Deep Learning for Natural Language Processing

Hien T. Nguyen
Director, Artificial Intelligence Laboratory
Faculty of Information Technology
Ton Duc Thang University, Vietnam

Email: hien@tdt.edu.vn or hiennguyenthanh@gmail.com

Outline

- Introduction
- Review on basic machine learning models
- Deep Learning
- Distributed Representation
- Deep Learning for NLP
  - Named Entity Recognition and Sequence Labelling tasks
  - Sentence Modeling
  - Sentiment Analysis
  - Question Answering
  - Machine Translation
  - Syntactic Parsing
  - Visual Recognition
  - Speech Synthesis
- Conclusion

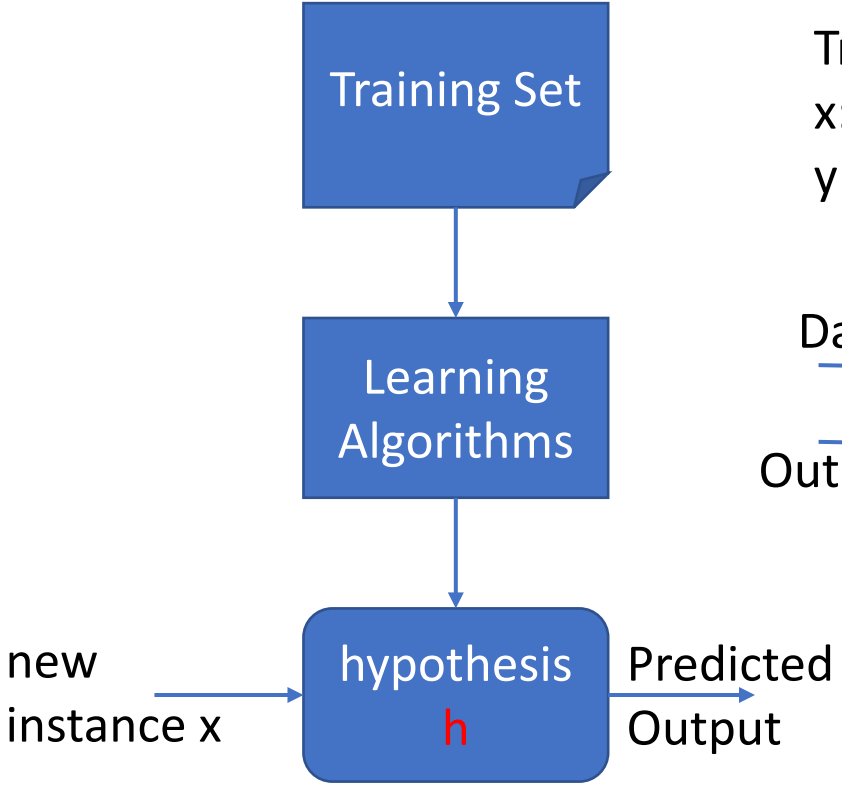
Review

- Learning models
- Logistic regression and Classification
- Perceptron
- Softmax regression
- Basic Neural Networks

Learning models

- Supervised learning
- Unsupervised learning
- Reinforcement learning

Supervised Learning



Training set: $\{(x^{(i)}, y^{(i)}); i=1,2,\ldots, m\}$

x: a vector of features

y: output

- Classification
- Regression

Supervised Learning

Living area (feet^2)	#bedrooms	Price (1000$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
⋮	⋮	⋮

Linear Regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x \quad (x_0=1)$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

(loss function)

Learning using Gradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

learning rate

Supervised Learning

- Gradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

  - Batch gradient descent
  - Stochastic gradient descent
  - Minibatch
- Loss function
  - Cross-entropy
  - Square error
  - Max-margin (Hinge loss)

Supervised vs. Unsupervised Learning

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a *function* to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

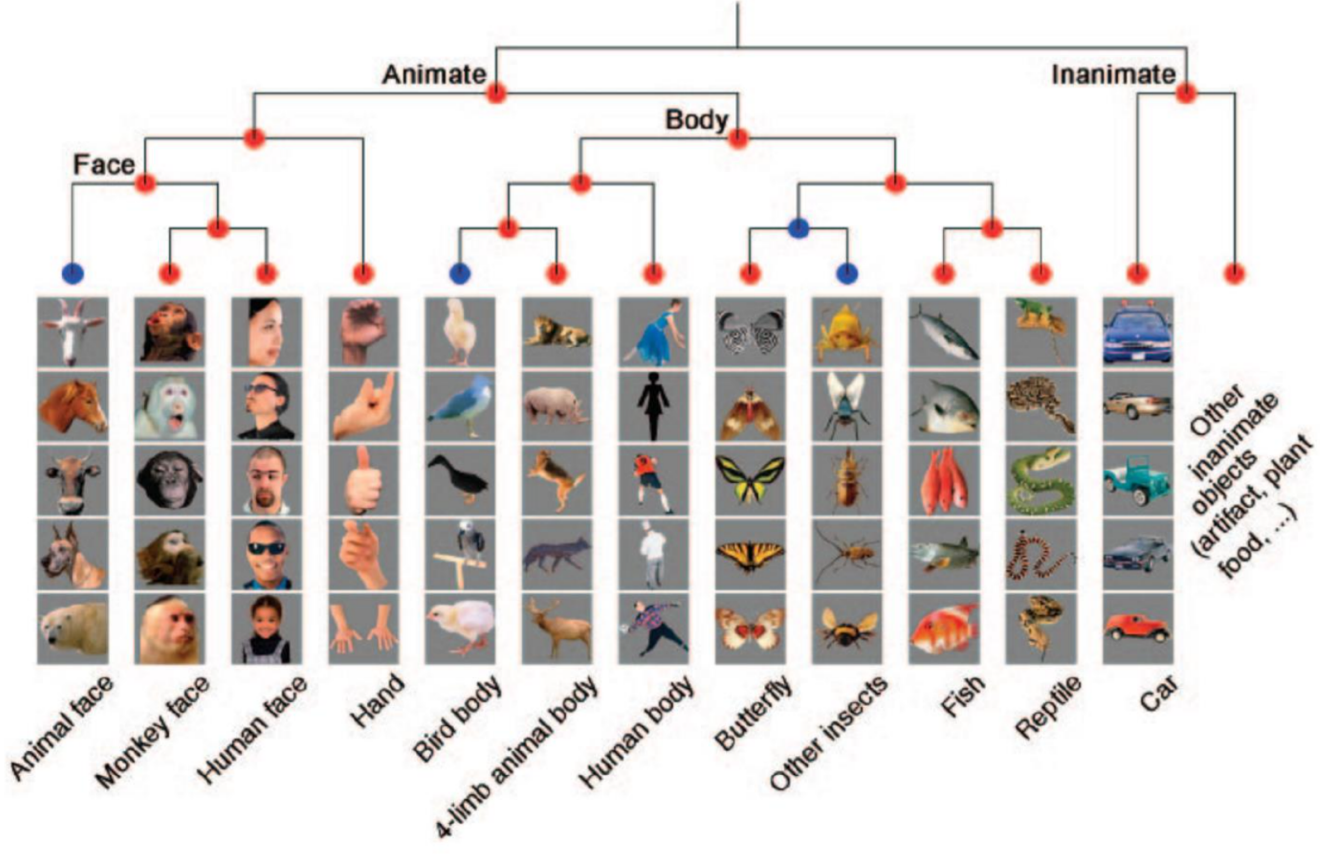
Unsupervised Learning

Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

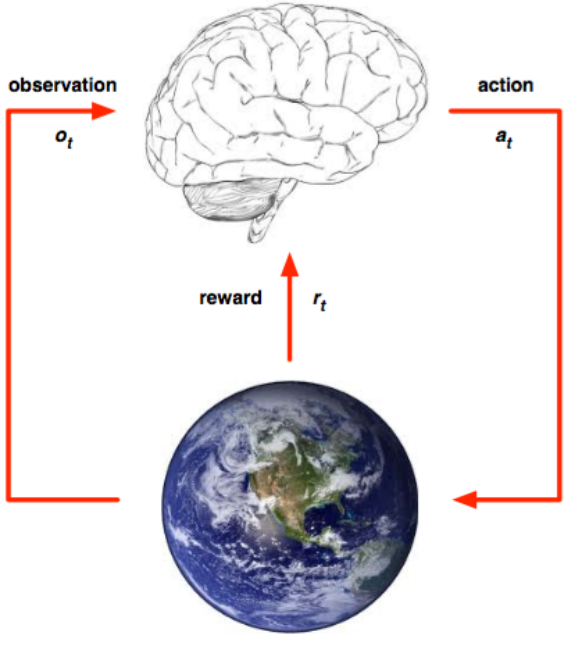
Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Unsupervised Learning



Kiani *et. al.* 2007

Reinforcement Learning



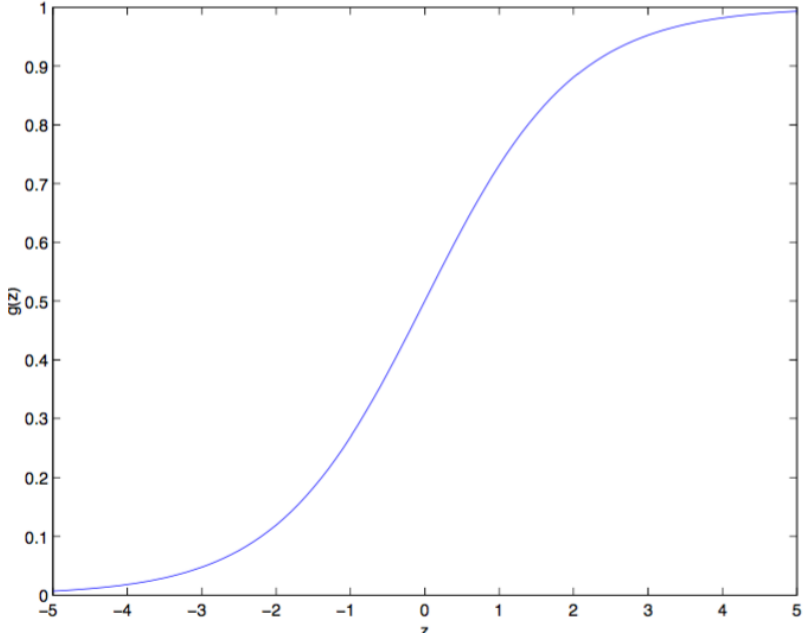
- At each step t the agent:
  - Executes action a_t
  - Receives observation o_t
  - Receives scalar reward r_t
- The environment:
  - Receives action a_t
  - Emits observation o_{t+1}
  - Emits scalar reward r_{t+1}

David Silver's slide

Logistic Regression and Classification

- Logistic Function (sigmoid)

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



Logistic Regression and Classification

- Logistic Function (sigmoid)

$$
\begin{aligned}
g'(z) &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\
&= \frac{1}{(1+e^{-z})^2}\left(e^{-z}\right) \\
&= \frac{1}{(1+e^{-z})}\cdot\left(1-\frac{1}{(1+e^{-z})}\right) \\
&= g(z)(1-g(z)).
\end{aligned}
$$

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

- Our hypotheses $h_\theta(x)$

Logistic Regression and Classification

Let us assume that

$$\begin{aligned} P(y = 1 \mid x; \theta) &= h_\theta(x) \\ P(y = 0 \mid x; \theta) &= 1 - h_\theta(x) \end{aligned}$$

Note that this can be written more compactly as

$$p(y \mid x; \theta) = (h_\theta(x))^y \, (1 - h_\theta(x))^{1-y}$$

Assuming that the m training examples were generated independently, we can then write down the likelihood of the parameters as

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X; \theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1 - h_\theta(x^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

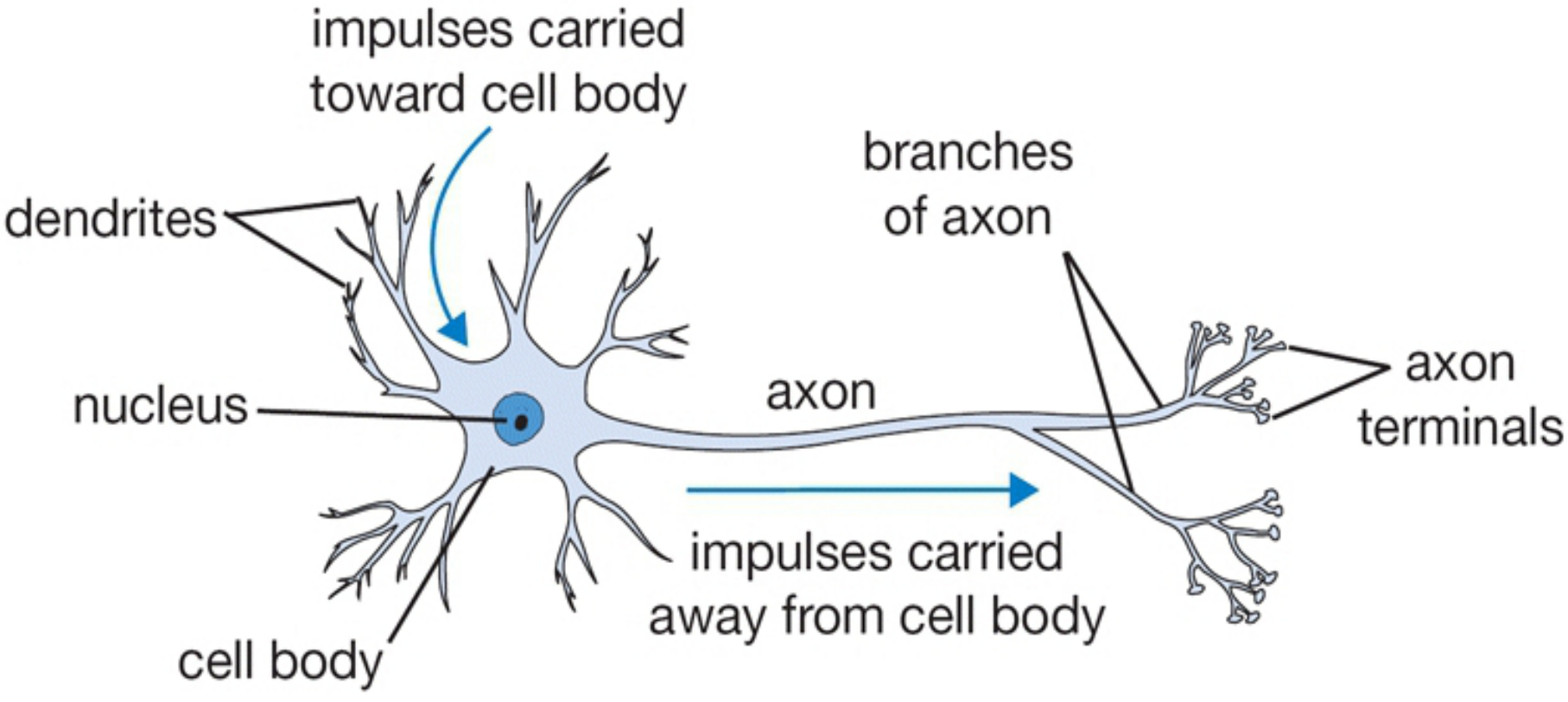
Logistic Regression and Classification

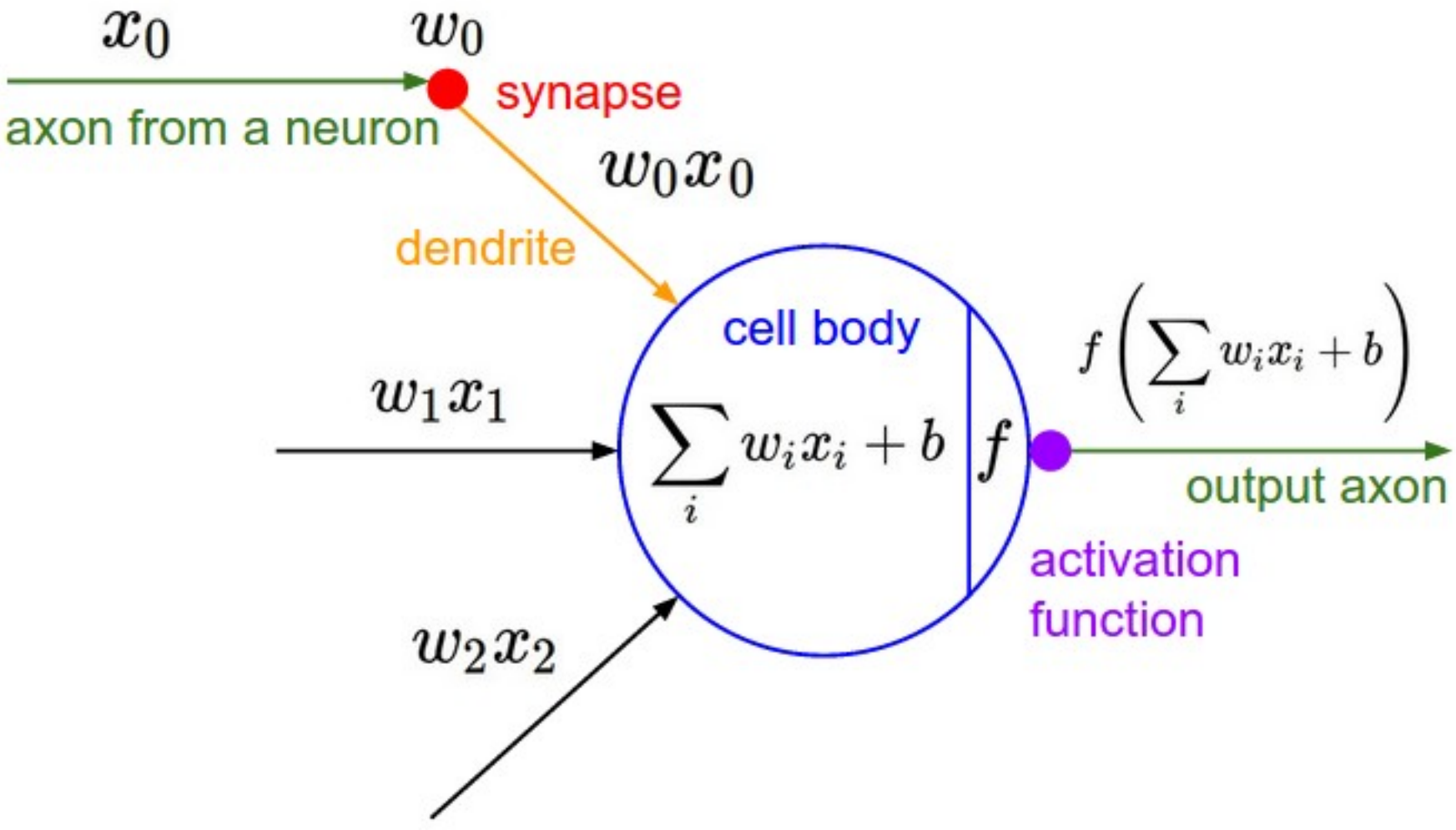
- Log likelihood

$$
\begin{aligned}
\ell(\theta) &= \log L(\theta) \\
&= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))
\end{aligned}
$$

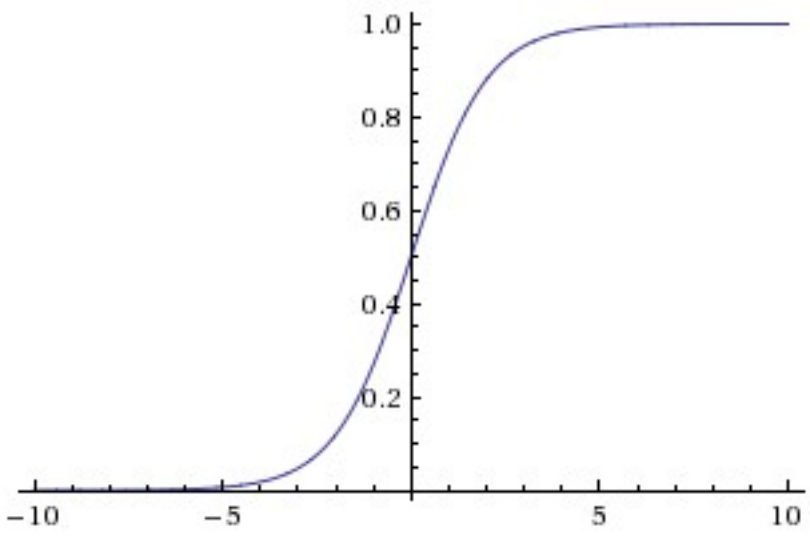
$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} \ell(\theta) &= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) \frac{\partial}{\partial \theta_j} g(\theta^T x) \\
&= \left(y \frac{1}{g(\theta^T x)} - (1 - y) \frac{1}{1 - g(\theta^T x)} \right) g(\theta^T x)(1 - g(\theta^T x) \frac{\partial}{\partial \theta_j} \theta^T x \\
&= \left(y(1 - g(\theta^T x)) - (1 - y) g(\theta^T x) \right) x_j \\
&= (y - h_\theta(x)) \, x_j
\end{aligned}
$$

Perceptron

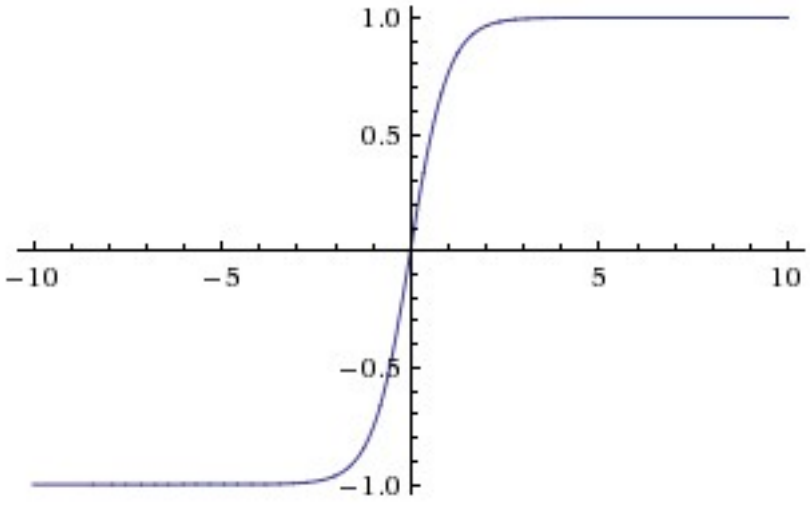

impulses carried
toward cell body
dendrites
branches
of axon
nucleus
axon
axon
terminals
impulses carried
away from cell body
cell body


x_0
w_0
synapse
axon from a neuron
$w_0 x_0$
dendrite
cell body
$w_1 x_1$
$\sum_i w_i x_i + b$
f
$f\left(\sum_i w_i x_i + b\right)$
output axon
activation function
$w_2 x_2$

Activation function

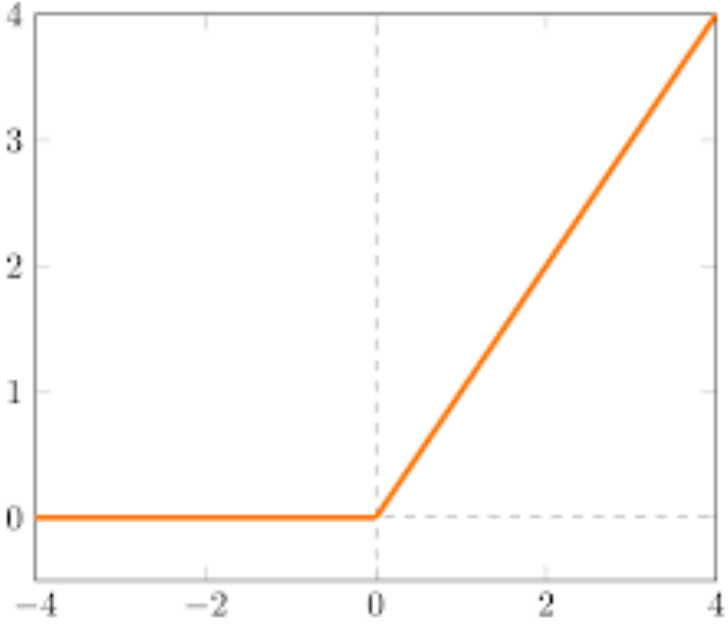


Sigmoid function



Tanh (x) = 2sigmoid(2x)-1

Activation function

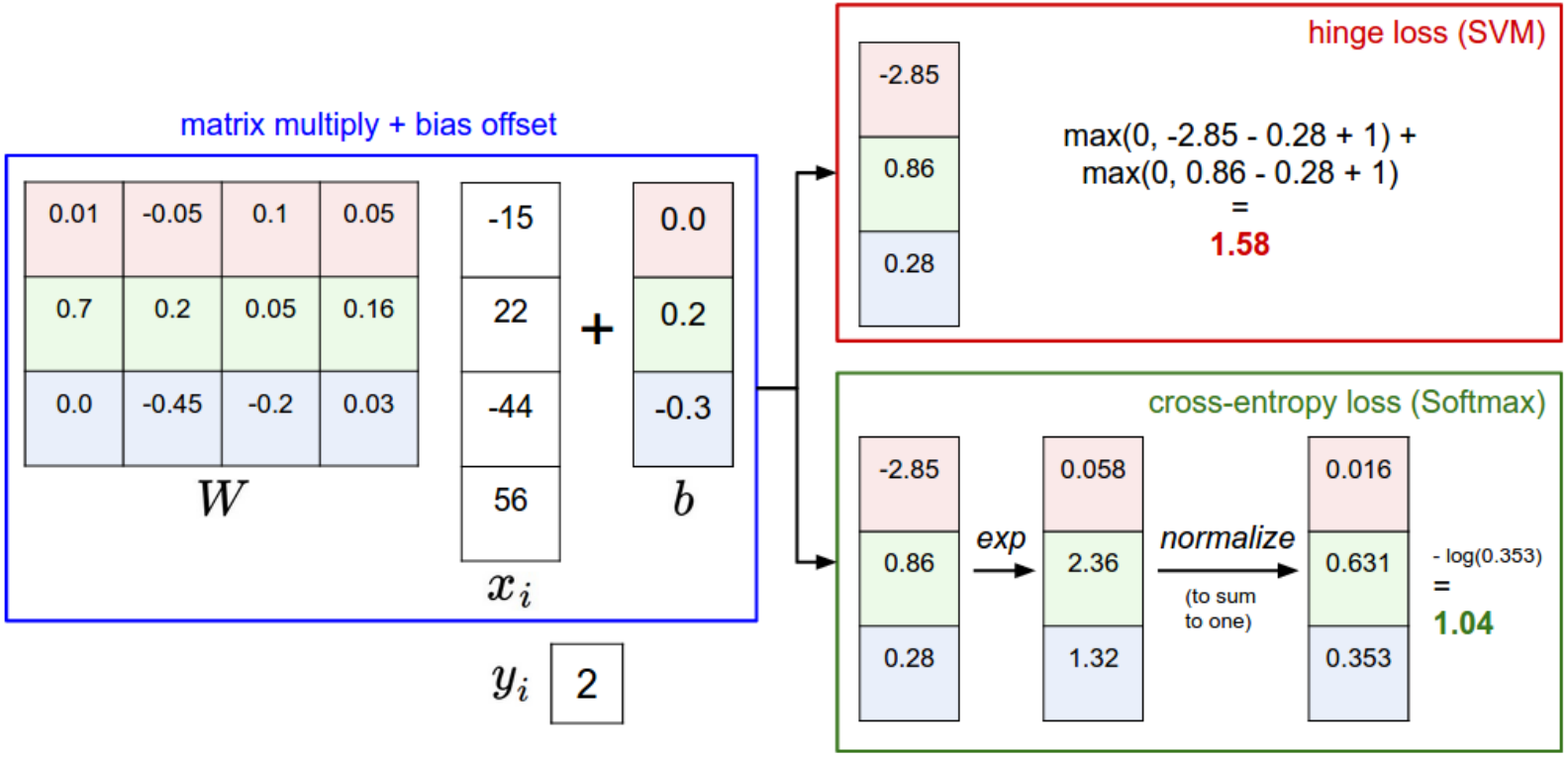


Rectified Linear Unit (ReLu)

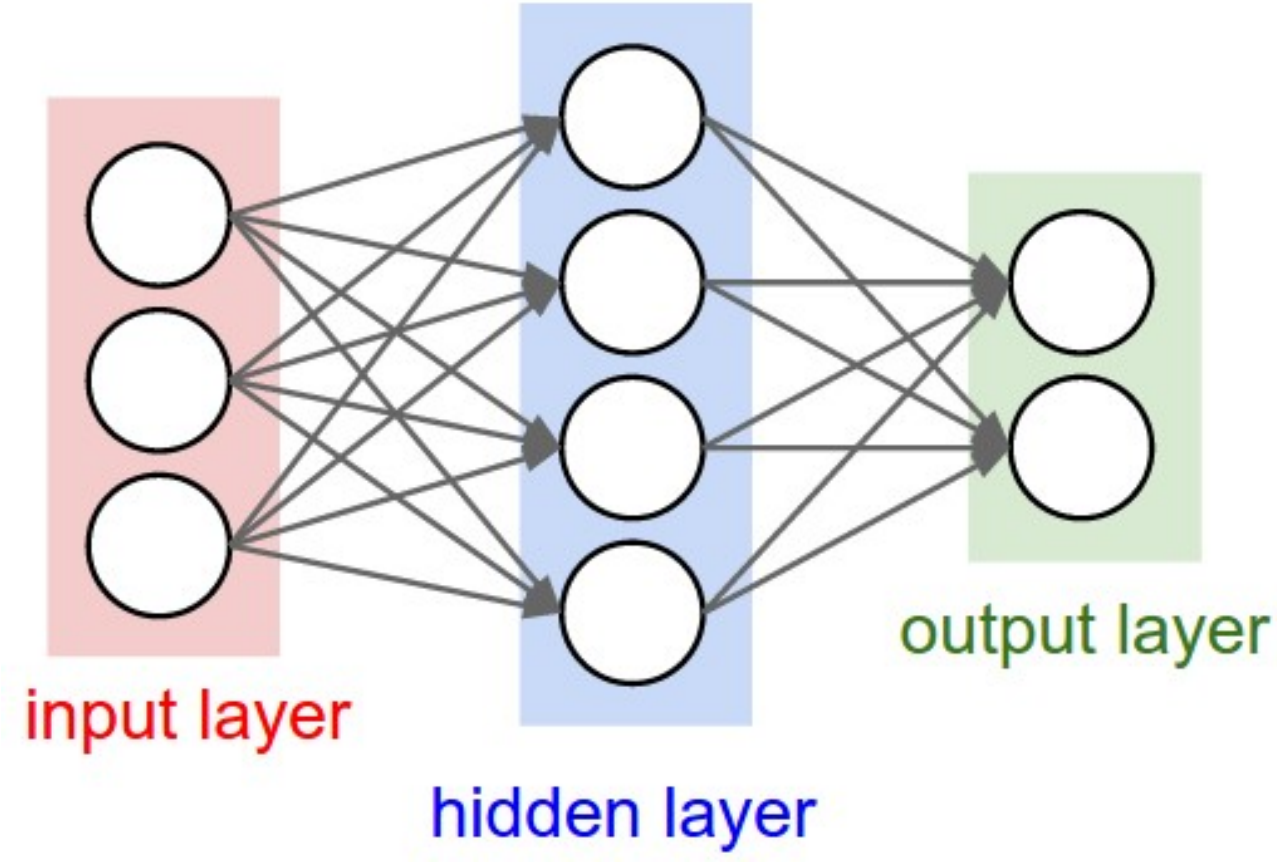
ReLU (x) = max(0,x)

Softmax

$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$



Basic Neural Networks



Derivatives

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

$$f(x+h) = f(x) + h\frac{df(x)}{dx}$$

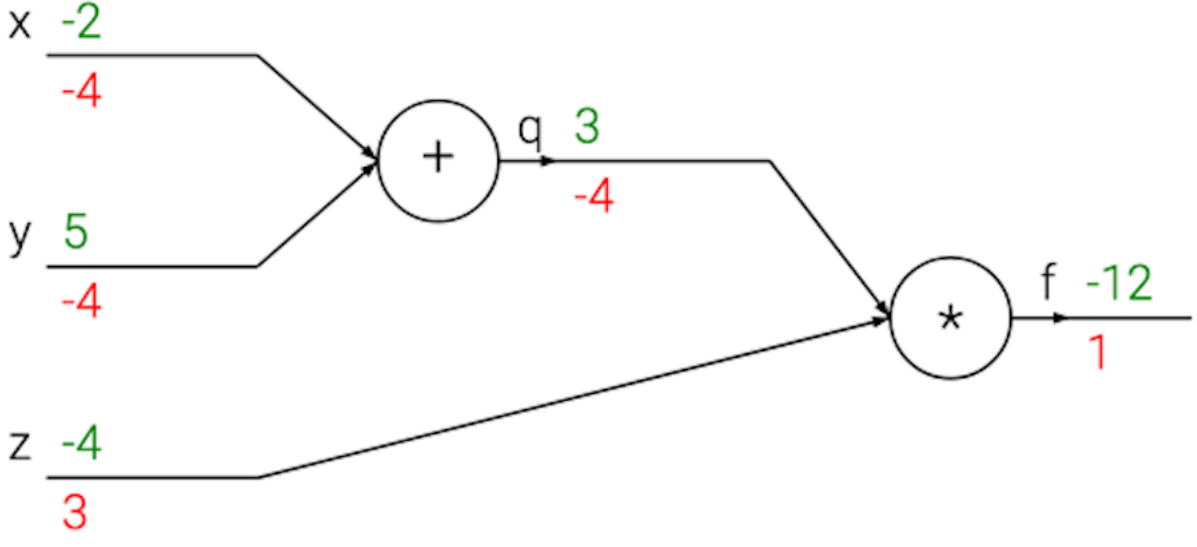
Chain rule

- f (x,y,z) = (x+y)z
- q (x,y) = x + y

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

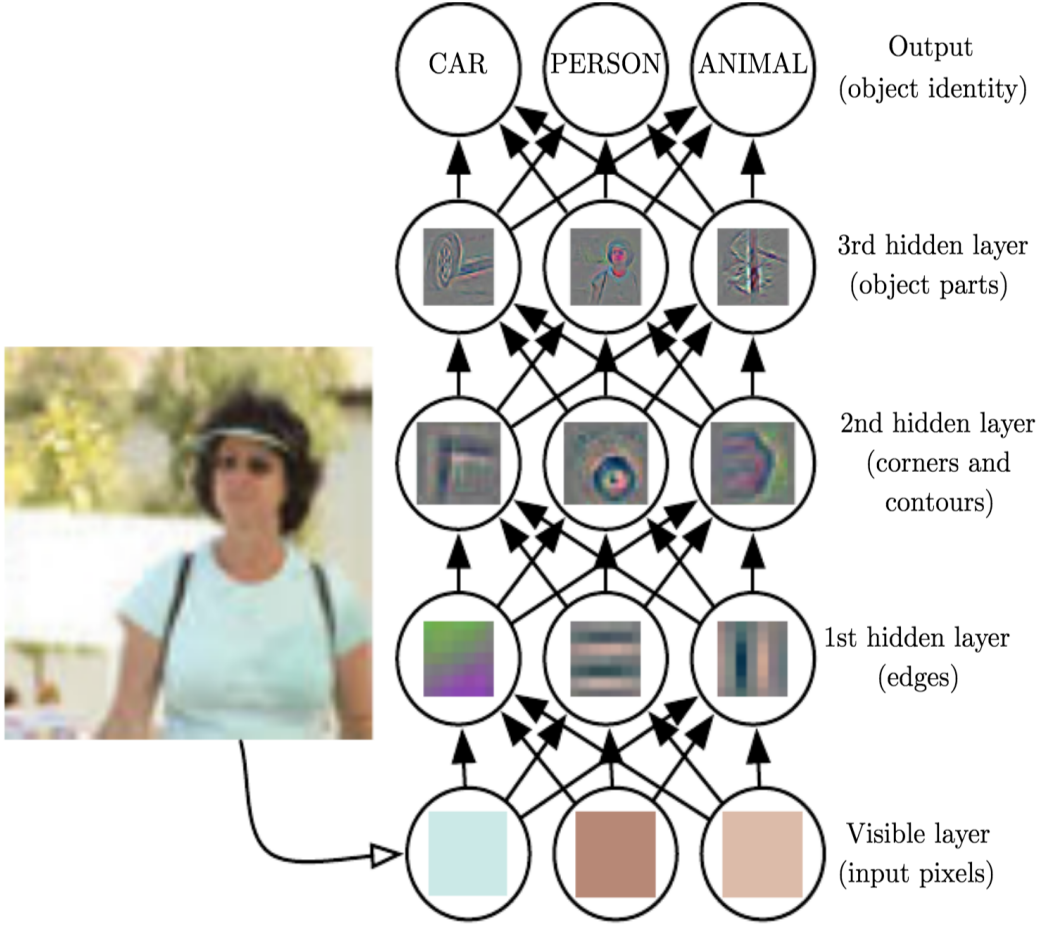
Backpropagation

- Forward pass
- Backward pass

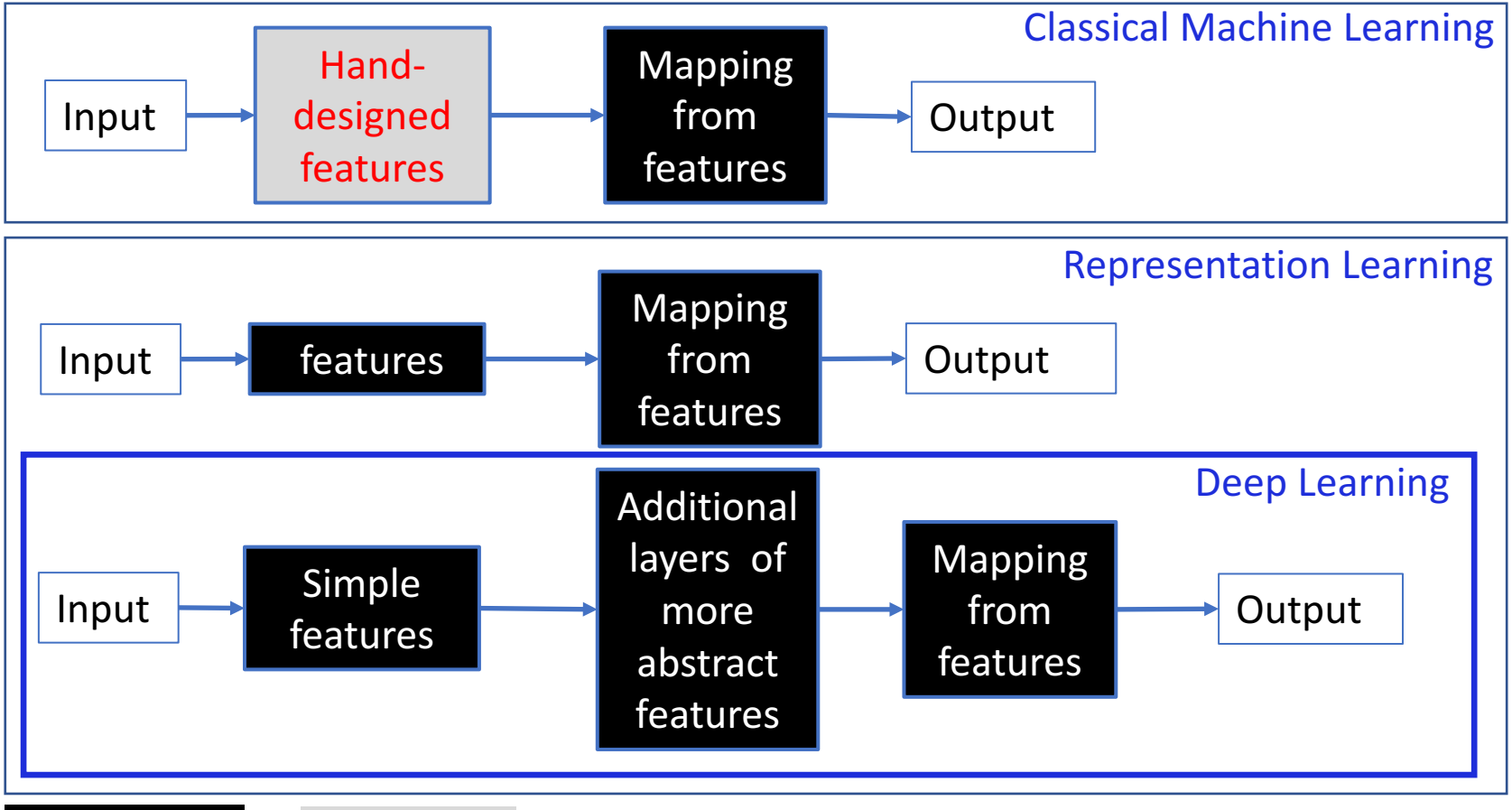


Deep Learning

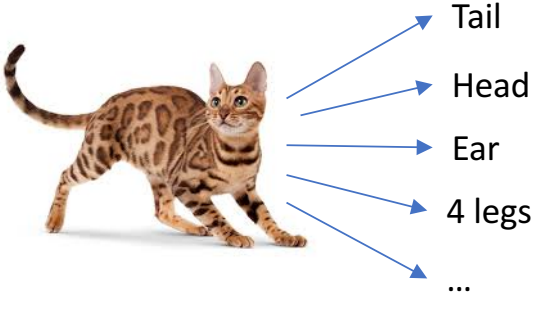
- Automatically learn features or representations at multiple levels of abstraction
- Use computational models composing of multiple processing layers

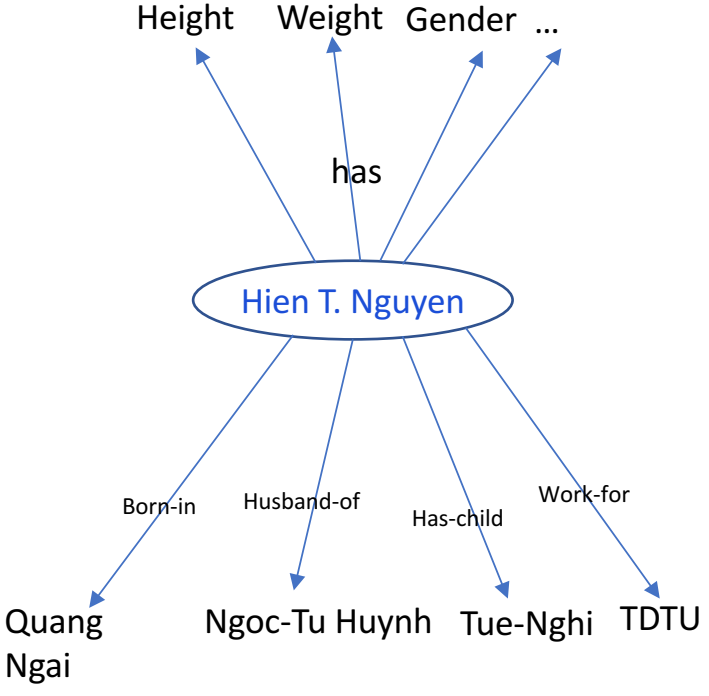


Deep Learning vs. Classis ML



Representation Learning



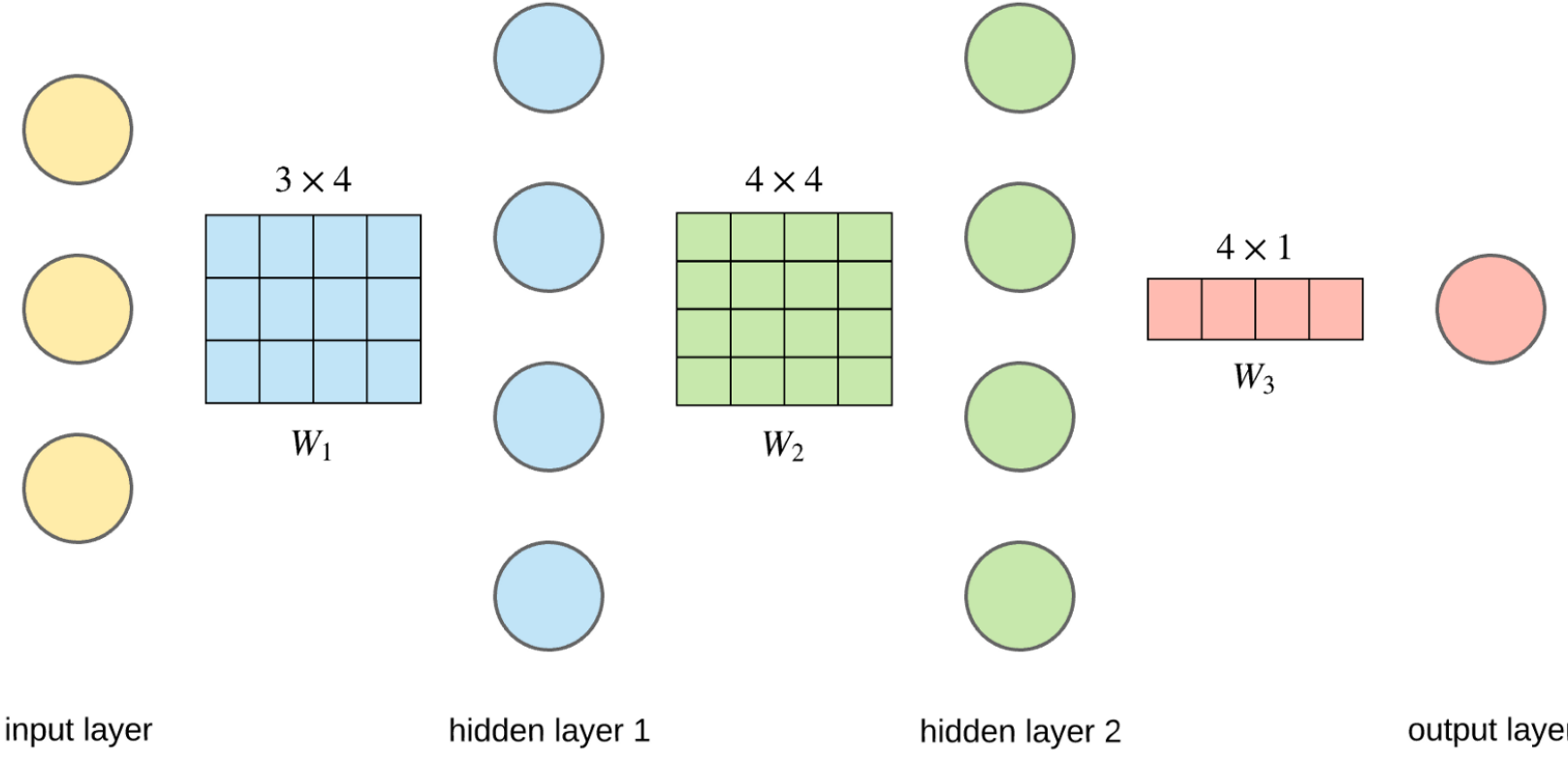


Triple: Subject – relation – Object

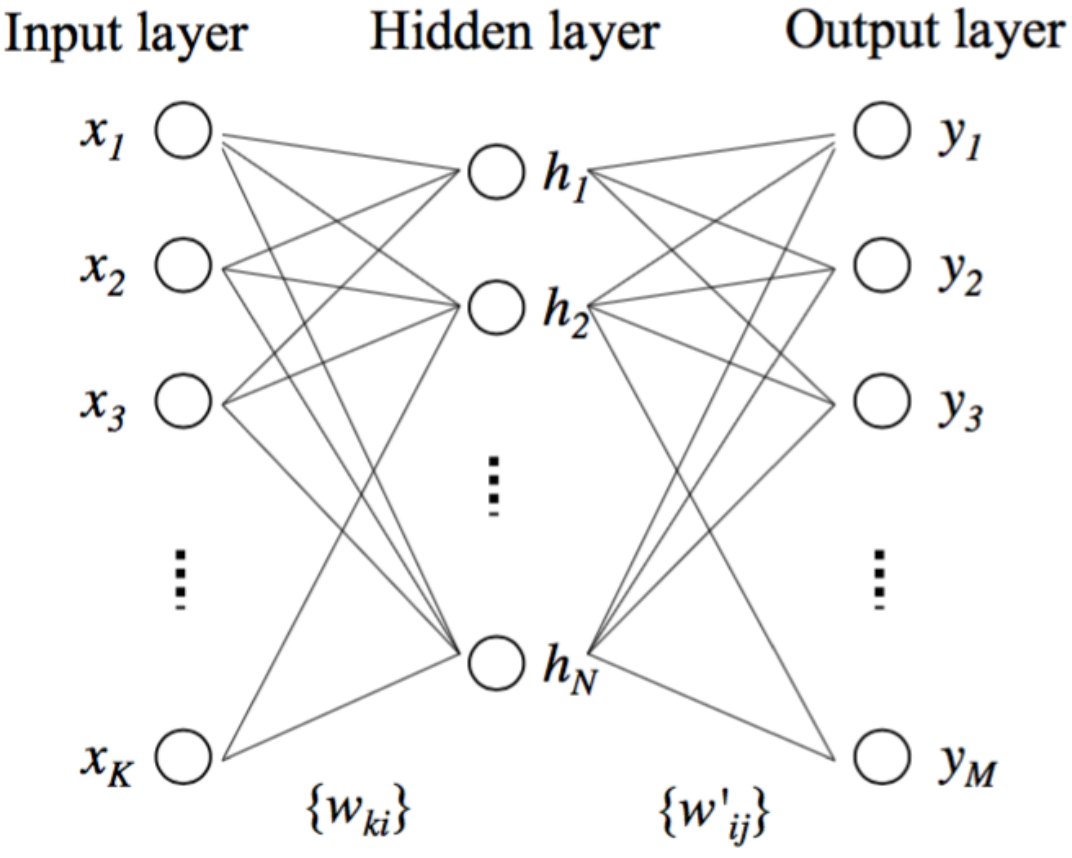
Deep Neural Networks

- Feed-Forward Neural Networks (FFNN)
- Convolutional Neural Networks (CNN)
- Recurrent Neural Networks (RNN)
  - Long-Short Term Memory (LSTM)
  - Gated Recurrent Unit (GRU)

Multi-Layer Networks



Multi-Layer Networks



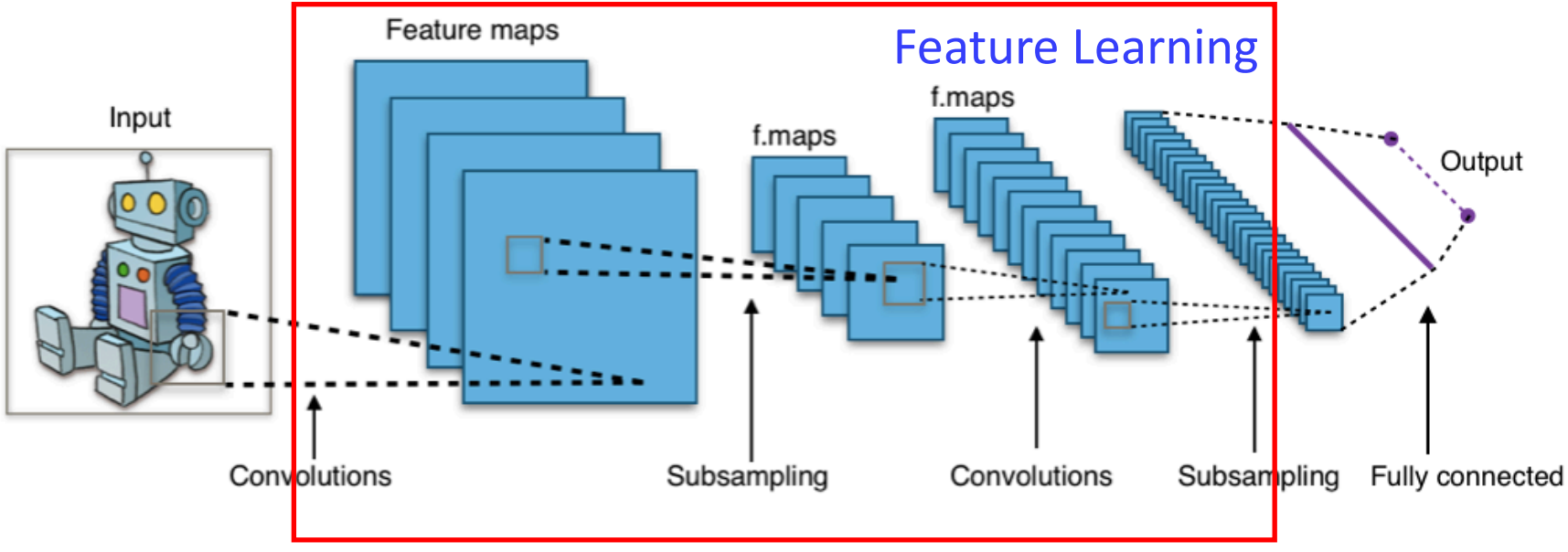
Loss Function

$$E(\mathbf{x}, \mathbf{t}, \mathbf{W}, \mathbf{W}') = \frac{1}{2}\sum_{j=1}^{M}(y_j - t_j)^2$$

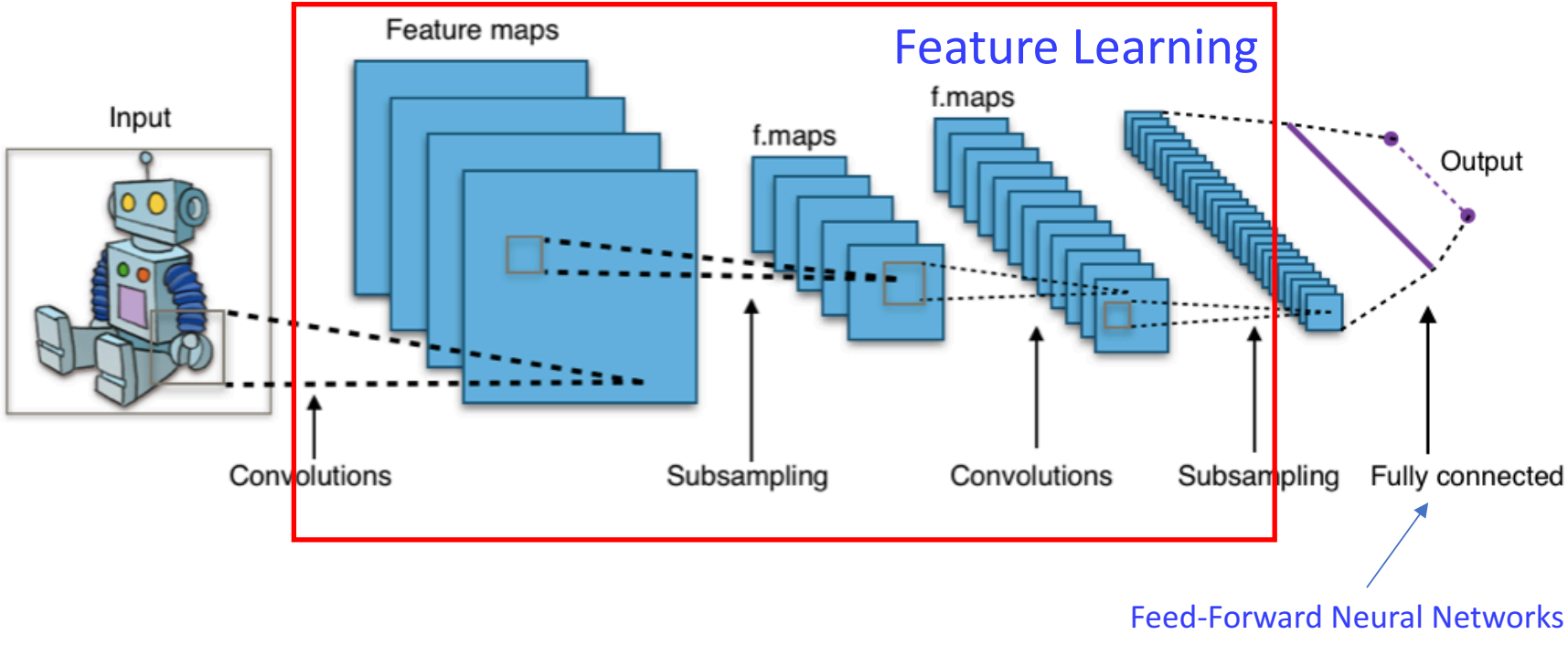
$$h_i = \sigma(u_i) = \sigma\left(\sum_{k=1}^{K} w_{ki}x_k\right)$$

$$y_j = \sigma(u'_j) = \sigma\left(\sum_{i=1}^{N} w'_{ij}h_i\right)$$

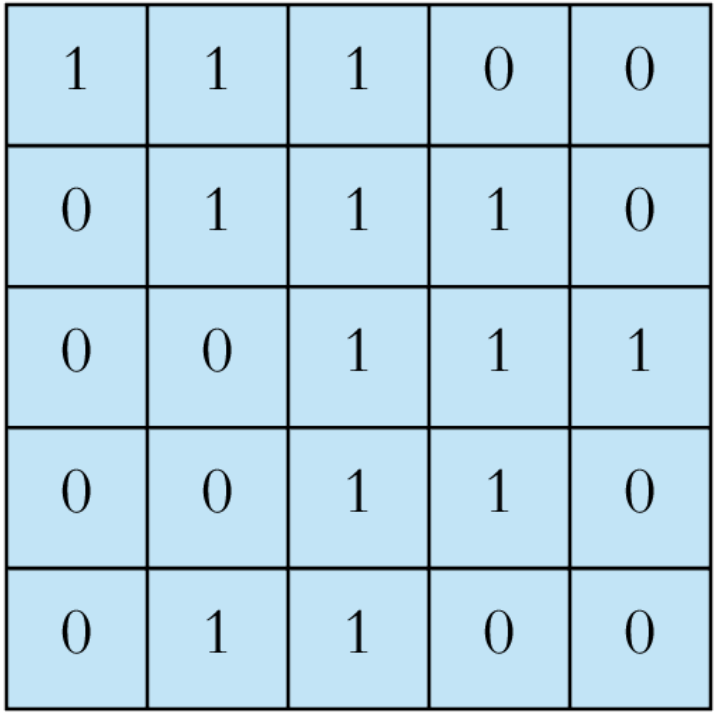
Convolutional Neural Networks



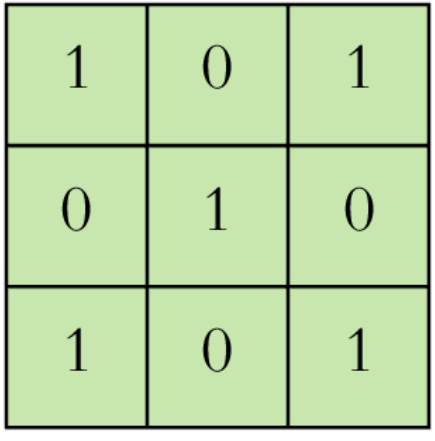
Convolutional Neural Networks



Convolutional Neural Networks

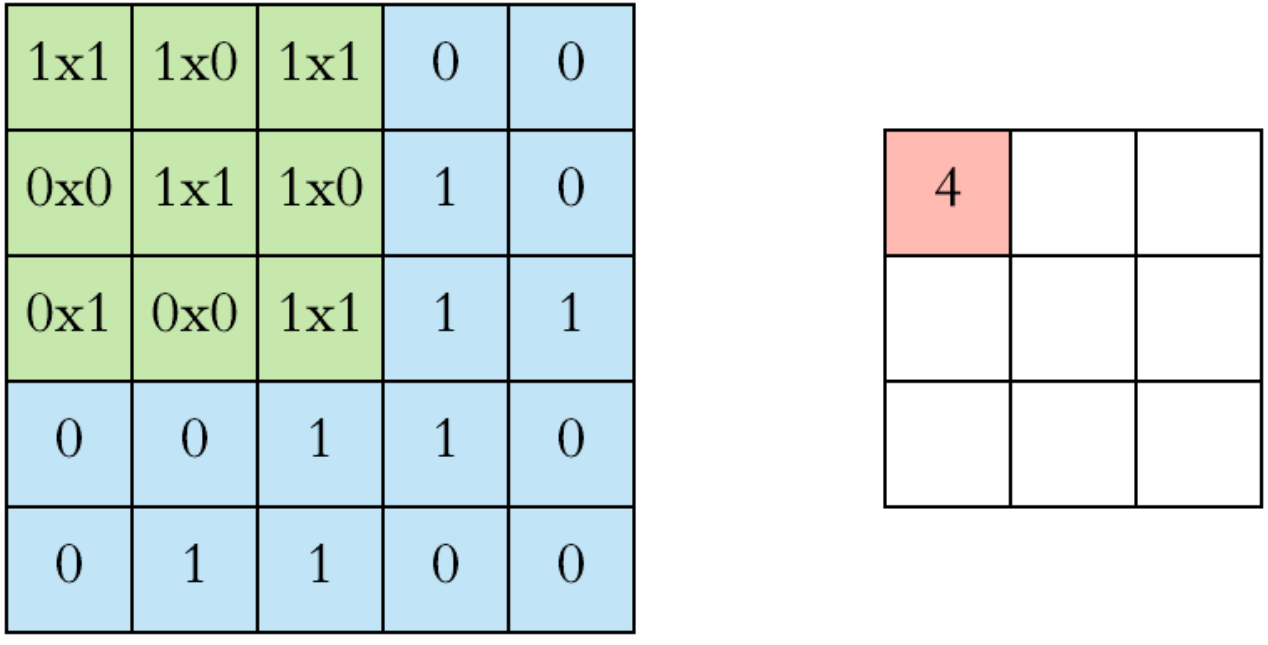


Input



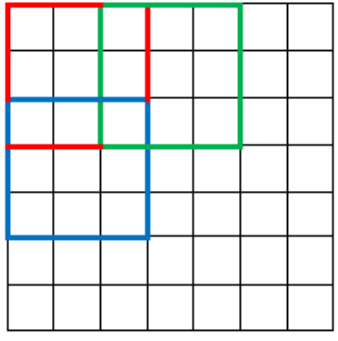
Filter / Kernel

Convolutional Neural Networks

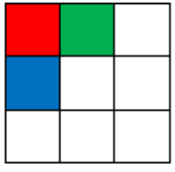


Convolutional Neural Networks

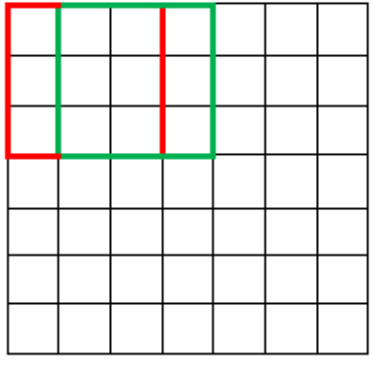
7 x 7 Input Volume



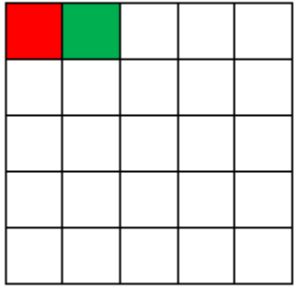
3 x 3 Output Volume



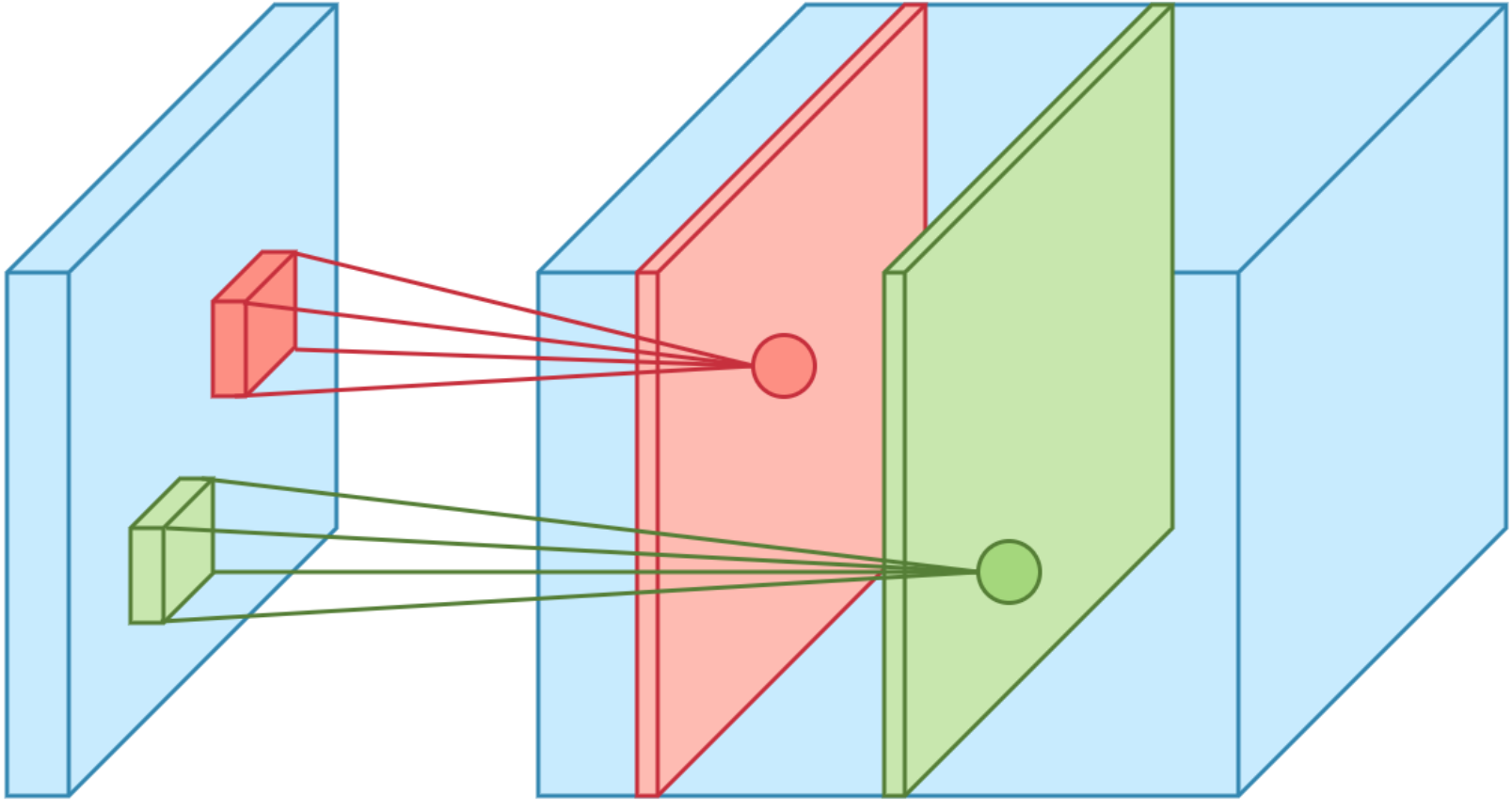
7 x 7 Input Volume



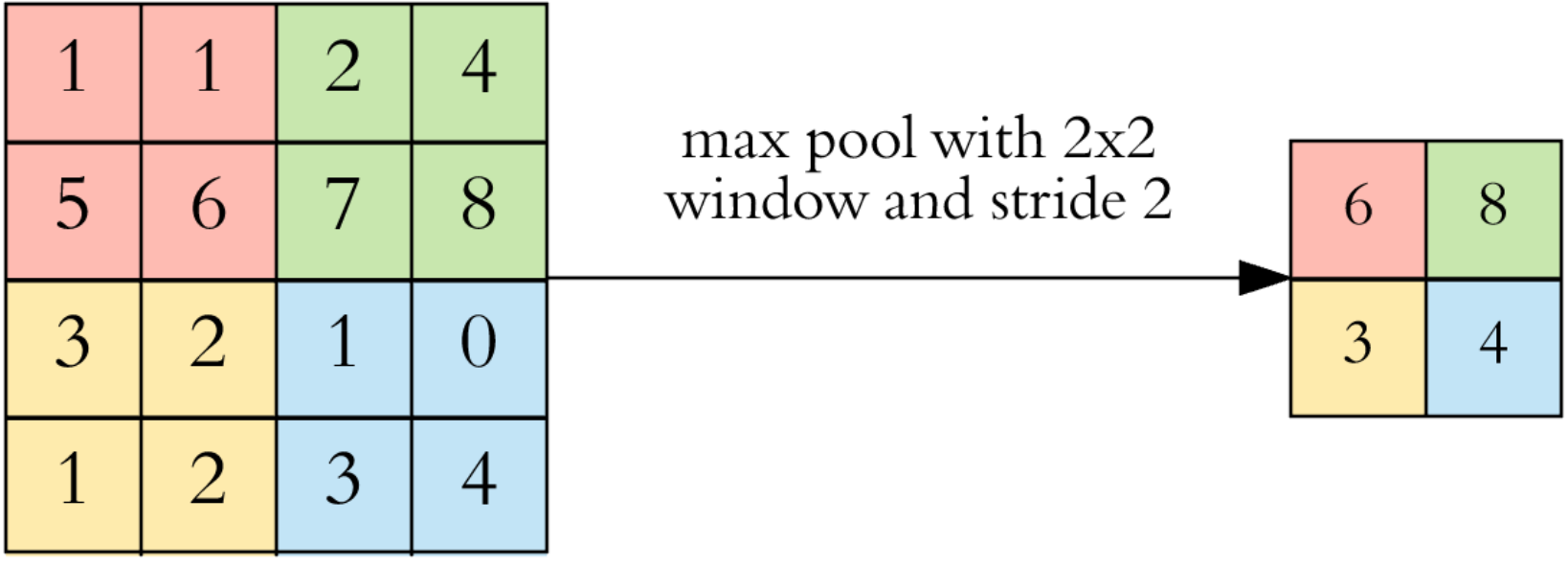
5 x 5 Output Volume



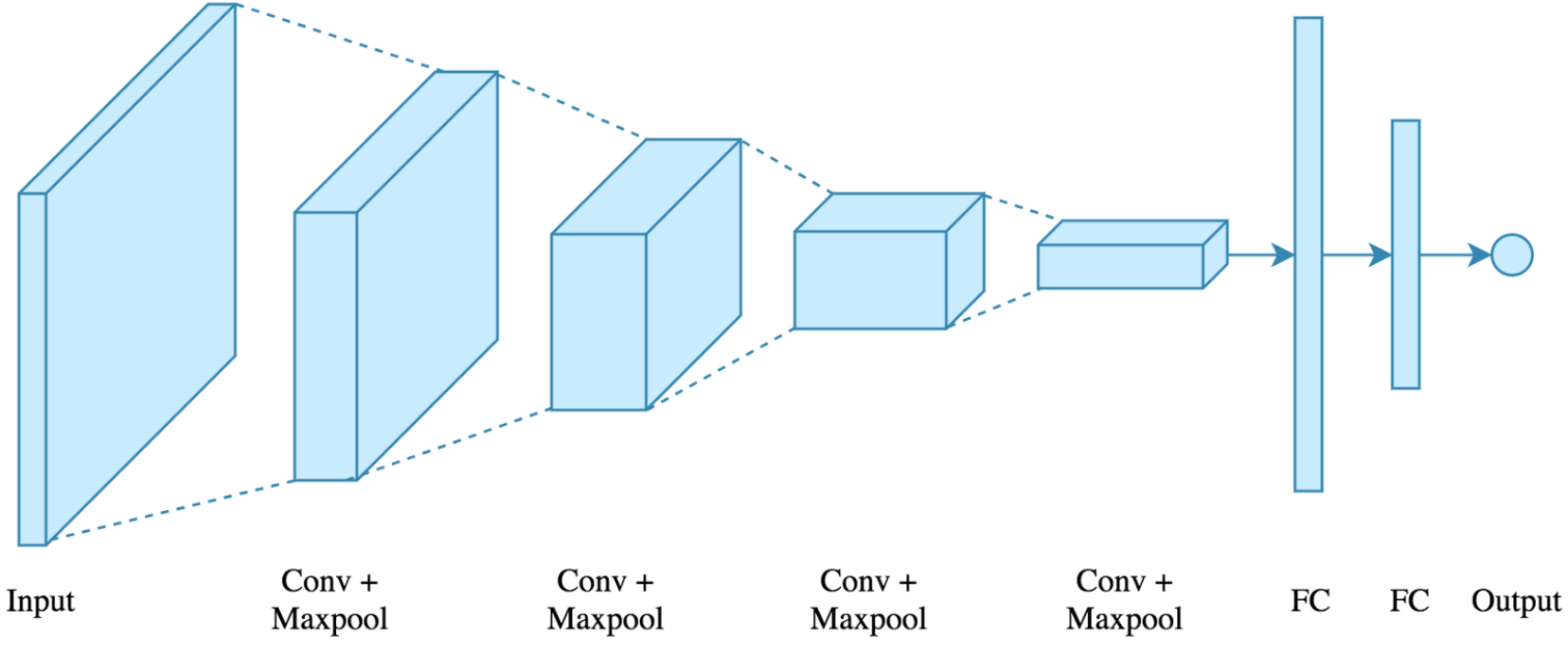
Convolutional Neural Networks



Convolutional Neural Networks

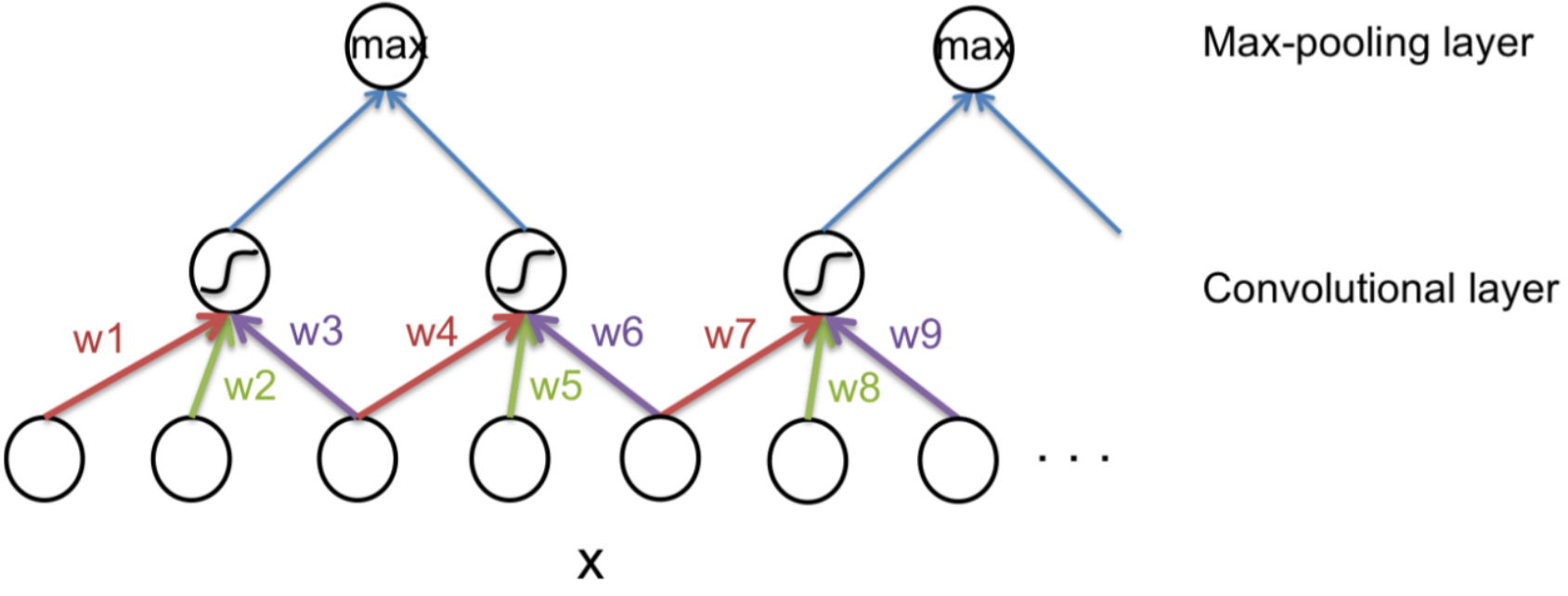


Convolutional Neural Networks



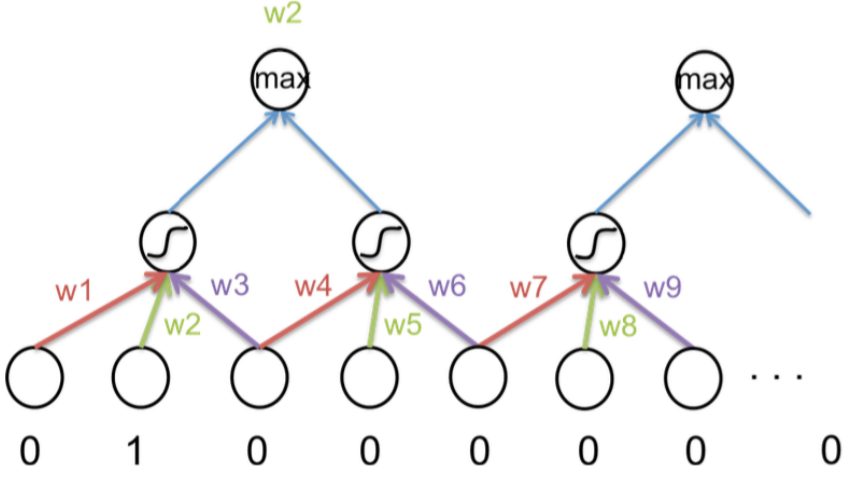
Convolutional Neural Networks

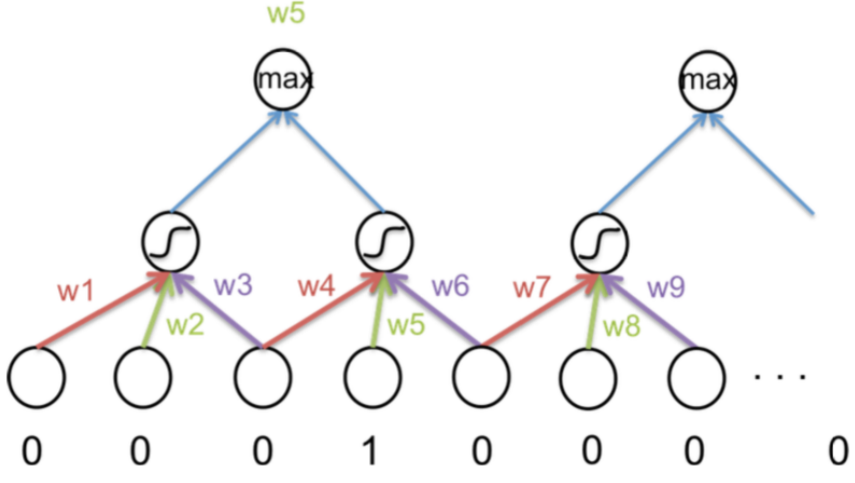
- Weight Sharing



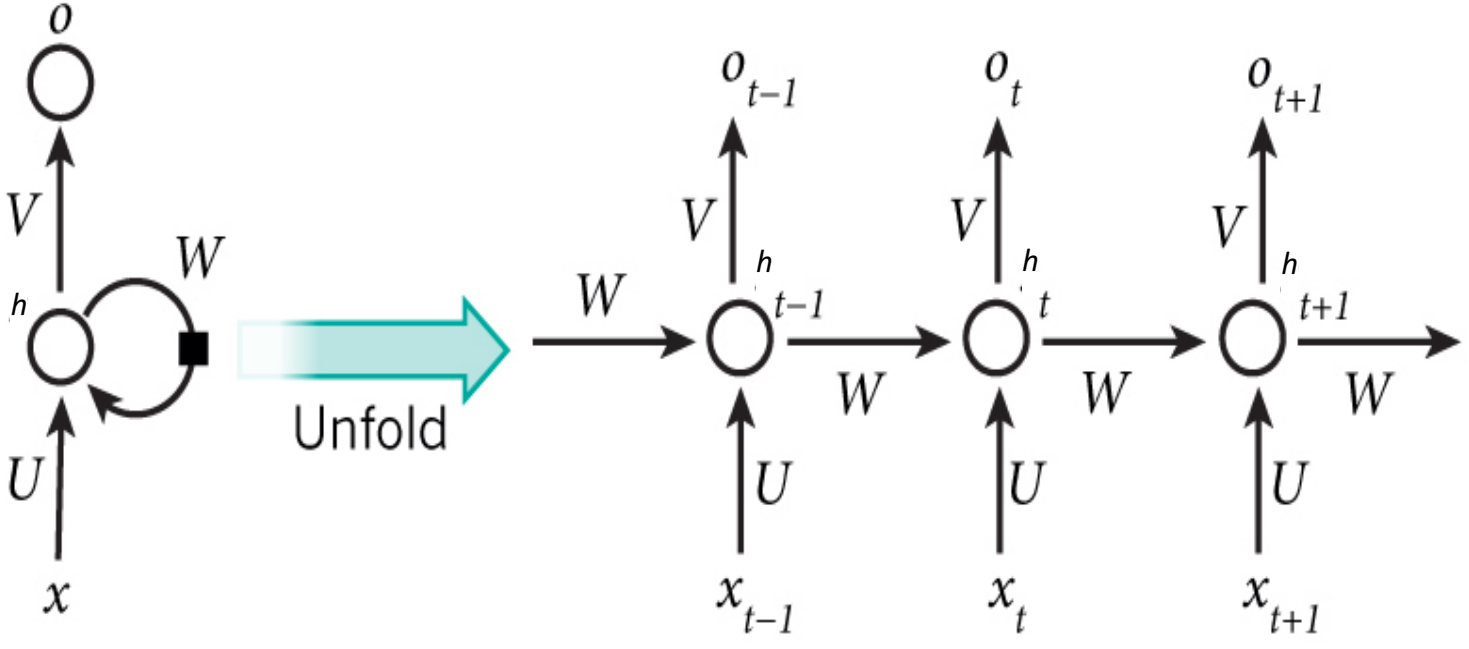
Convolutional Neural Networks

- The max-pooling neurons are invariant to shifts in the inputs

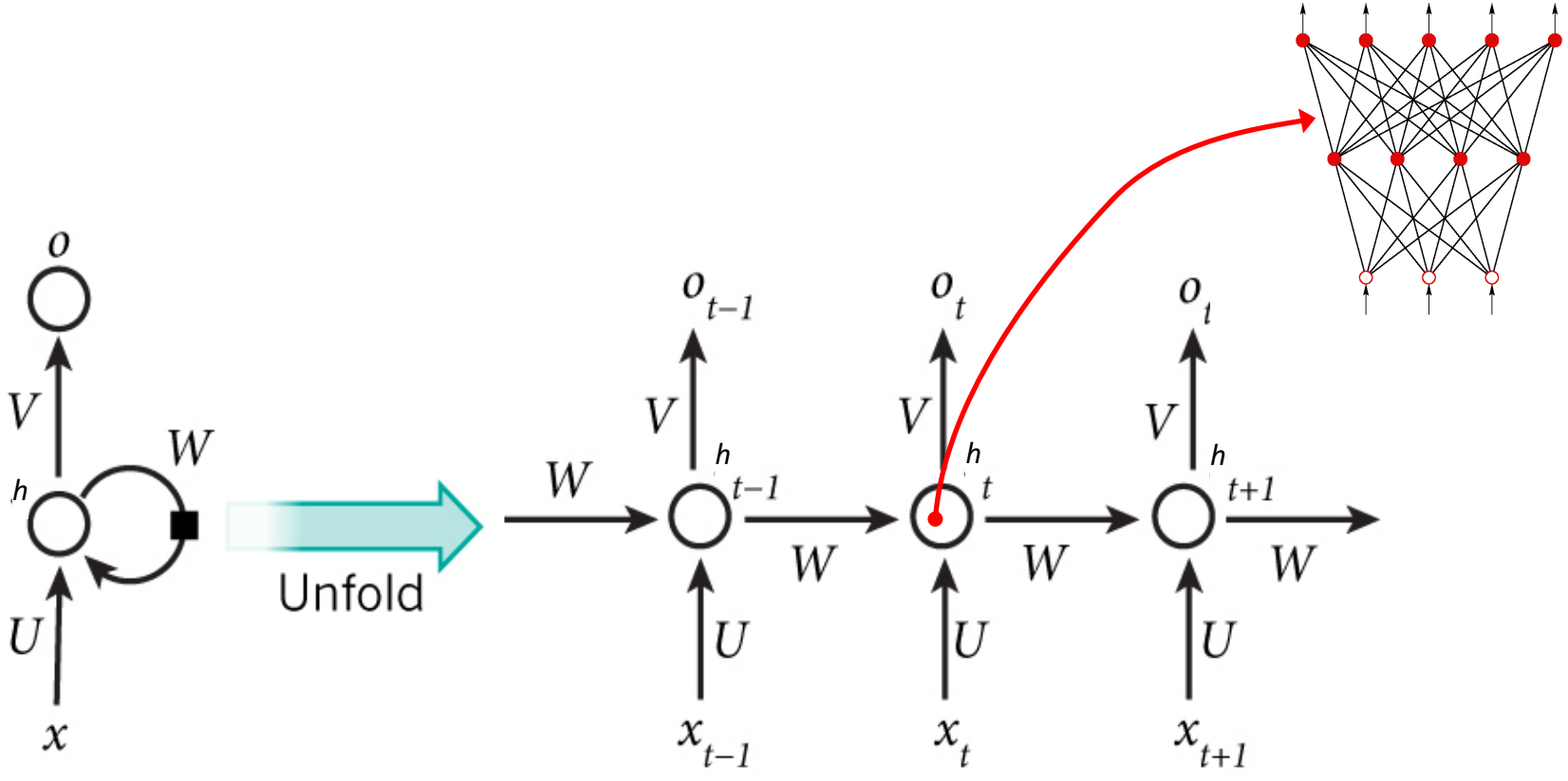




Recurrent Neural Networks

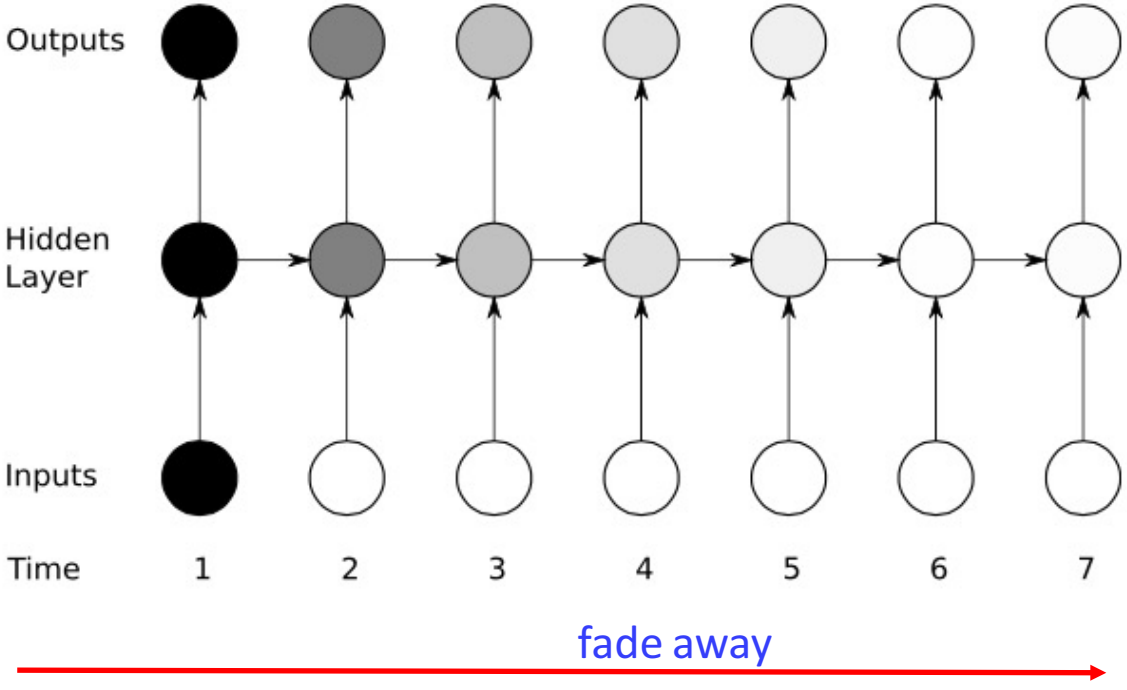


Recurrent Neural Networks



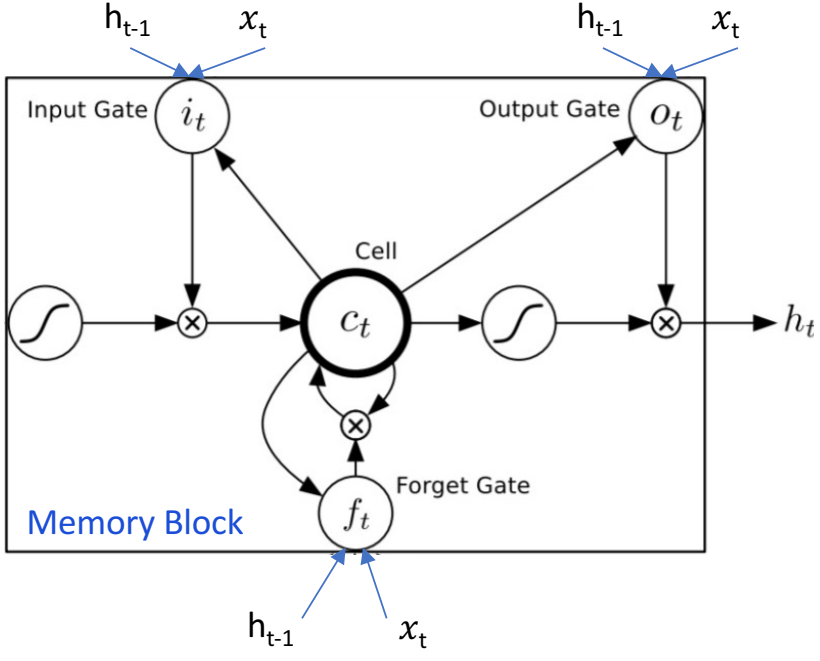
Recurrent Neural Networks

- Limitations of RNN: Vanishing and Exploding



Gated Recurrent Neural Networks

- Long-Short Term Memory (LSTM)



$$f_t = \sigma(W_f[x_t, h_{t-1}] + b_f)$$
$$i_t = \sigma(W_i[x_t, h_{t-1}] + b_i)$$
$$\tilde{c}_t = tanh(W_c[x_t, h_{t-1}] + b_c)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$
$$o_t = \sigma(W_o[x_t, h_{t-1}] + b_o)$$
$$h_i = o_i \odot tanh(c_t)$$

- Gated Recurrent Unit (GRU): a special case of LSTM

Residual Network

- Building block

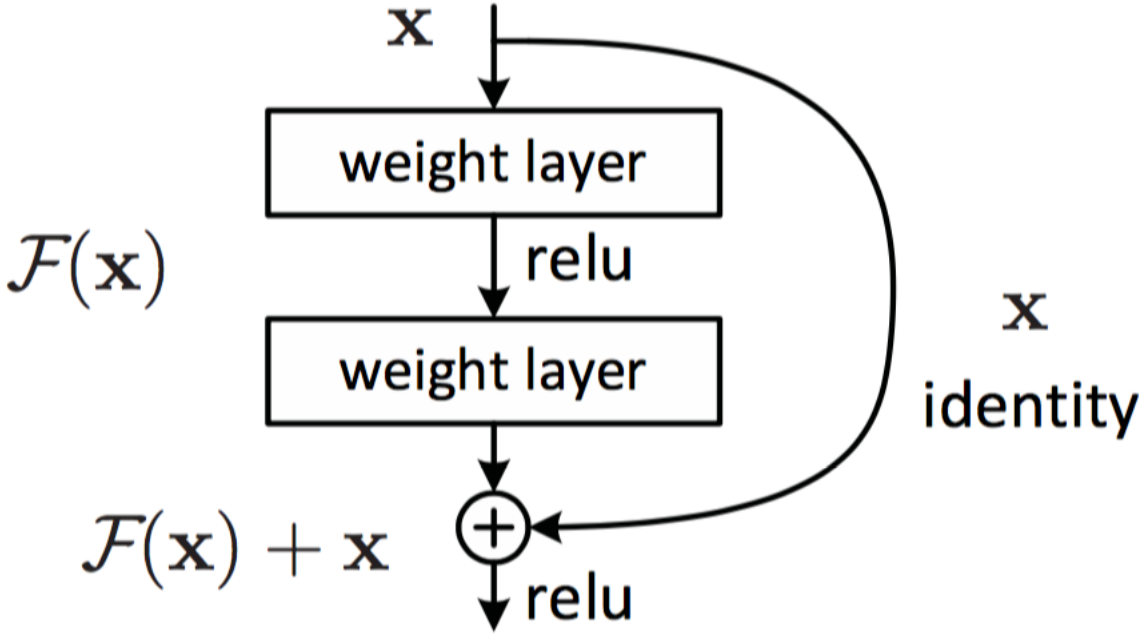


Image taken from He et. al. 2015

Residual Network

- Building block

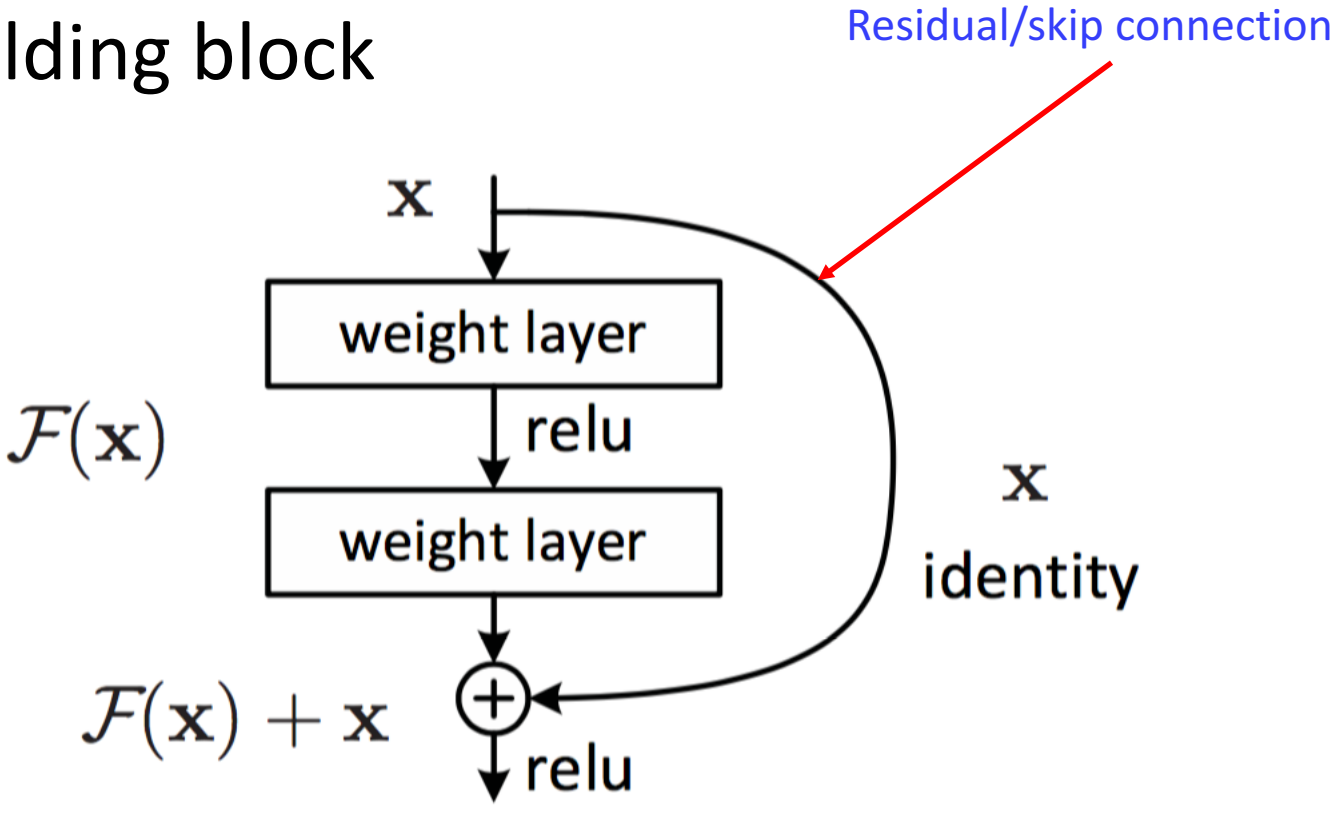


Image taken from He et. al. 2015

Residual Network

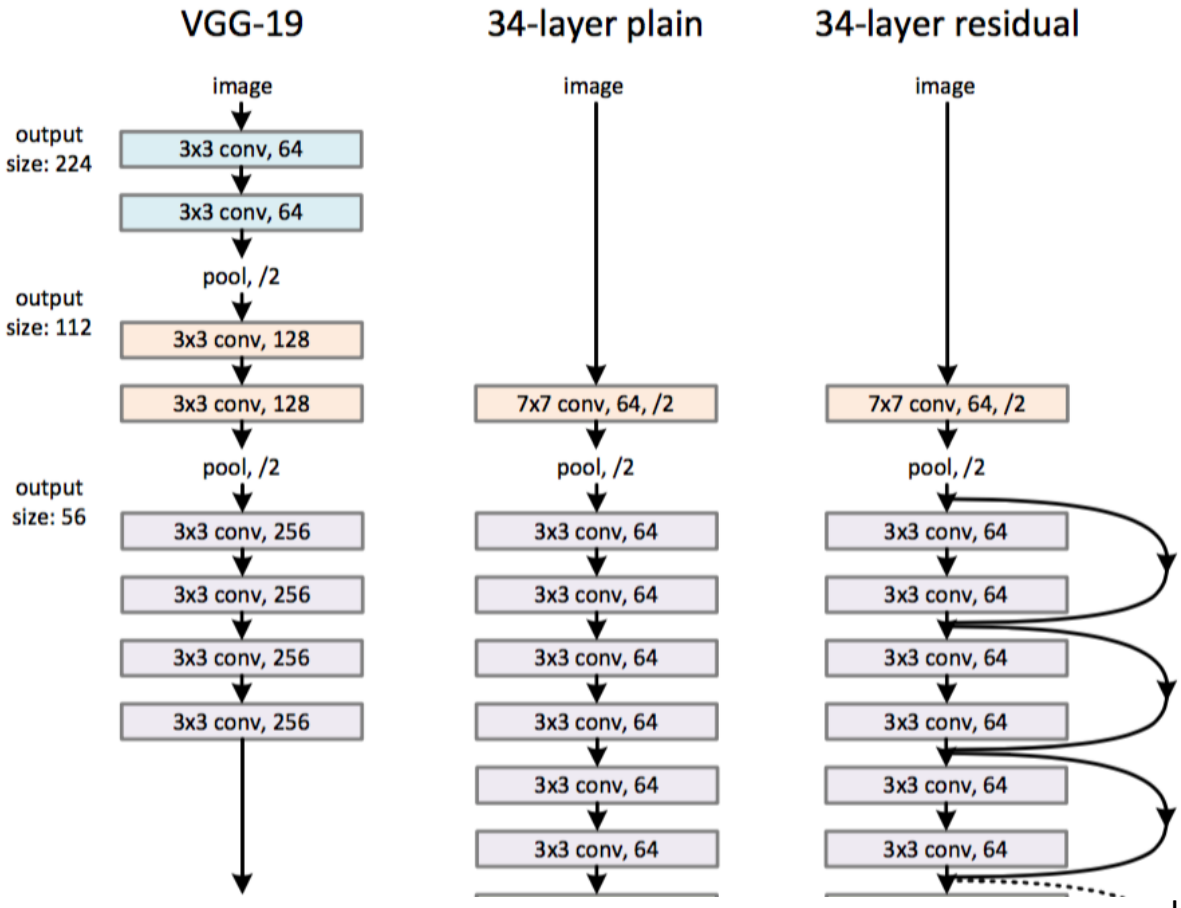
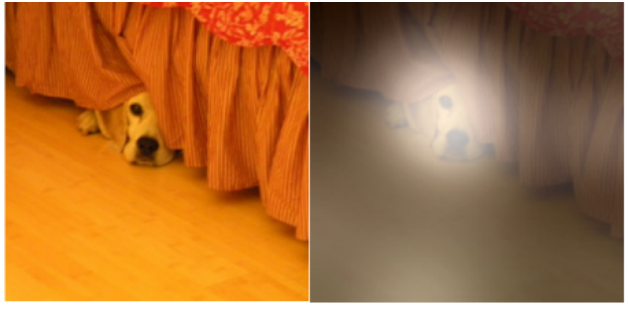


Image taken from He et. al. 2015

Attention Mechanism

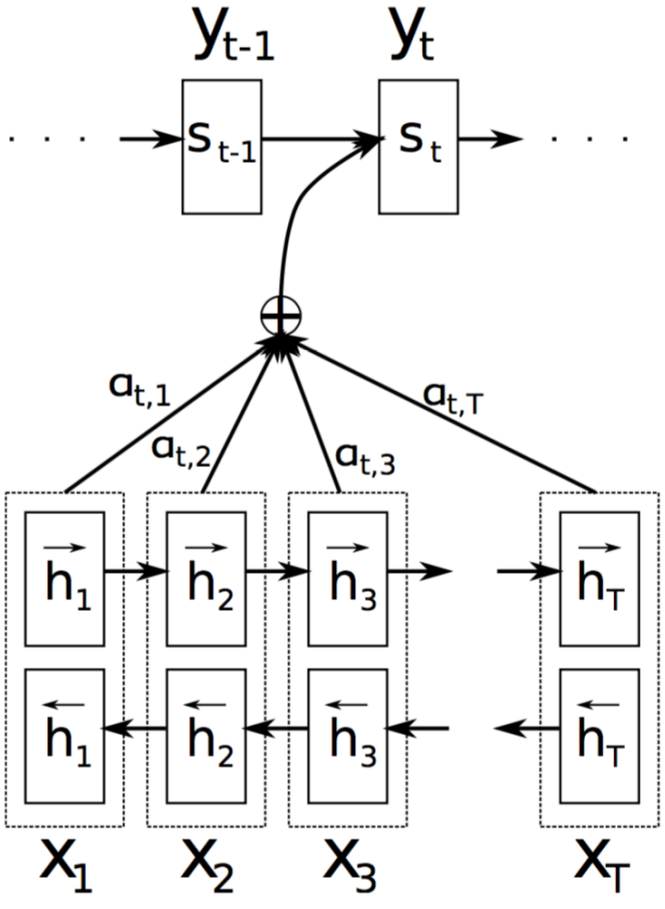
A woman is throwing a frisbee in a park.

A stop sign is on a road with a mountain in the background.

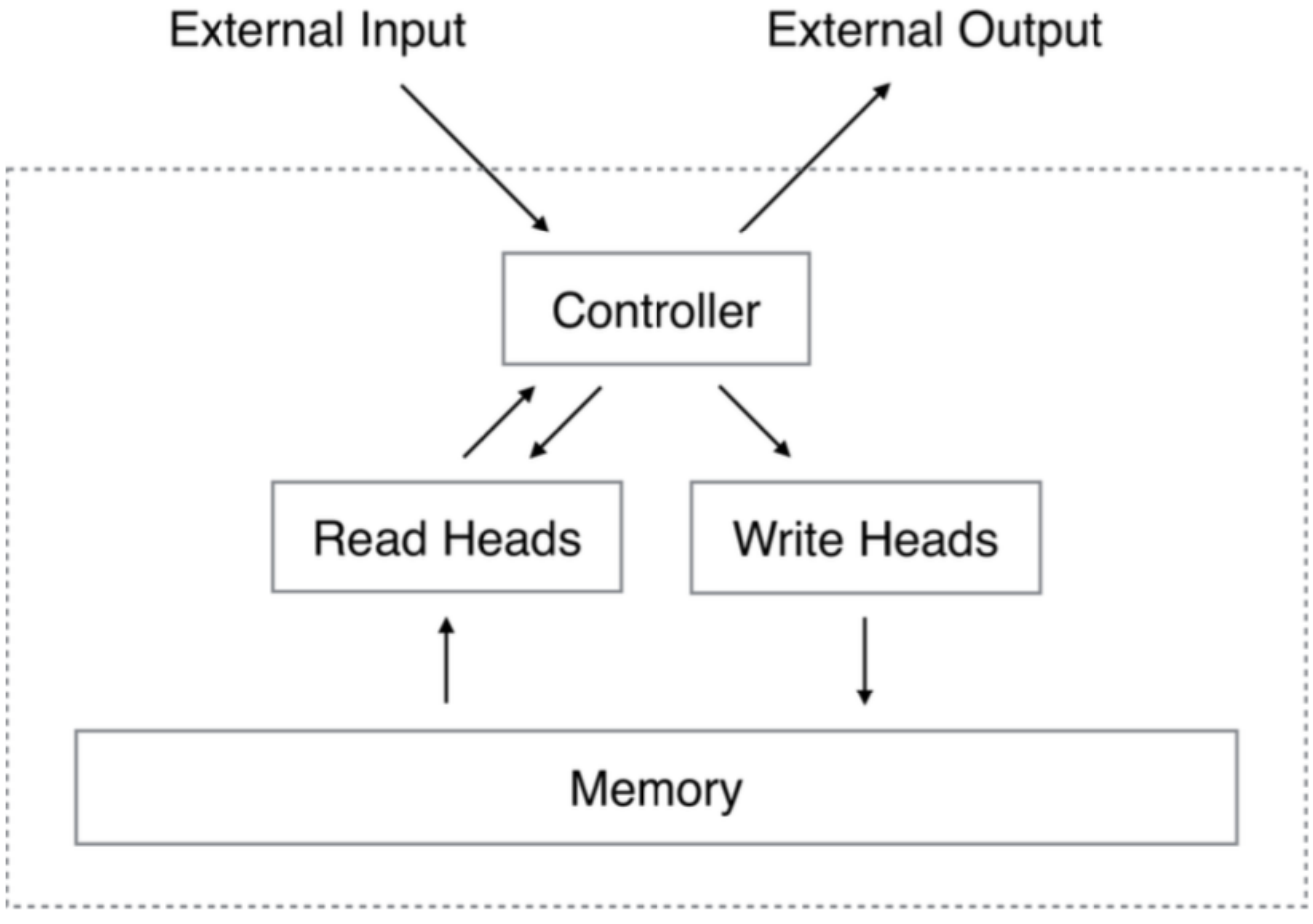


A dog is standing on a hardwood floor.

Attention Mechanism



Neural Turing Machine



Natural Language Processing

- Named Entity Recognition (NER)
  - identifying precisely entity mentions
  - classifying them into the corresponding entity types according to their context
  - E.g., Extract entities of a small set of types: people (PER), organizations (ORG), locations (LOC), and MISC.

    "**Germany**'s representative to the **European Union**'s veterinary committee **Werner Zwingman** said on Wednesday consumers should …"

Natural Language Processing

- IOB scheme for Sequence Labelling
  - NER: B-PER, I-PER, B-ORG, I-ORG, B-LOC, I-LOC, B-MISC, I-MISC, O

"**Germany**'s representative to the **European Union**'s veterinary . . .
B-LOC O O O B-ORG I-ORG O

  - Part-of-Speech Tagging
  - Chunking
  - Semantic Role Labeling
  - Etc.

Natural Language Processing

- Sentiment Analysis: negative, neutral, positive
- Question Answering (QA)
  - Model
    - Retrieval-based
    - Generative
  - Domain: open and close
- Machine Translation: translate a sentence in a source language into a target language
- Visual Recognition
  - Image Captioning
  - Visual QA
- Speech Synthesis: generate audio from text

Distributed Representation

- Many-to-many relationship between two types of representations
  - A word is represented as a vector and a vector encodes words as its entries
- Multi-level Representation
  - Word embedding
  - Character-level representation of words
  - Mention and context representation
  - Knowledge representation of entities

Word Embedding

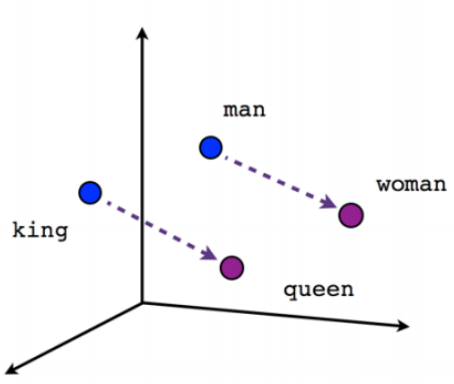
"You shall know a word by the company it keeps"

(J. R. Firth 1957: 11)

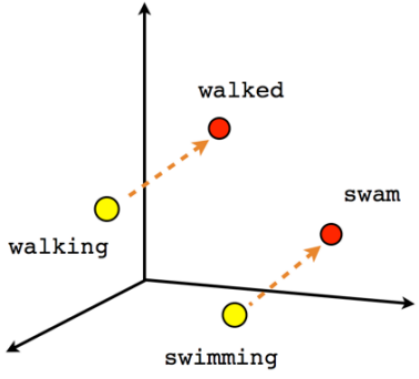
Word Embedding

- SENNA' word embedding
  - https://ronan.collobert.com/senna/
- Word2vec
  - https://github.com/dav/word2vec
- Glove
  - https://nlp.stanford.edu/projects/glove/

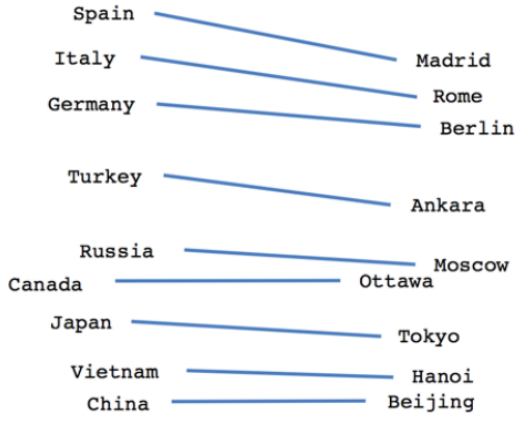
Word Embedding



Male-Female



Verb tense

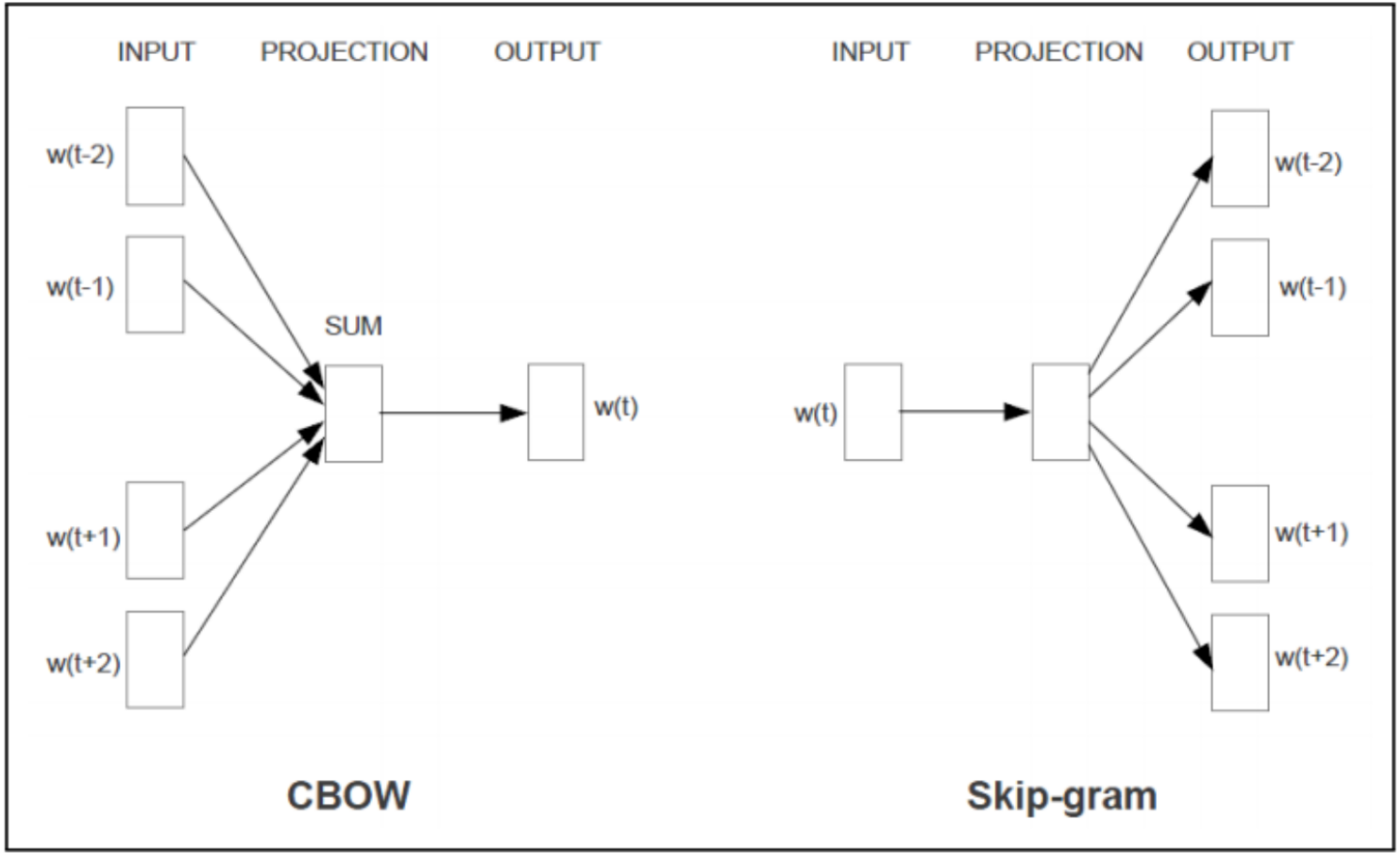


Country-Capital

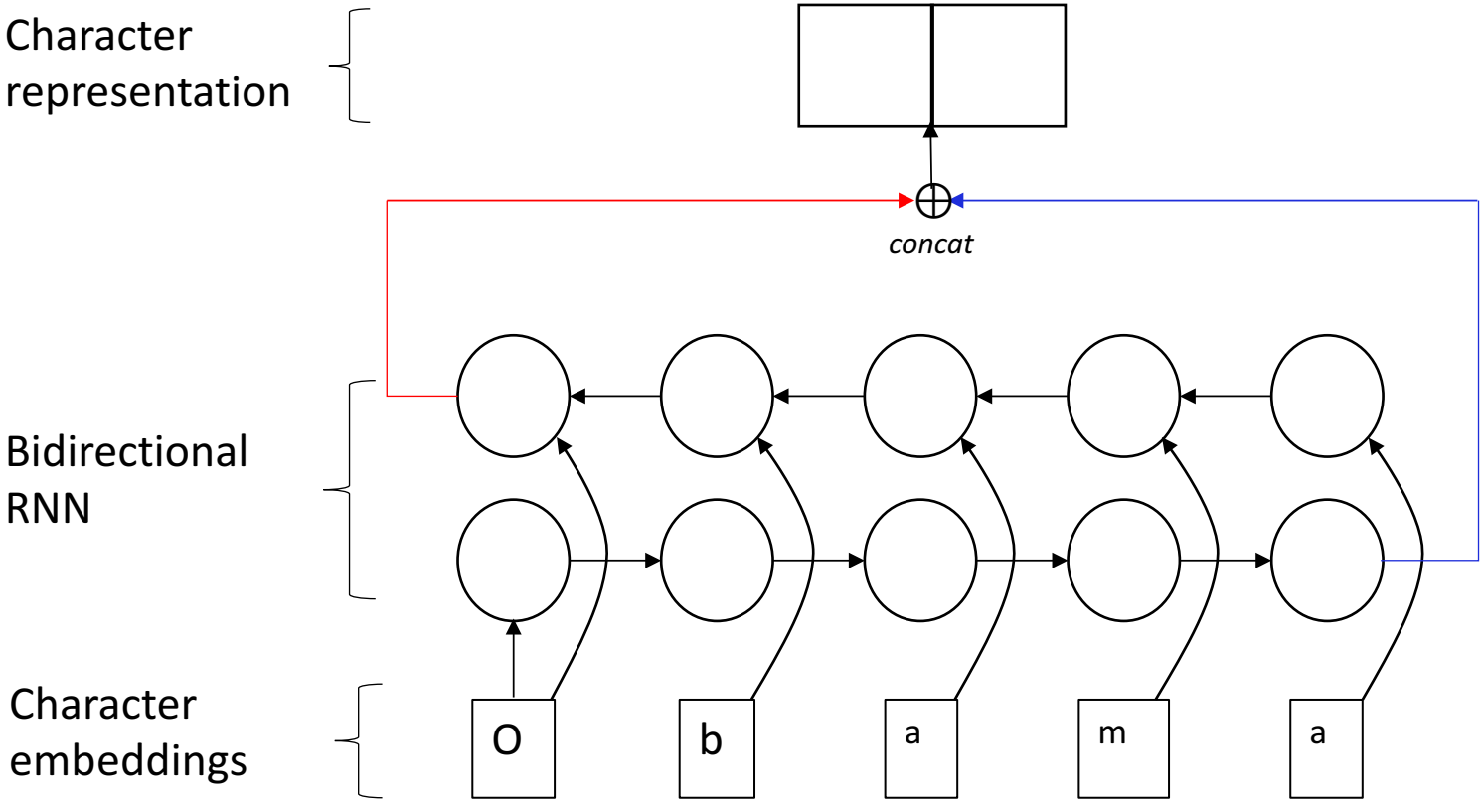
'king' - 'man' + 'woman' is close to 'queen'

https://www.tensorflow.org/images/linear-relationships.png

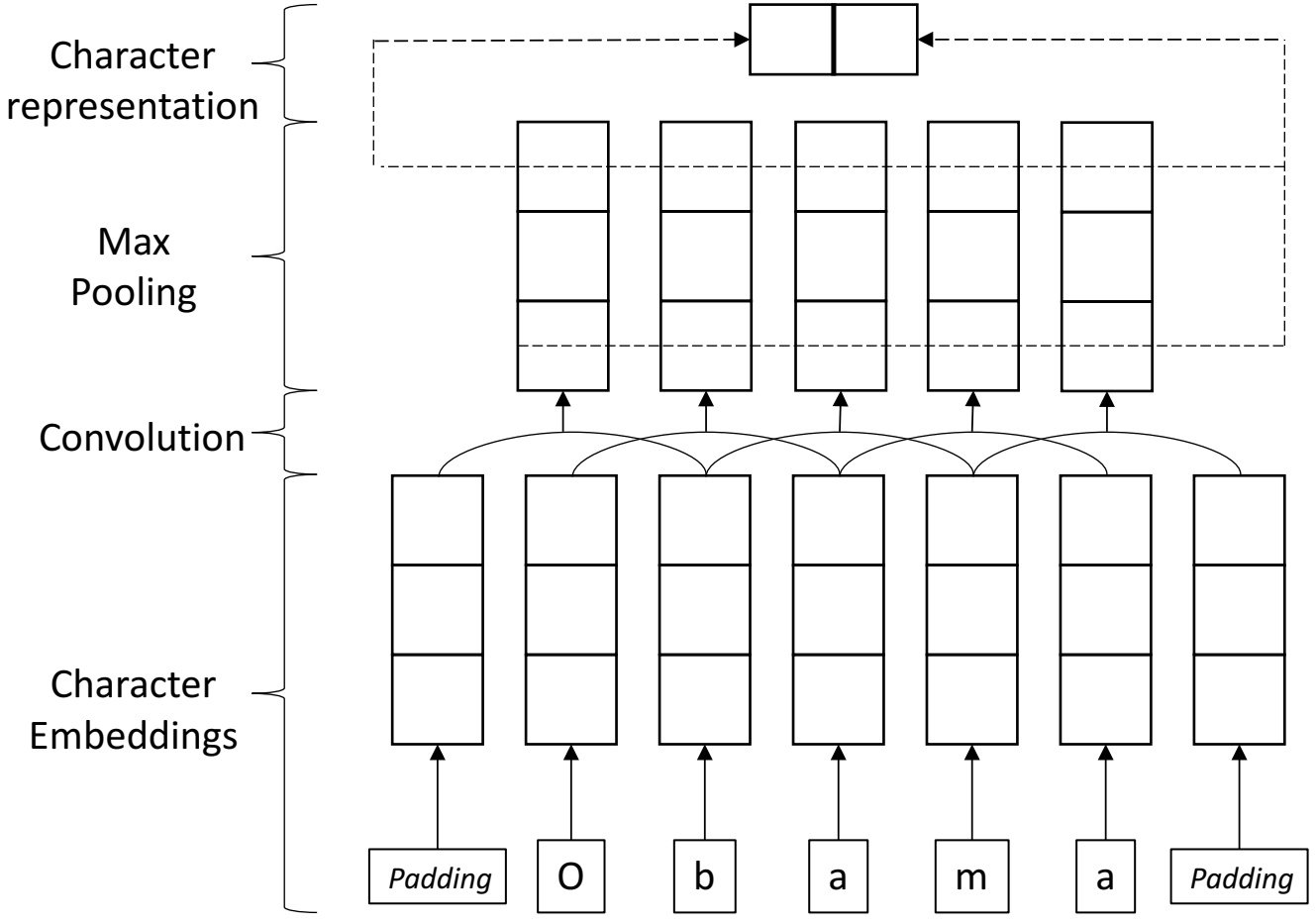
Word2vec


INPUT
PROJECTION
OUTPUT
w(t-2)
w(t-1)
SUM
w(t)
w(t+1)
w(t+2)
CBOW
INPUT
PROJECTION
OUTPUT
w(t)
w(t-2)
w(t-1)
w(t+1)
w(t+2)
Skip-gram

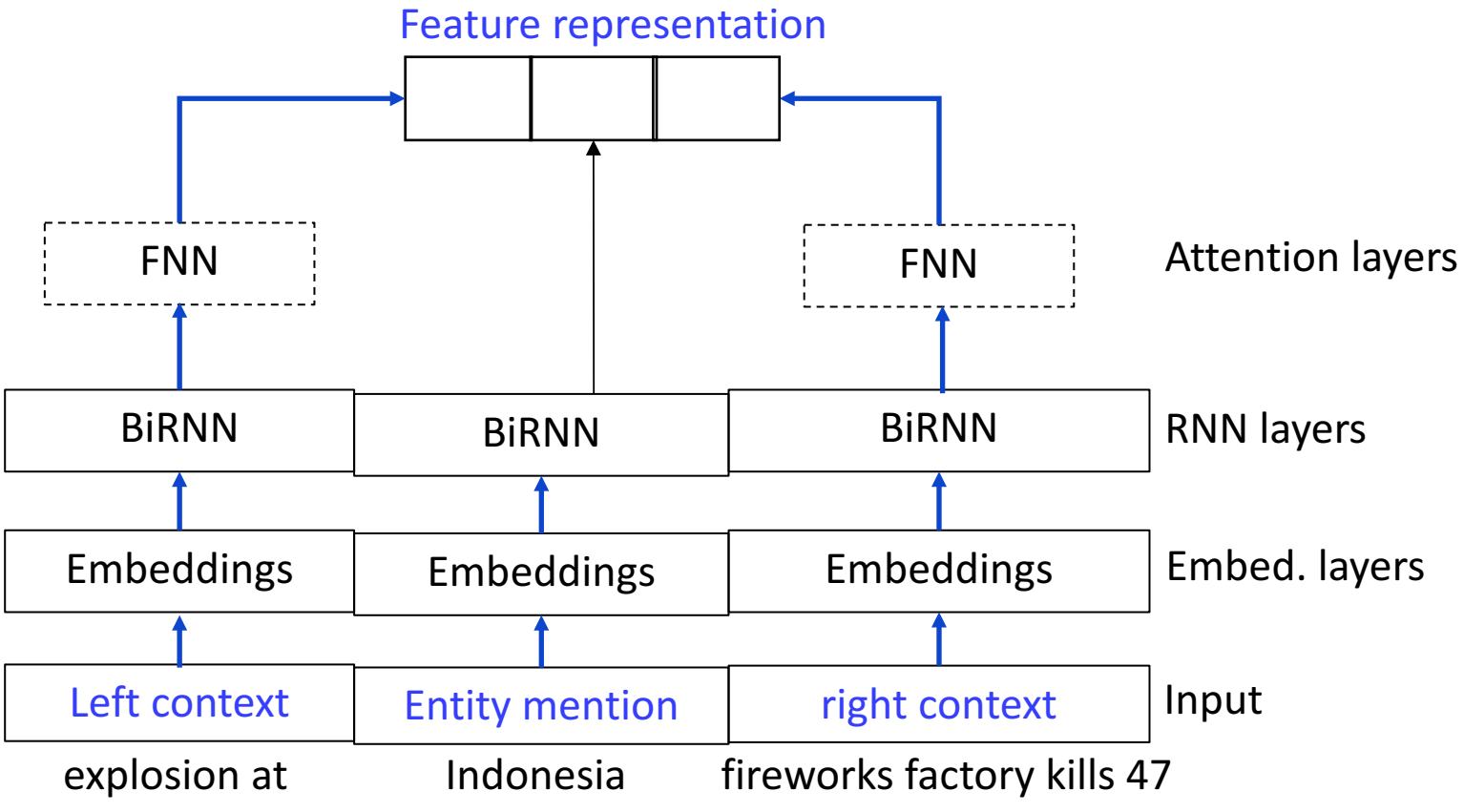
Char.-level Representation of words using LSTM



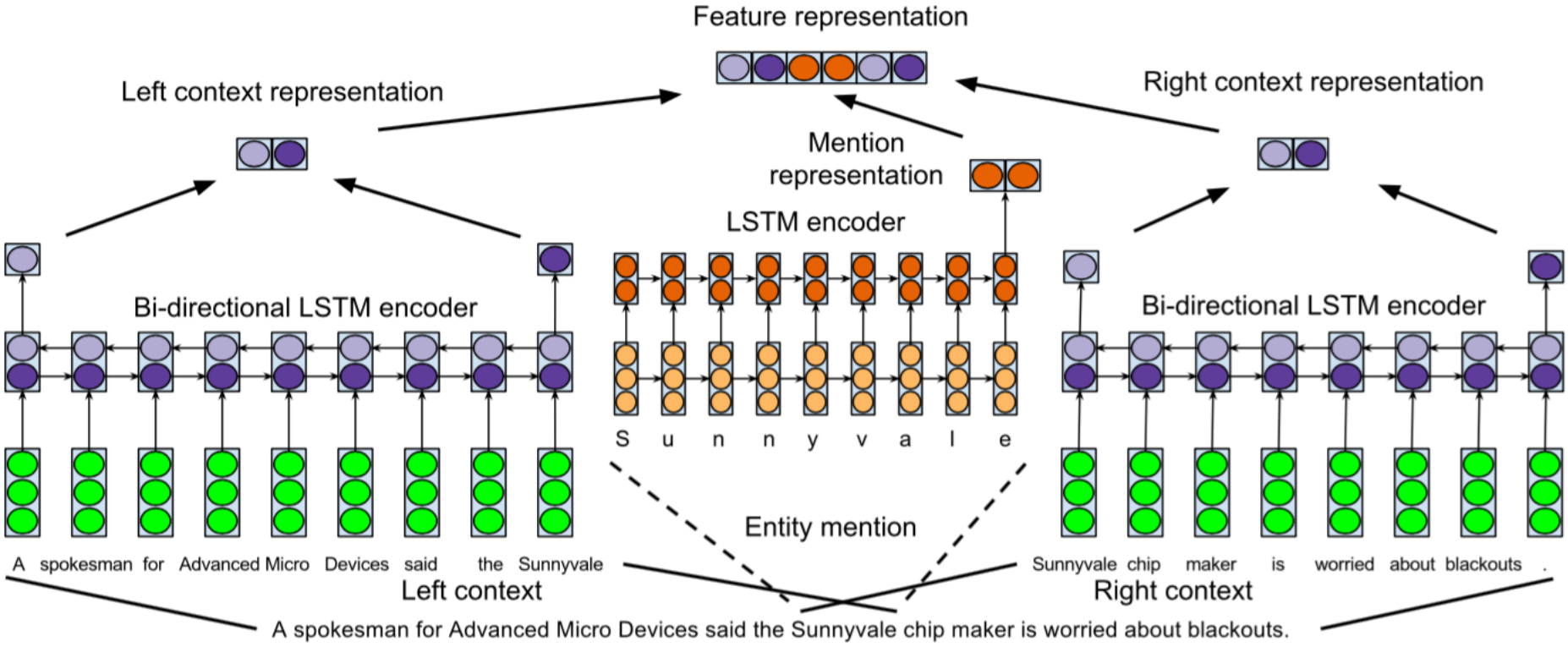
Char.-level Representation of words Using CNN



Mention and Context Representation



Mention and Context Representation



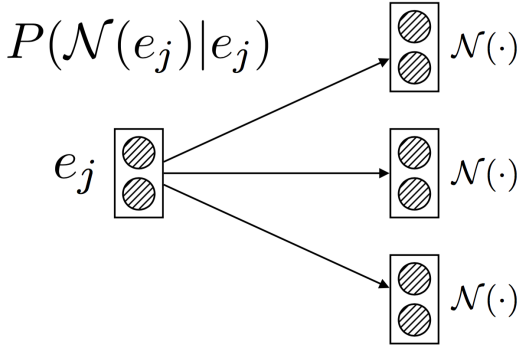
Abhishek and Awekar, EACL 2017

Knowledge representation of entities

- Entities sharing more common neighbors tend to have similar representations
- Given a $\mathcal{KB}$, let $\mathcal{E} = \{e_j\}$ be a set of entities
  - maximizing the log probability

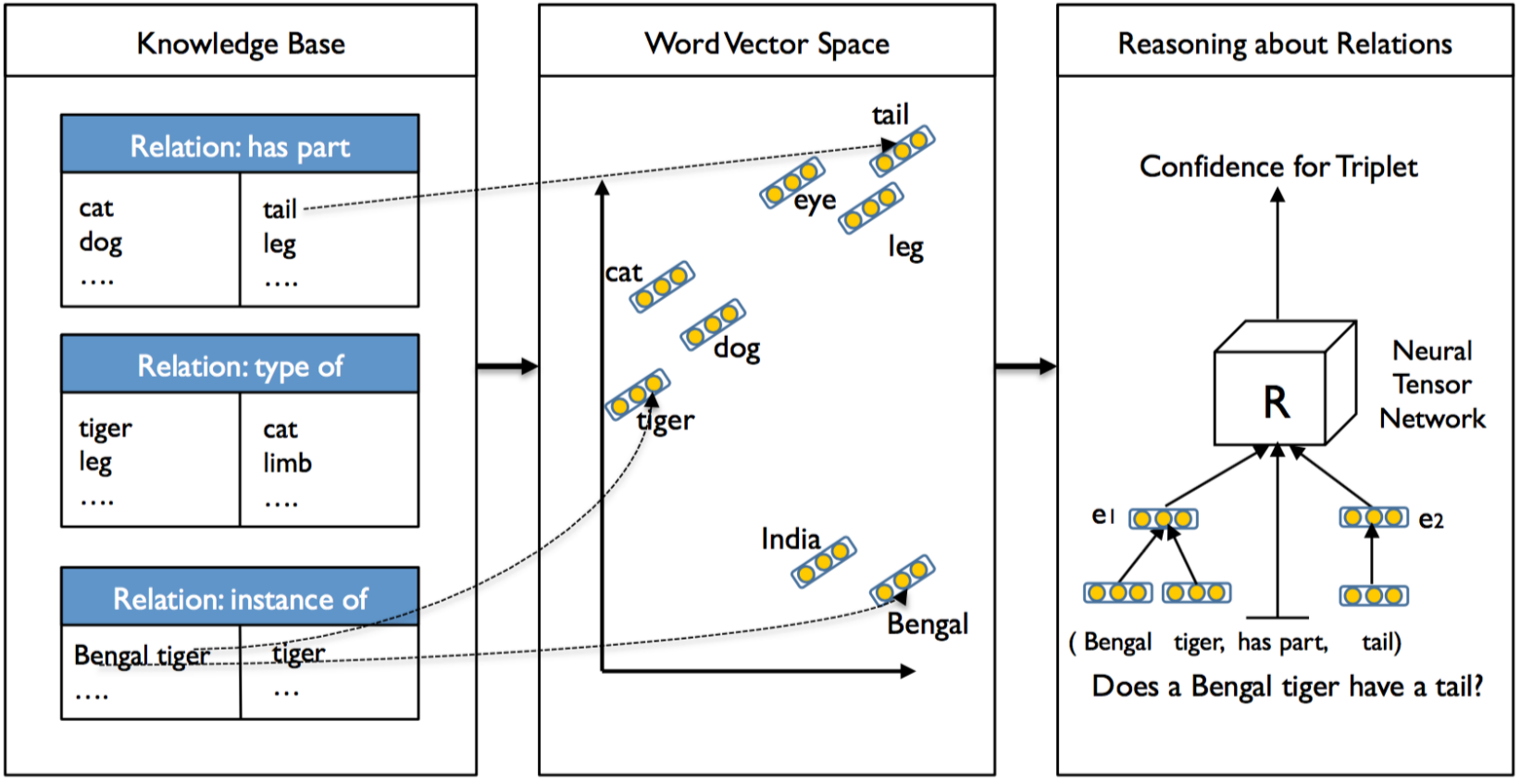
$$\mathcal{L}_e = \sum_{e_j \in \mathcal{E}} \log P(\mathcal{N}(e_j)|e_j)$$

Neighbor Entities



Huang *et. at.,* Big Data, 5(1), 19-31, 2017

Knowledge representation of entities



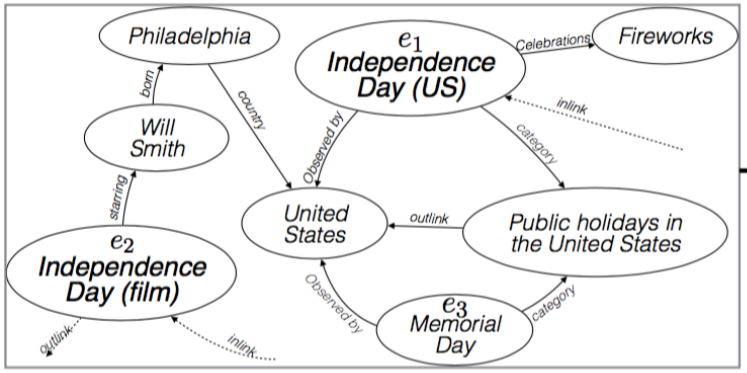
Socher et. al., NIPS 2013

Sentic LSTM

$$f_i = \sigma(W_f[x_i, h_{i-1}, \mu_i] + b_f)$$
$$I_i = \sigma(W_I[x_i, h_{i-1}, \mu_i] + b_I)$$
$$\widetilde{C}_i = tanh(W_C[x_i, h_{i-1}] + b_C)$$
$$C_i = f_i * C_{i-1} + I_i * \widetilde{C}_i$$
$$o_i = \sigma(W_o[x_i, h_{i-1}, \mu_i] + b_o)$$
$$o_i^c = \sigma(W_{co}[x_i, h_{i-1}, \mu_i] + b_{co})$$
$$h_i = o_i * tanh(C_i) + o_i^c * tanh(W_c\mu_i)$$

Ma et. al., AAAI 2018

Unify Text and Knowledge into a semantic space



Knowledge Base

bands played it during public events, such as [[Independence Day (US)|July 4th]] celebrations

Anchor

Mention Sense Mapping

$g(July\ 4th, e_1)$

played it during public events, such as $[[s^*_{Independence\ Day\ (US)}]]$ celebrations

d_1 … In the 1996 action film [[Independence Day (film)|Independence Day]], the United States military uses alien technology captured …

d_2 … holds annual [[Independence Day (US)|Independence Day]] celebrations and other festivals …

d_3 … early Confederate [[Memorial Day]] celebrations were simple, somber occasions for veterans and their families to honor the dead …

Text

Entity Representation Learning

$P(\mathcal{N}(e_j)|e_j)$

e_j → $\mathcal{N}(\cdot)$

Mention Representation Learning

$P(e_j|\mathcal{C}(m_l), s^*_j)$

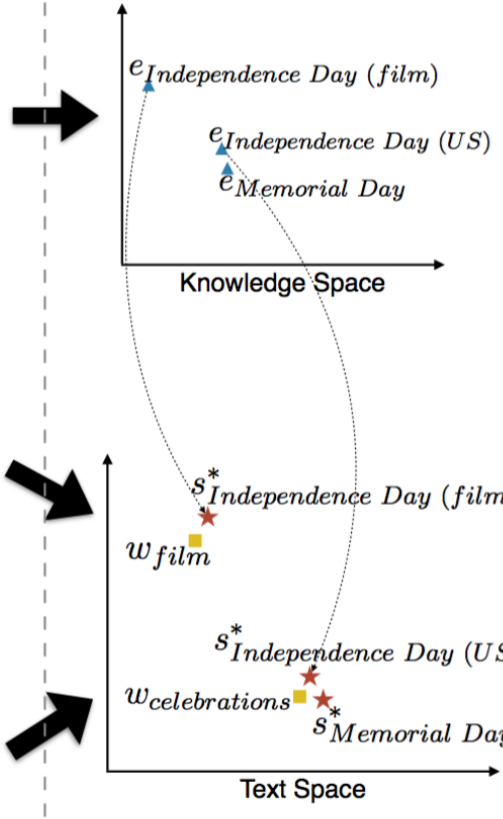
$\mathcal{C}(\cdot)$, $\mathcal{C}(\cdot)$, $\mathcal{C}(\cdot)$, s^*_j → e_j

Text Representation Learning

$P(\mathcal{C}(w_i)|w_i) \cdot P(\mathcal{C}(m_l)|s^*_j)$

w_i/s^*_j → $\mathcal{C}(\cdot)$

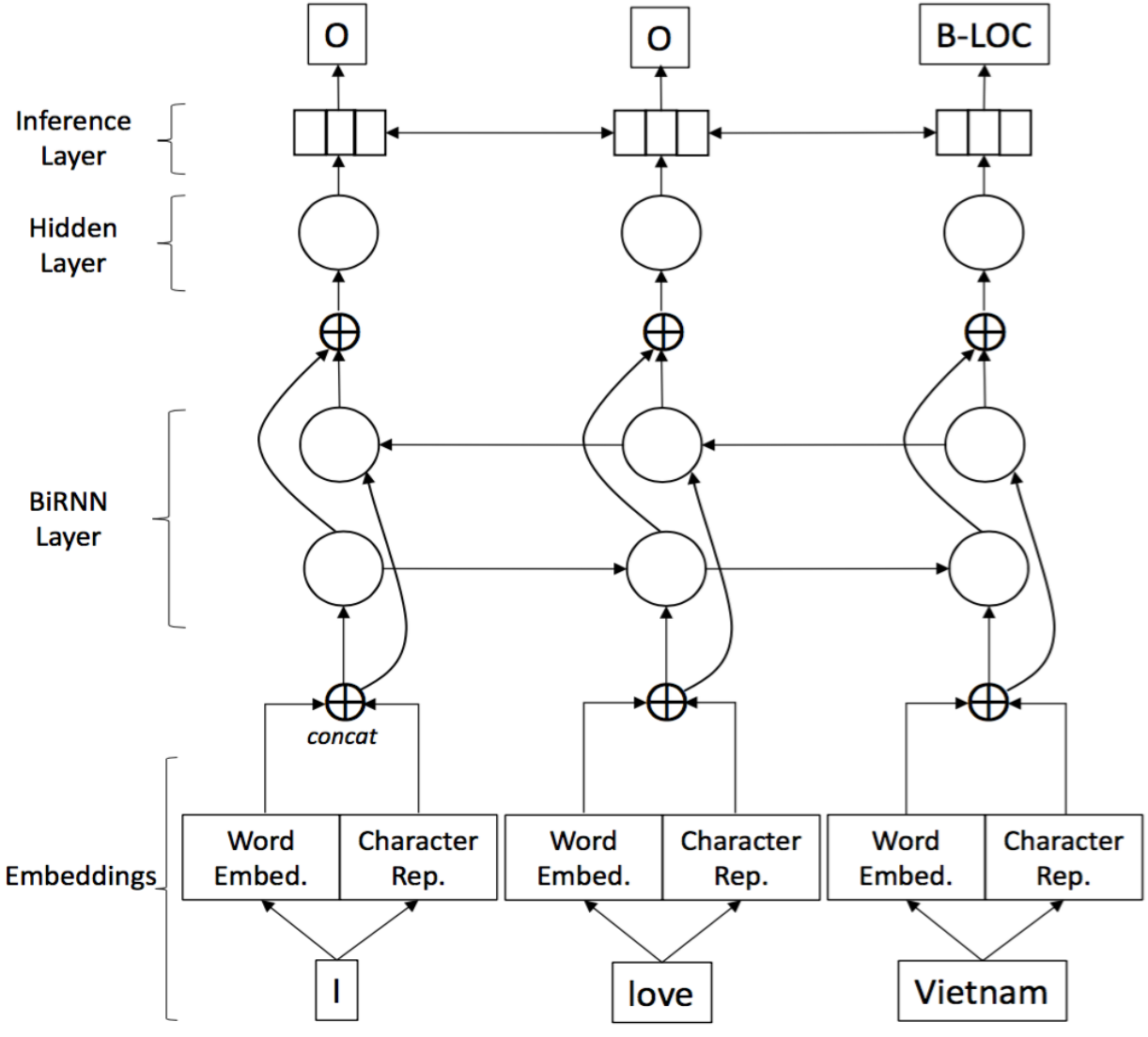
Representation Learning



Cao et. al., ACL 2017

State-of-the-Art NER Systems

- BiRNN-CRF with some variants



State-of-the-Art NER Systems

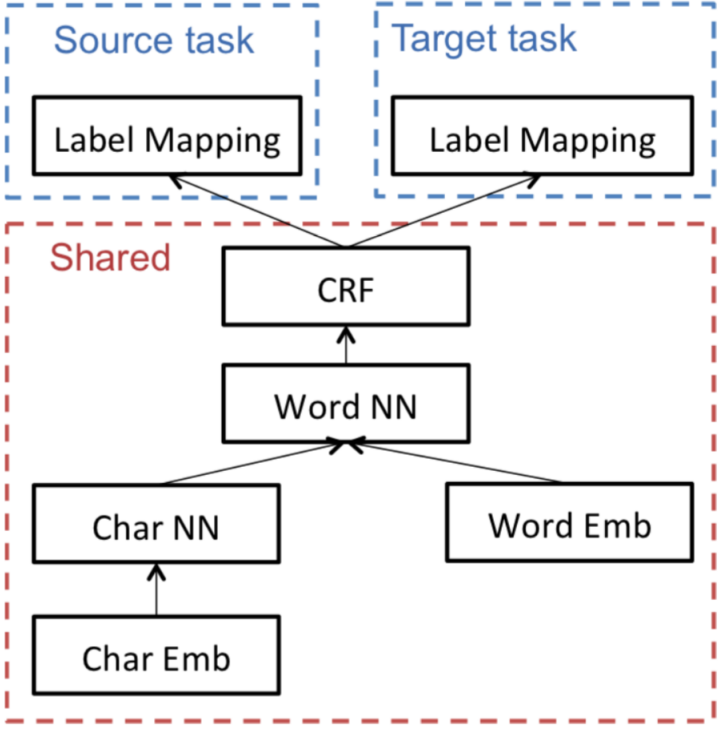
- English CoNLL 2003 dataset

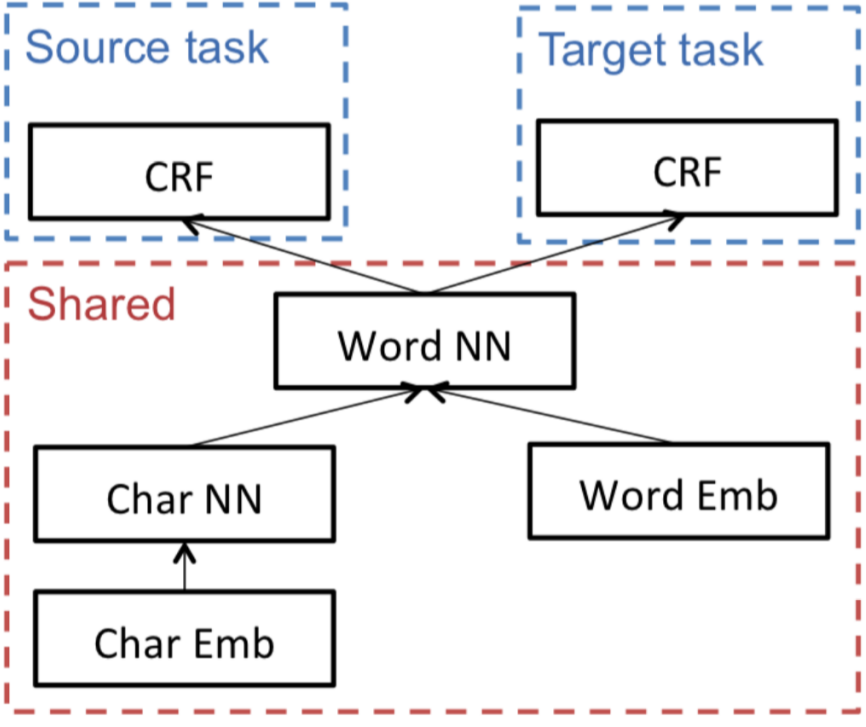
	No.	Method	F1-score
BiRNN - CRF	1	BiLSTM-CRF (Huang *et. al.*, 2015) [31]	90.10%
	2	BiLSTM-CNN-CRF (Ma *et. al.*, 2016) [23]	91.21%
	3	BiLSTM-CRF, char-BiLSTM (Lample *et. al.*, 2016) [21]	90.94%
	4	BiLSTM-CRF, char-CNN (Chiu *et. al.*, 2016) [19]	91.62%
	5	Neural Reranking (Yang *et. al.*, 2017) [18]	91.62%
BiRNN-CRF+	6	Transfer Learning with RNN (Yang *et. al.*, 2017) [17]	91.26%
FOFE+FNN	7	FOFE+FNN (Xu *et. al.*, 2017) [16]	90.85%
Res-RNN	8	3 Res-RNN + foreLM + backLM + bias (Tran *et. al.*, 2017) [12]	91.69%
ID-CNN	9	ID-CNN (Strubell *et. al.*, 2017) [10]	90.65%
BiRNN-CRF + LM	10	TagLM (Peters *et. al.*, 2017) [8]	91.93%
	11	LM-BiLSTM-CRF (Liu *et. al.*, 2017) [7]	91.71%

My paper at ACDT'2017, Tokyo

State-of-the-Art CGER Systems

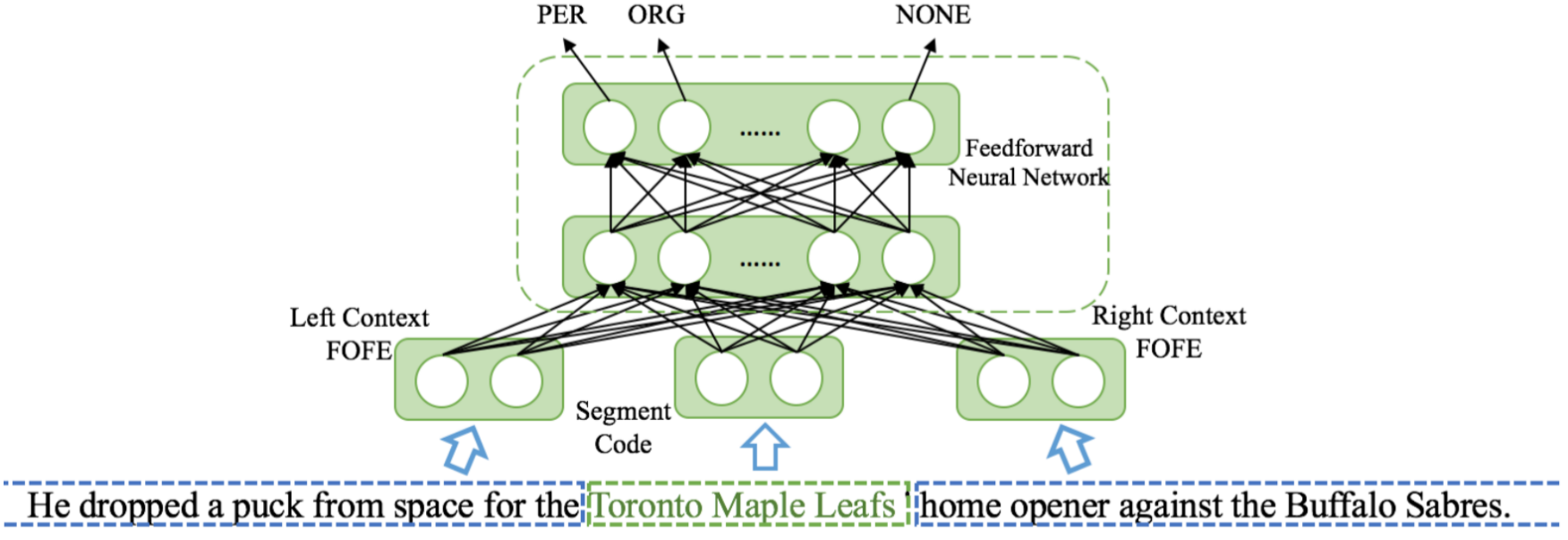
- Transfer Learning





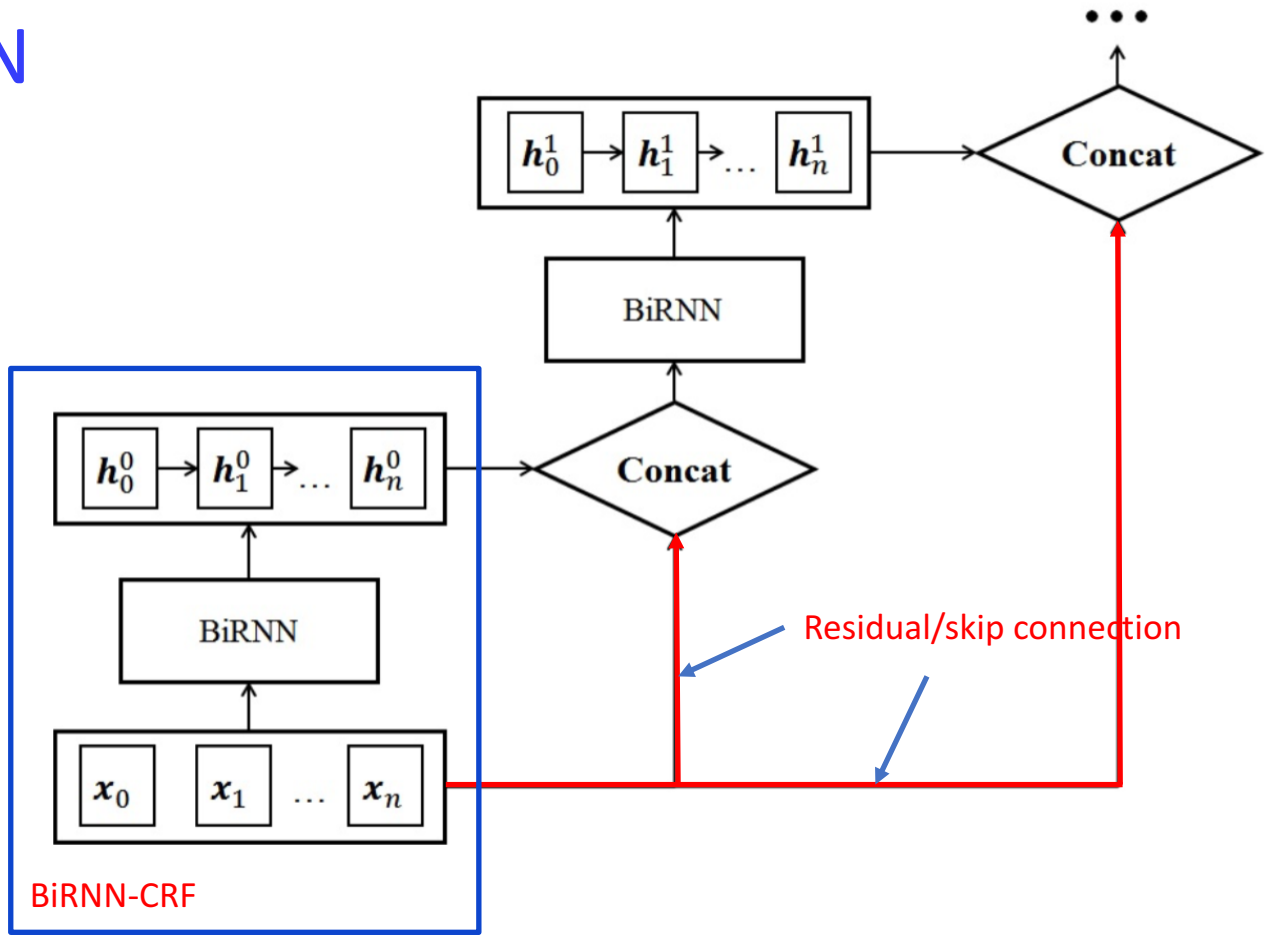
State-of-the-Art CGER Systems

- FOFE+FNN



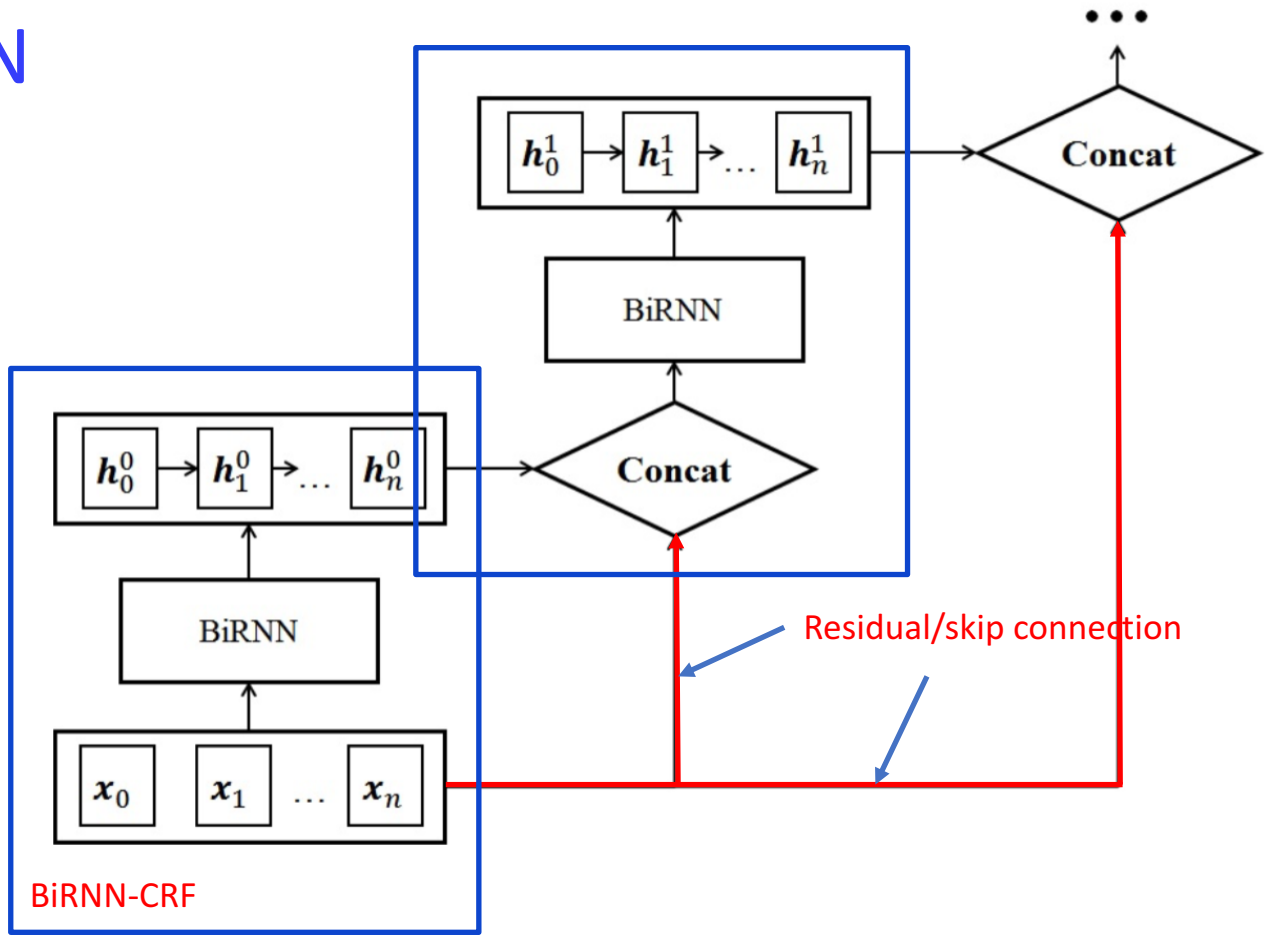
State-of-the-Art CGER Systems

- Res-RNN



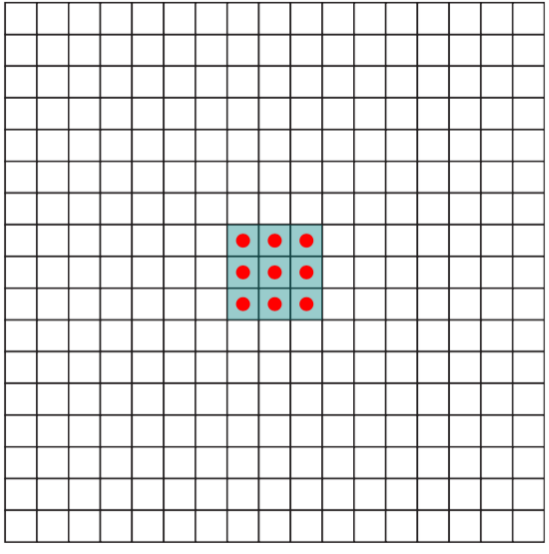
State-of-the-Art CGER Systems

- Res-RNN

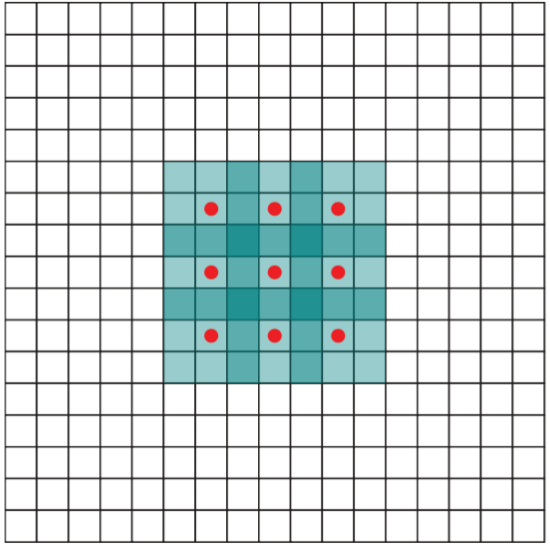


State-of-the-Art CGER Systems

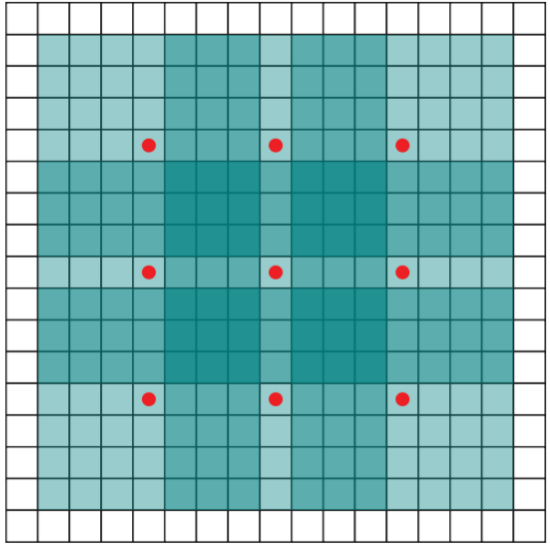
- ID-CNN (Iterated Dilated CNNs)



1-dilated convolution



2-dilated convolution



4-dilated convolution

Yu and Koltun, ICRL 2016

State-of-the-Art CGER Systems

- ID-CNN (Iterated Dilated CNNs)

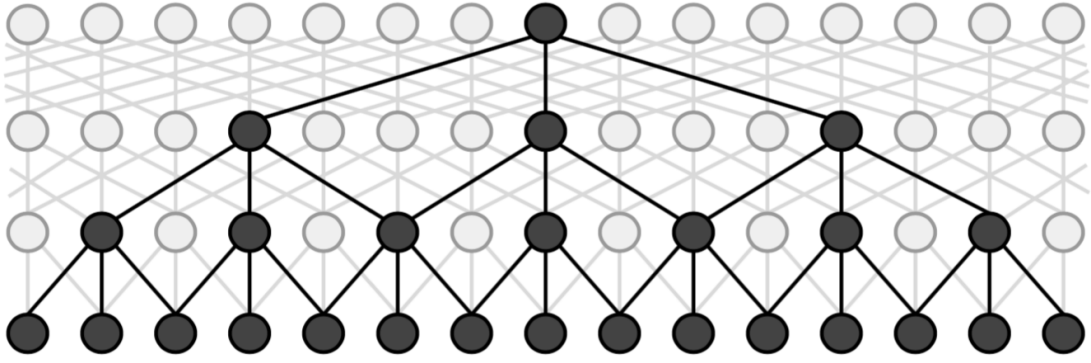


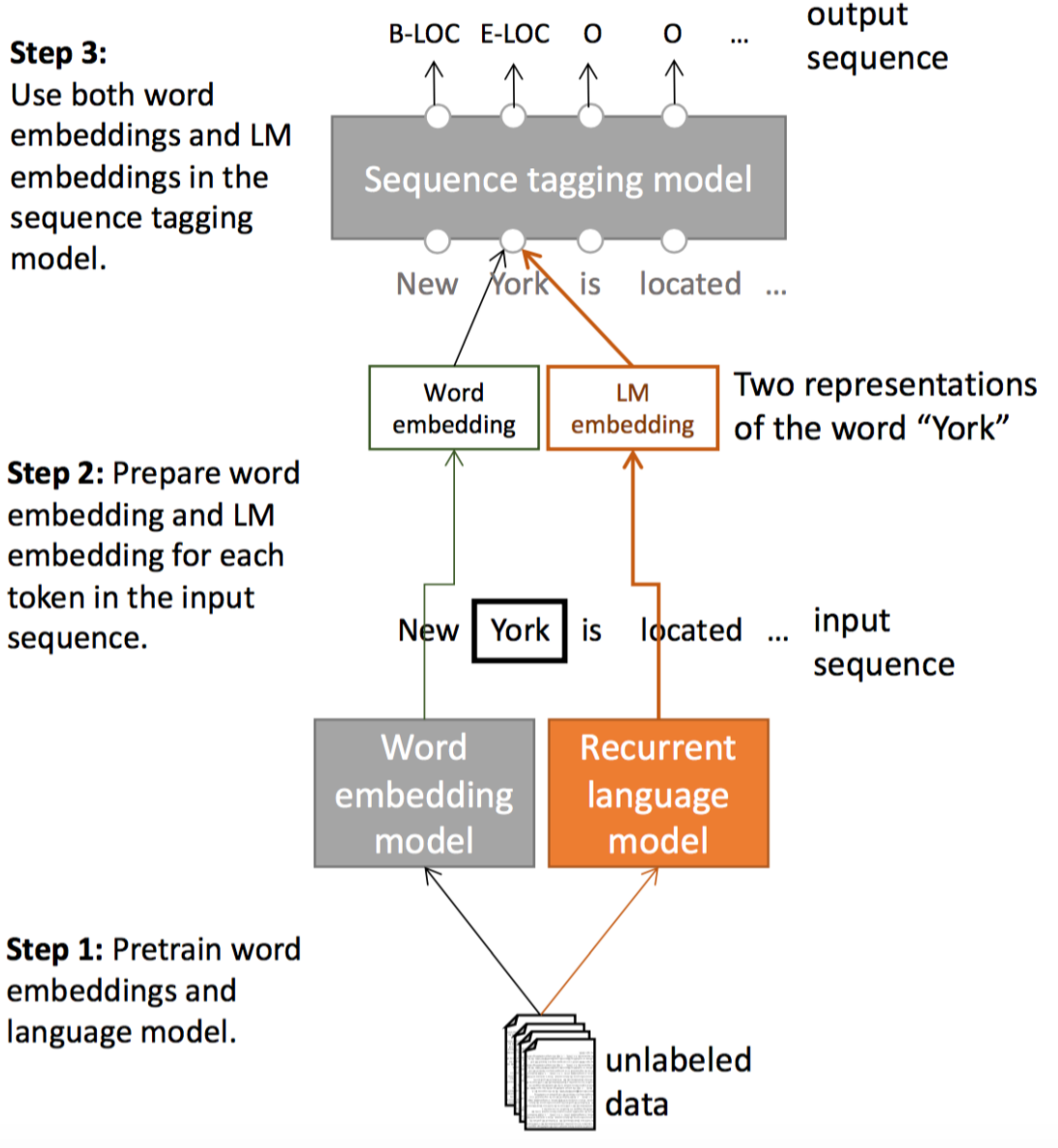
Figure 1: A dilated CNN block with maximum dilation width 4 and filter width 3. Neurons contributing to a single highlighted neuron in the last layer are also highlighted.

Strubell et. al., EMNLP 2017

State-of-the-Art CGER Systems

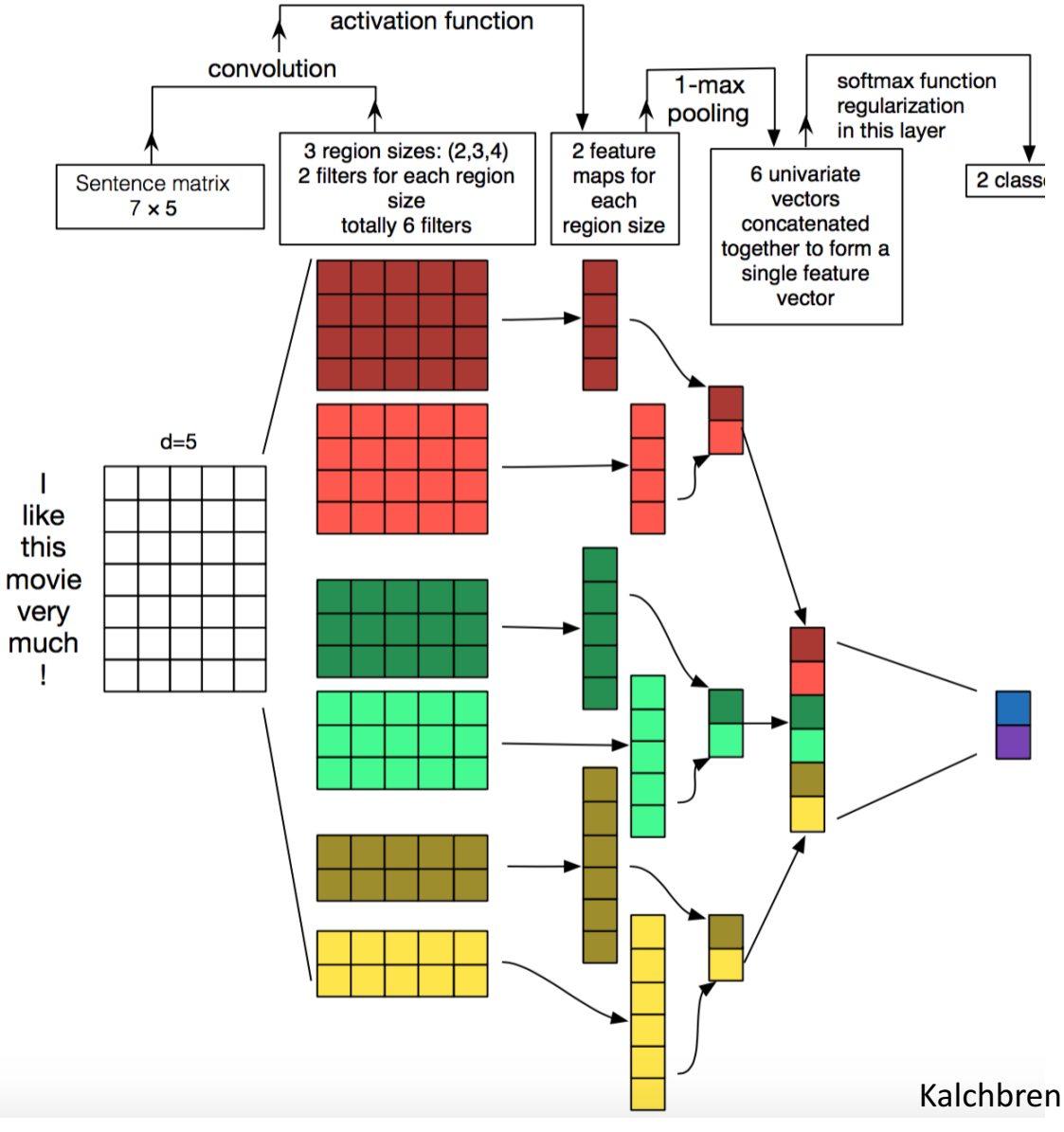
- BiRNN-CRF

+ LM



Peters *et. al.*, 2017

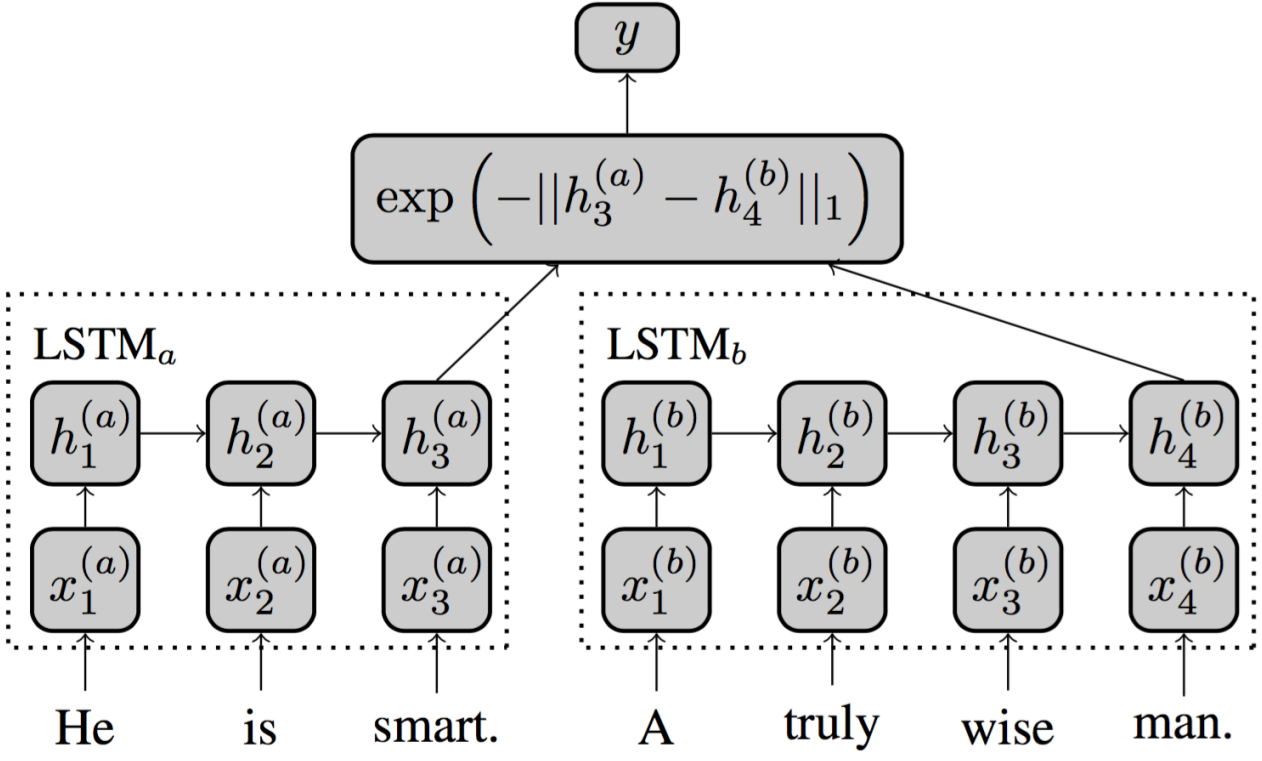
Sentence Modelling



Kalchbrenner et. al., ACL 2014

Sentence Modelling

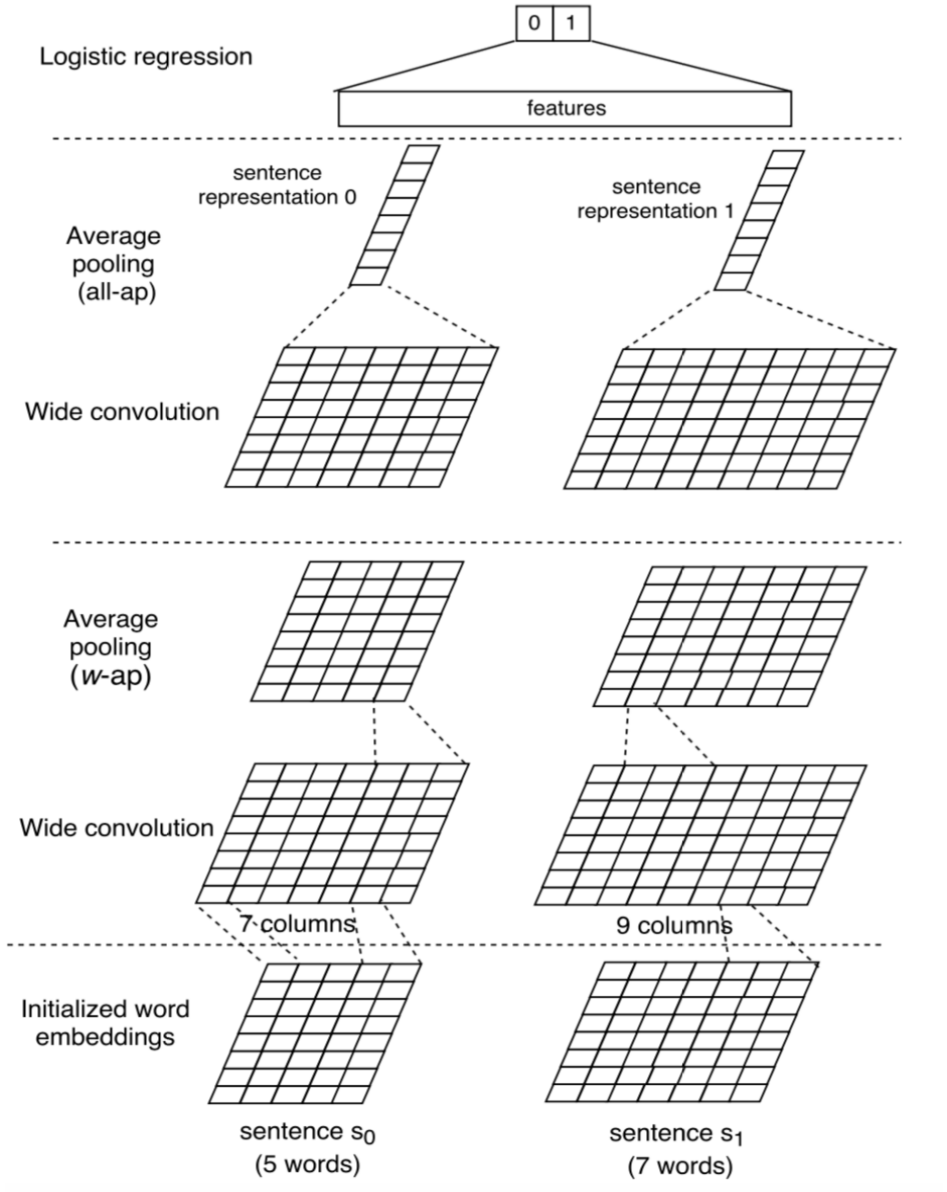
- Siamese Recurrent Architectures (RNN)



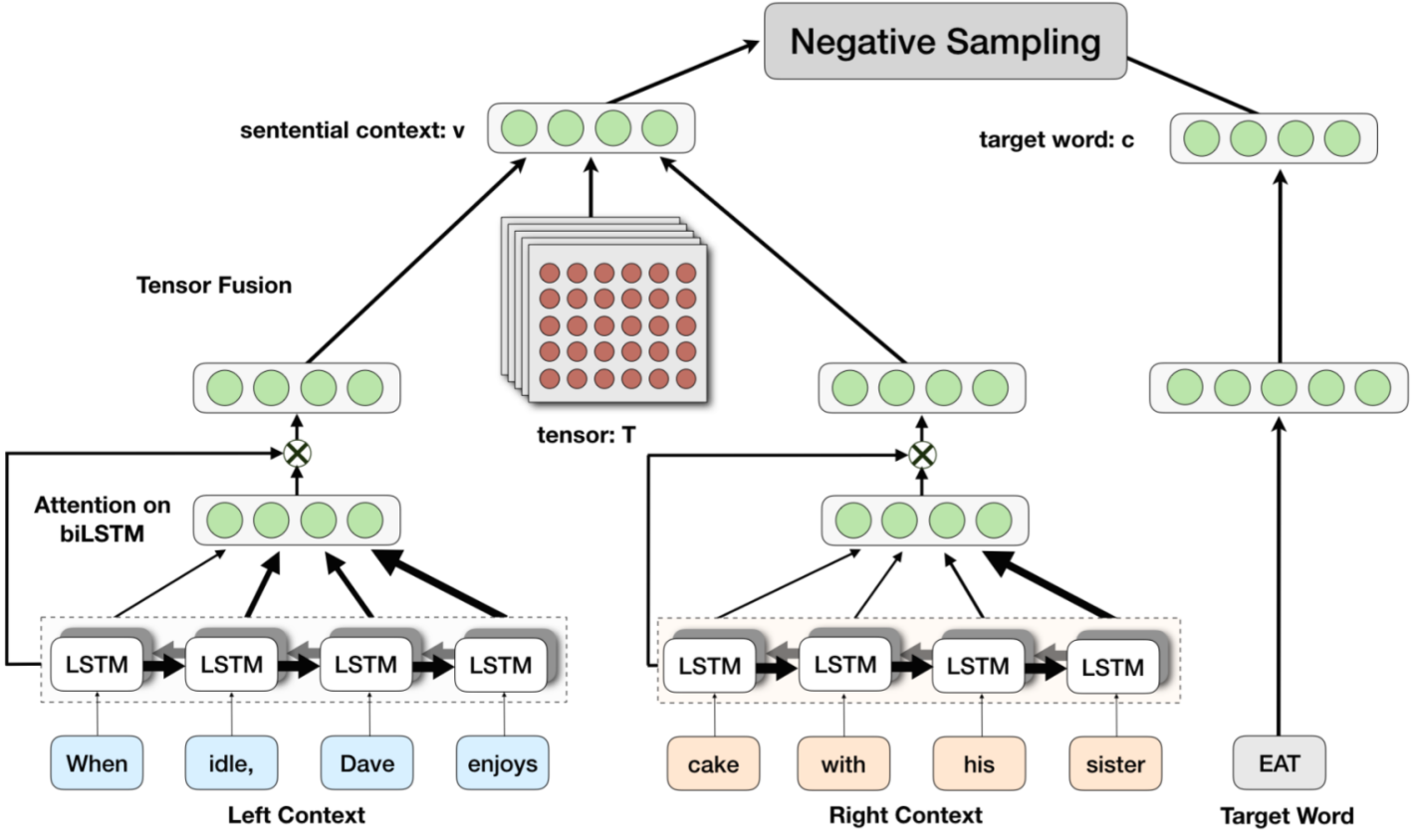
Mueller and Aditya, AAAI 2017

Sentence Modelling

- Siamese Recurrent Architectures (CNN)

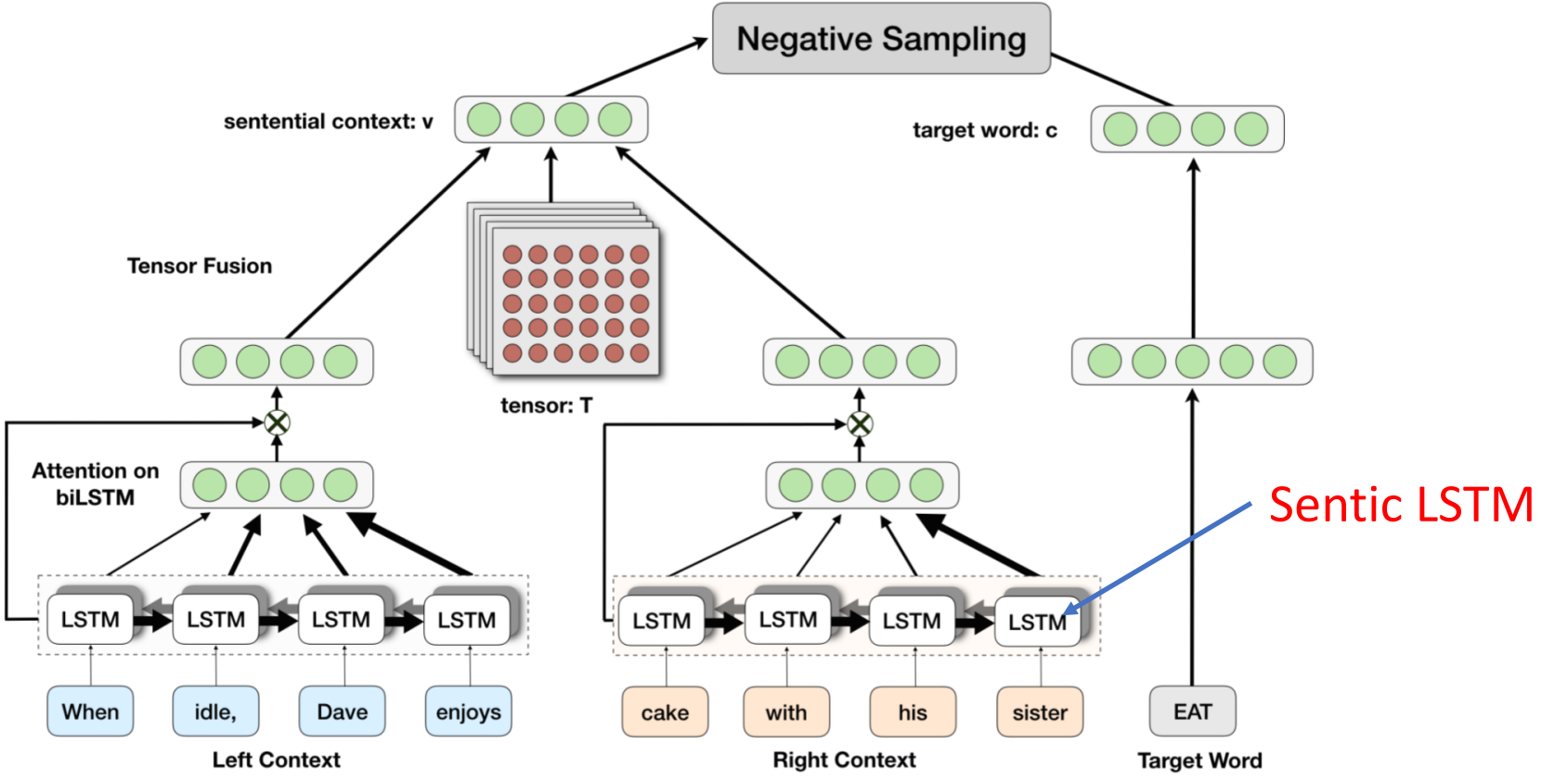


Learning Representation for Sentiment Analysis



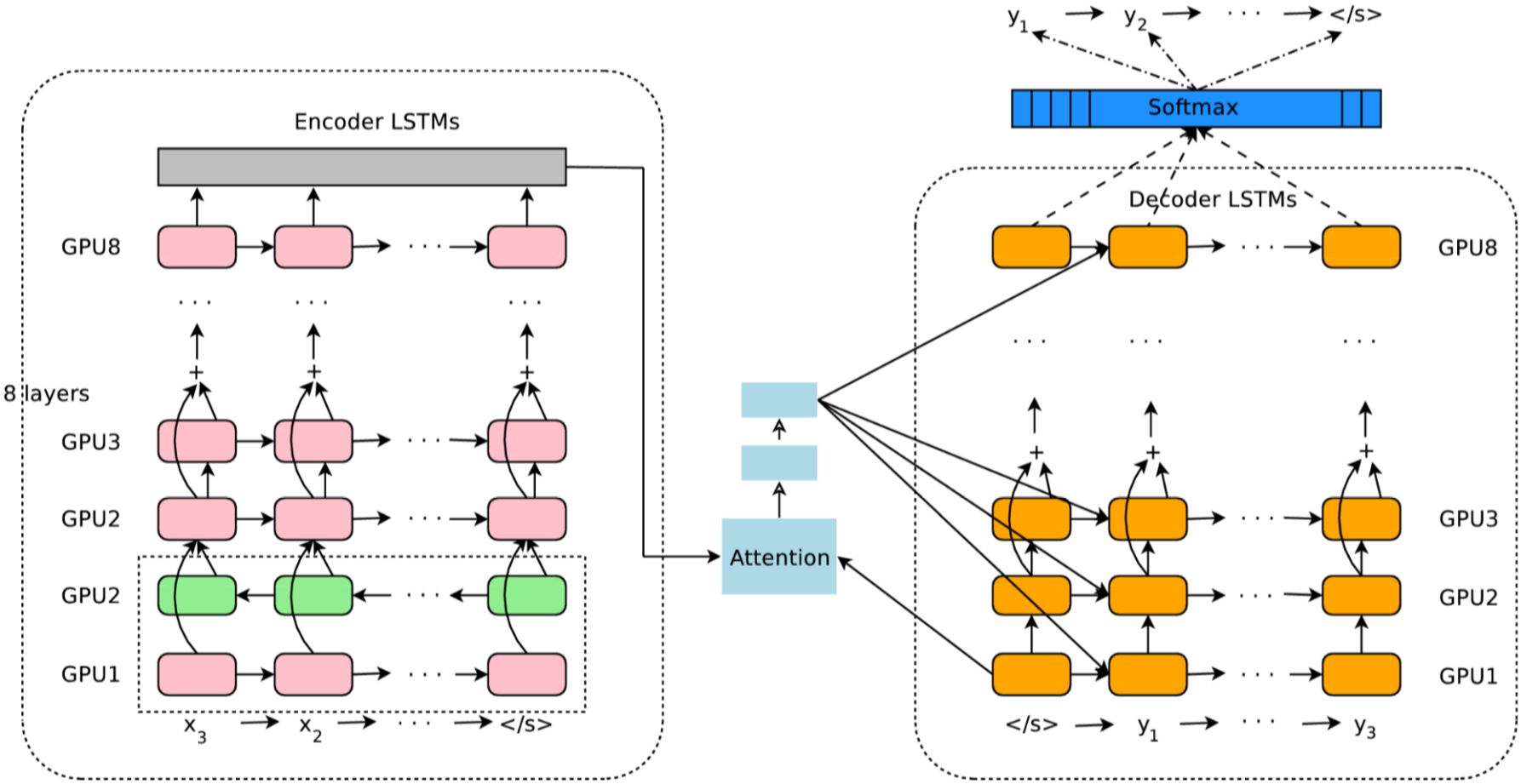
Cambria et. al., AAAI 2018

Learning Representation for Sentiment Analysis



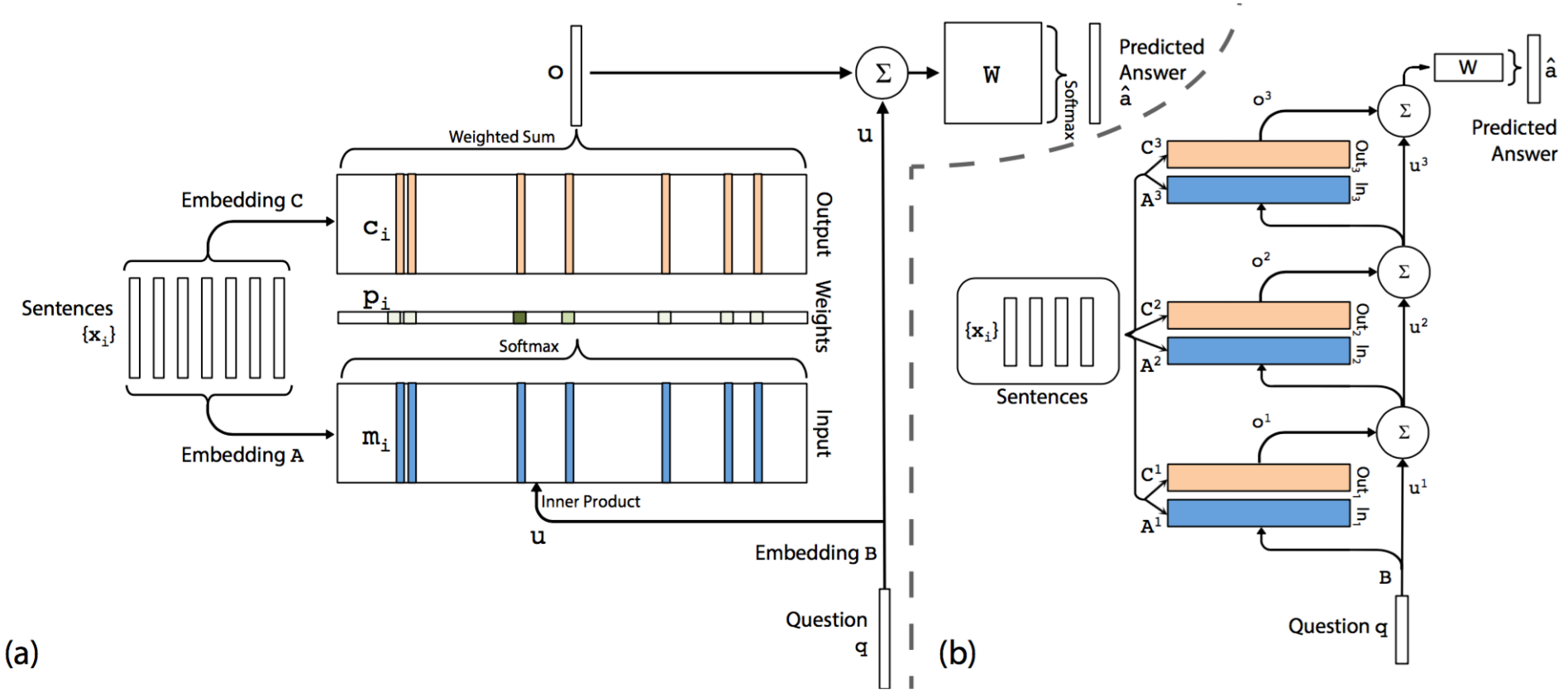
Cambria et. al., AAAI 2018

Google's Neural Machine Translation



Wu et. al, 2016

End-To-End Memory Networks



Sukhbaatar et. al., NIPS 2015

Sequence to Sequence Learning

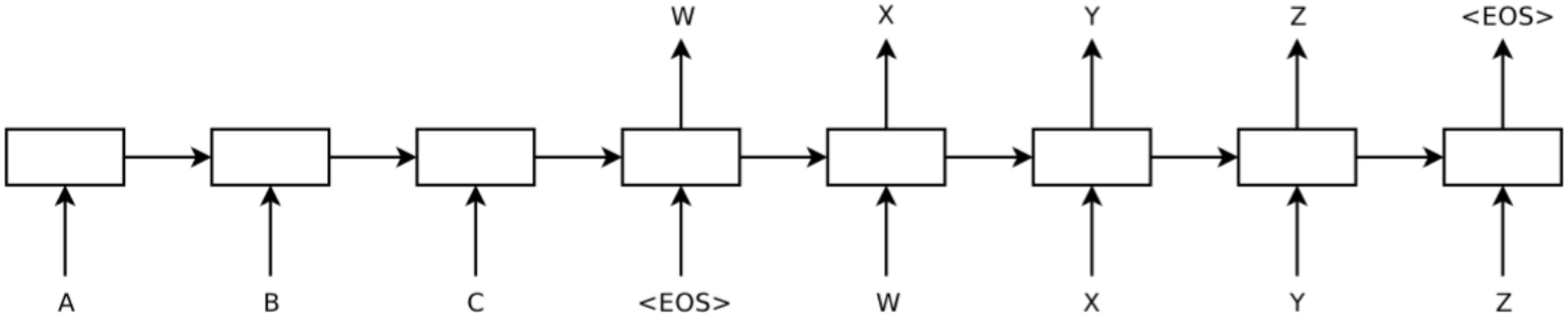
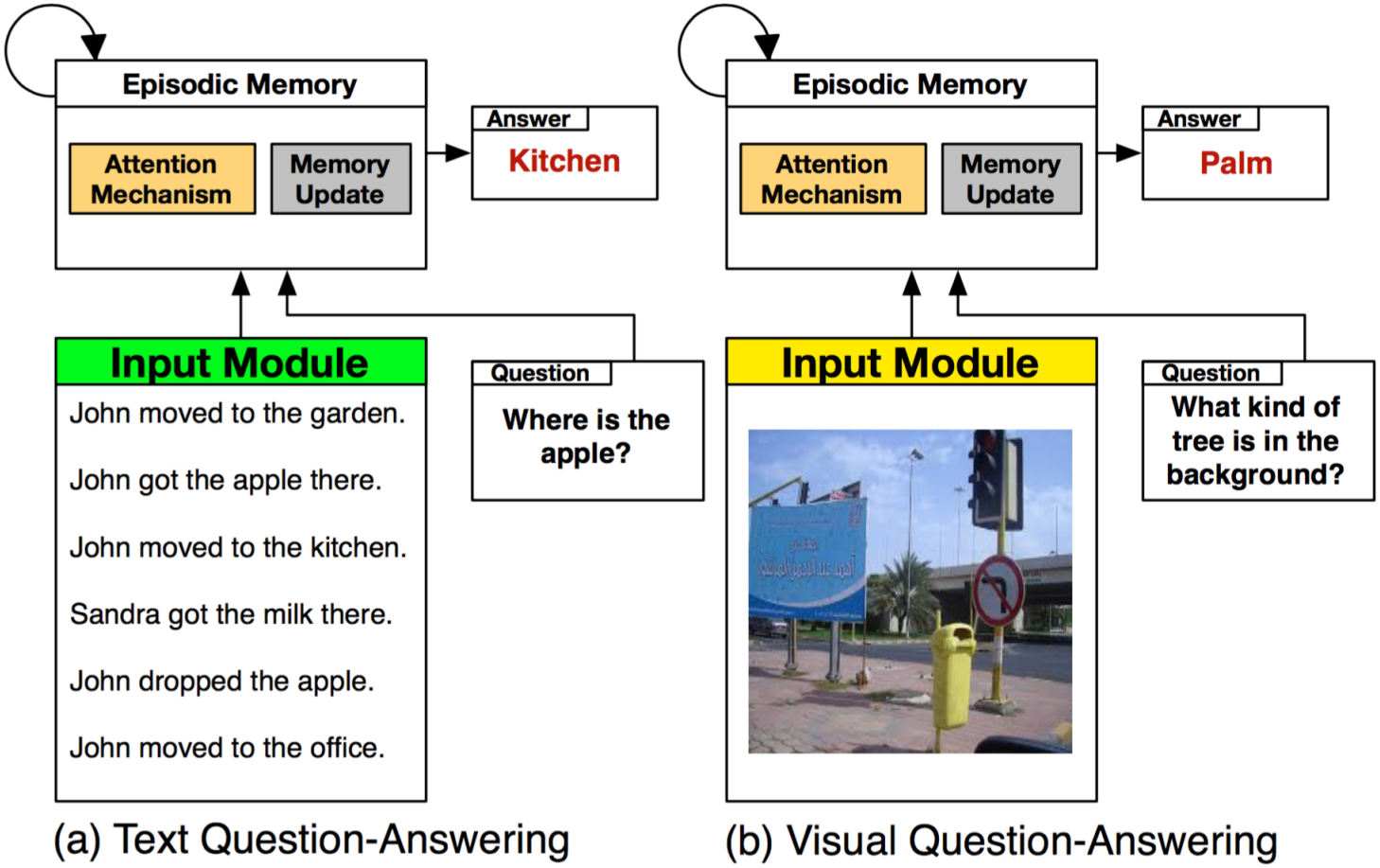


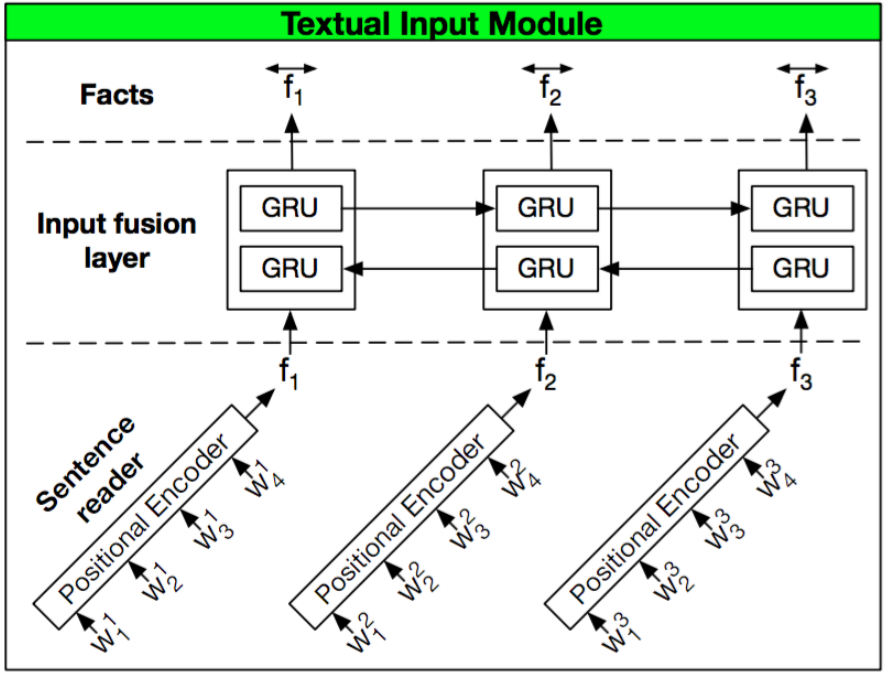
Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence. The model stops making predictions after outputting the end-of-sentence token. Note that the LSTM reads the input sentence in reverse, because doing so introduces many short term dependencies in the data that make the optimization problem much easier.

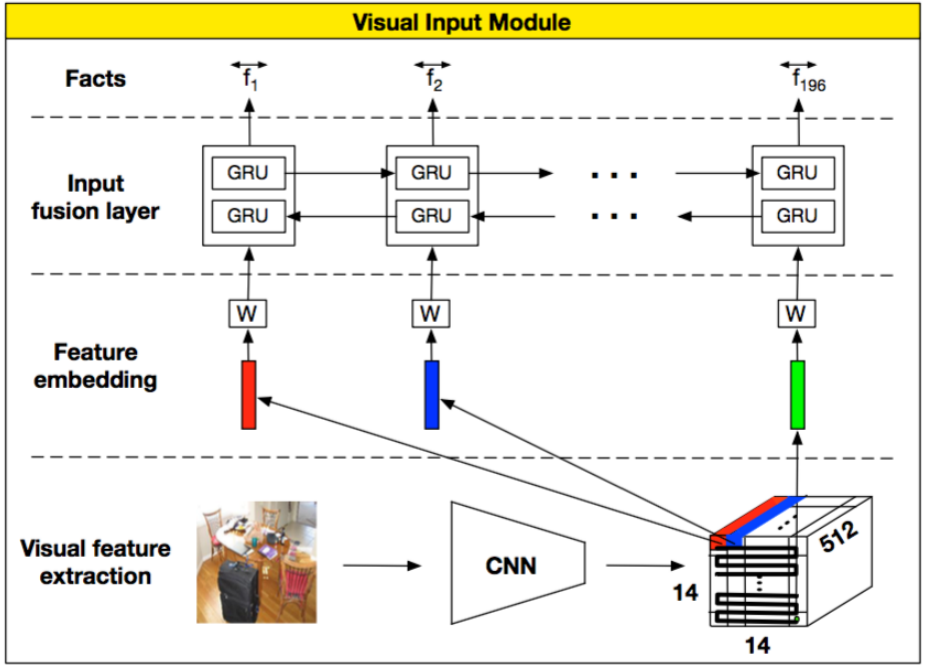
Dynamic Memory Networks



Xiong et. al., ICML 2016

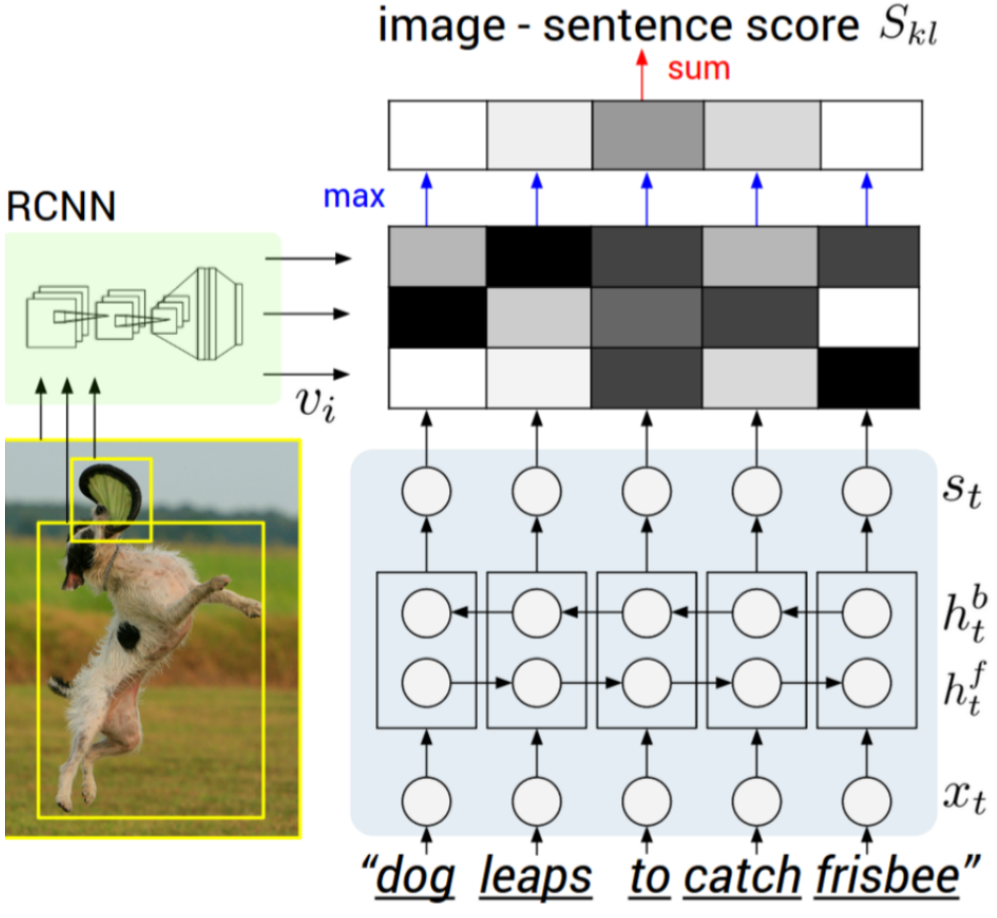
Dynamic Memory Networks





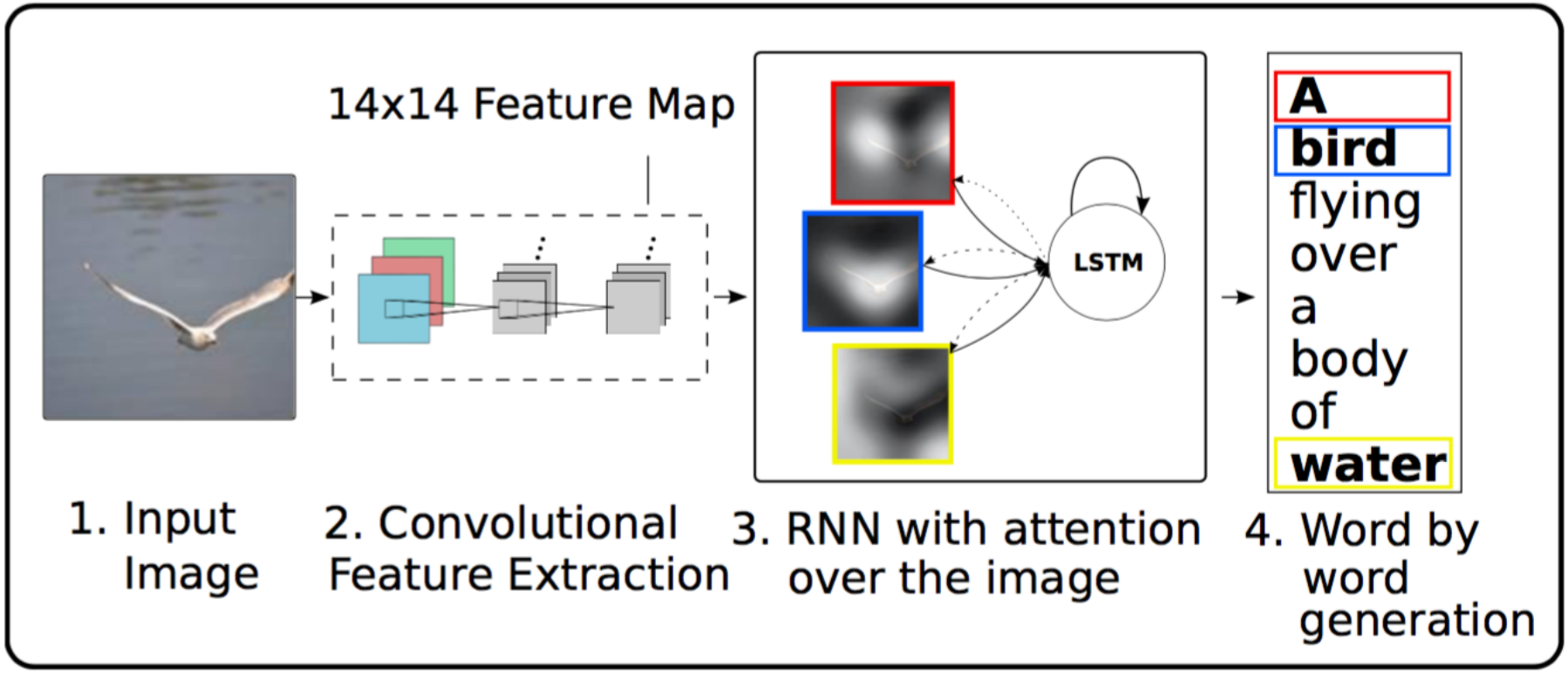
Xiong et. al., ICML 2016

Image Captioning



Karpathy and Fei-Fei, CVPR 2015

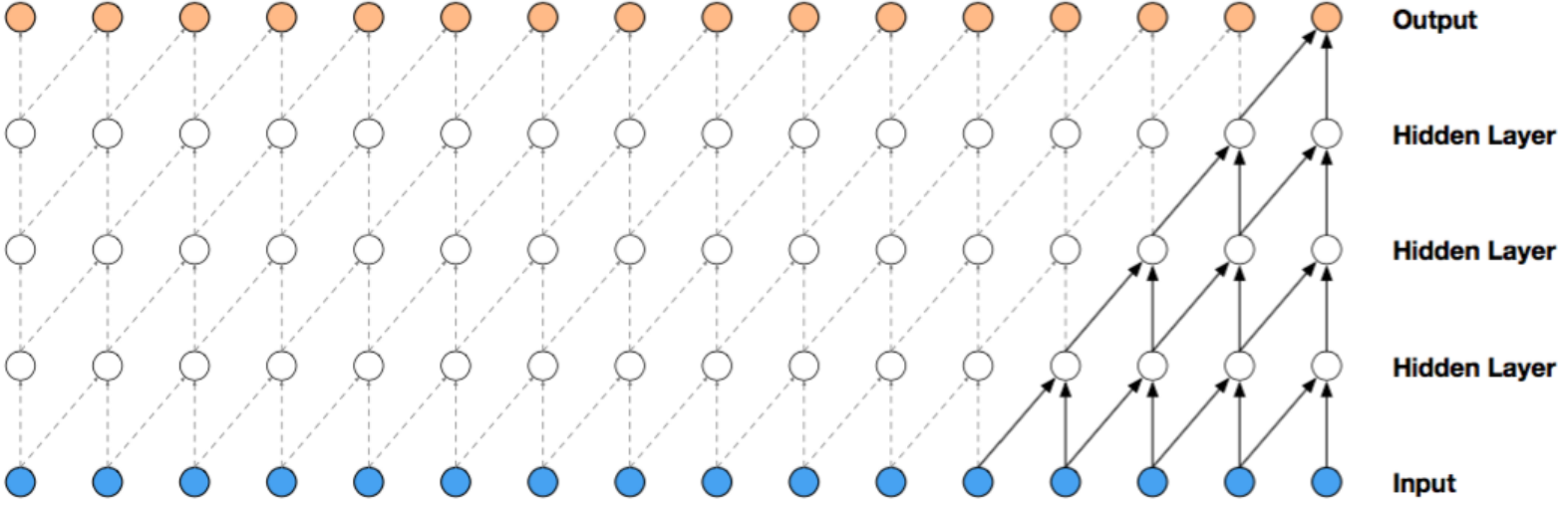
Image Captioning



Xu et. al., ICML 2015

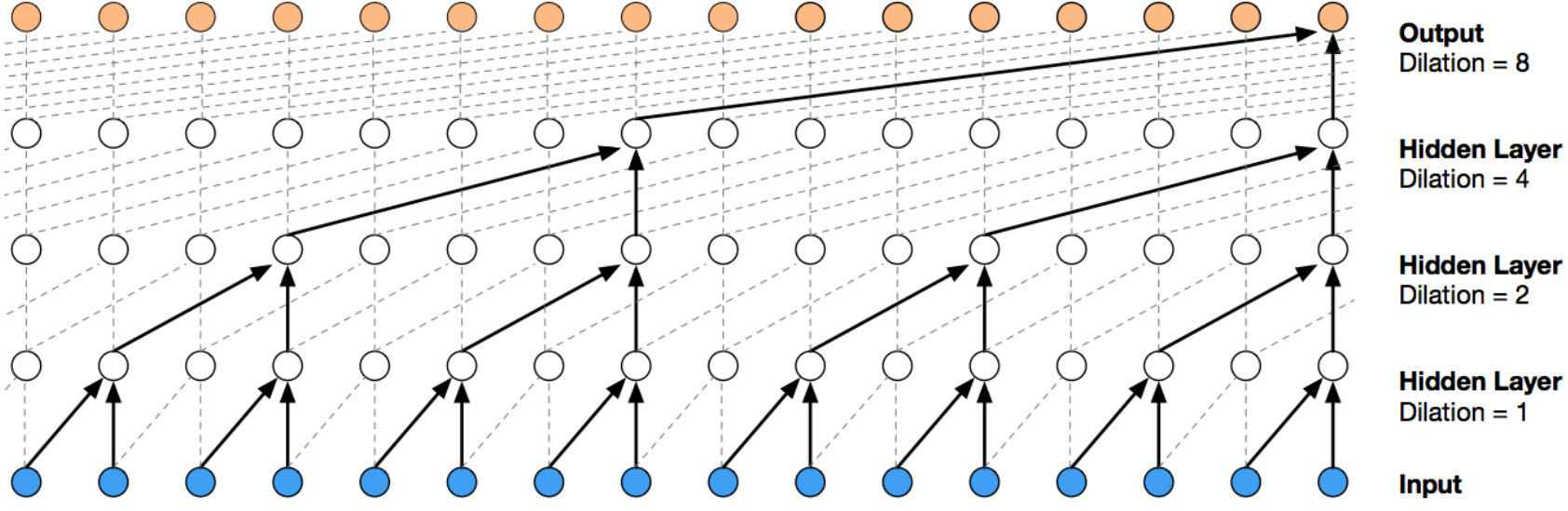
WaveNet

- Stack of causal convolutional layers



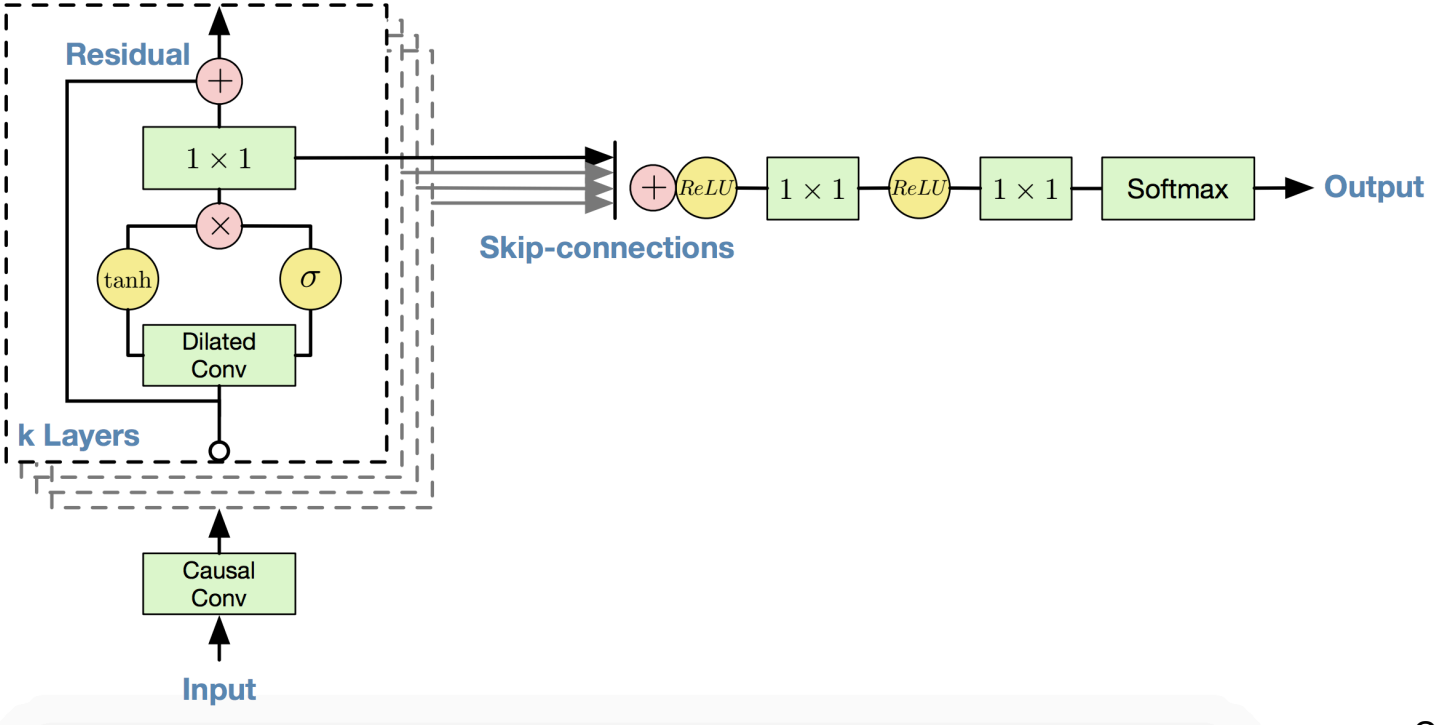
WaveNet

- Stack of dilated convolutional layers



WaveNet

- Residual block and overall architecture



Oord et. al., 2016

Thank you.