Deep Learning for Healthcare

Jimeng Sun, Cao (Danica) Xiao, Edward Choi

August 7, 2018

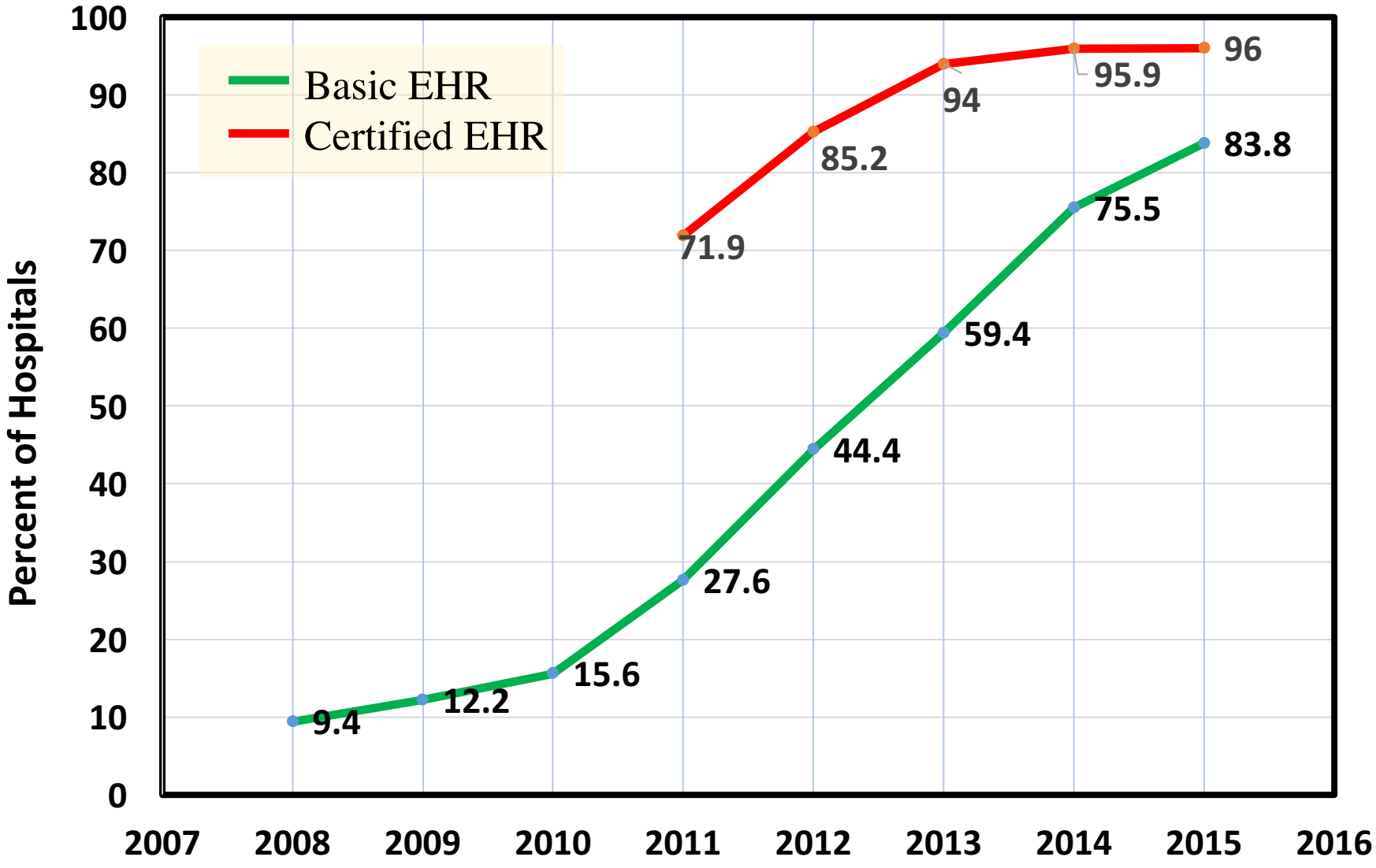
Agenda

- Background
  - Healthcare data
  - Analytical tasks
  - Why deep learning models
  - Deep learning architectures
- Success of Deep Learning in Computational Healthcare
  - Medical Classification
  - Sequential Prediction
  - Concept Embedding
  - Data Augmentation
- Open Challenges
- Q&A

Agenda

- Background
  - Healthcare data
  - Analytical tasks
  - Why deep learning models
  - Deep learning architectures
- Success of Deep Learning in Computational Healthcare
  - Disease Classification
  - Sequential Prediction
  - Concept Embedding
  - Data Augmentation
- Open Challenges (10 min)
- Q&A

Adoption of EHR Systems among U.S. Hospitals



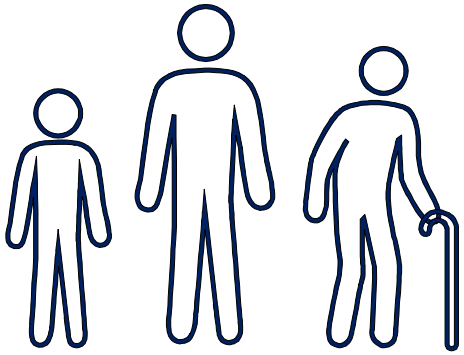
Source: American Hospital Association Annual Survey

Sun, Xiao, Choi, dl4health.org

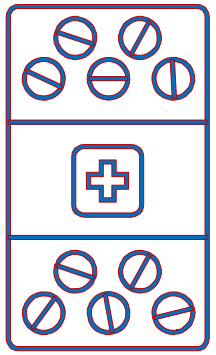
The Growing Availability of Health Data

Sun, Xiao, Choi, dl4health.org

Multiple Data Modalities in the EHR Systems

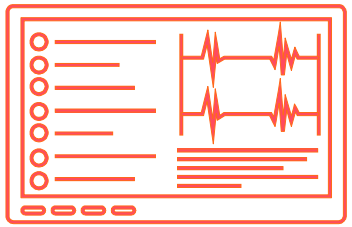


Demographics



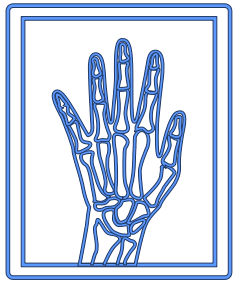
Medications

Clinical Notes and Reports



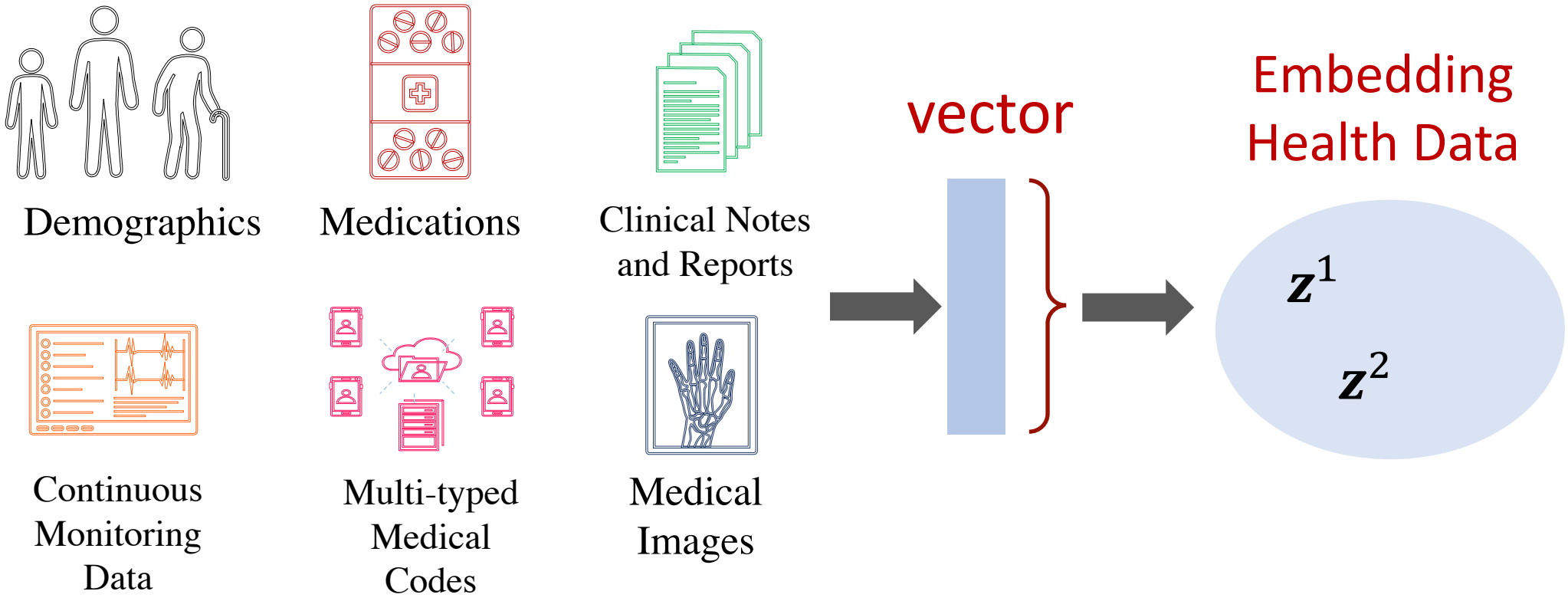
Continuous Monitoring Data

Multi-typed Medical Codes



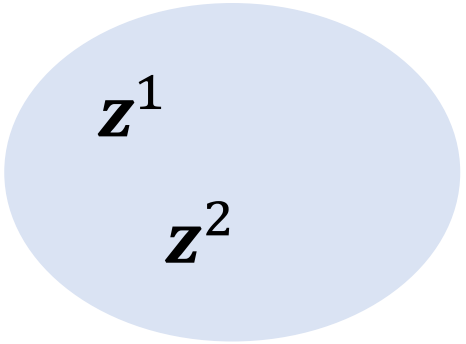
Medical Images

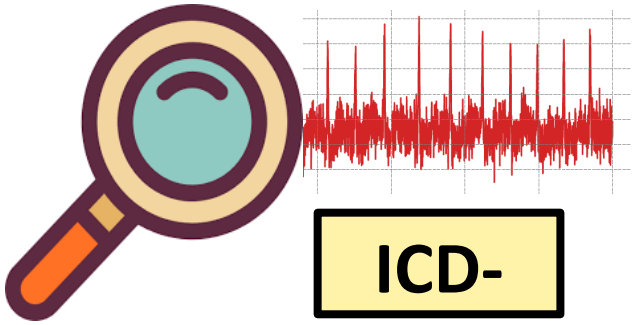
Representations Learning from Health Data



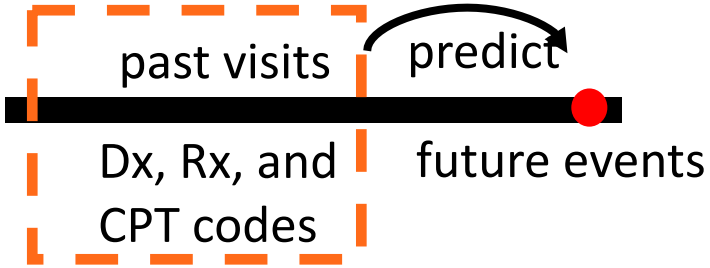
Analytics Tasks using EHR Data

Embedding Health Data

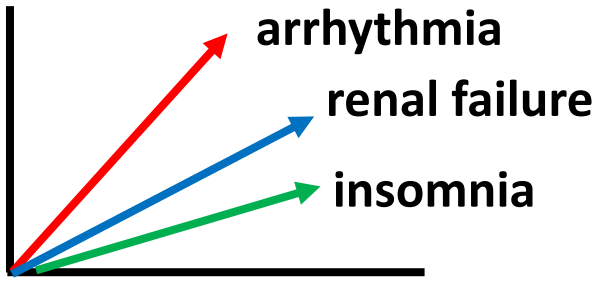




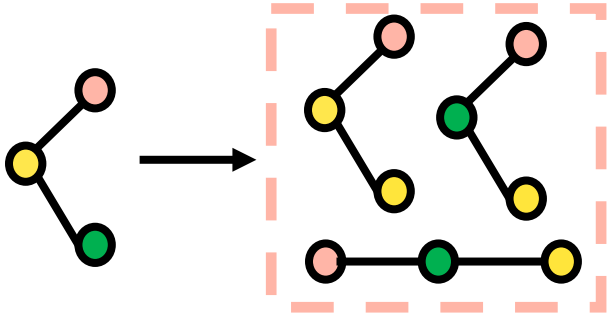
Disease Classification



Sequential Prediction

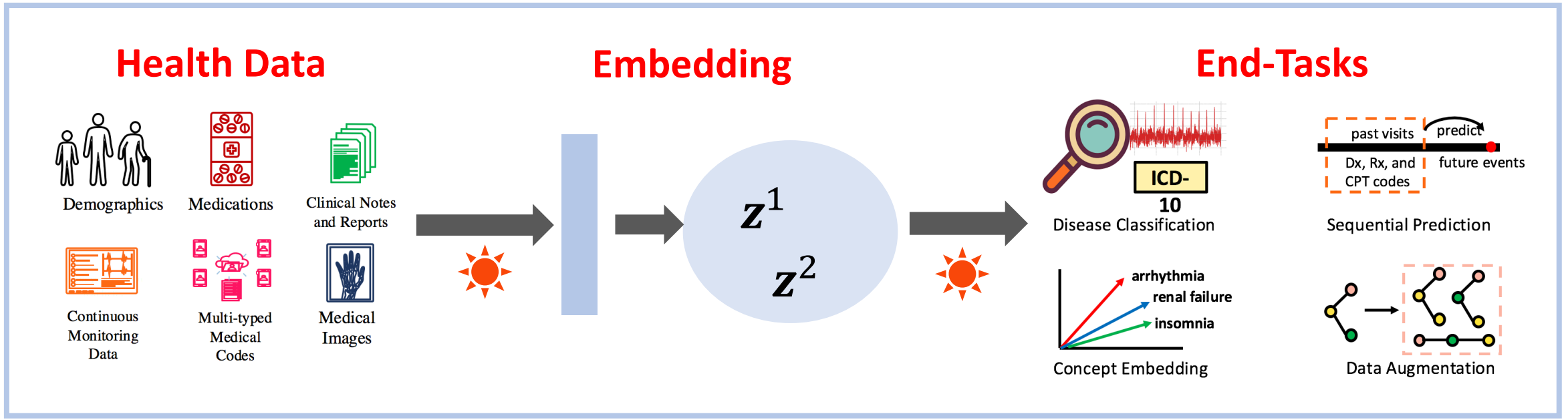


Concept Embedding



Data Augmentation

Deep Learning as Effective Tool



Deep learning models are effective tools for all phases of health data modeling.

Check Our Survey and DL4HC Website!

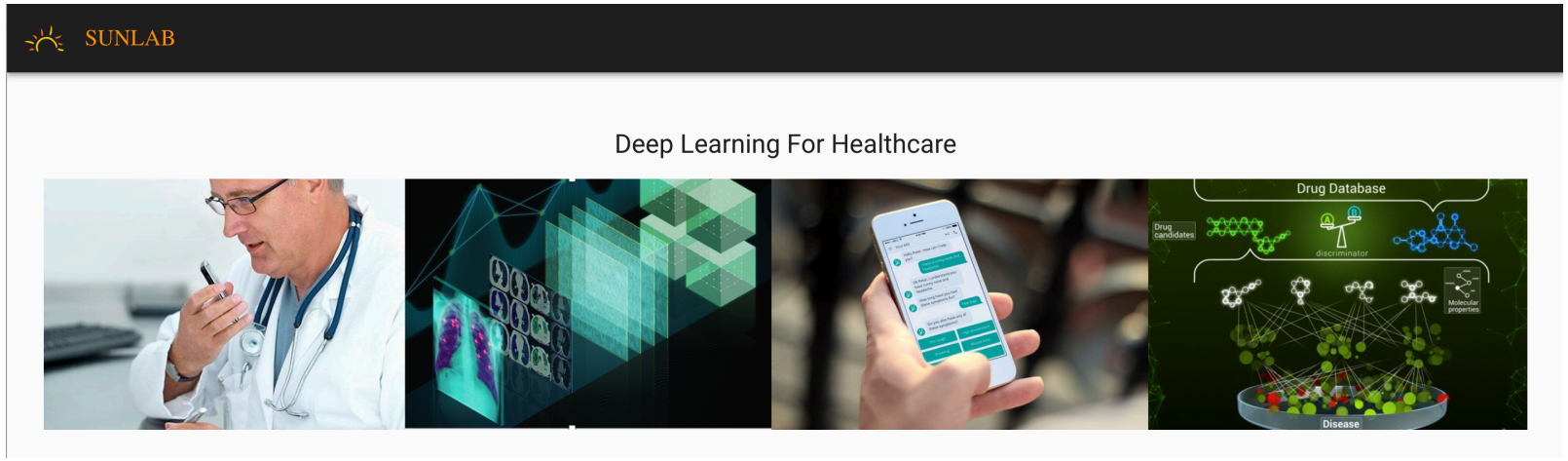
JAMIA 2018

dl4health.org

Review

Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review

Cao Xiao,[1] Edward Choi[2] and Jimeng Sun[2]

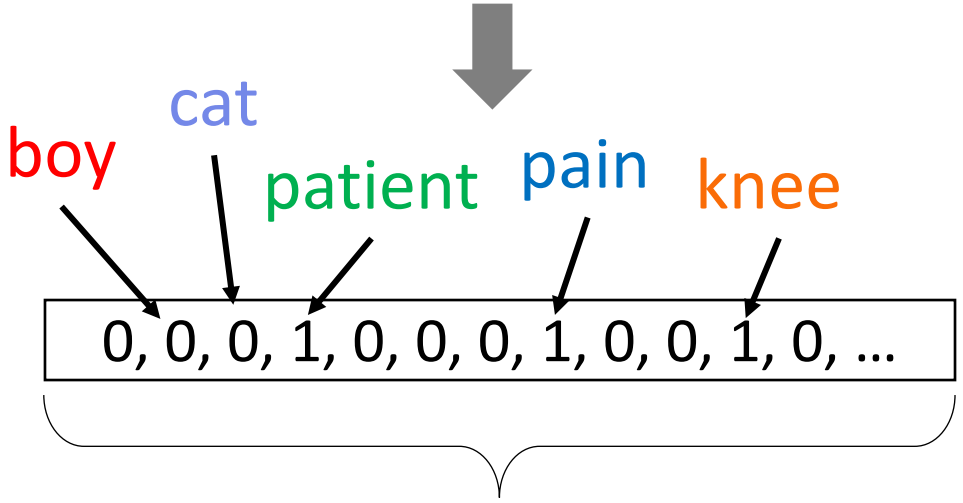


Basic Methods: Neural Networks Basics

Neural Networks Basics

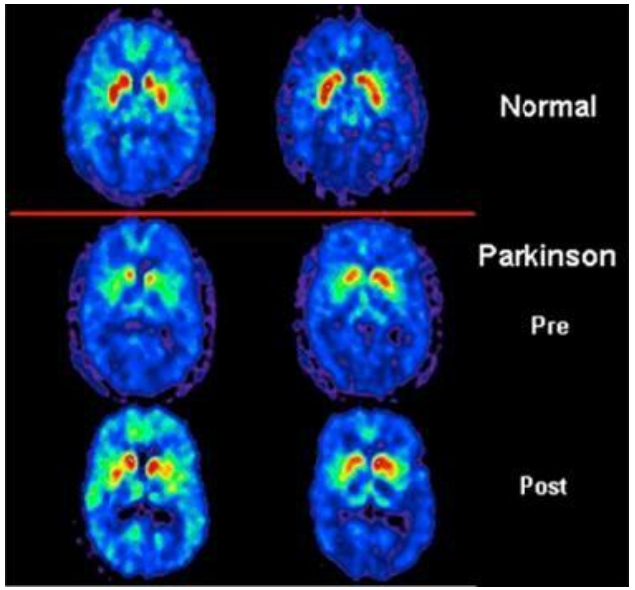
- **x**: Input sample in a vector form
  - Diagnosis classification (arthritis or not?)
  - Sentence in a vector form

"The patient reported a severe pain in the knee ."



Neural Networks Basics

- $\boldsymbol{x}$: Input sample in a vector form
  - Image classification (stages of Parkinson's disease)
  - Pixels in a vector form

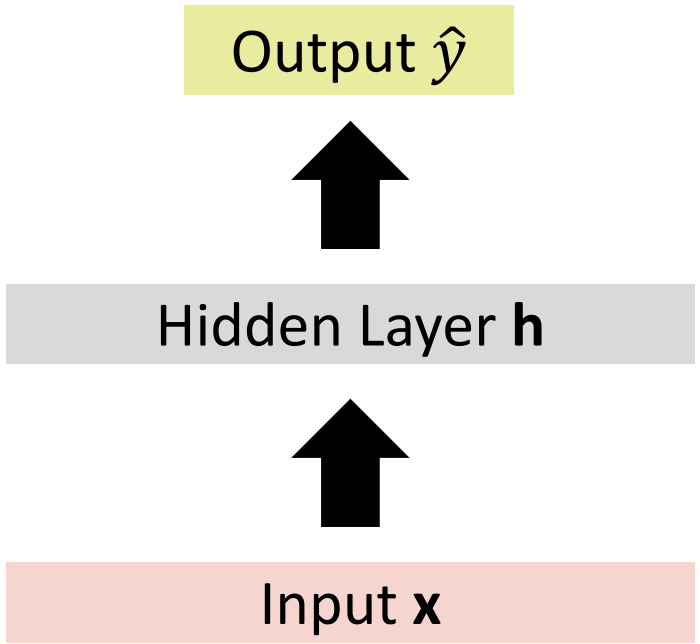


…, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, …

Number of pixels in the image

Neural Networks Basics

- Let's start with a simple Multi-layer Perceptron (MLP)
  - Binary classification



$\hat{y} = \boldsymbol{\sigma}(\mathbf{w_o}^{\top}\mathbf{h})$ (generate an outcome 0.0~1.0)

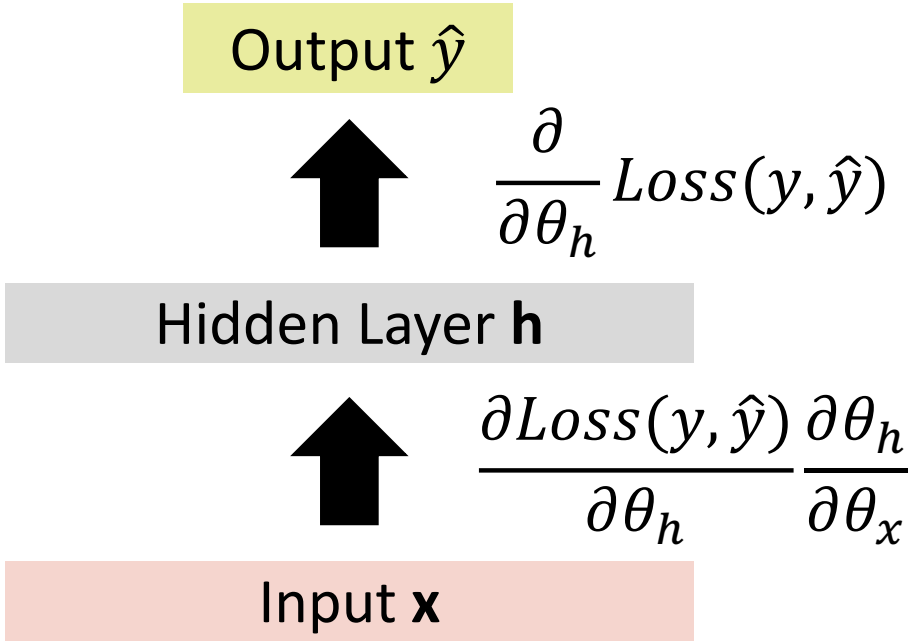
$\mathbf{h} = \boldsymbol{\sigma}(\mathbf{W_h}^{\top}\mathbf{x})$ (transform **x** for an easier prediction)

x (a vector representation of the input sample)

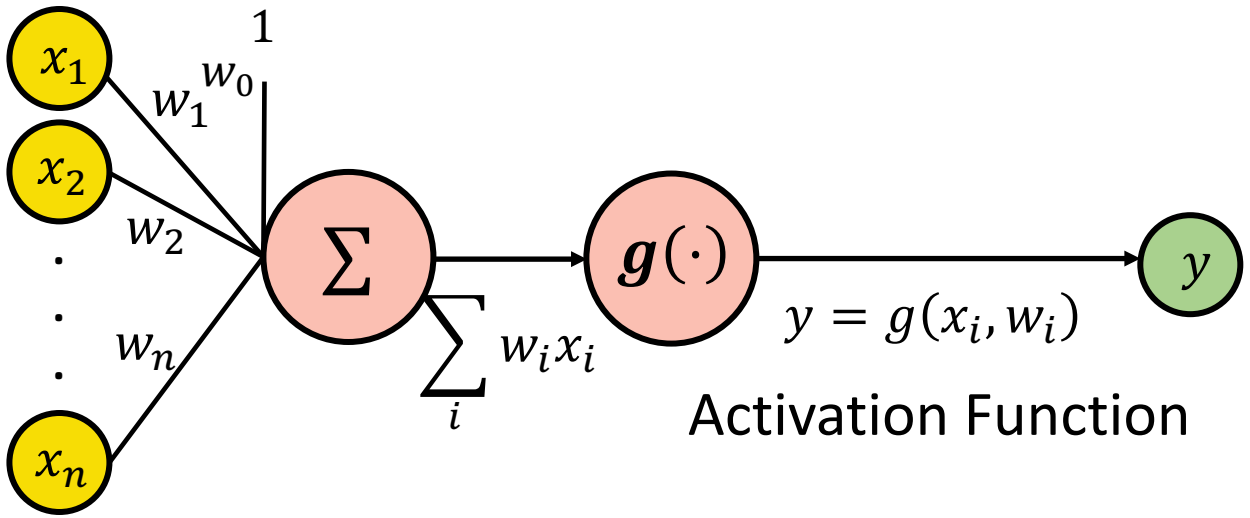
Neural Networks Basics

- Learning the model parameters
  - Backpropagation + Gradient descent

$$Loss(y, \hat{y}) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y}))$$



Neural Networks Basics



Adding non-linearity

Neural Networks Basics

Sigmoid

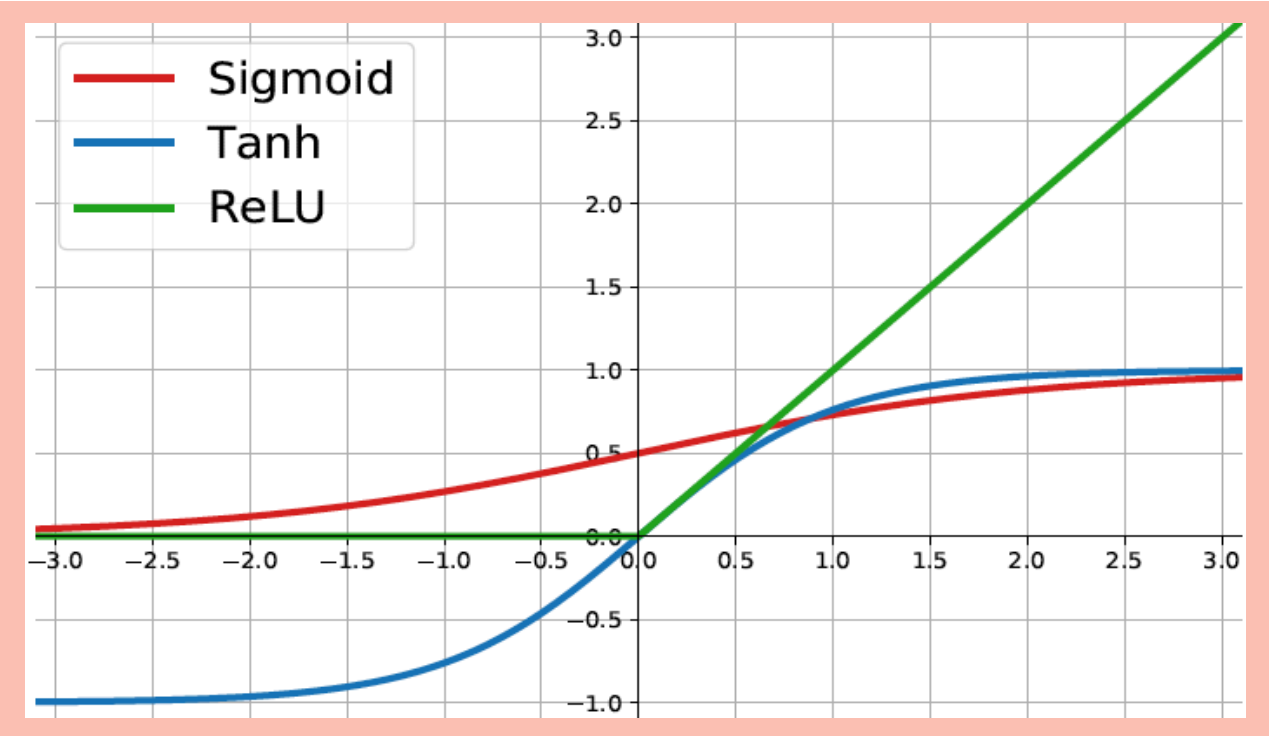
$$g(x) = \frac{1}{1 + e^{-x}}$$

Tanh

$$g(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

ReLU

$$g(x) = \max(0, x)$$



The introduction of ReLU makes it possible to train deep nets in a purely supervised way for the first time (Glorot & Bengio AISTATS 2011).

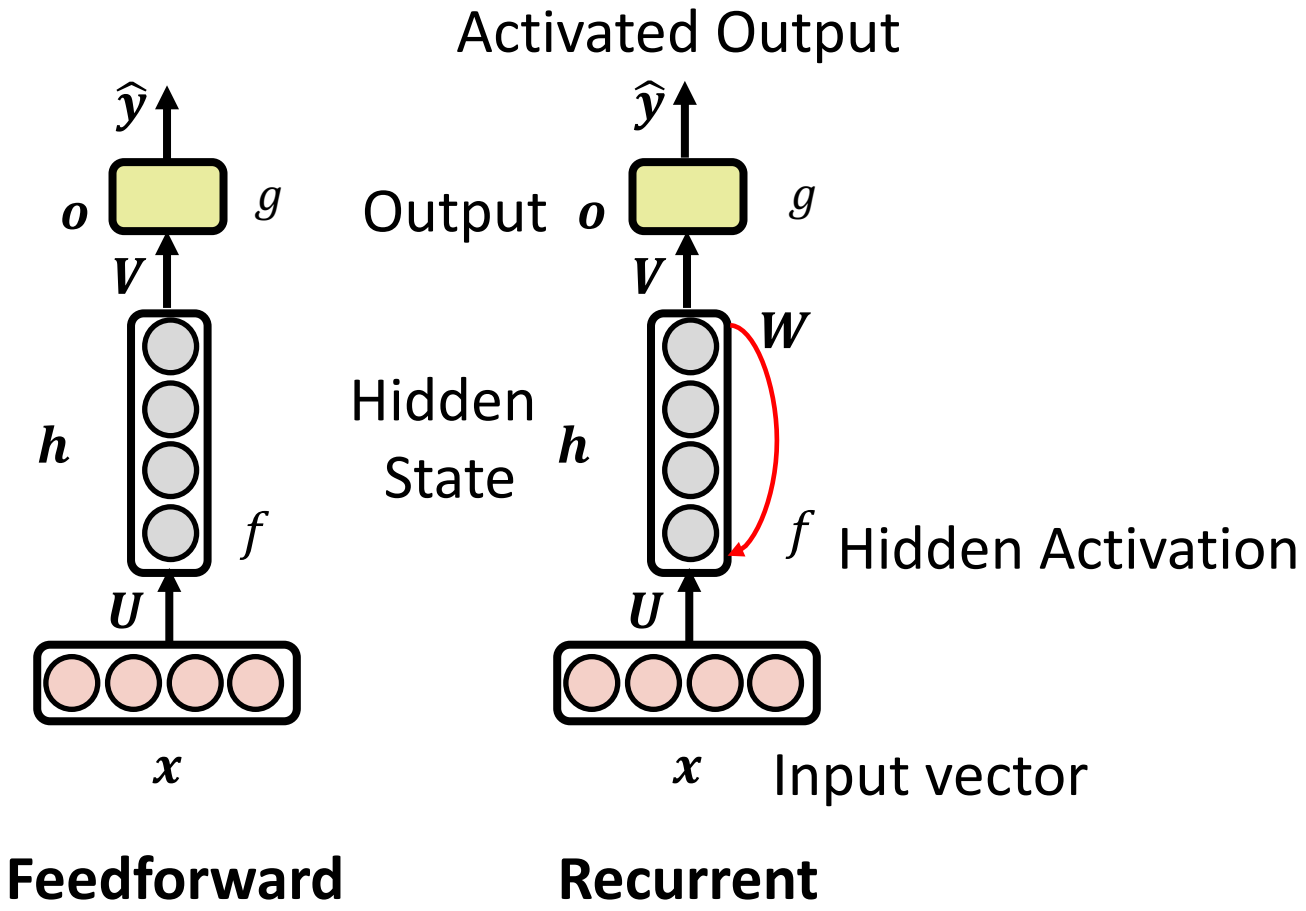
Basic Methods:
Recurrent Neural Networks

Recurrent Neural Networks (RNN)

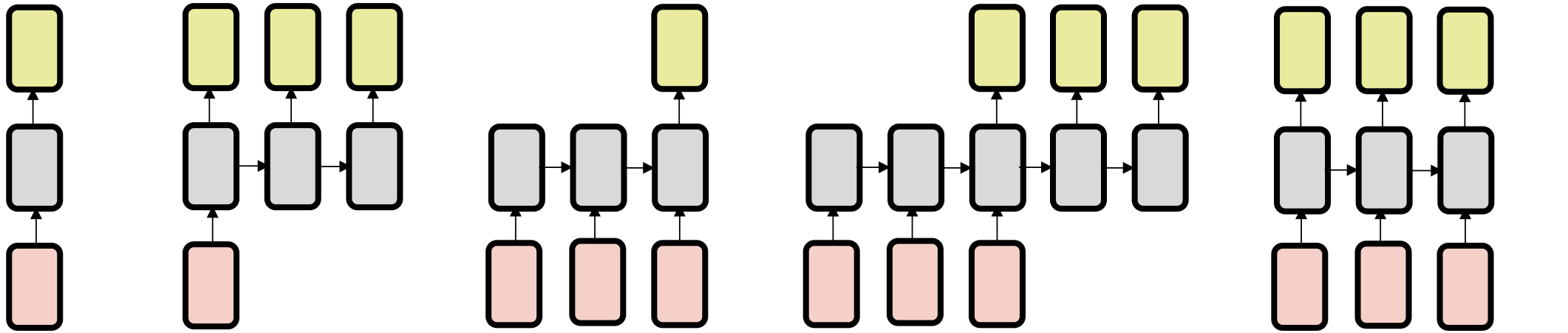
- What if the input sample is a sequence?
  - “The patient reported a severe pain in the knee .”
  - A sequence of 10 elements

- MLP loses sequence information
- RNN retains sequence information!

Recurrent Neural Networks



Basic RNN Structure



One-to-One (e.g., image classification)

One-to-Many (e.g., image to text)

Many-to-One (e.g., sequence classification)

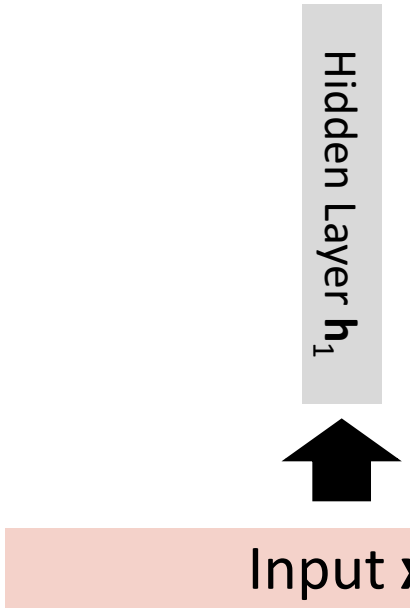
Many-to-Many (e.g., seq2seq)

Many-to-Many (e.g., sequential prediction)

The figure is inspired by page 12 on http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf

Recurrent Neural Networks

- Recurrent Neural Network (RNN)
  - Binary classification

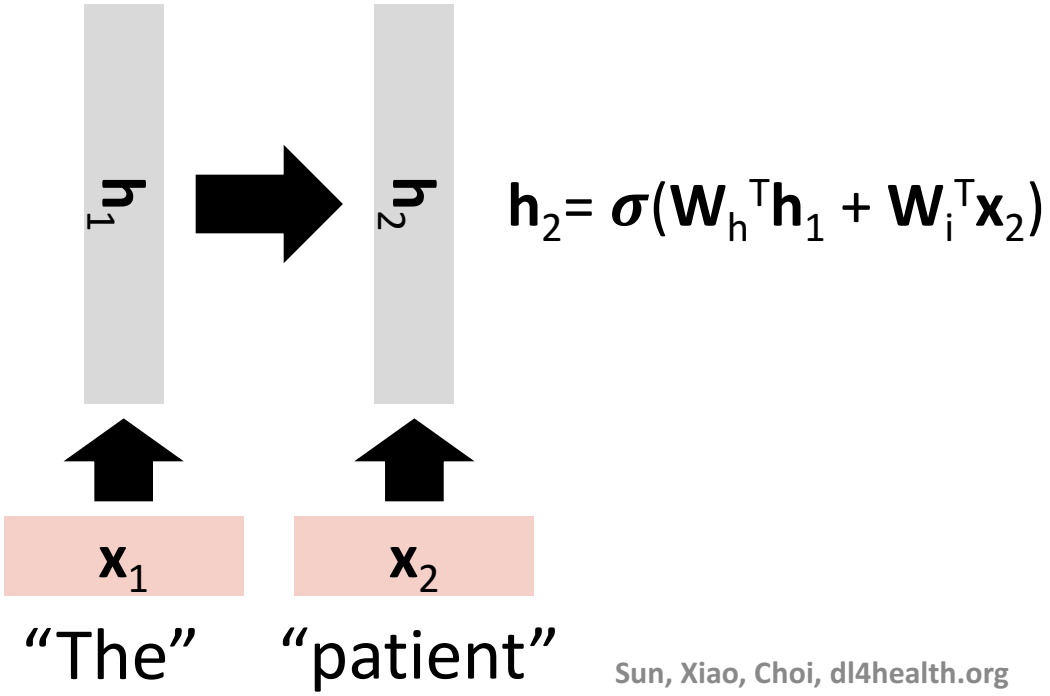


$\mathbf{h}_1 = \sigma(\mathbf{W}_i^T \mathbf{x}_1)$

$\mathbf{x}_1$ (First element in the sequence “The”)

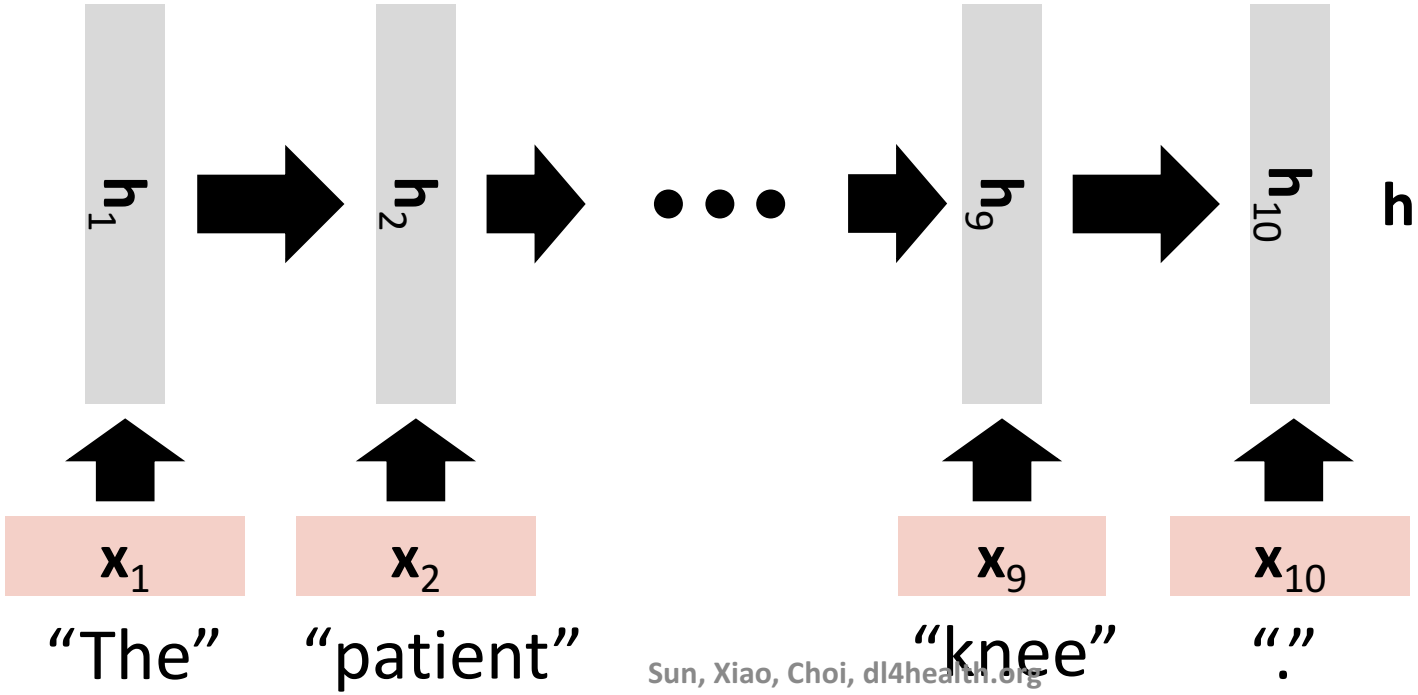
Recurrent Neural Networks

- Recurrent Neural Network (RNN)
  - Binary classification



Recurrent Neural Networks

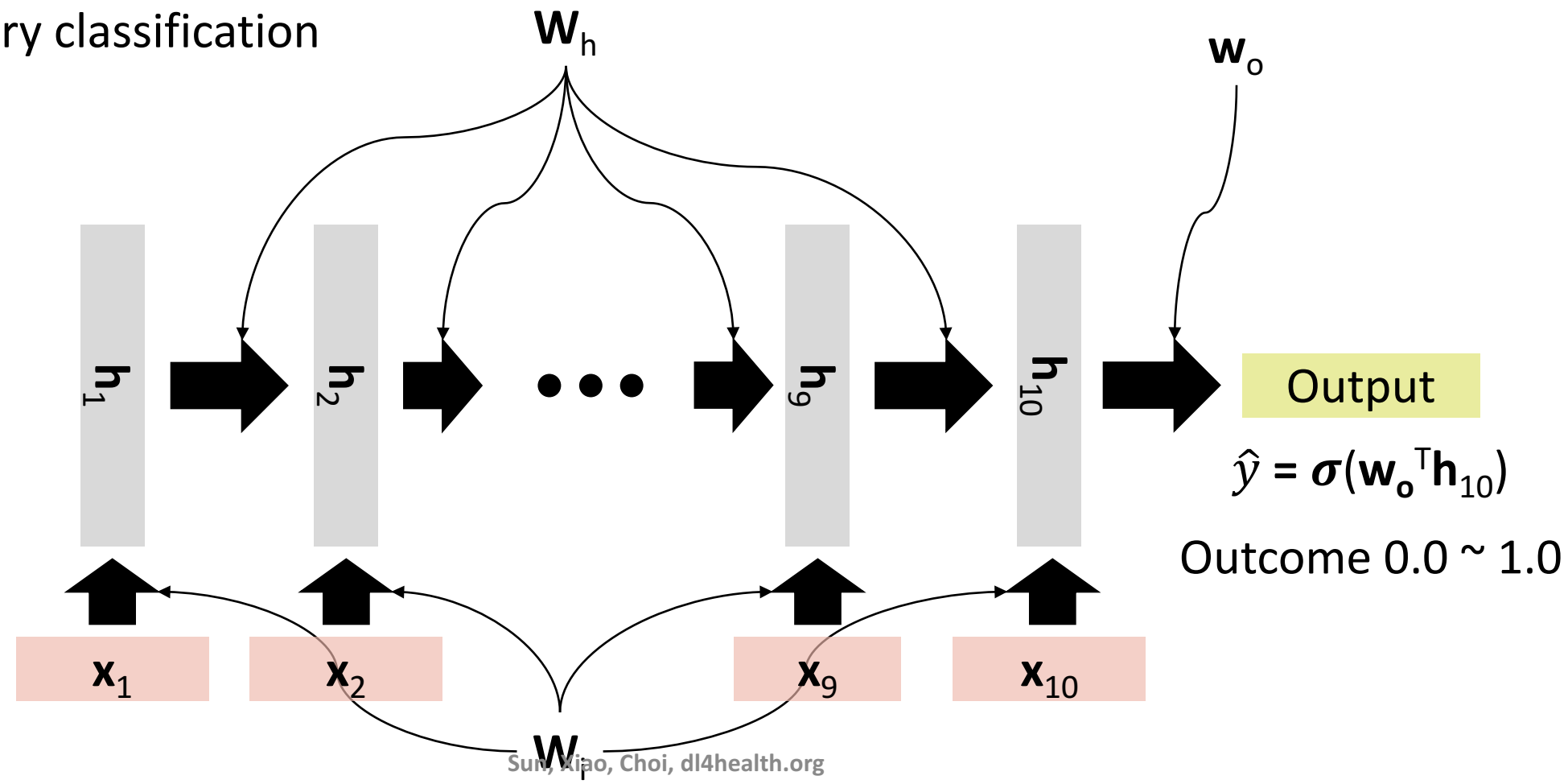
- Recurrent Neural Network (RNN)
  - Binary classification



$$\mathbf{h}_{10} = \sigma(\mathbf{W}_h^T \mathbf{h}_9 + \mathbf{W}_i^T \mathbf{x}_{10})$$

Recurrent Neural Networks

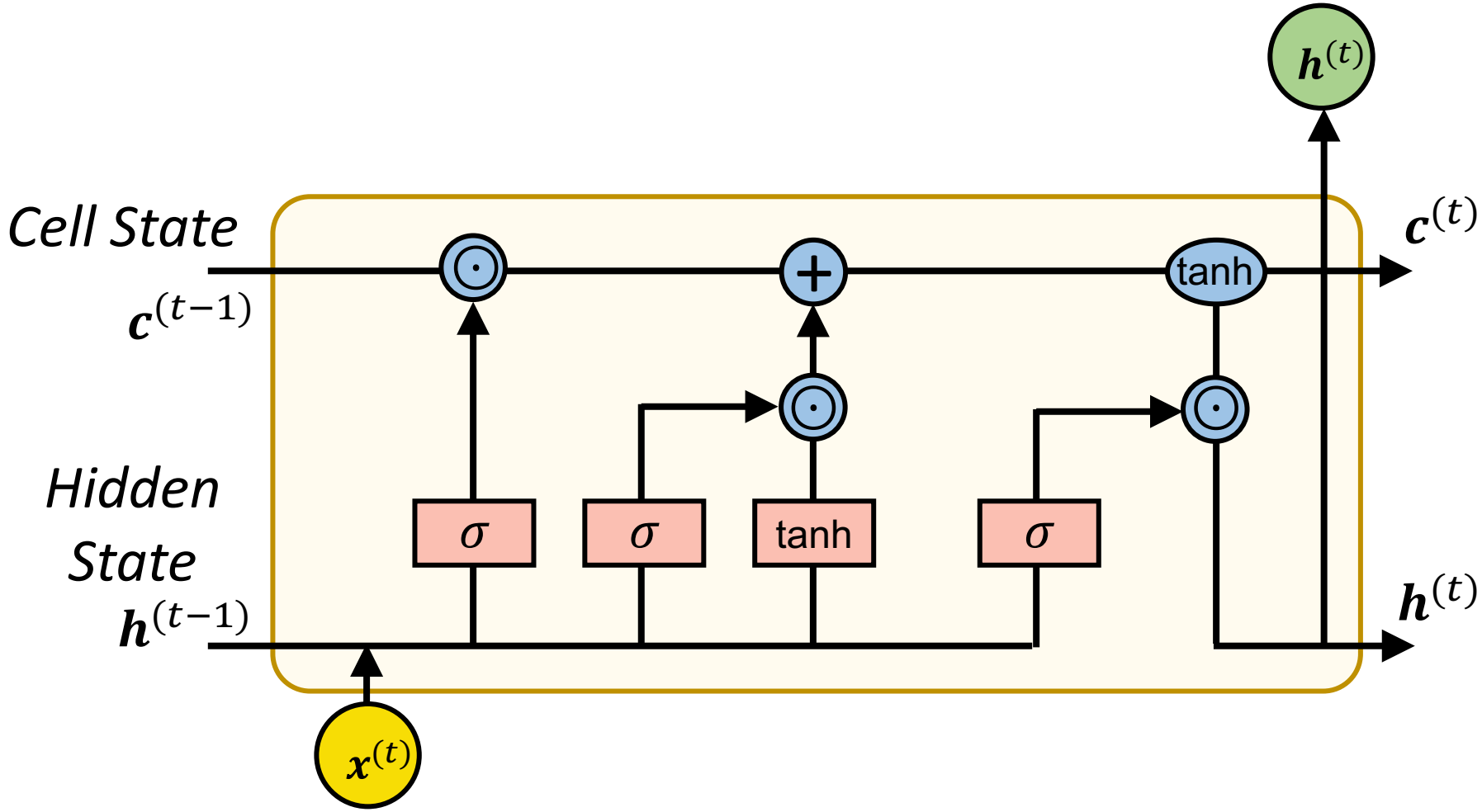
- Recurrent Neural Network (RNN)
  - Binary classification



RNN Variants

- Vanishing gradient problem
  - Vanilla RNN does not handle long sequences too well
- Two most popular RNN variants
  - Long Short-Term Memory (LSTM)
  - Gated Recurrent Units (GRU)
- Both use gates to adjust the flow of information across time

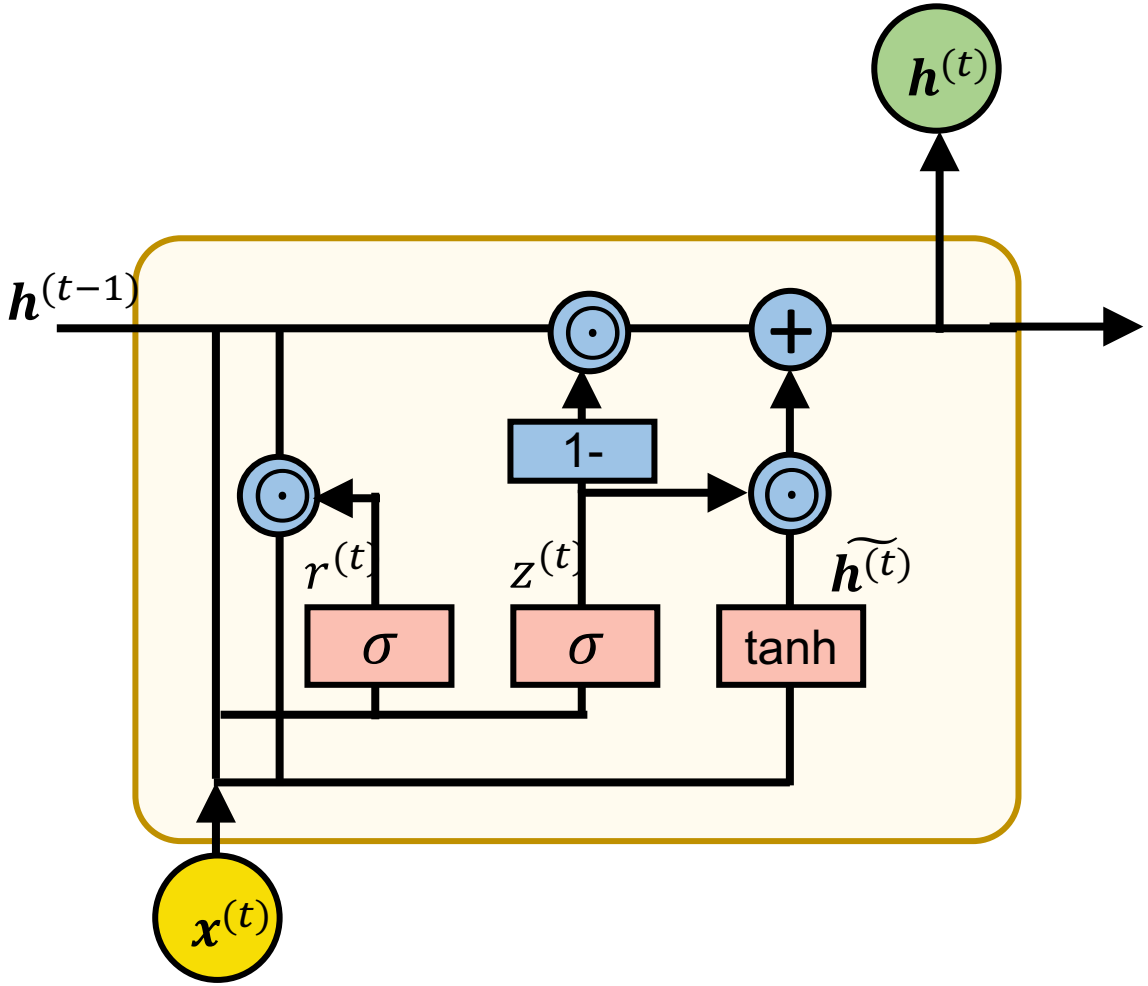
LSTM: Cell Structure



Hochreiter, S., and J. Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8): 1735–80.

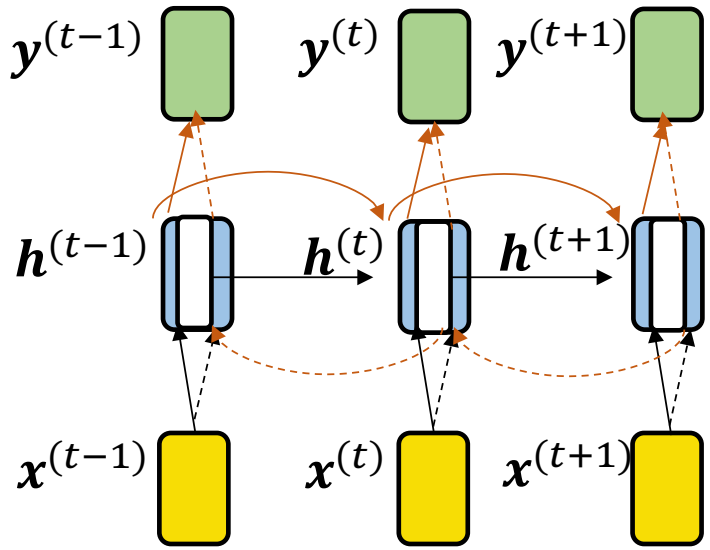
Sun, Xiao, Choi, dl4health.org

GRU: Cell Structure



Cho, Kyunghyun et al. 2014. "Learning Phrase Representations Using RNN Encoder--Decoder for Statistical Machine Translation." *EMNLP*, 1724–34.

Bidirectional RNN

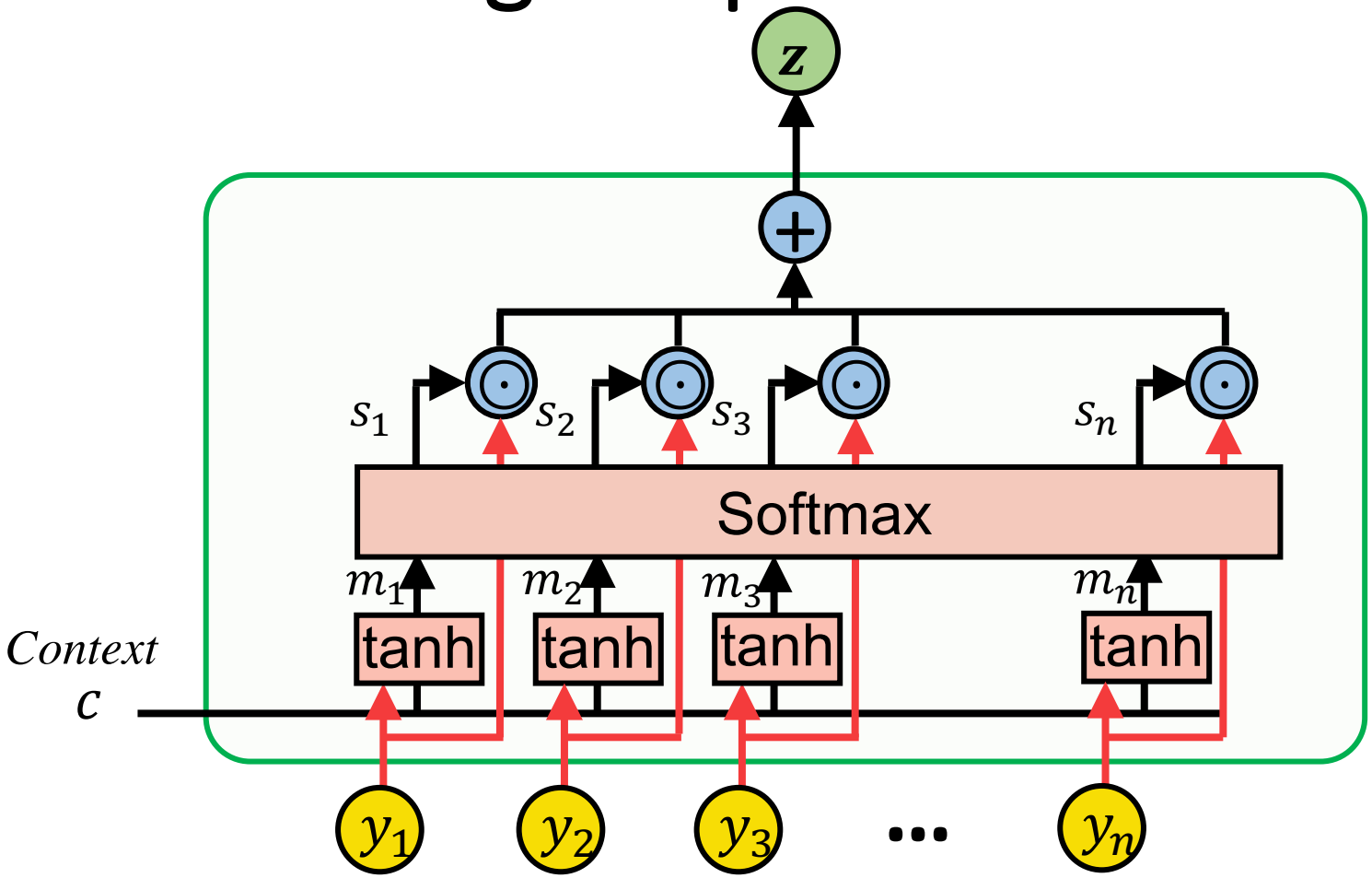


$$\overrightarrow{\boldsymbol{h}}^t = f(\boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_1)$$
$$\overleftarrow{\boldsymbol{h}}^t = f(\boldsymbol{U}\boldsymbol{x}^{(t)} + \boldsymbol{W}\boldsymbol{h}^{(t+1)} + \boldsymbol{b}_1)$$
$$y^{(t)} = g(\boldsymbol{V}\left[\overrightarrow{\boldsymbol{h}}^t; \overleftarrow{\boldsymbol{h}}^t\right] + \boldsymbol{b}_2)$$

Schuster and Paliwal, Bidirectional recurrent neural networks, 1997

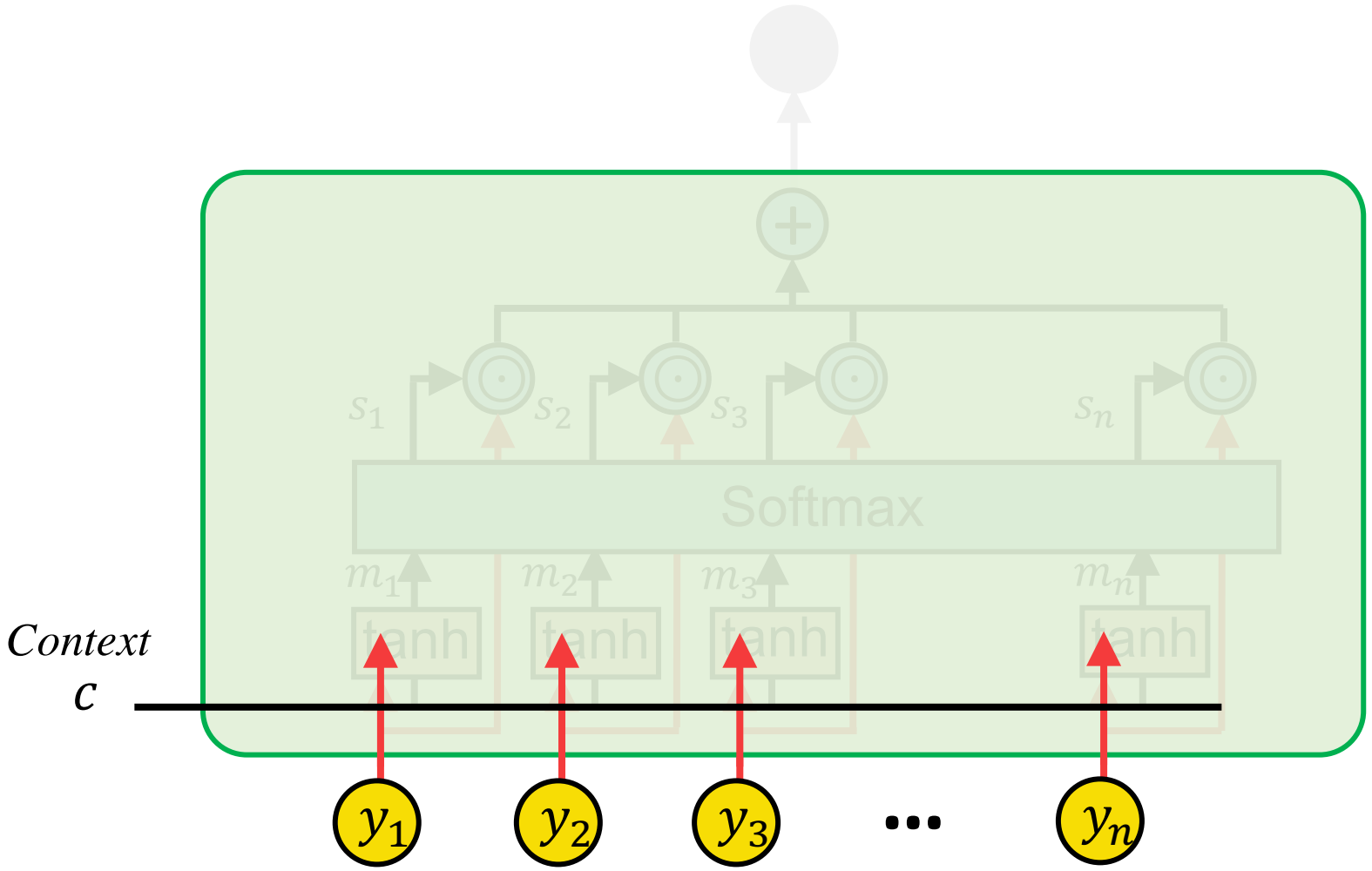
Basic Methods: Attention Mechanism

Attention for Image Caption Generation

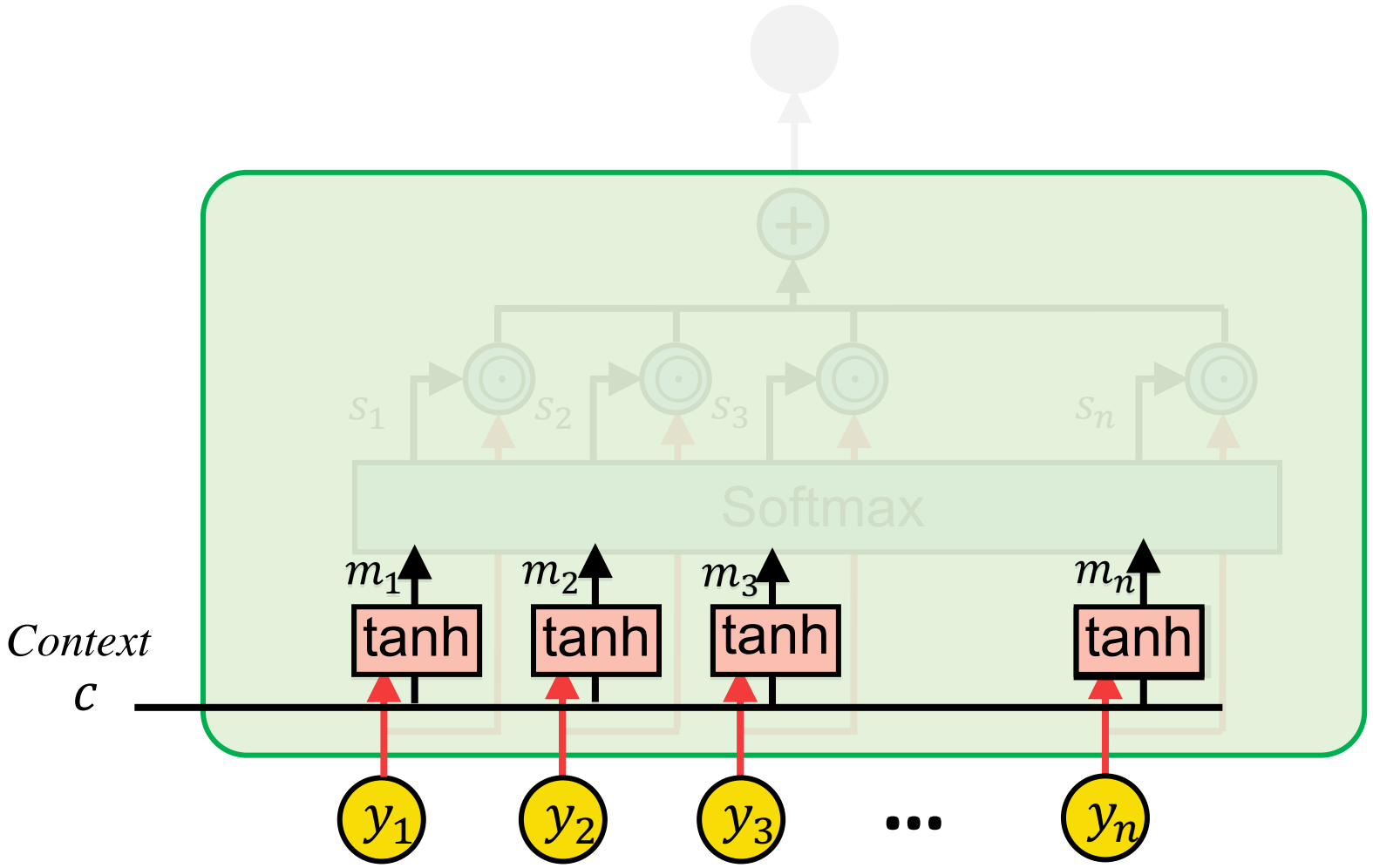


Xu et al., Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, ICML' 15

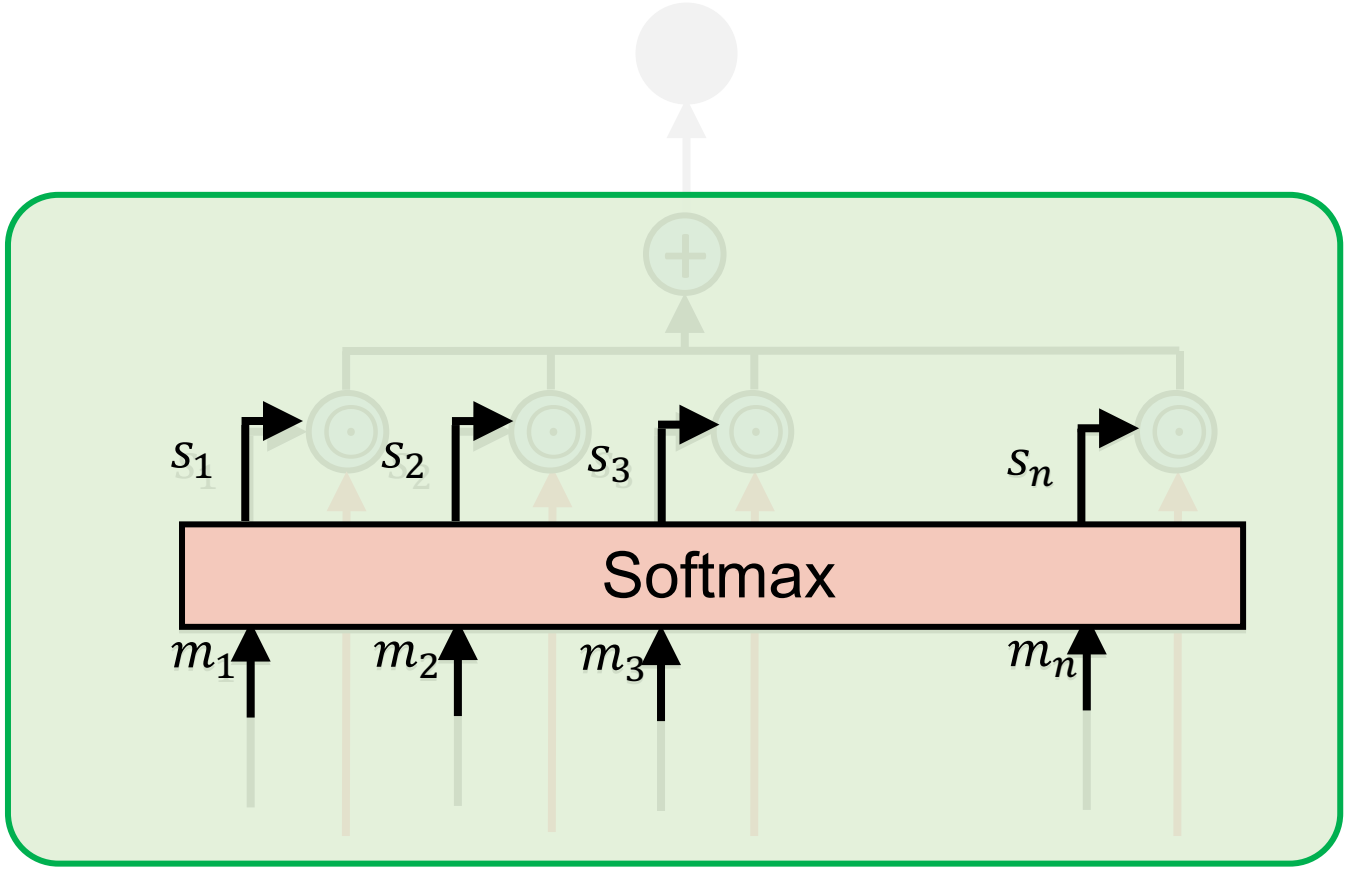
Input



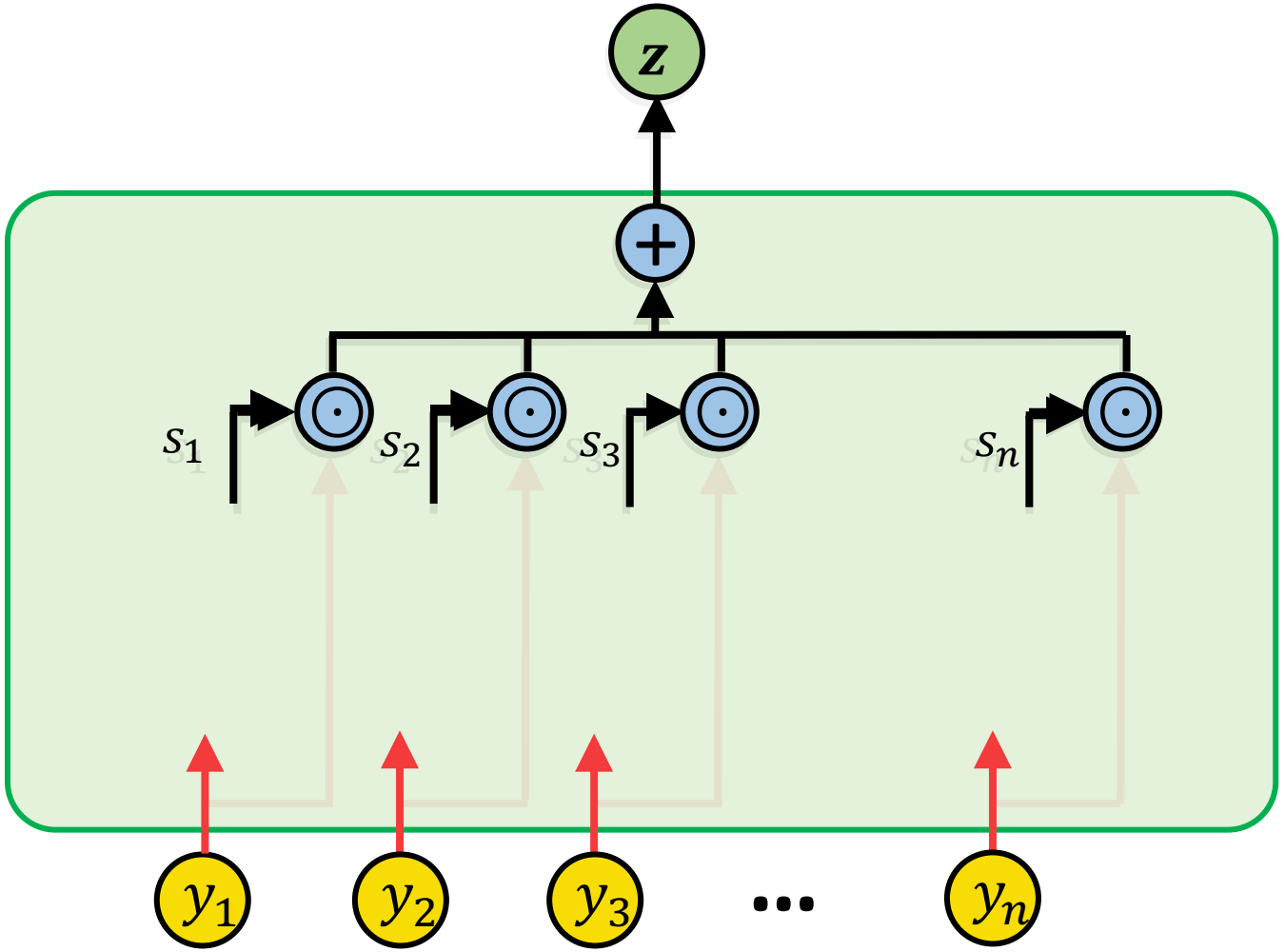
Aggregation



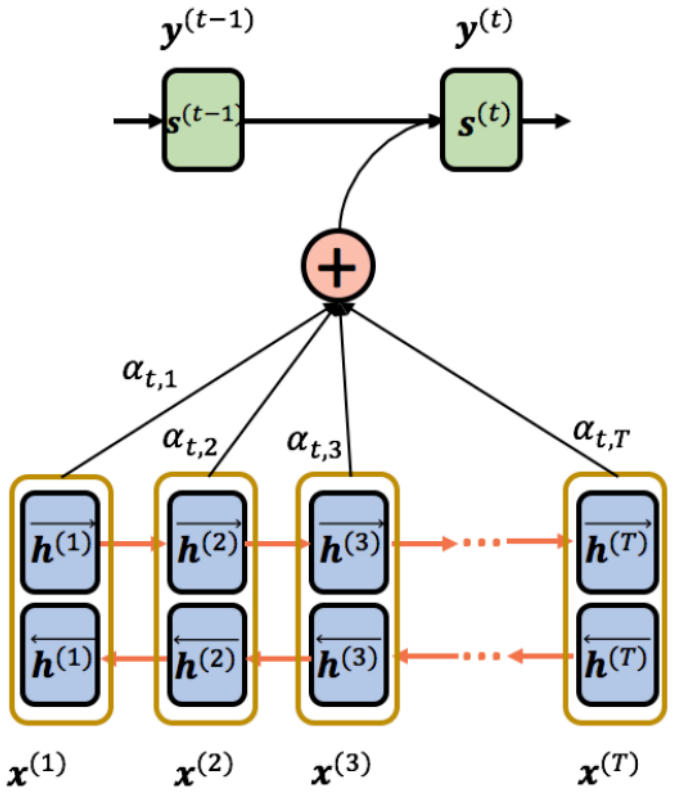
Relevance



Output



Attention for Neural Machine Translation

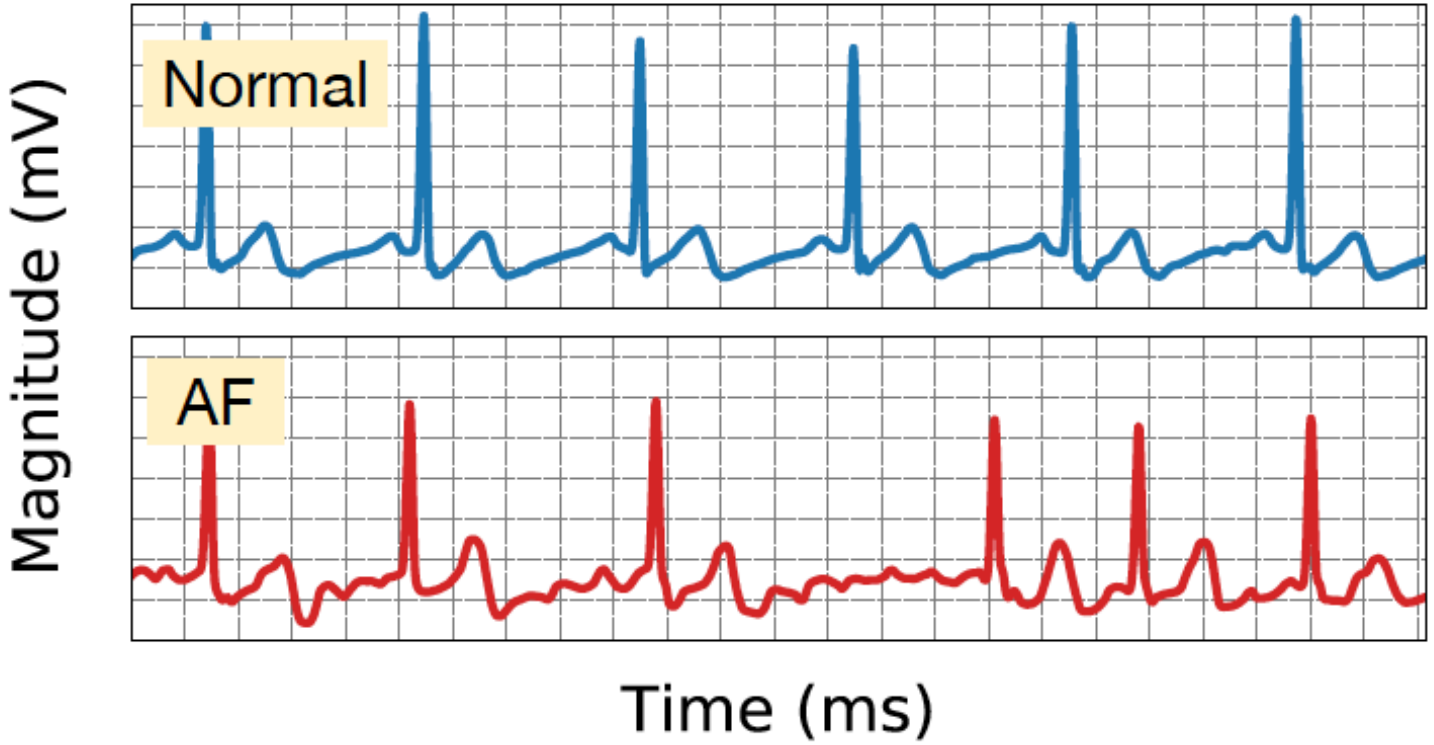


Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR' 15

Basic Methods: Convolutional Neural Nets

Convolutional Neural Networks (CNN)

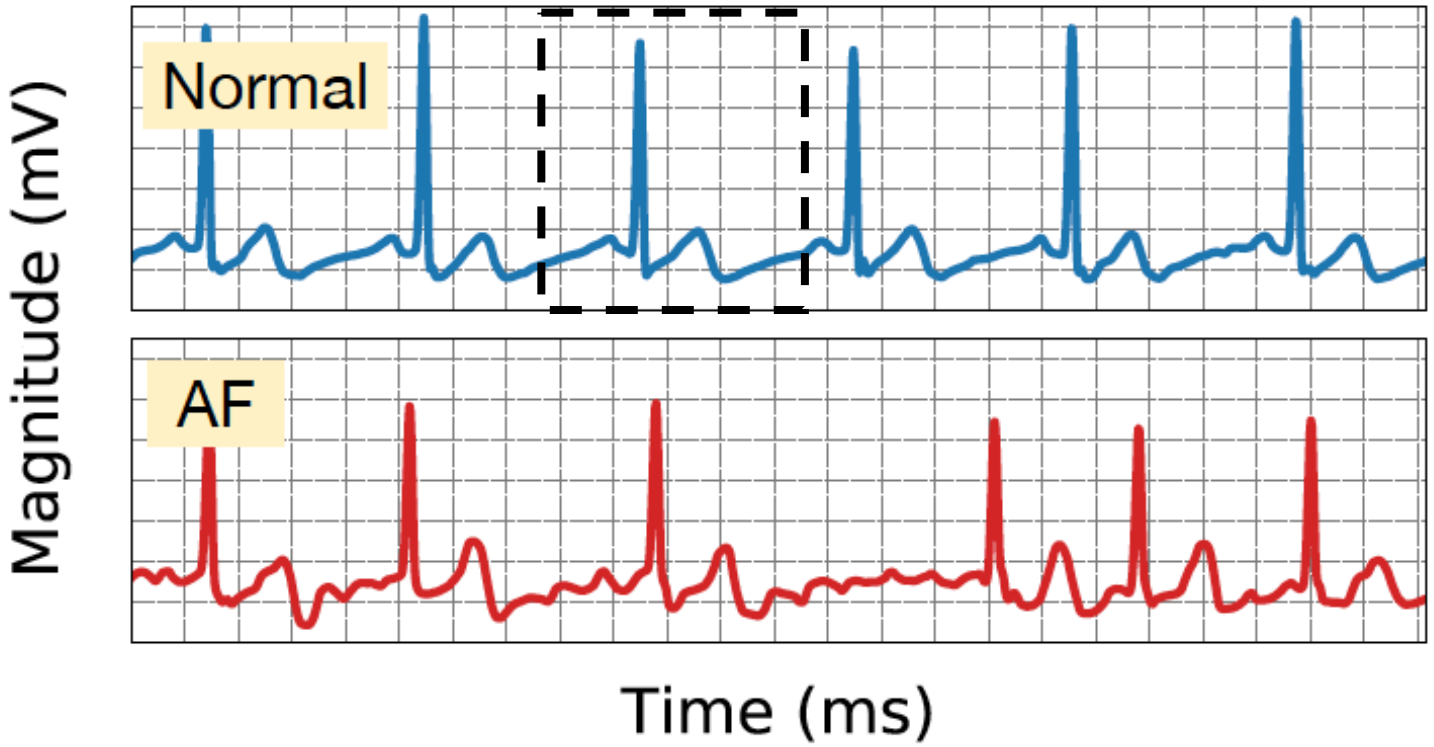
- How to diagnose atrial fibrillation (AF) from ECG signal?



Sun, Xiao, Choi, dl4health.org

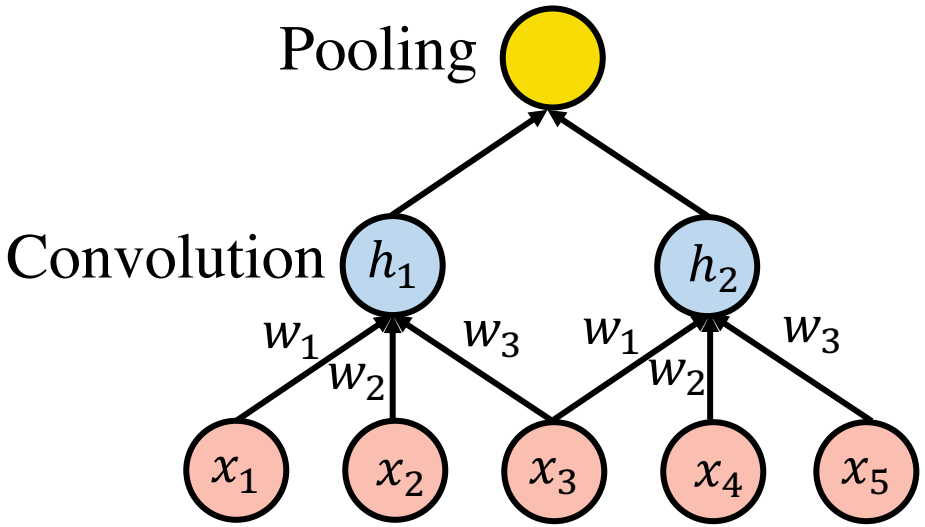
Convolutional Neural Networks (CNN)

- How to diagnose atrial fibrillation (AF) from ECG signal?
- Focus on the local features, build up global features



Sun, Xiao, Choi, dl4health.org

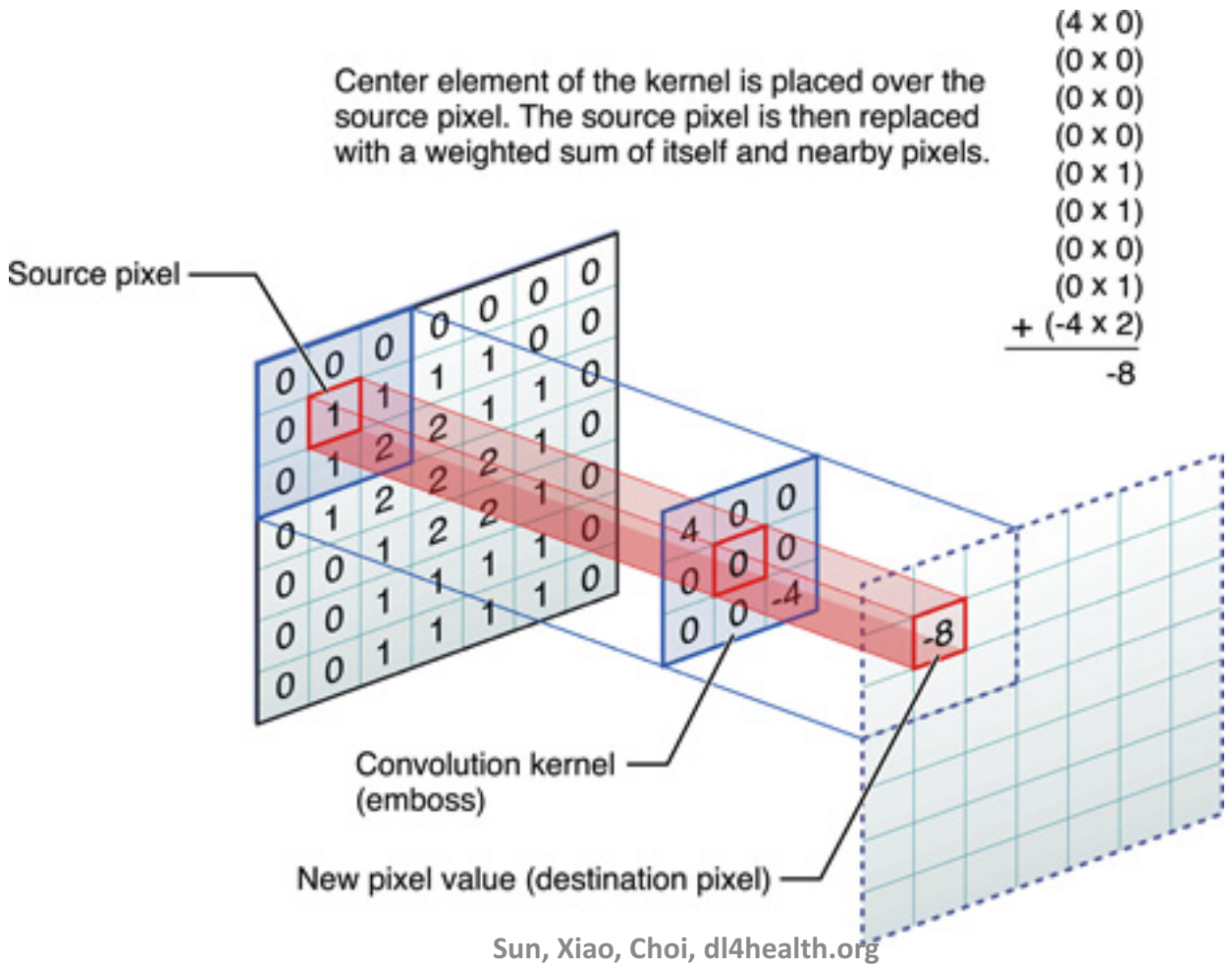
Convolutional Neural Networks (CNN)



- Process data that has a known grid-like structure (e.g., images, waveforms).
- Utilize a specialized linear operation – convolution.
- Advantages: sparse interactions, parameter sharing, and translational invariance.

Convolutional Neural Networks (CNN)

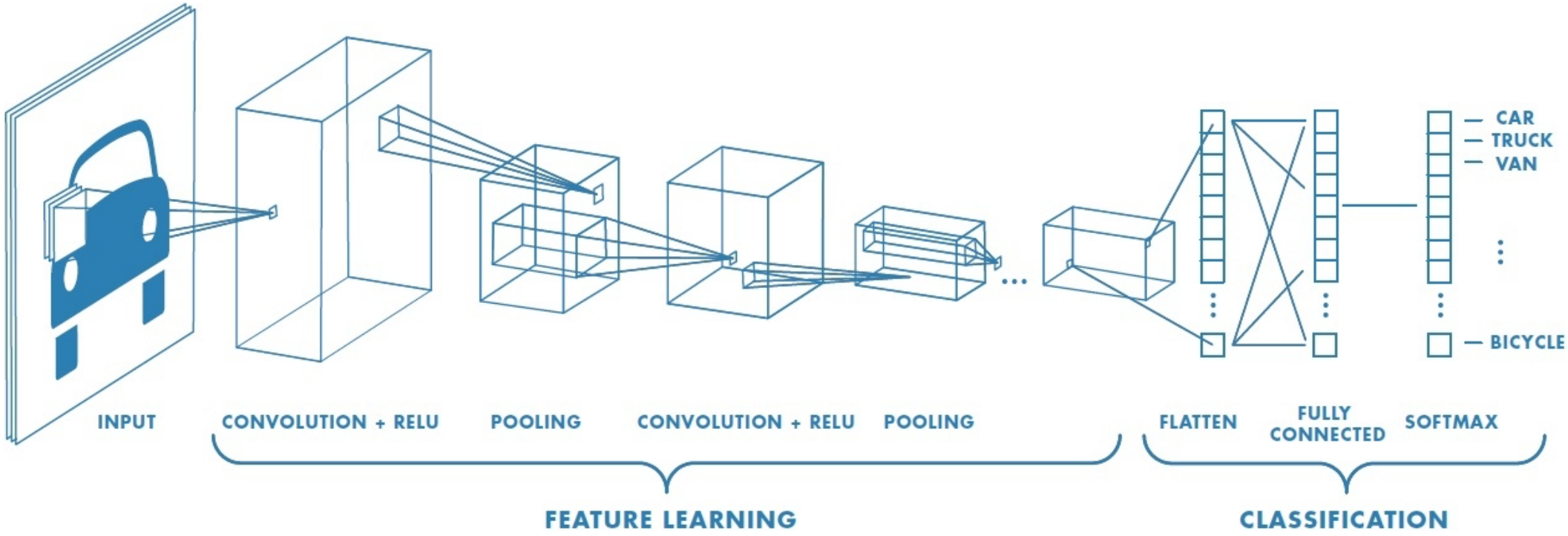
- Focus on the local features, build up global features



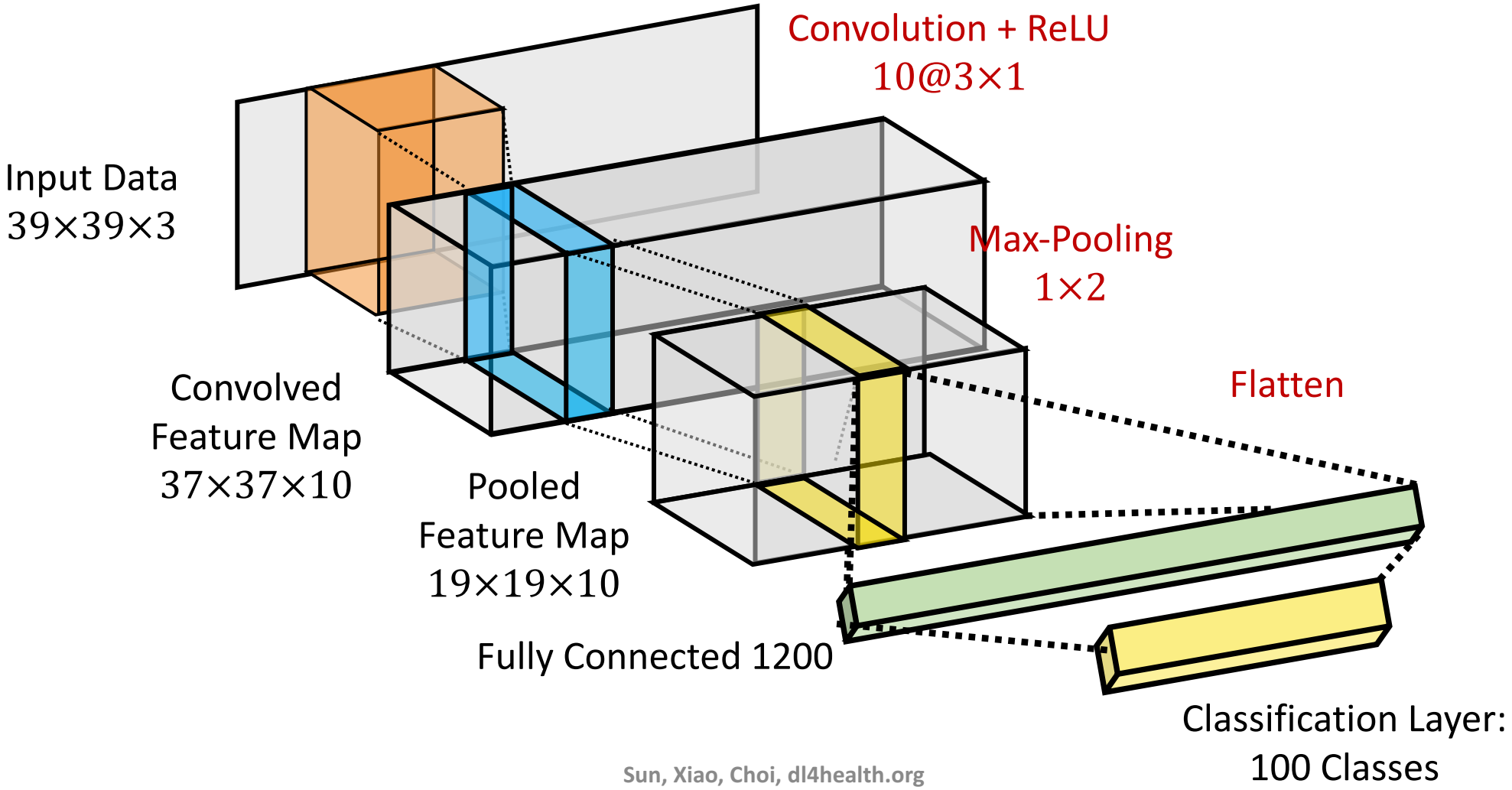
Sun, Xiao, Choi, dl4health.org

Convolutional Neural Networks (CNN)

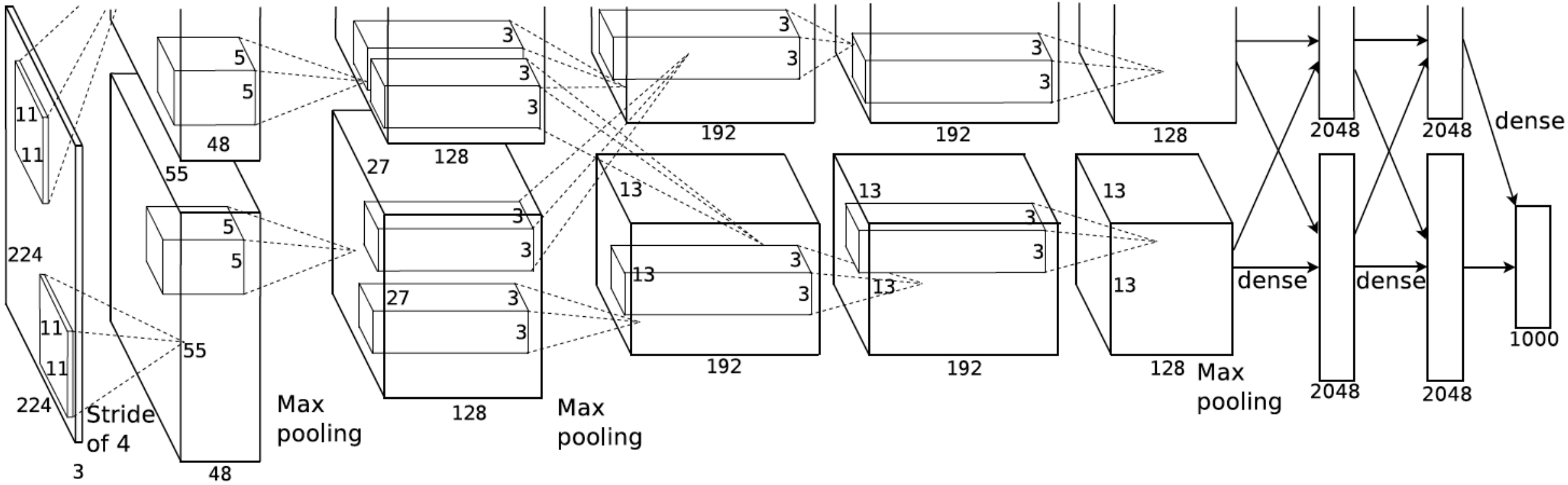
- Focus on the local features, build up global features



Convolutional Neural Networks (CNN)

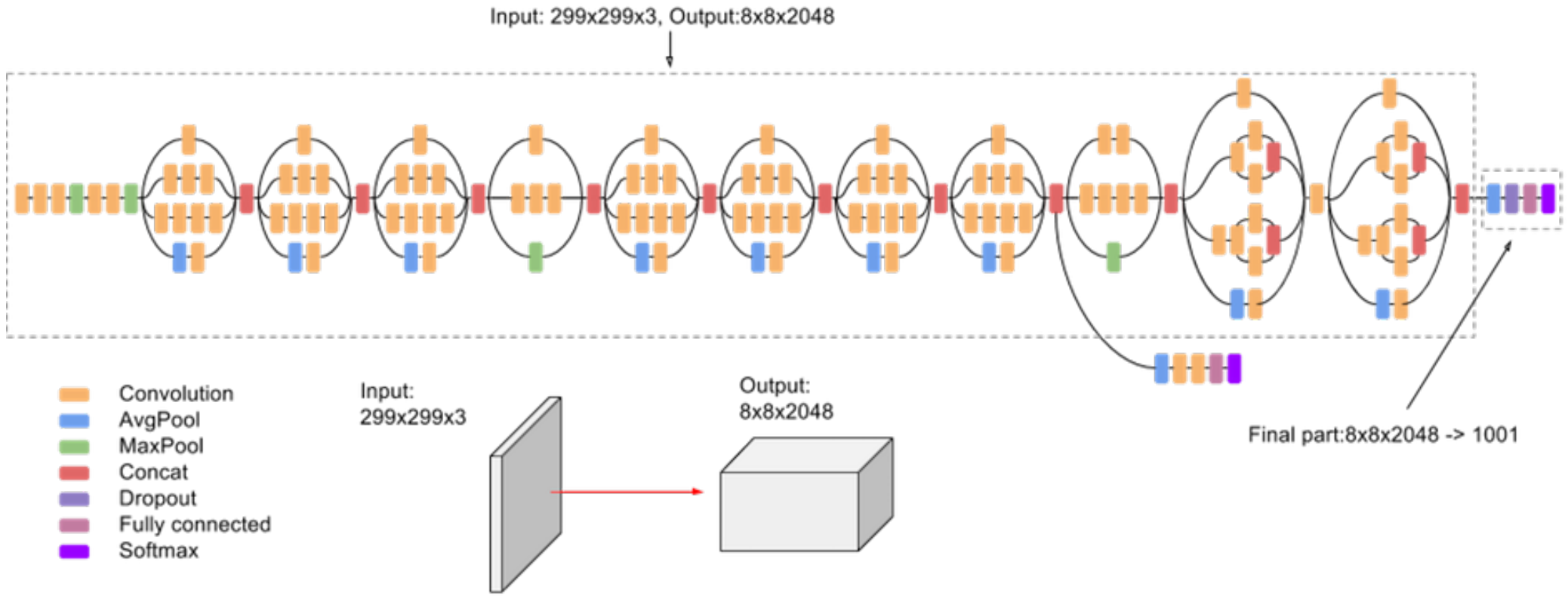


AlexNet



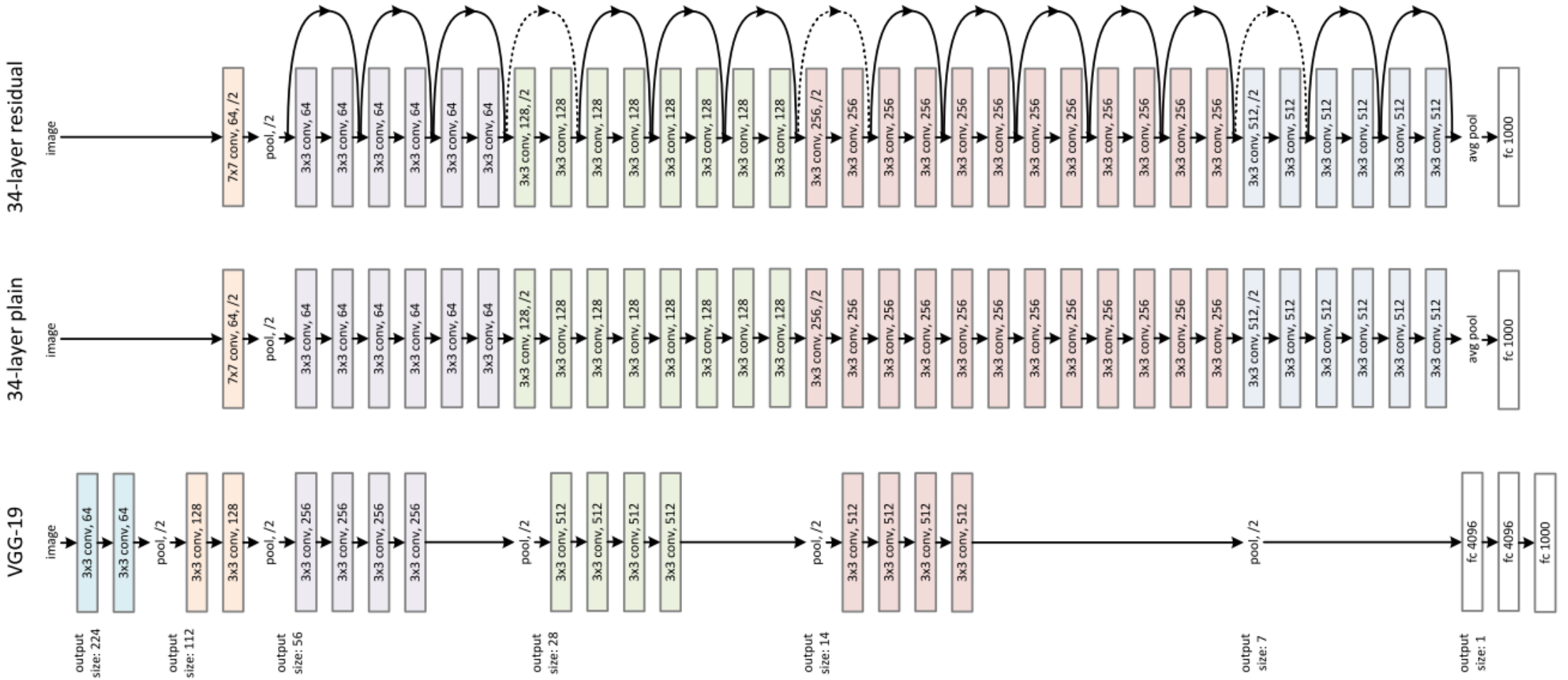
AlexNet: Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks.", NIPS 2012
VGGNet: Karen Simonyan, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition.", ICLR 2015

Inception



Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions.", CVPR 2015

ResNet



Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition", CVPR 2016

Development of CNN Architectures

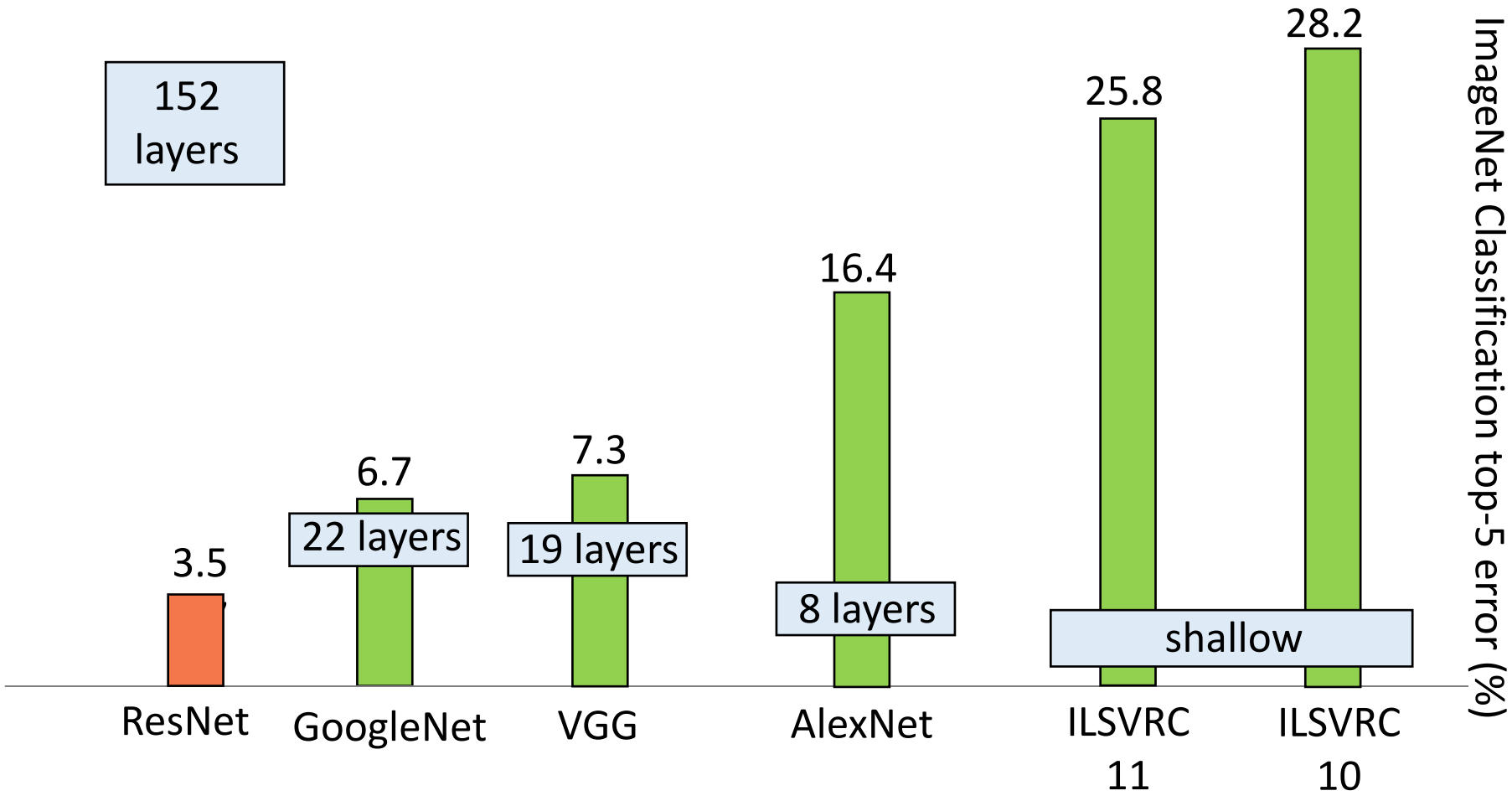


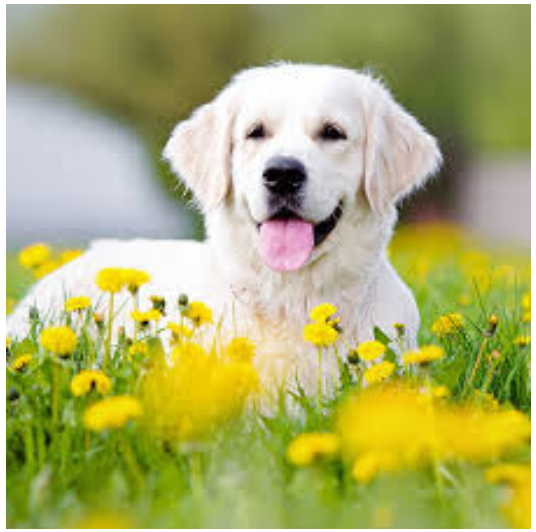
Figure idea borrowed from He *et. al.*, CVPR 2016

Basic Methods: Autoencoders

Autoencoders

- Compression & decompression
  - Learning the latent representation of a given sample **x**

256 X 256 dimensions



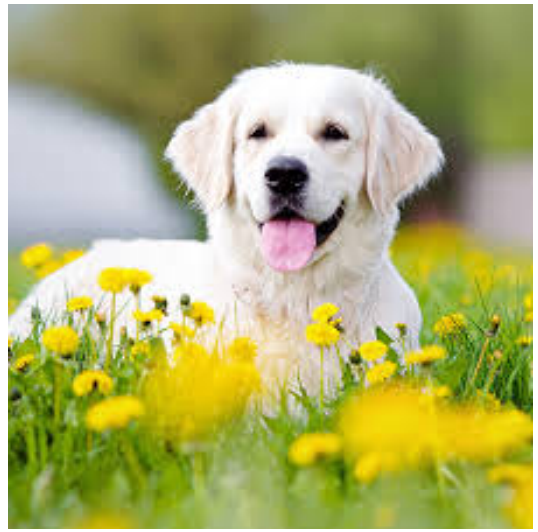
Compression

128 dimensions

0.1
0.2
-0.4
1.5
-2.1
0.2
…
…

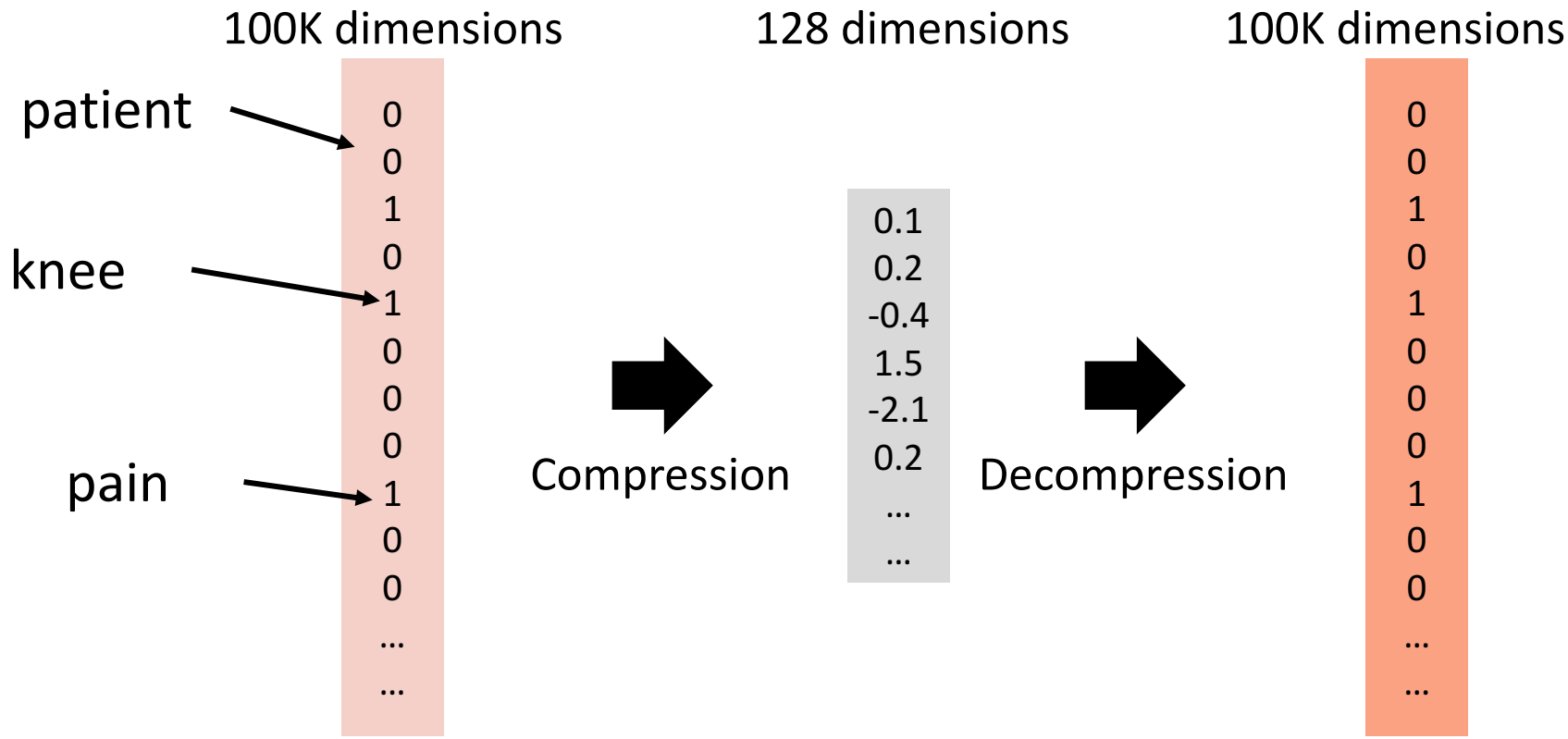
Decompression

256 X 256 dimensions



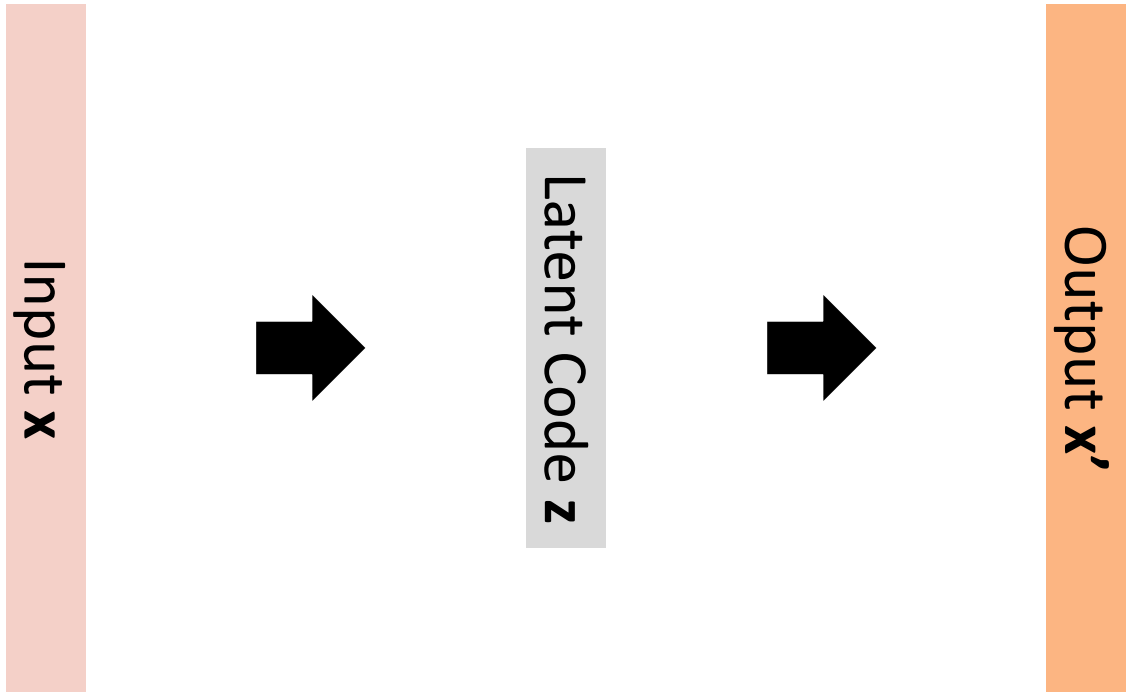
Autoencoders

- Compression & decompression
  - Learning the latent representation of a given sample **x**



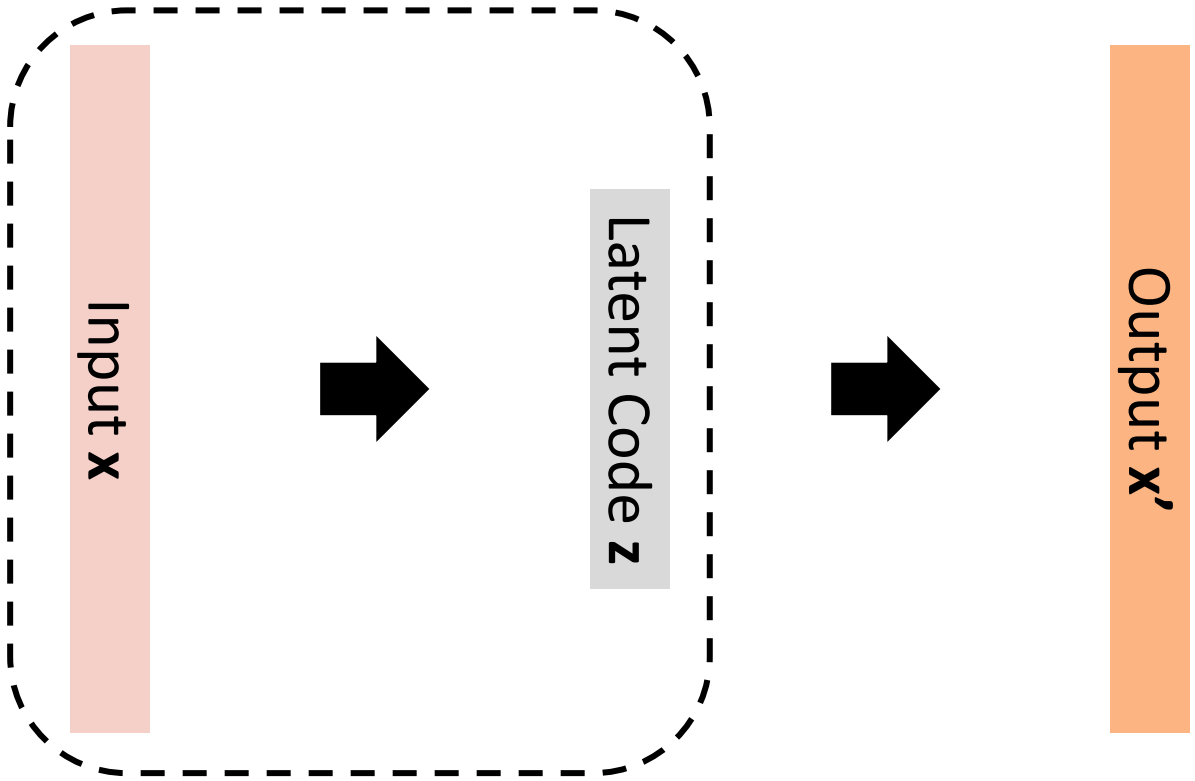
Autoencoders

- Compression & decompression
  - Learning the latent representation of a given sample **x**



Autoencoders

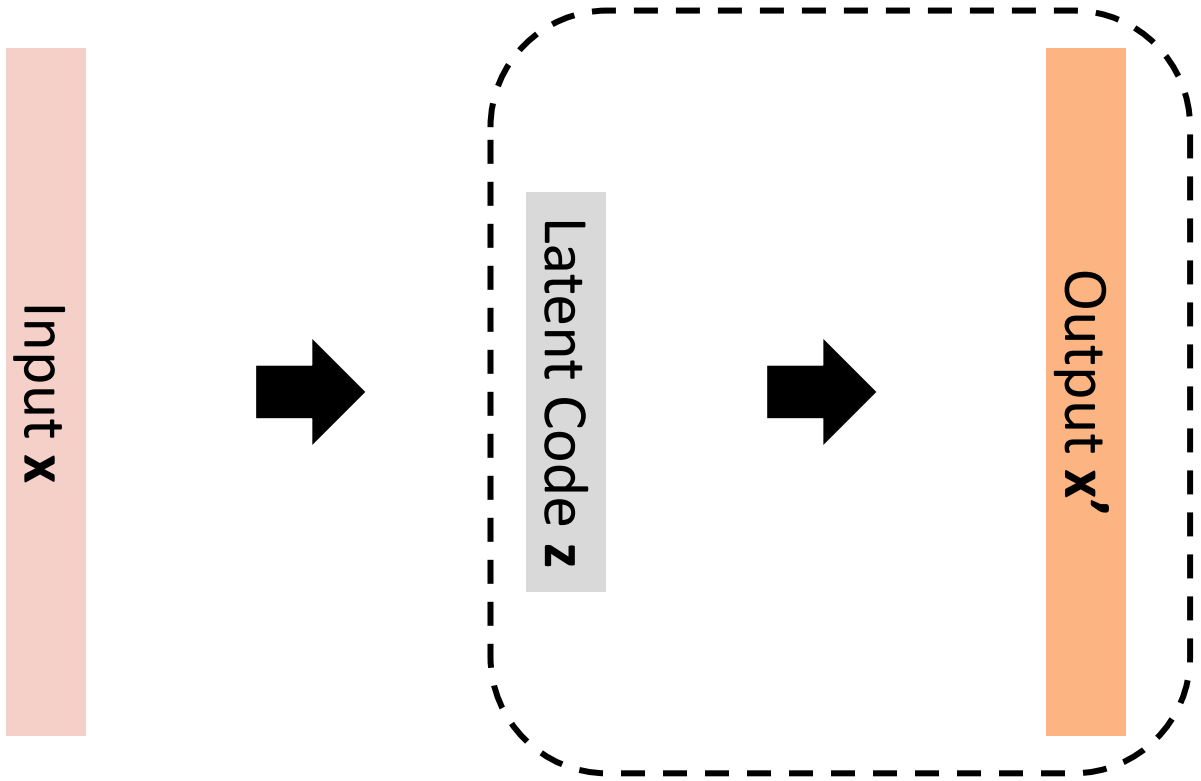
- Compression & decompression
  - Learning the latent representation of a given sample **x**



Encoding $\mathbf{z} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$

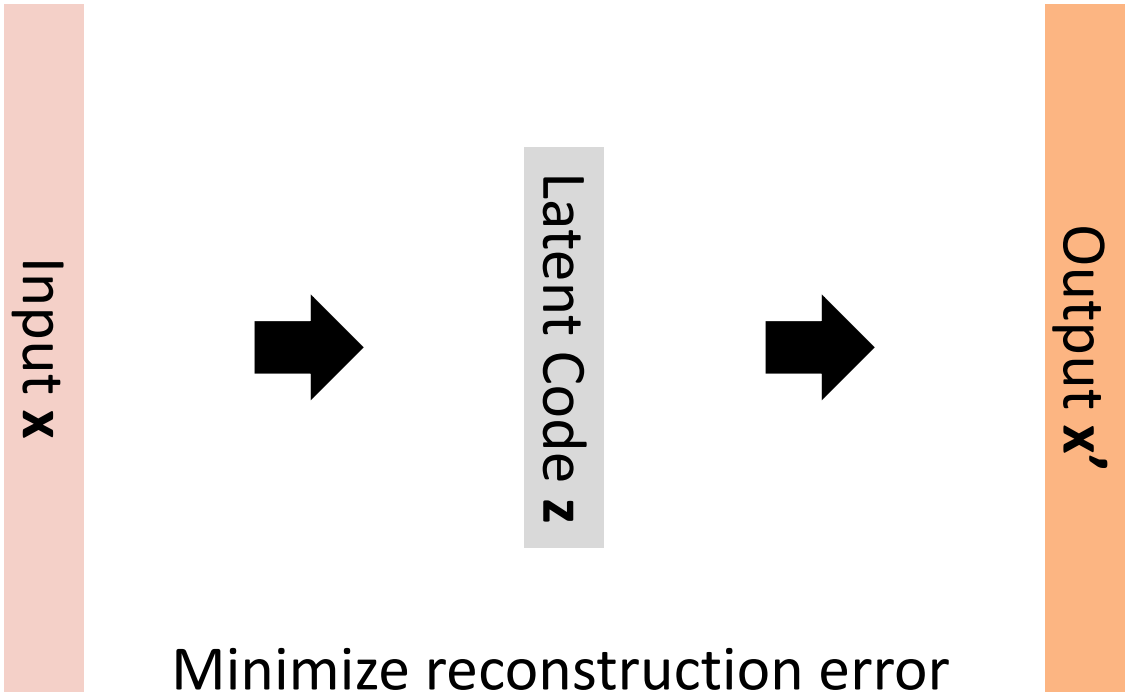
Autoencoders

- Compression & decompression
  - Learning the latent representation of a given sample **x**



Decoding $\mathbf{x}' = \sigma'(\mathbf{W}'\mathbf{z} + \mathbf{b}')$

Autoencoders



Minimize reconstruction error

$$\mathcal{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2$$

Autoencoders are designed to be ***unable*** to learn to copy perfectly. Usually they are **restricted** in ways that allow them to **copy only approximately**. Because the model is forced to **prioritize which aspects of the input should be copied**, it often learns useful properties of the data.

Deep Learning (Ian Goodfellow, Yoshua Bengio, Aaron Courville)

Properties of Autoencoders

Which of the following number sequences do you find the easiest to memorize?

- 40, 27, 25, 36, 81, 57, 10, 73, 19, 68
- 50, 25, 76, 38, 19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20

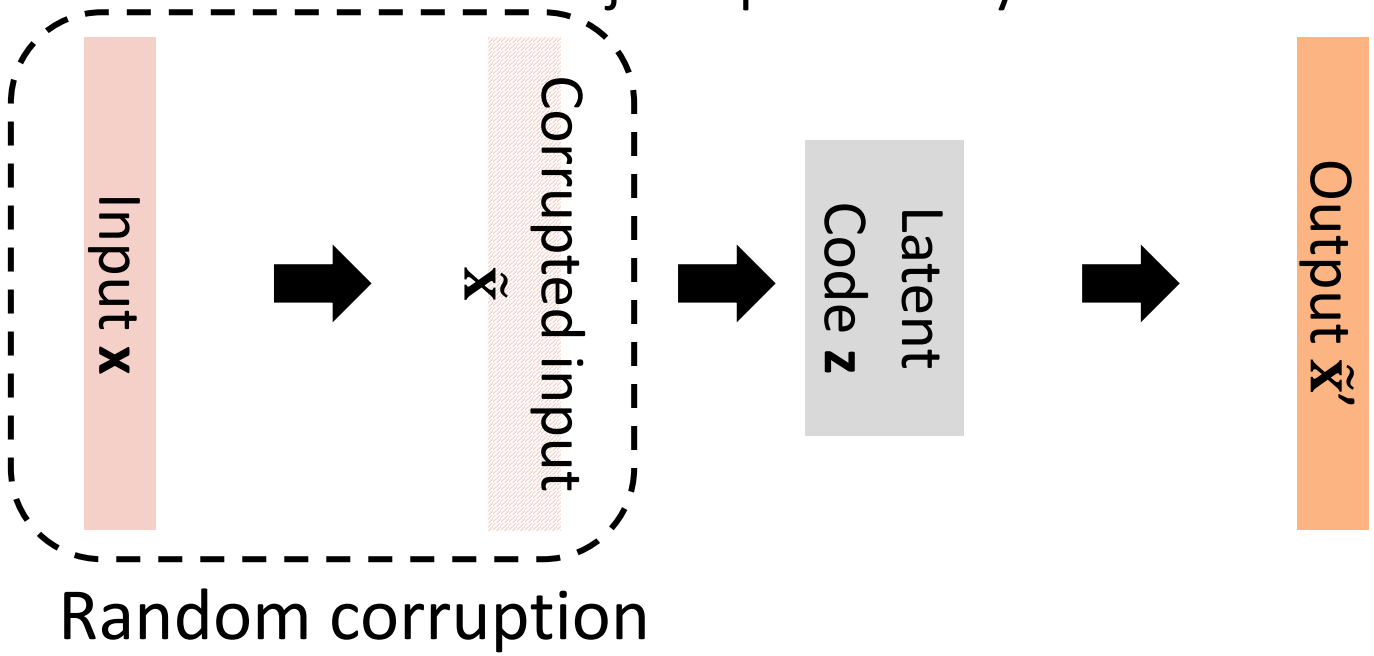
1. a. Capture the **intrinsic properties** of data → feed them into downstream applications
   b. Can be thought of as **patterns** in data → generate new data
2. Produce low-dimensional vectors (**efficient**/compact representations)
   a. Efficient for storage
   b. Efficient for downstream models
   c. May be **free of noise** in input
   d. Easier to **visualize** than high-dimensional data

Properties of Autoencoders

3. **Are flexible: Can be modified/guided/regularized in various ways**:
    a. Input data, e.g. add noise
    b. Output data, e.g. something different from the input
    c. Architecture, e.g. fully connected layer → convolutional layer
    d. Loss, e.g. add additional loss terms → capture other useful information from input
    e. Latent space, e.g. Gaussian (more later in VAE)
        i. **Enforce certain prior knowledge, usually through additional loss terms**
        ii. **Analyzing the latent space/representations is a trend (?),** e.g. debiasing word embeddings

Denoising Autoencoders

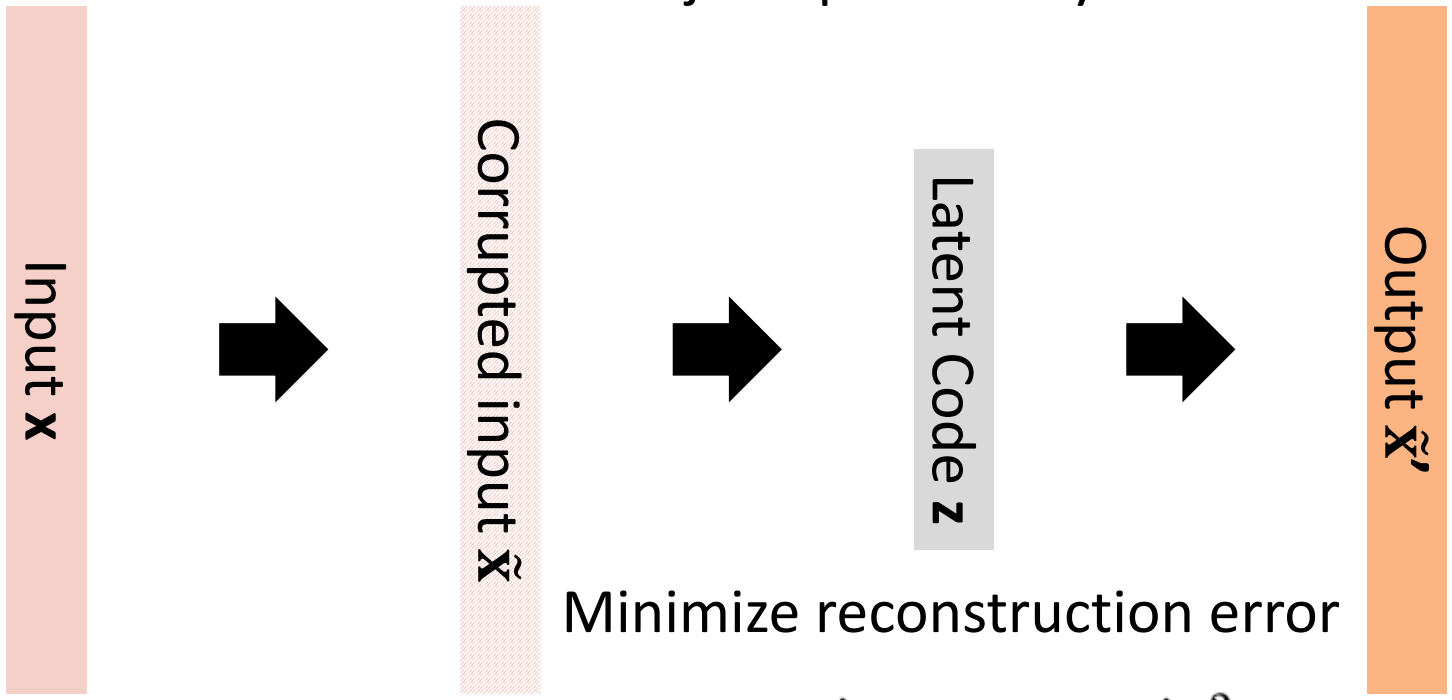
- Corrupt the input sample **x**
  - To learn a robust representation of **x**
  - The model strives to learn the joint probability of the dimensions of **x**



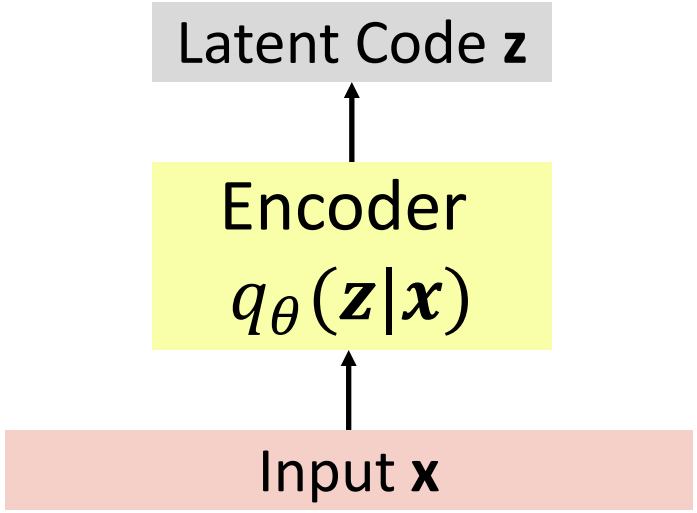
Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders, ICML' 08

Denoising Autoencoders

- Corrupt the input sample **x**
  - To learn a robust representation of **x**
  - The model strives to learn the joint probability of the dimensions of **x**



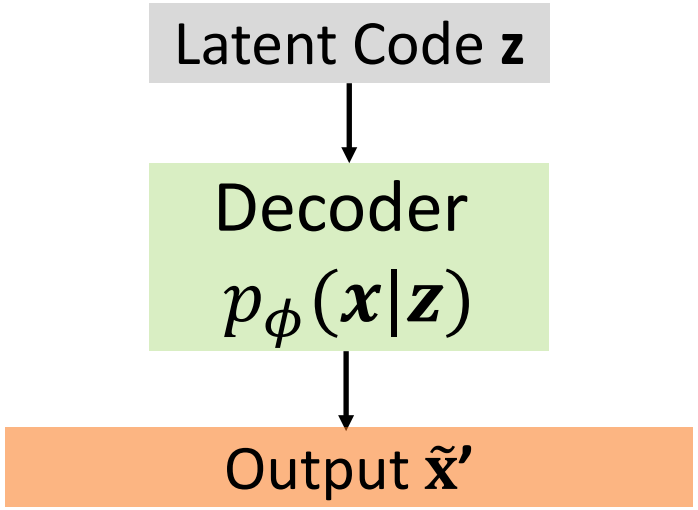
Variational Autoencoders: Encoder



- The encoder learn an efficient compression of the data into this lower-dimensional space.
- It outputs parameters to $q_\theta(\boldsymbol{z}|\boldsymbol{x})$, a Gaussian probability density.

Diederik P Kingma; Welling, Max (2013). Auto-Encoding Variational Bayes. arXiv:1312.6114

Variational Autoencoders: Decoder



- The decoder learn learned to reconstruct the input data given its latent representation.
- It achieves this via sampling from the output distribution of the encoder to get noisy values of the representations.

Variational Autoencoders: Key Idea

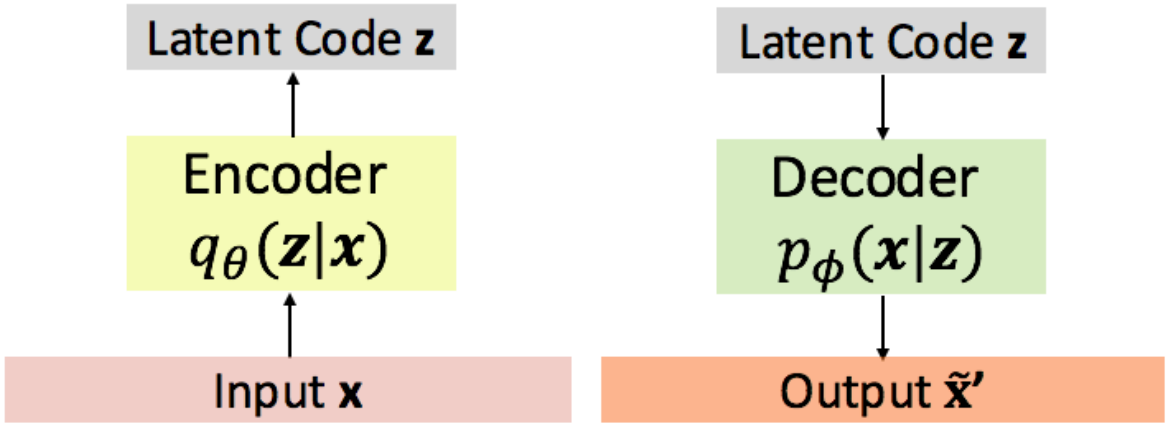
Assumption: there is latent mechanism for generating data $\boldsymbol{x}$
Challenge: marginal log-likelihood of data is intractable

$$\log p_\phi(\boldsymbol{x}|\boldsymbol{z}) = \log \int p_\phi(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z}$$

Solution:

- Approximate the posterior $p_\phi(\boldsymbol{z}|\boldsymbol{x})$ with a tractable and simpler distribution $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ (tightness condition: $p_\phi(\boldsymbol{z}|\boldsymbol{x}) = q_\theta(\boldsymbol{z}|\boldsymbol{x})$)
- Cast inference as optimization problem over parameters of the model and the approximate posterior.

Variational Autoencoders: Evidence Lower Bound (ELBO)



$$\log p_\phi(\boldsymbol{x}, \boldsymbol{z})$$

$$= \log \int q_\theta(\boldsymbol{z}|\boldsymbol{x}) \frac{p_\phi(\boldsymbol{x}, \boldsymbol{z})}{q_\theta(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z}$$

$$\geq \int q_\theta(\boldsymbol{z}|\boldsymbol{x}) \log \frac{p_\phi(\boldsymbol{x}, \boldsymbol{z})}{q_\theta(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z}$$

This is called **ELBO**$(\boldsymbol{X}; \phi, \theta)$, the new variational objective to be optimized, for the marginal likelihood of $\boldsymbol{x}$.

Variational Autoencoders: Variational Bayes

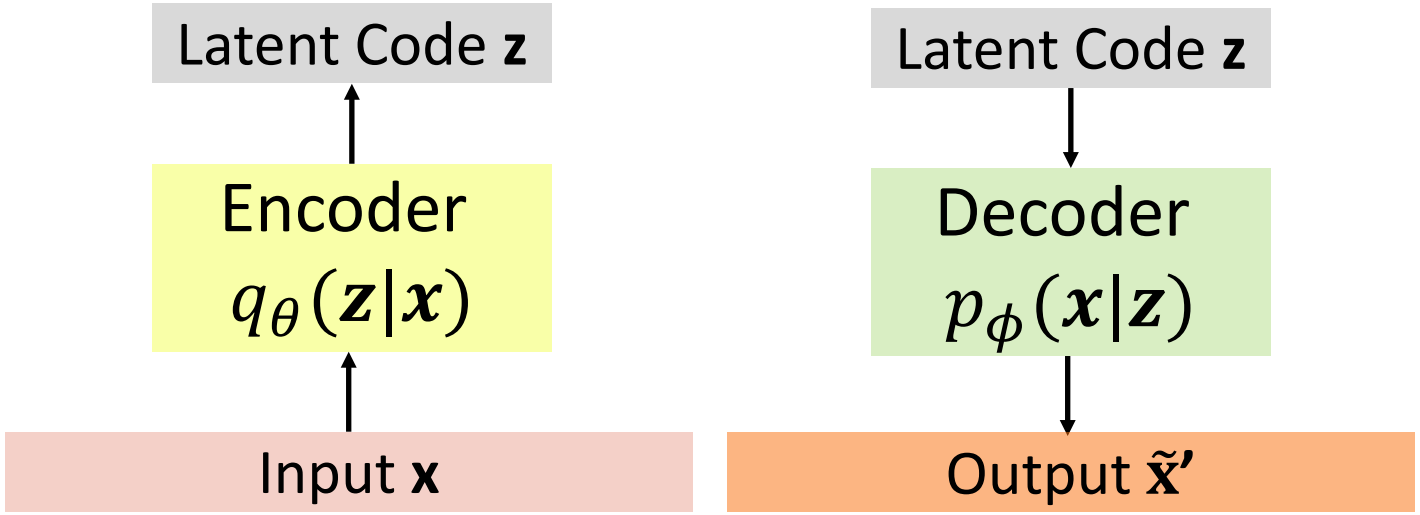
ELBO$(\boldsymbol{X}; \phi, \theta)$

Kullback-Leibler (KL) divergence for the information loss of variational approximation

$$\log p_\phi(\boldsymbol{x}) = E_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[\log\left(\frac{p_\phi(\boldsymbol{x},\boldsymbol{z})}{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\right)\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x})||p_\phi(\boldsymbol{x}|\boldsymbol{z}))$$

$$\textbf{ELBO}(\boldsymbol{X}; \phi, \theta) = E_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[\log(p_\phi(\boldsymbol{x}|\boldsymbol{z}))\right] - KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}), p_\phi(\boldsymbol{z}))$$

Variational Autoencoders: Loss Function



$$L(\theta,\phi) = -E_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[\log(p_\phi(\boldsymbol{x}|\boldsymbol{z}))\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}), p_\phi(\boldsymbol{z}))$$

reconstruction loss

penalty for information loss

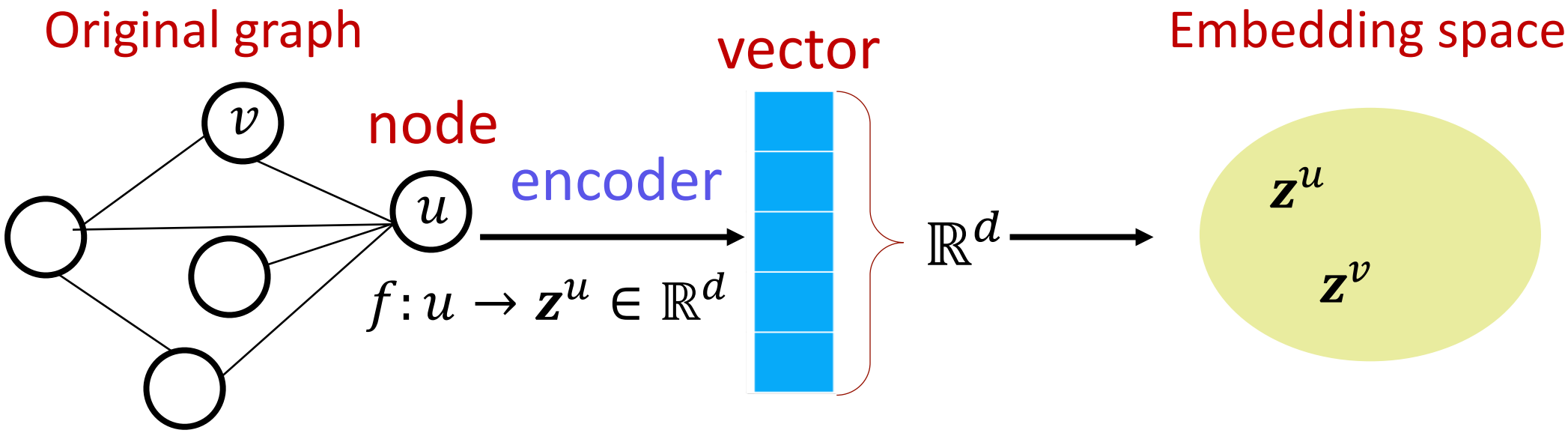
Basic Methods: Graph Convolutional Networks (GCN)

GCN: Motivation

- Graphs are universal representations of pairwise relationships (gene expression networks, knowledge graphs, biomedical networks).
- Grid-like data can be viewed as regular graphs, where CNN models demonstrated good performance.
- GCNs are the generalization of CNN to irregular graphs.

Kipf & Welling (ICLR 2017), Semi-Supervised Classification with Graph Convolutional Networks
Defferrard et al. (NIPS 2016), Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering

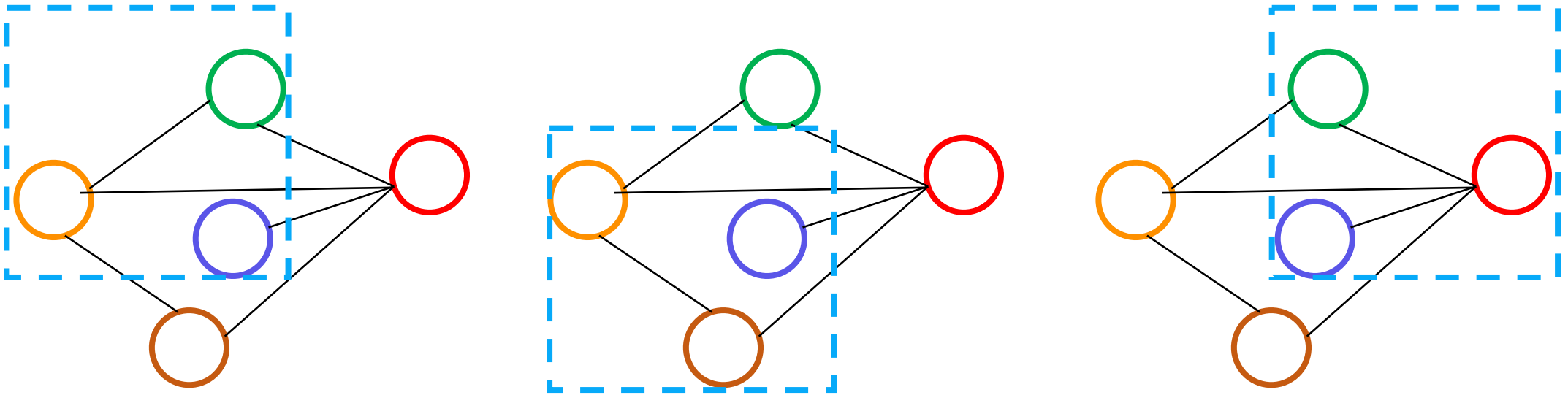
GCN: Task-free Node Embedding



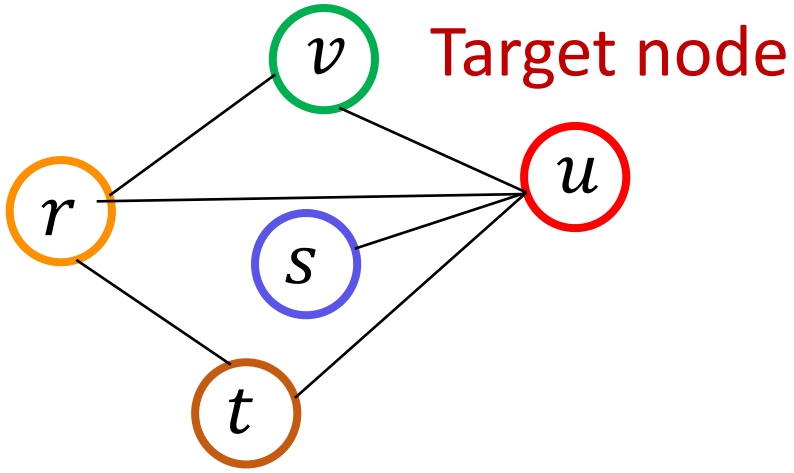
GCN encodes nodes so that similarity in the embedding space approximates similarity in the original graph.

GCN: Convolutions on Graphs

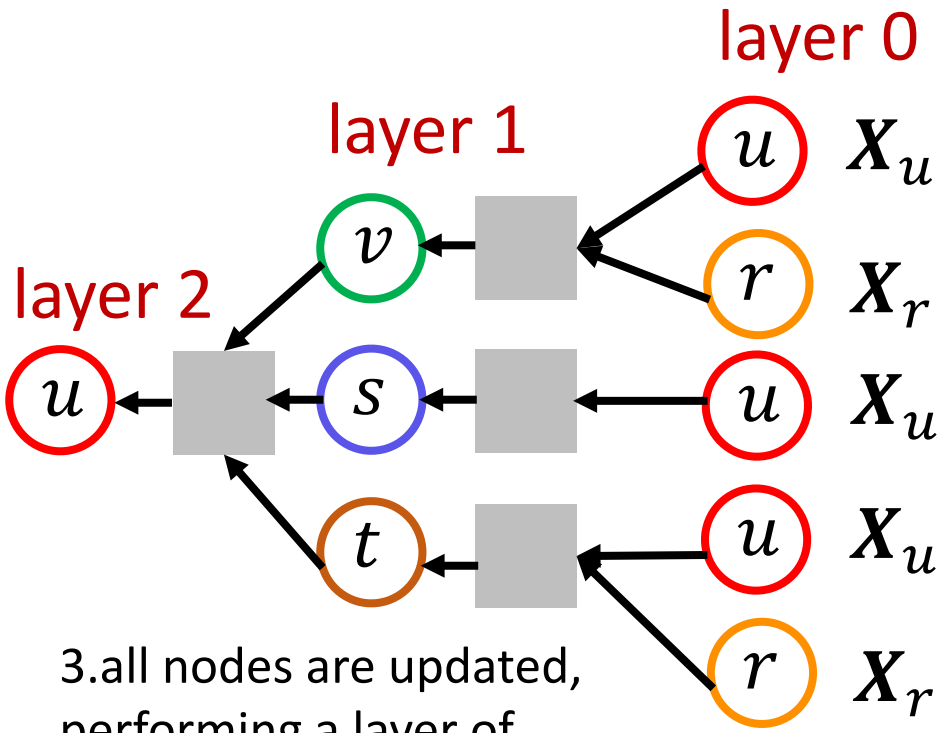
- Challenge: local neighborhood connectivity patterns are different for irregular graphs.
- A spectral view : graph convolutions as pointwise multiplication of the spectra of signals in Fourier-domain.



GCN: Neighborhood Aggregation



- The deeper the network, the larger the local neighborhood
- Propagate information through neighborhoods so that global information is disseminated to each graph node.



1. Each node is initially attached with a feature vector

3.all nodes are updated, performing a layer of forward propagation

2. For each node, the vectors of the neighbors are summed (with weights and transforms) into it.

GCN: Approximation

- Computing the spectrum of signal over general graphs is computationally expensive.
- approximating the convolution kernels makes computation for efficient.
- GCN (Kipf and Weiling 2016) used first order approximation. In the Fourier domain, this restricts convolutions to kernels whose spectrum is an affine function of eigenvalues.

GCN (Kipf and Welling, 2016)

- Input
  - Node features $\boldsymbol{X}$
  - Adjacency matrix $\boldsymbol{A}$ that represents graph structure
- Output: Node embedding $\boldsymbol{Z}$
- Method

$$\boldsymbol{H}^{(l+1)} = f(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$$

where $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$, $\boldsymbol{D}$ is a diagonal matrix such that $D_{ii} = \sum_j \tilde{A}_{ij}$, $\boldsymbol{W}^{(l)}$ is layer-specific parameter matrix, $\boldsymbol{H}^{(l)}$ is node representation of lth layer.

Efficient GCN Learning via Sampling

Challenges of Learning with GCN: requires full expansion of neighborhood, thus is costly

- GraphSAGE (William Hamilton, Rex Ying, Jure Leskovec, NIPS' 17)
- FastGCN (Jie Chen, Tengfei Ma, Cao Xiao, ICLR' 18)

FastGCN

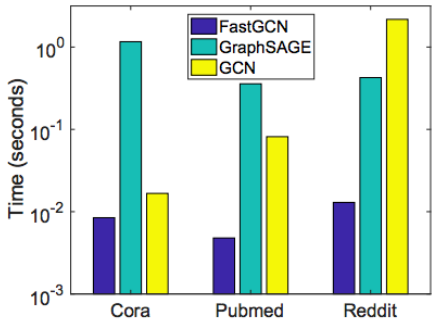
GCN layer propagation rule

$$\boldsymbol{H}^{(l+1)} = f(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$$

Generalize it to integral transformation under a probability measure P.

Instead of fully expansion, sample fixed number of neighbors.

Monte Carlo approximation of the integral under measure P yields a consistent estimator of the loss



Micro F1 Score

	Cora	Pubmed	Reddit
FastGCN	0.850	0.880	0.937
GraphSAGE-GCN	0.829	0.849	0.923
GraphSAGE-mean	0.822	0.888	0.946
GCN (batched)	0.851	0.867	0.930
GCN (original)	0.865	0.875	NA

orders of magnitude faster

theory

Tasks in Healthcare Modeling Supported by GCN

- Node classification
  - Classify functions of proteins in the interactome

  William Hamilton, Rex Ying, Jure Leskovec. *Inductive Representation Learning on Large Graphs*, NIPS' 17
- Link prediction
  - Predict whether drug nodes and adverse reaction nodes are connected.

  Marinka Zitnik , Monica Agrawal, Jure Leskovec. *Modeling Polypharmacy Side Effects with Graph Convolutional Networks*. Bioinformatics 2018
- Network similarity
  - How similar are two nodes or two (sub)networks

  Tengfei Ma, Cao Xiao, Jiayu Zhou, Fei Wang. *Drug Similarity Integration Through Attentive Multi-view Graph Auto-Encoders*. IJCAI' 18