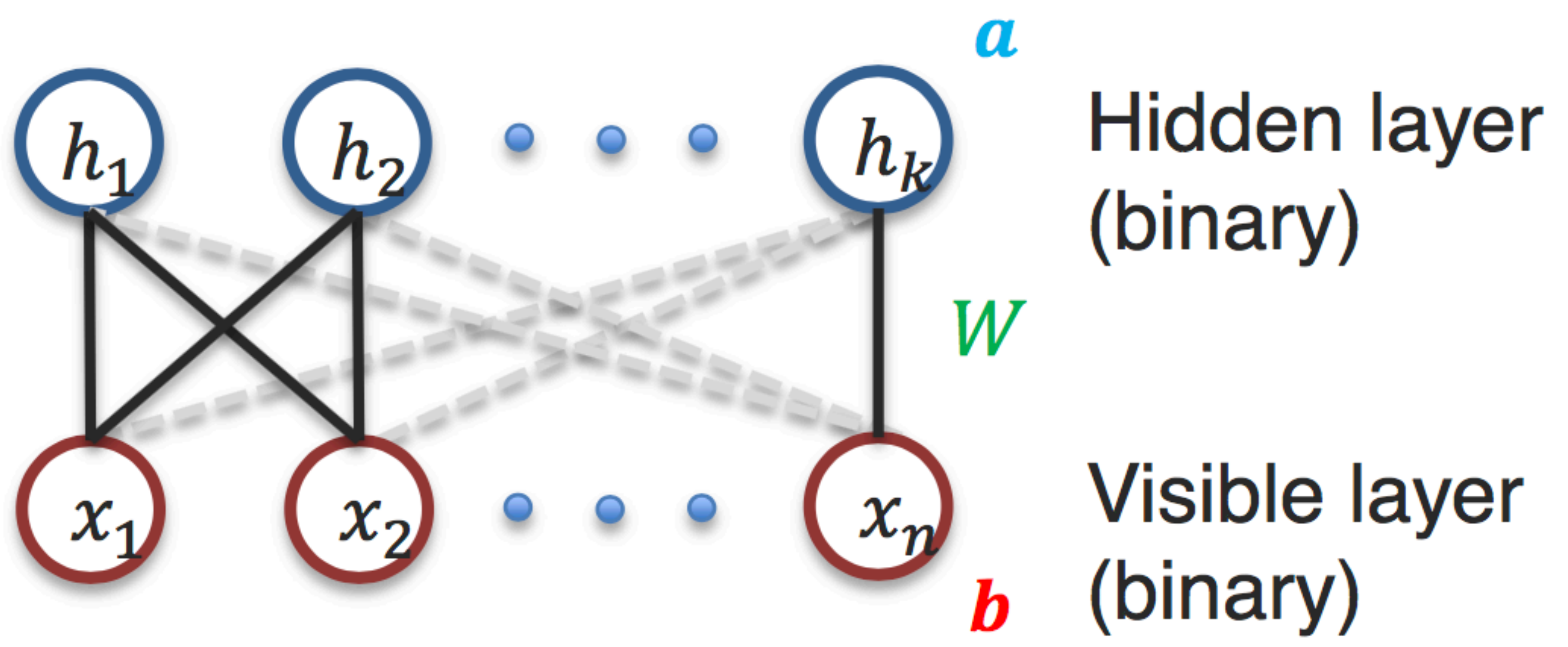
Model the **joint probability** of hidden state and observation

$$p(\boldsymbol{x}, \boldsymbol{h}; \theta) = \frac{\exp(-\mathrm{E}(\boldsymbol{x}, \boldsymbol{h}; \theta))}{Z}$$

$$Z = \sum_{\boldsymbol{x}} \sum_{\boldsymbol{h}} \exp(-\mathrm{E}(\boldsymbol{x}, \boldsymbol{h}; \theta))$$

$$\mathrm{E} = -\boldsymbol{x}W\boldsymbol{h} - \boldsymbol{b}^T\boldsymbol{x} - \boldsymbol{a}^T\boldsymbol{h}$$

$$\mathrm{E} = -\underbrace{\sum_i \sum_j w_{i,j} x_i h_j}_{\text{Interaction term}} - \underbrace{\sum_i b_i x_i - \sum_j a_j h_j}_{\text{Bias terms}}$$




Objective, maximize likelihood of the data

*slide from Louis-Philippe Morency

# Autoencoders

*slide from Louis-Philippe Morency

# Autoencoders

Self (i.e. self-encoding)

*slide from Louis-Philippe Morency

# Autoencoders

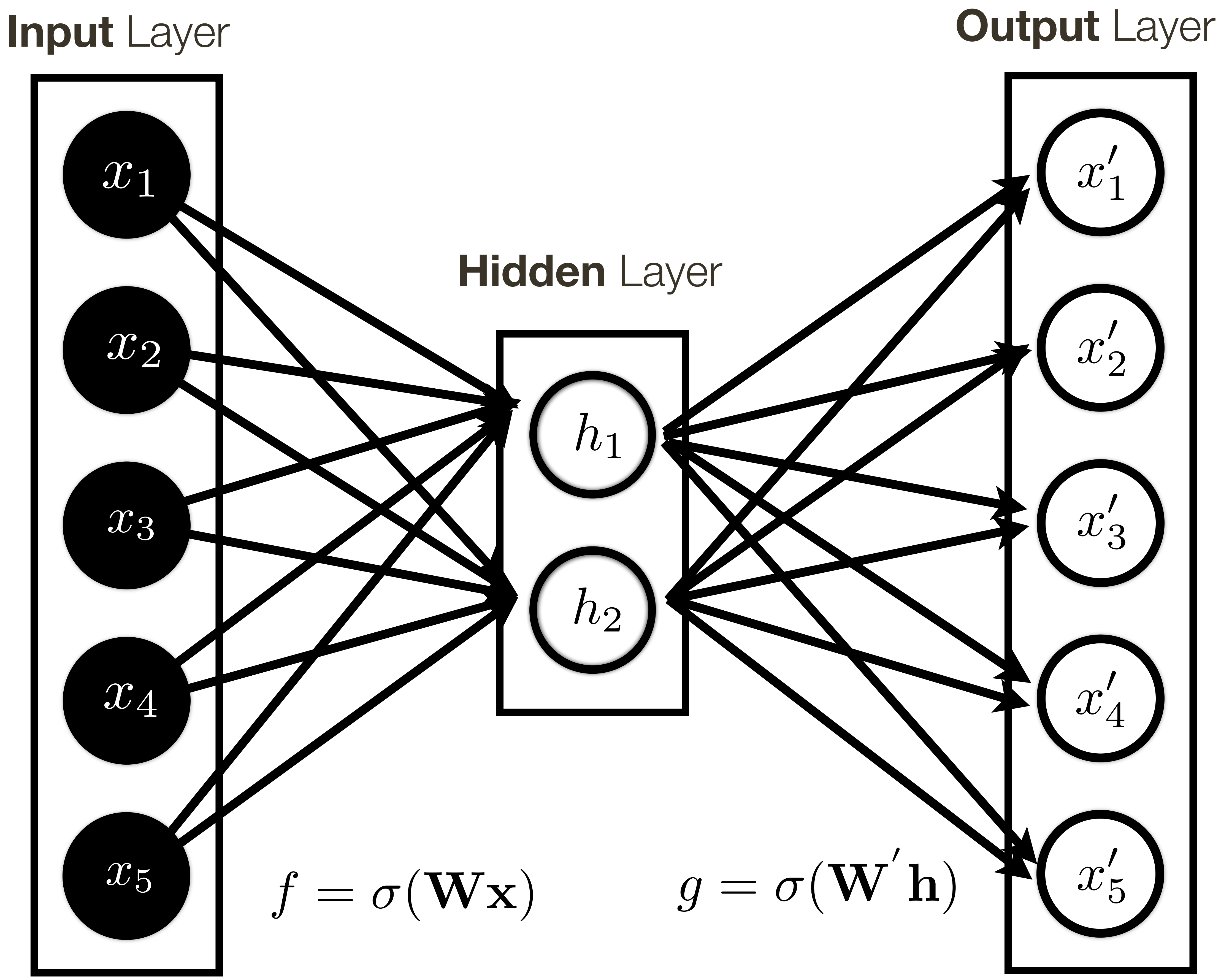
Self (i.e. self-encoding)

— Feed forward network intended to reproduce the input

— Encoder/Decoder architecture

Encoder: $f = \sigma(\mathbf{W}\mathbf{x})$

Decoder: $g = \sigma(\mathbf{W}'\mathbf{h})$




*slide from Louis-Philippe Morency

# Autoencoders

Self (i.e. self-encoding)

— Feed forward network intended to reproduce the input

— Encoder/Decoder architecture

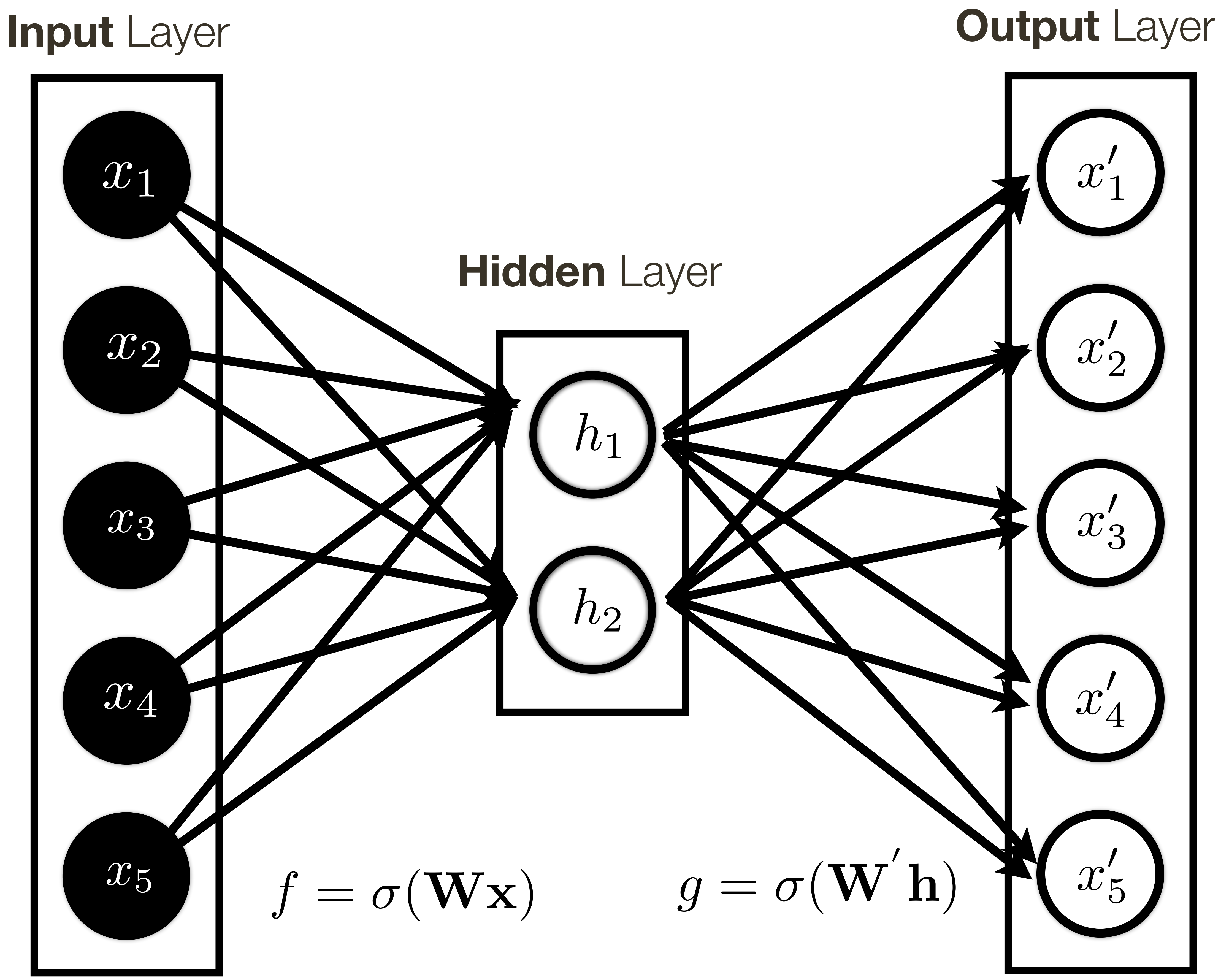
Encoder: $f = \sigma(\mathbf{W}\mathbf{x})$

Decoder: $g = \sigma(\mathbf{W}^{'}\mathbf{h})$

— Score function

$$\mathbf{x}' = f(g(\mathbf{x}))$$

$$\mathcal{L}(\mathbf{x}', \mathbf{x})$$




*slide from Louis-Philippe Morency

# Autoencoders

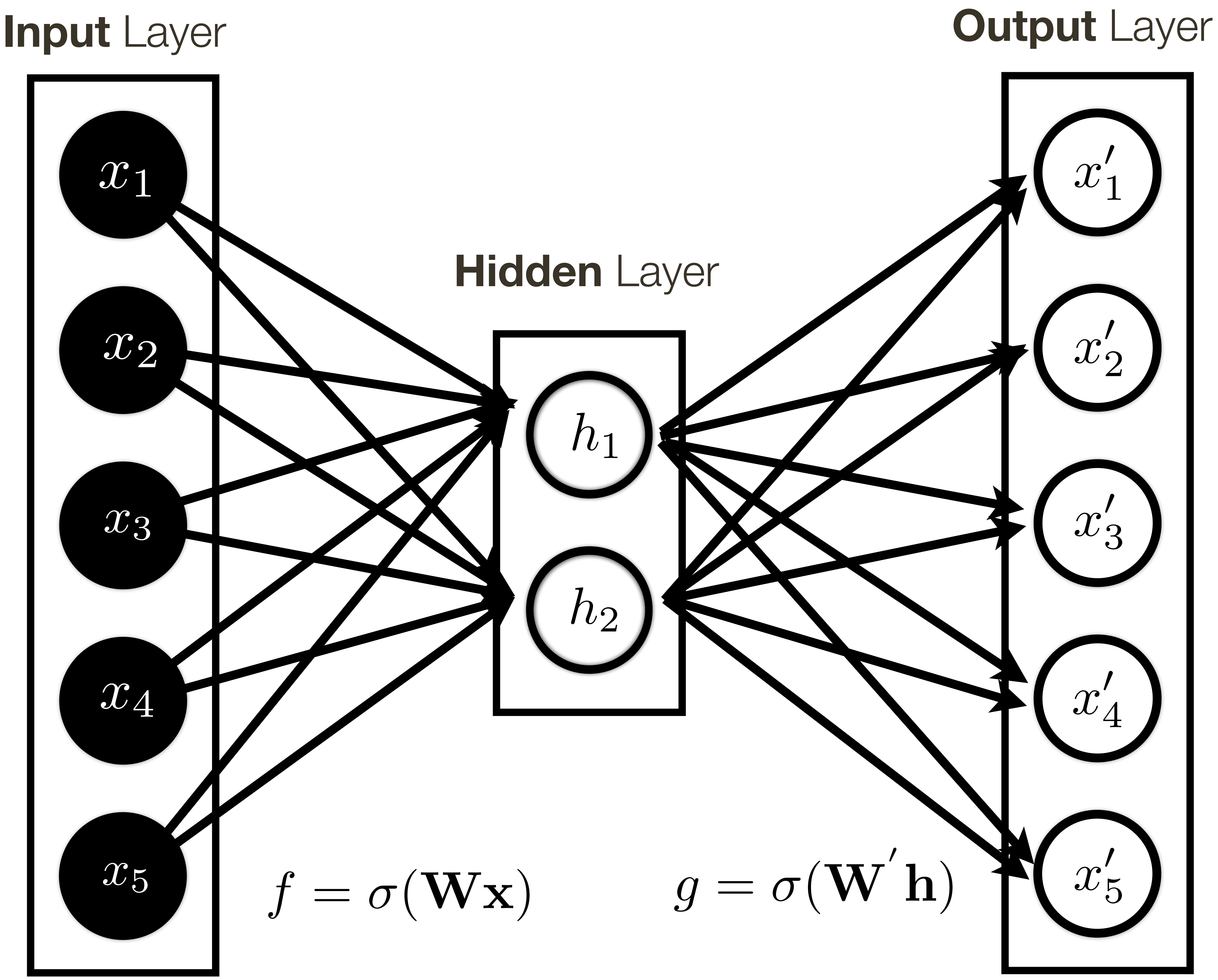
A standard neural network architecture (linear layer followed by non-linearity)

— Activation depends on type of data

(e.g., sigmoid for binary; linear for real valued)

— Often use tied weights

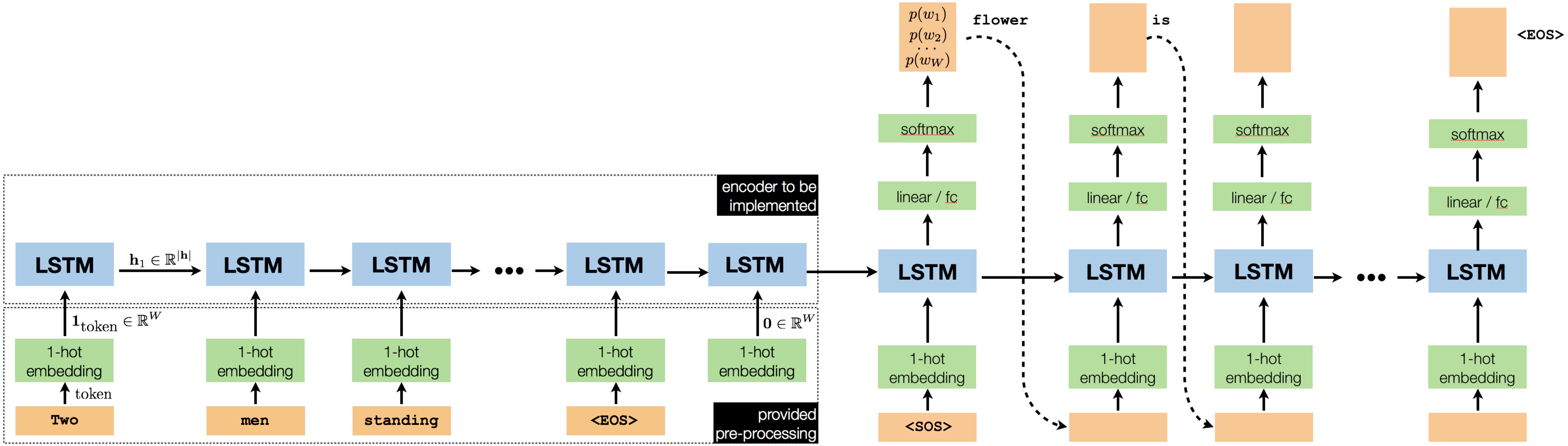
$$\mathbf{W}' = \mathbf{W}$$




*slide from Louis-Philippe Morency

# Autoencoders

Assignment 3 can be interpreted as a language autoencoder



# **Autoencoders**: Hidden Layer Dimensionality

**Smaller** than the input

— Will compress the data, reconstruction of the data far from the training distribution will be difficult

— Linear-linear encoder-decoder with Euclidian loss is actually equivalent to PCA (under certain data normalization)

*slide from Louis-Philippe Morency

# **Autoencoders**: Hidden Layer Dimensionality

**Smaller** than the input

— Will compress the data, reconstruction of the data far from the training distribution will be difficult

— Linear-linear encoder-decoder with Euclidian loss is actually equivalent to PCA (under certain data normalization)

Side note, this is useful for **anomaly detection**

*slide from Louis-Philippe Morency

# **Autoencoders**: Hidden Layer Dimensionality

**Smaller** than the input

— Will compress the data, reconstruction of the data far from the training distribution will be difficult

— Linear-linear encoder-decoder with Euclidian loss is actually equivalent to PCA (under certain data normalization)

**Larger** than the input

— No compression needed

— Can trivially learn to just copy, no structure is learned (unless you regularize)

— Does not encourage learning of meaningful features (unless you regularize)

*slide from Louis-Philippe Morency