Foundations of Deep Learning and its applications in Natural Language Processing

Md Shad Akhtar

Research Scholar

AI-NLP-ML Group

Department of Computer Science & Engineering

Indian Institute of Technology Patna

Email: shad.akhtar@gmail.com, shad.pcs15@iitp.ac.in

Web: https://iitp.ac.in/~shad.pcs15/

Outline

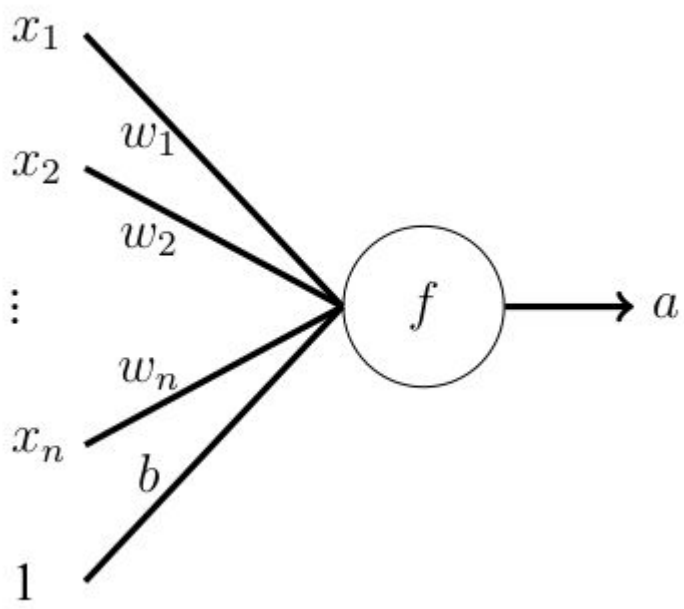
- Neural Network
- Deep Learning
  - Convolutional Neural Network
  - Recurrent Neural Network (RNN)
    - Long Short Term Memory (LSTM)
    - Gated Recurrent Unit (GRU)
  - Attention Mechanism
- Natural Language Processing
  - POS Tag
  - NER
  - Sentiment Analysis
  - Machine Translation
  - Question - Answering

Artificial Neural Network

Artificial Neural Network [Rosenblatt, 1957]

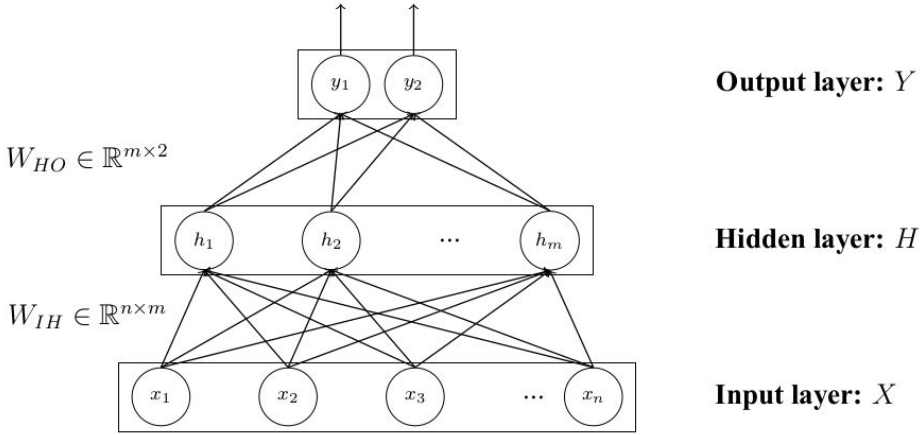
- Network of artificial neurons (or simply neuron)
- Neuron:
  - Receives inputs from inlinks
  - Computes the weighted sum
  - Performs an operation (activation function: f)
  - Forwards the output on its outlink.

$$a = f(\sum_{i}^{n} w_i x_i + b)$$



Multi-layer Perceptron or Feed-forward Network

- A layered architecture
  - No restriction of number of layers
  - Each layer can have any number of neurons (except input and output layer)
- Two constraints
  - Each neuron at a layer l must have a link with every other neurons at the upper layer $l + 1$
  - All the neurons at the same layer must be disconnected with each other.



Training a neural network and Backpropagation*

- Weights of a neural network is a learnable parameter.
- Given an input and output tuple <x, y>, i.e., a training set, the neural network learns a weight W that maps the input x to output y.

Algorithm 1 Backpropagation

Input: Training set = $[X, Y]$
Output: Weight W

Initialize network N with random weights W.
repeat
  For each train sample $< x, y > \in [X, Y]$ **do**
    $o \leftarrow \text{Forward}(x, N, W)$
    $e \leftarrow \text{ComputeError}(y, o)$
    For each layer $l \in N$ **do**
      Adjust weight W to compensate e.
    end for
  end for
until termination condition is reached.

* Rumelhart, Hinton and Williams, 1986

Few key terms..

- Layers
  - *Input, Output and Hidden*
- Activation functions
  - *Sigmoid, Tanh, Relu*
- Softmax
- Weight matrices
  - *Input → Hidden, Hidden → Hidden, Hidden → Output*
- Backpropagation
  - Optimizers
    - Gradient Descent (GD), Stochastic Gradient Descent (SGD), Adam etc.
  - Error (Loss) functions
    - MSE, Cross-Entropy etc.
  - Gradient of error
  - Passes: Forward pass and Backward pass

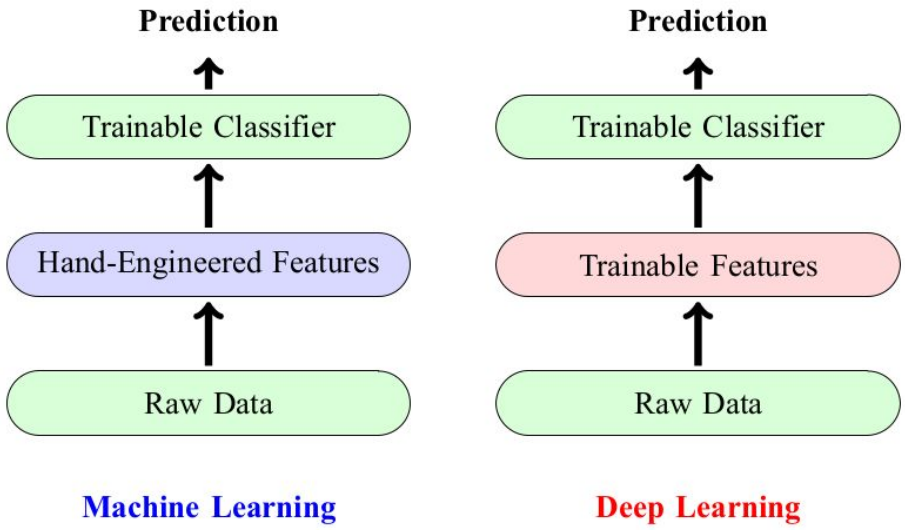
Deep Learning
or
Deep Neural Network

History of Neural Network and Deep learning

- Perceptron: Rosenblatt, 1957
- Backpropagation: Rumelhart, Hinton and Williams, 1986
  - Theoretically, a neural network can have any number of hidden layers.
  - But, in practice, it rarely had more than one layer hidden layers.
    - Computational issue: Limited computing power
    - Algorithmical issues: Vanishing gradient and Exploding gradient.
- Beginning of Deep learning: Late 1990's and early 2000's
  - **Solutions:**
    - Computational issue
      - Advance computing powers such as GPUs, TPUs
    - Algorithmical issues
      - Pre-training (e.g., AutoEncoder, RBM)
      - Better architectures (e.g., LSTM)
      - Better activation functions (e.g., Relu)

Deep Learning vs Machine Learning Paradigm

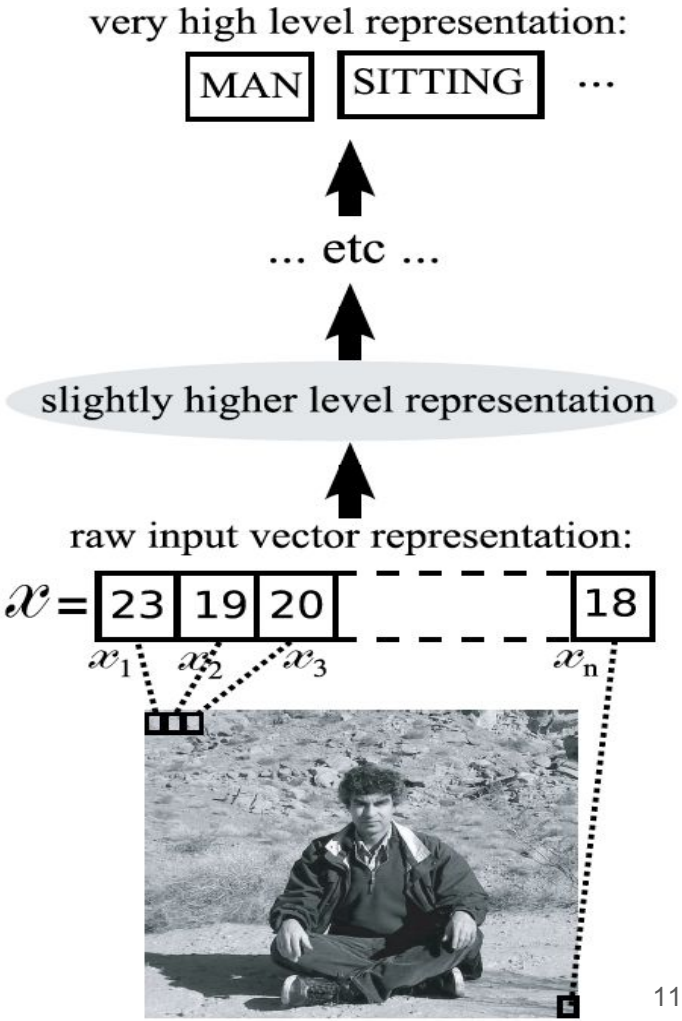
- The main advantage of deep learning based approaches is the trainable features, i.e., it extracts relevant features, on its own, during training.
- Requires minimal human intervention.



Why Deep Learning?

- Recall, artificial neural network tries to mimic the functionality of a brain.
- In brain, computations happen in layers.

- View of representation
  - As we go up in the network, we get high-level representations ⇒ Assists in performing more complex tasks.



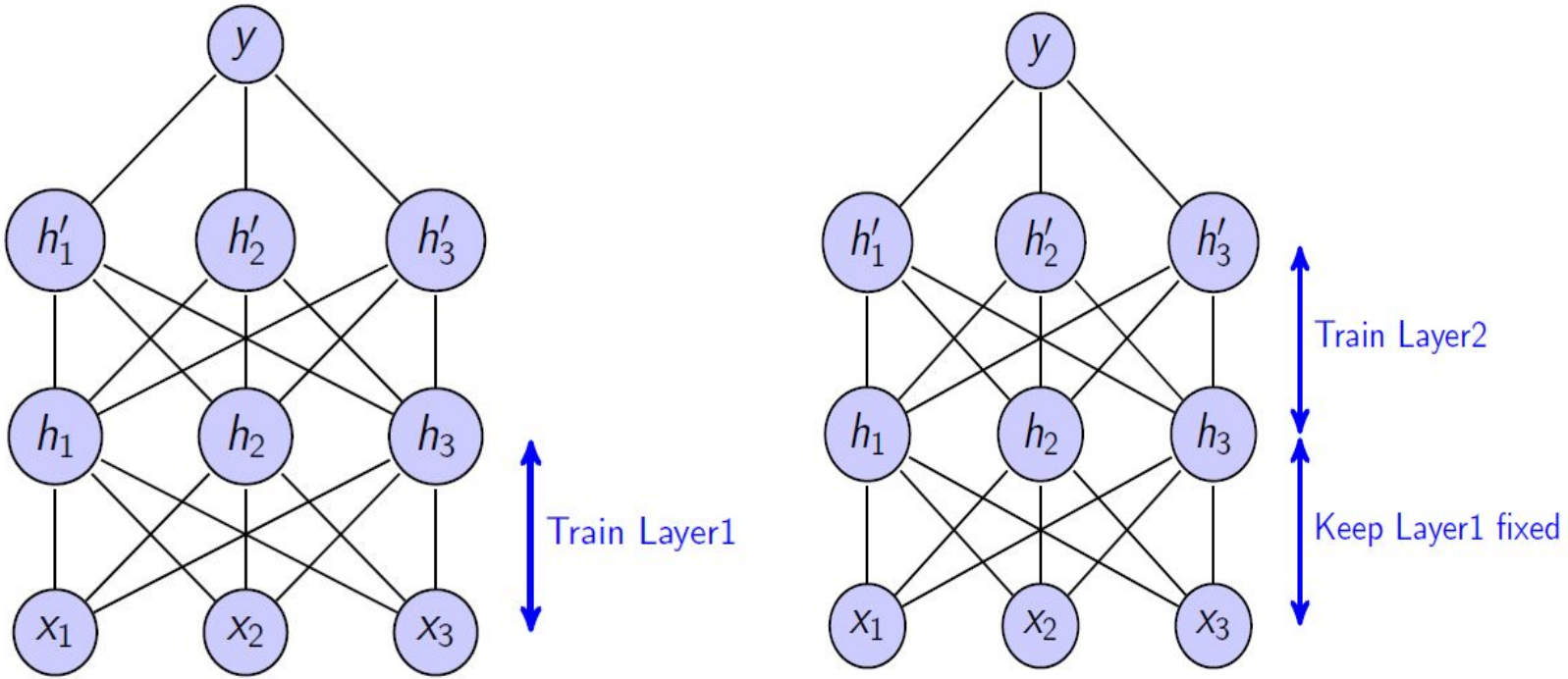
Why Deep Architectures were hard to train?

- General weight-updation rule

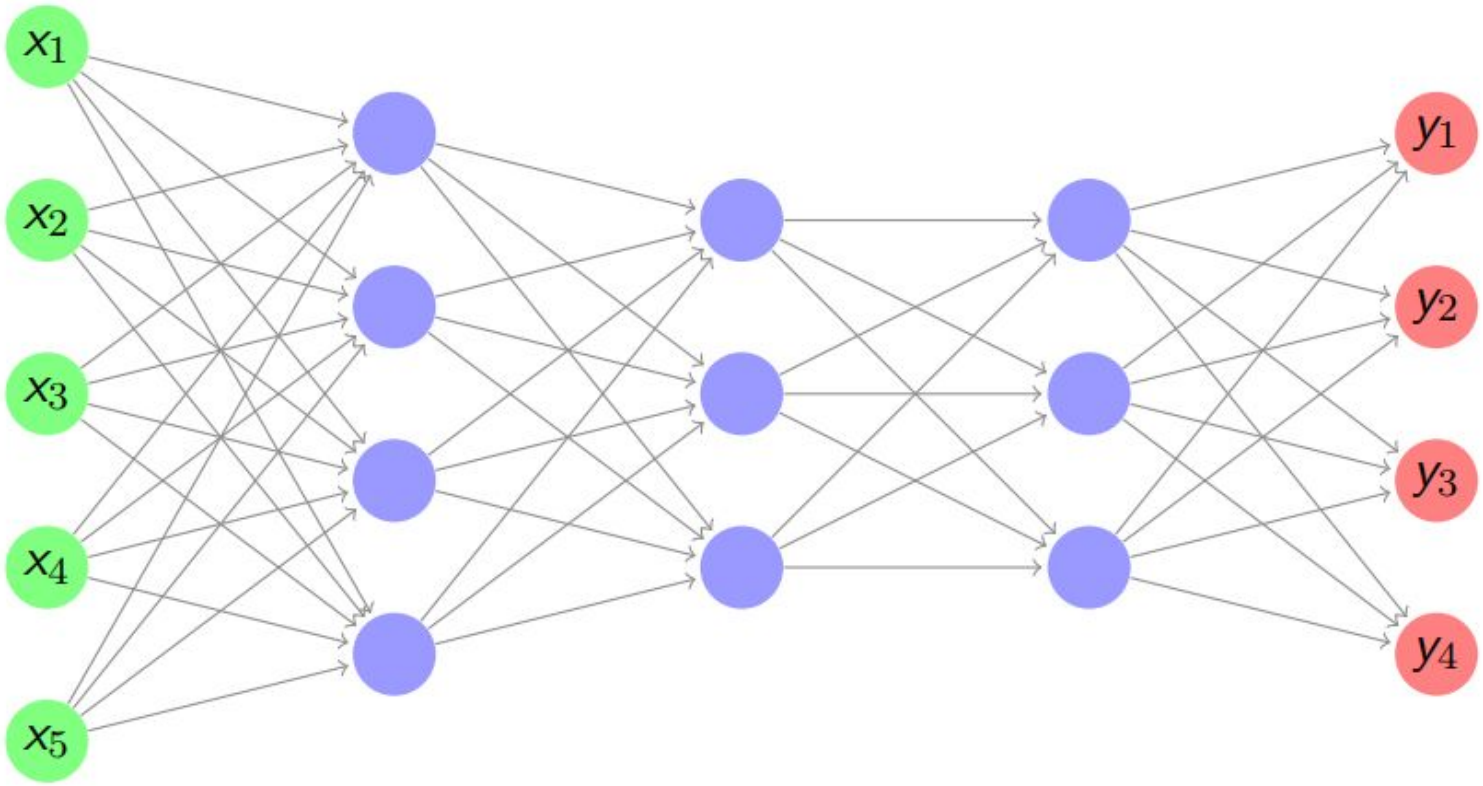
$$
\begin{aligned}
w_{ij} &= w_{ij} + \Delta w_{ij} \\
\Delta w_{ij} &= -\eta \frac{\partial E}{\partial w_{ij}} \\
&= -\eta \delta_j o_i \\
\delta_j &= \delta_k w_{jk} \sigma'
\end{aligned}
$$

- For lower-layers in deep architecture
  - δ_j will vanish, if it is less than 1
  - δ_j will explode, if it is more than 1

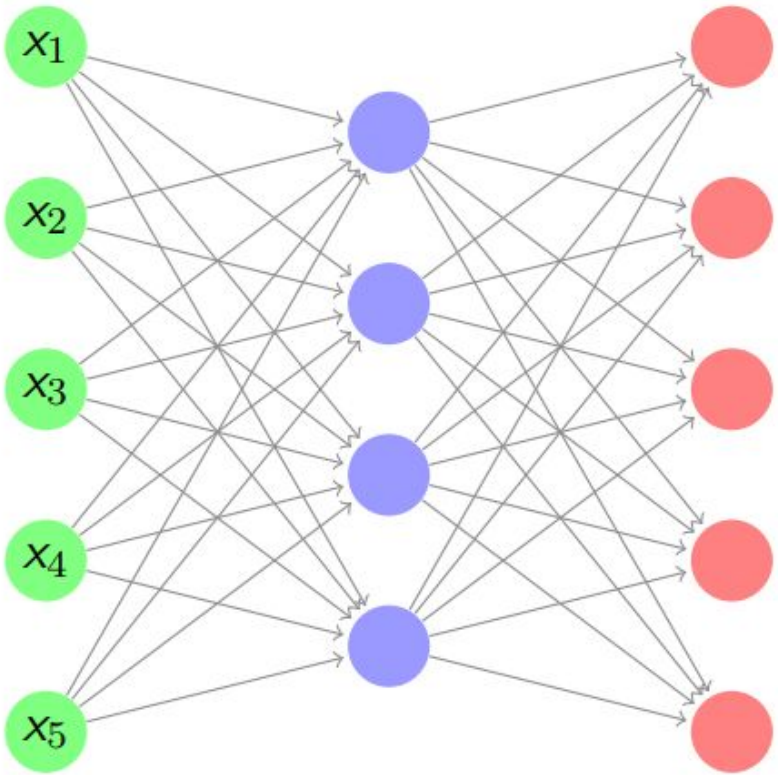
Layer-wise pre-training



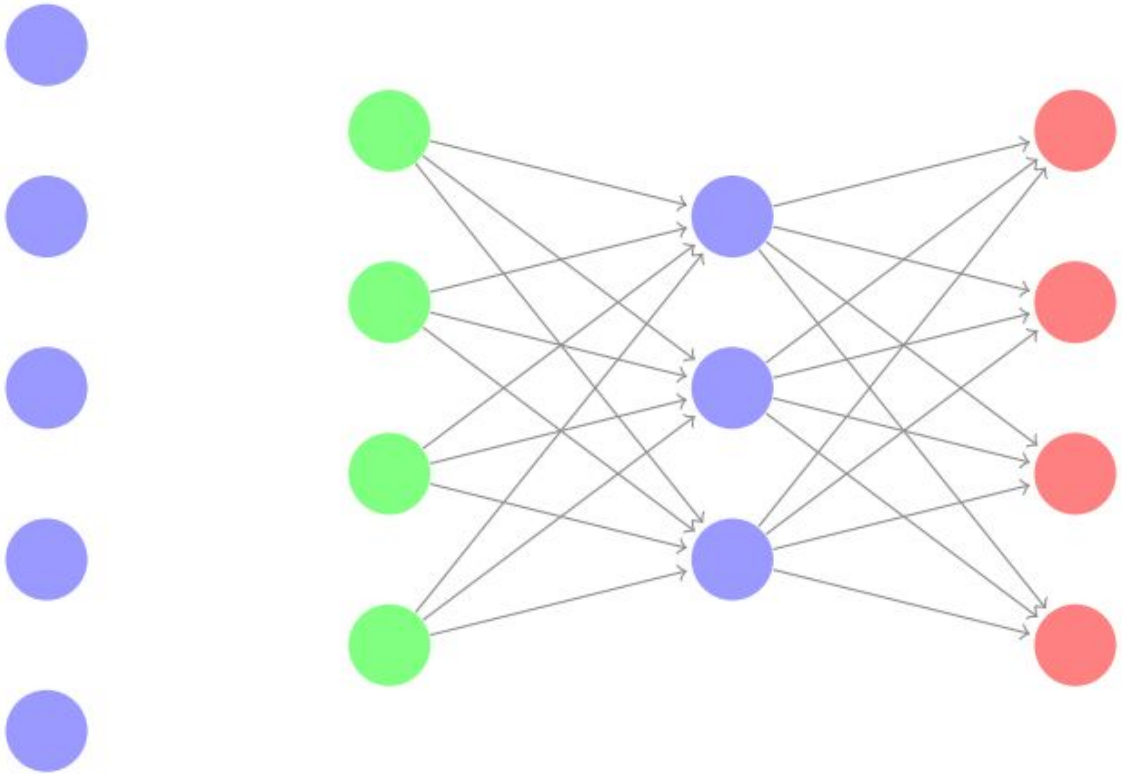
AutoEncoder



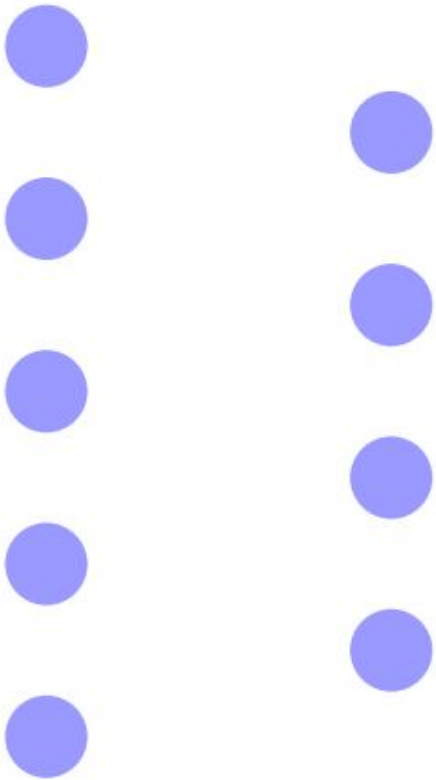
AutoEncoder: Layer 1

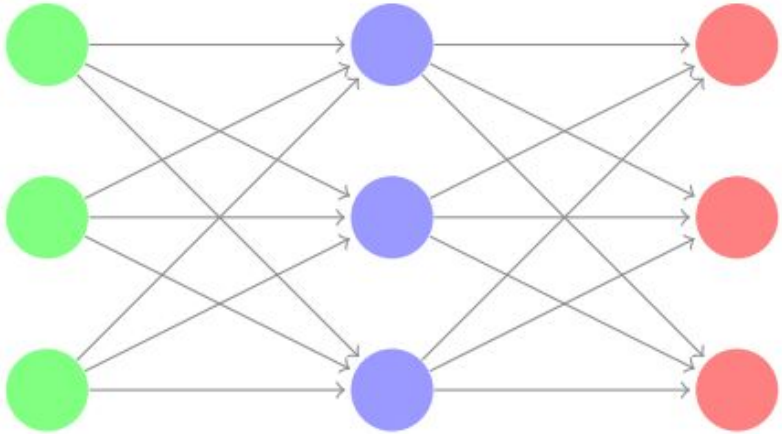


AutoEncoder: Layer 2

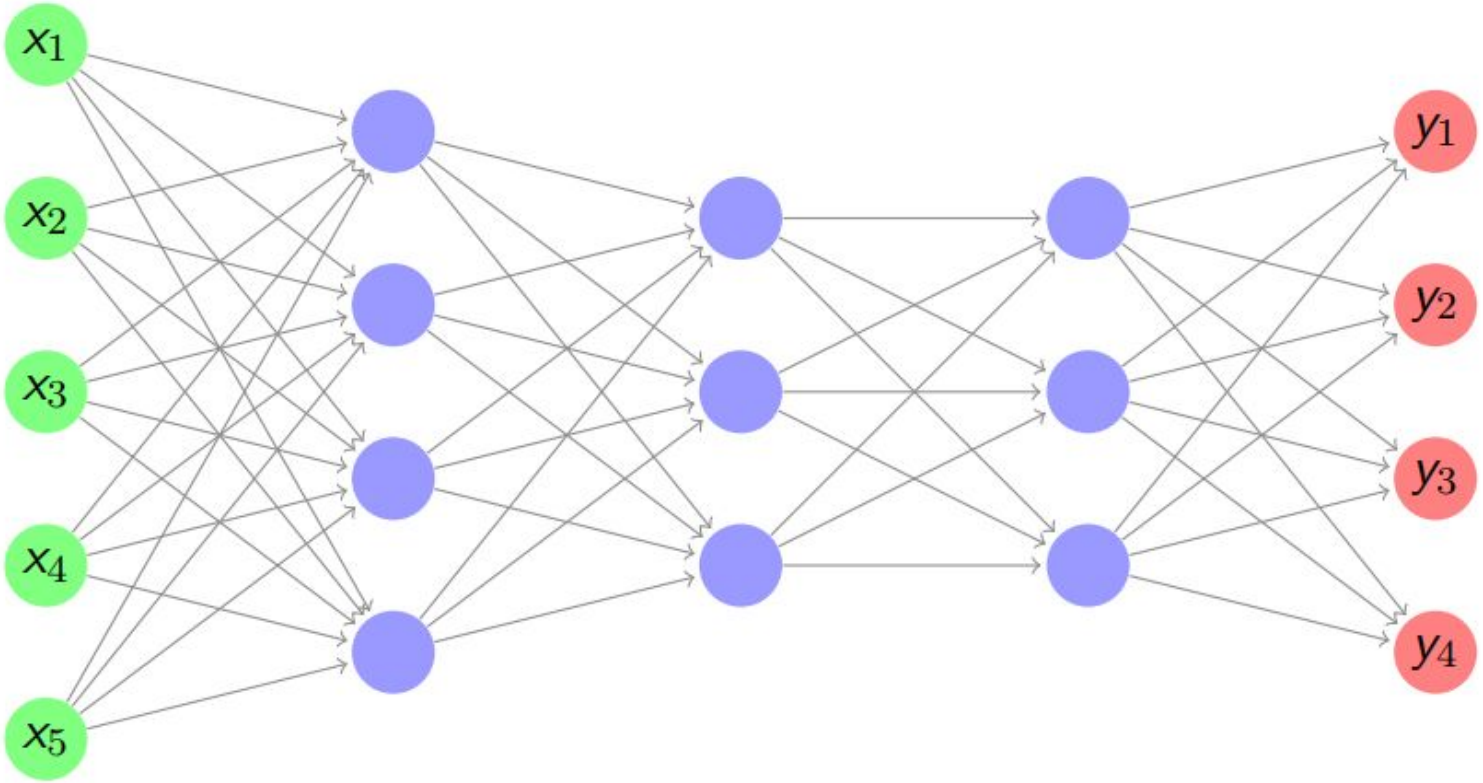


AutoEncoder: Layer 3





AutoEncoder: Pre-trained network



Deep Learning Architectures

- Convolutional neural network (CNN)
  - Aims to extract the local spatial features
- Recurrent neural network (RNN)
  - Exploits the sequential information of a sentence (sentence is a sequence of words).

Convolutional Neural Network

CNN

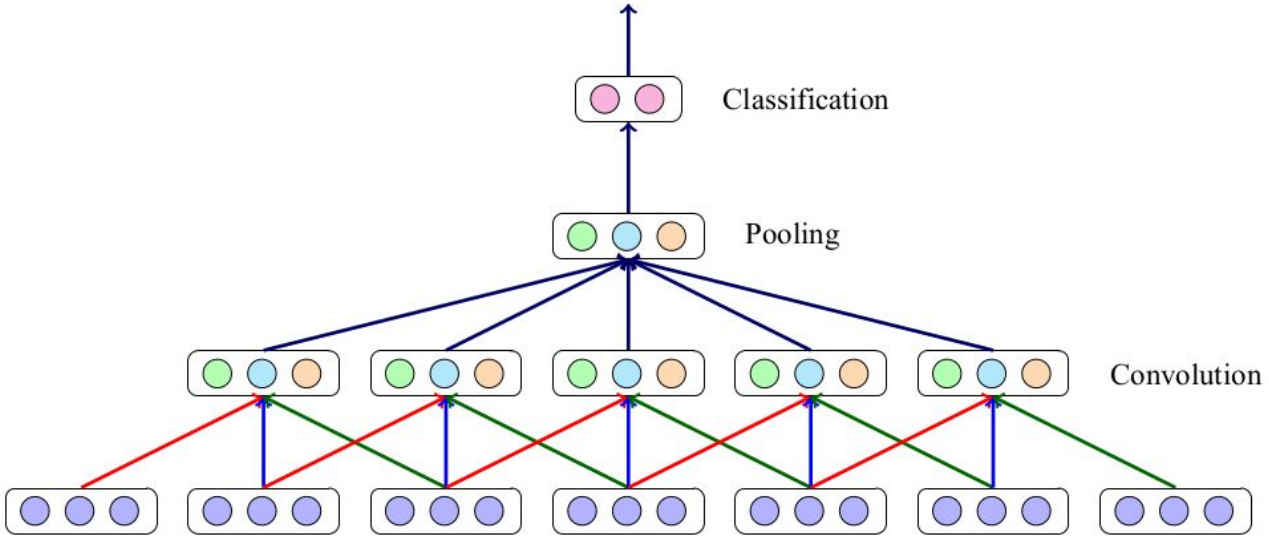
- A CNN consists of a series (≥ 1) of convolution layer and pooling layer.
- Convolutional operation extracts the feature representations from the input data.
  - Shares the convolution filters over different spatial locations, in a quest of extracting location-invariant features in the input.
  - Shape and weights of the convolution filter determine the features to be extracted from the input data.
  - In general, multiple filters of different shapes are used to ensure the diversity in the extracted features.
- Pooling operation extracts the most relevant features from the convoluted features. Similar to downsampling in image-processing.

CNN

For an input $X \in \mathbb{R}^{n \times d}$ and filter $F \in \mathbb{R}^{m \times d}$,

$$conv(x_i) = \sum_{j=1}^{m} \overrightarrow{f_j} \cdot \overrightarrow{x}_{i+j-1} \qquad \forall i \in [1, 2, ..., n-m+1]$$

$$Conv(X) = [conv(x_1), conv(x_2), ..., conv(x_{n-m+1})]$$

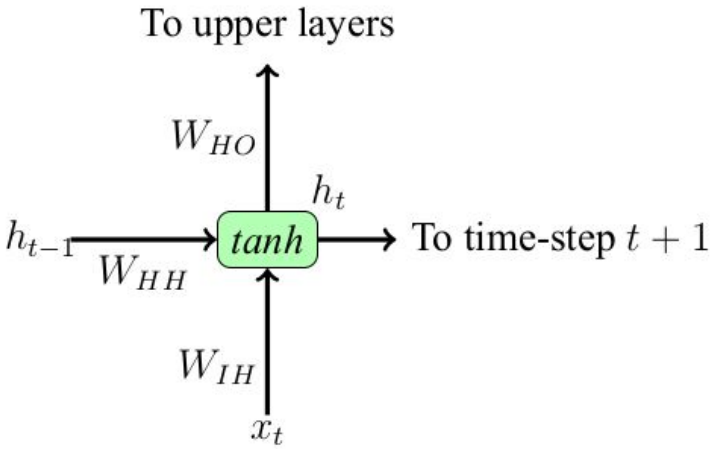


Recurrent Neural Network

Recurrent Neural Network (RNN)

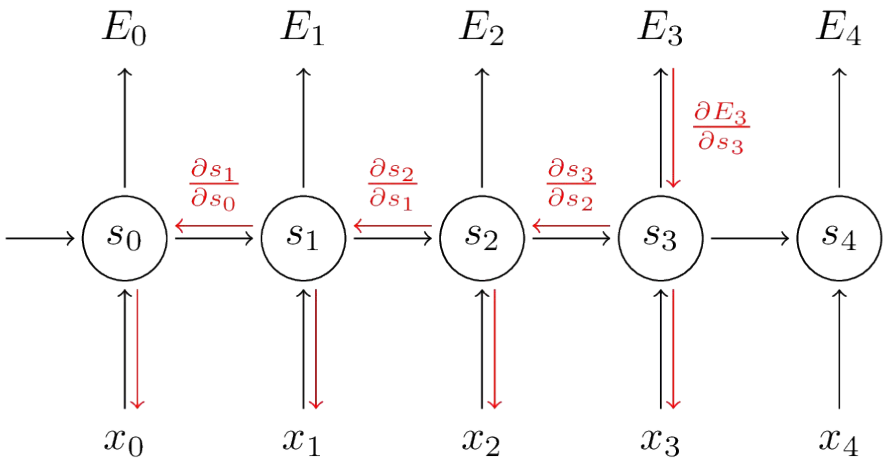
- A neural network with feedback connections
- Enable networks to do temporal processing
- Good at learning sequences
- Acts as memory unit

$$
\begin{aligned}
h_t &= tanh(W_{IH} \cdot x_t + W_{HH} \cdot h_{t-1} + b) \\
&= tanh([W_{IH}, W_{HH}] \cdot [x_t, h_{t-1}] + b)
\end{aligned}
$$



How to train RNNs?

- Typical FFN
  - Backpropagation algorithm
- RNNs
  - A variant of backpropagation algorithm namely Back-Propagation Through Time (BPTT).



Visualization of RNN through Feed-Forward Neural Network

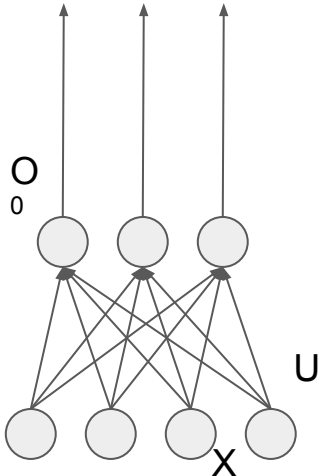
Problem, Data and Network Architecture

- Problem:
  - I/p sequence (X) : $X^0, X^1, \ldots, X^T$
  - O/p sequence (O) : $O^0, O^1, \ldots, O^T$

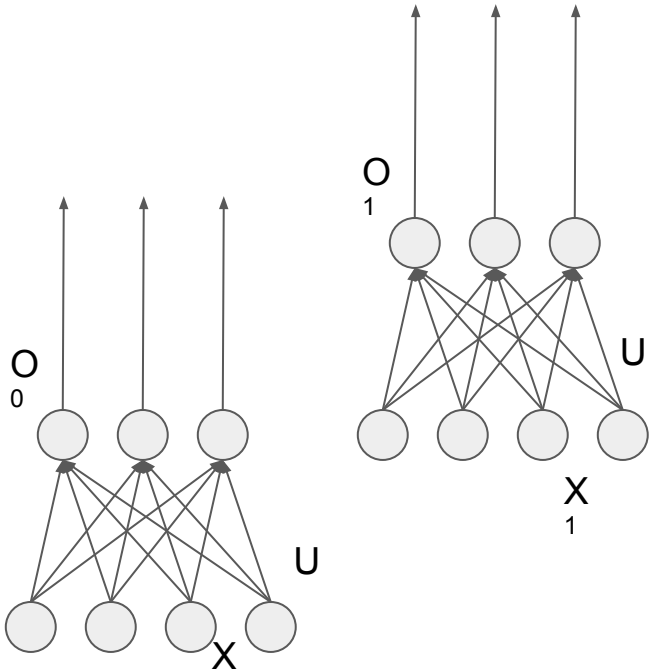
- Representation of data:
  - I/p dimension : 4
    - $X^0 \rightarrow 0\ 1\ 1\ 0$
  - O/p dimension : 3
    - $O^0 \rightarrow 0\ 0\ 1$

- Network Architecture
  - Number of neurons at I/p layer : 4
  - Number of neurons at O/p layer : 3
  - Do we need hidden layers?
    - If yes, number of neurons at each hidden layers

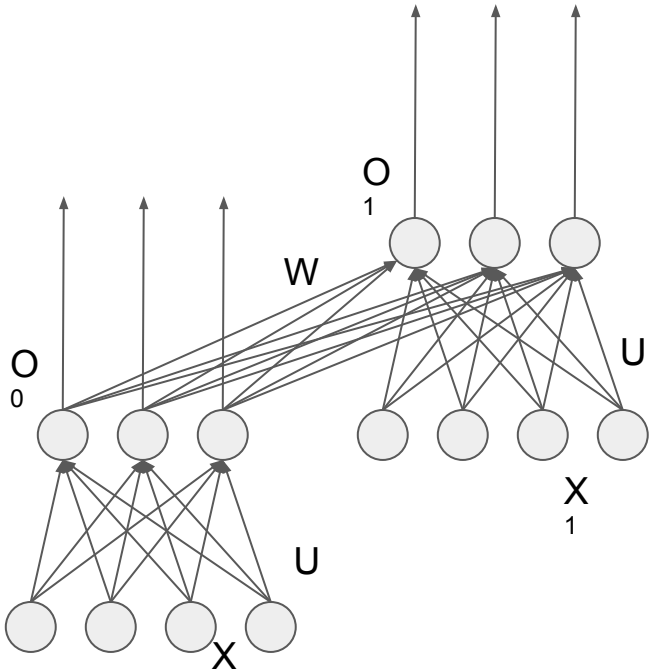
Network @ $t = 0$



Network @ *t* = 1



Network @ *t* = 1



Network @ t = 1

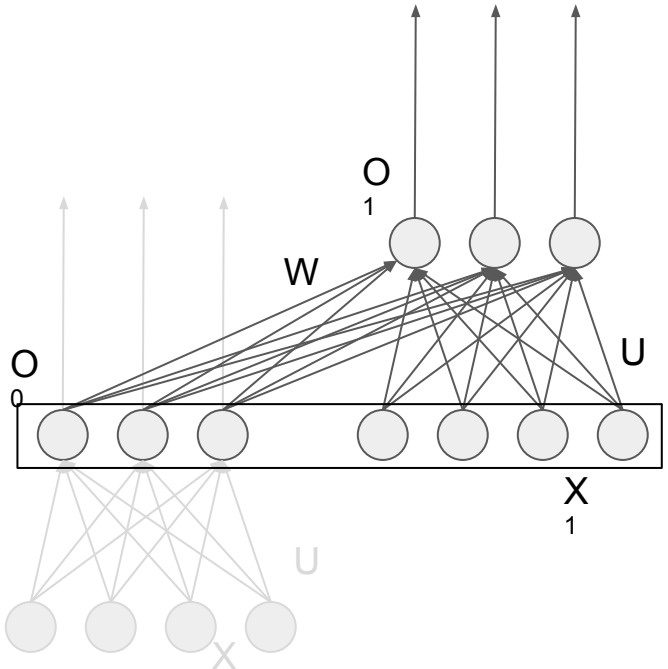
t

1

0

$O^1 = f(W.O^0 + U.X^1)$

$= f([W, U] . [O^0, x^1])$



Network @ $t = 2$

t

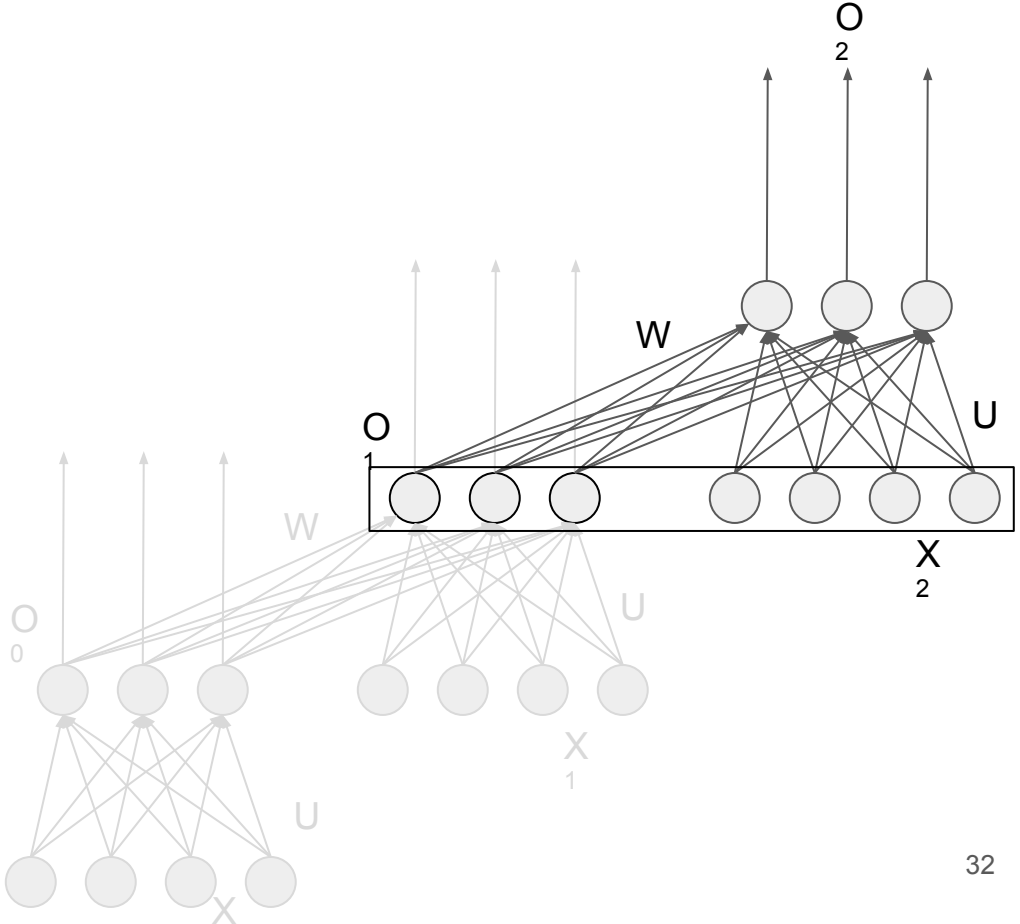
2

1

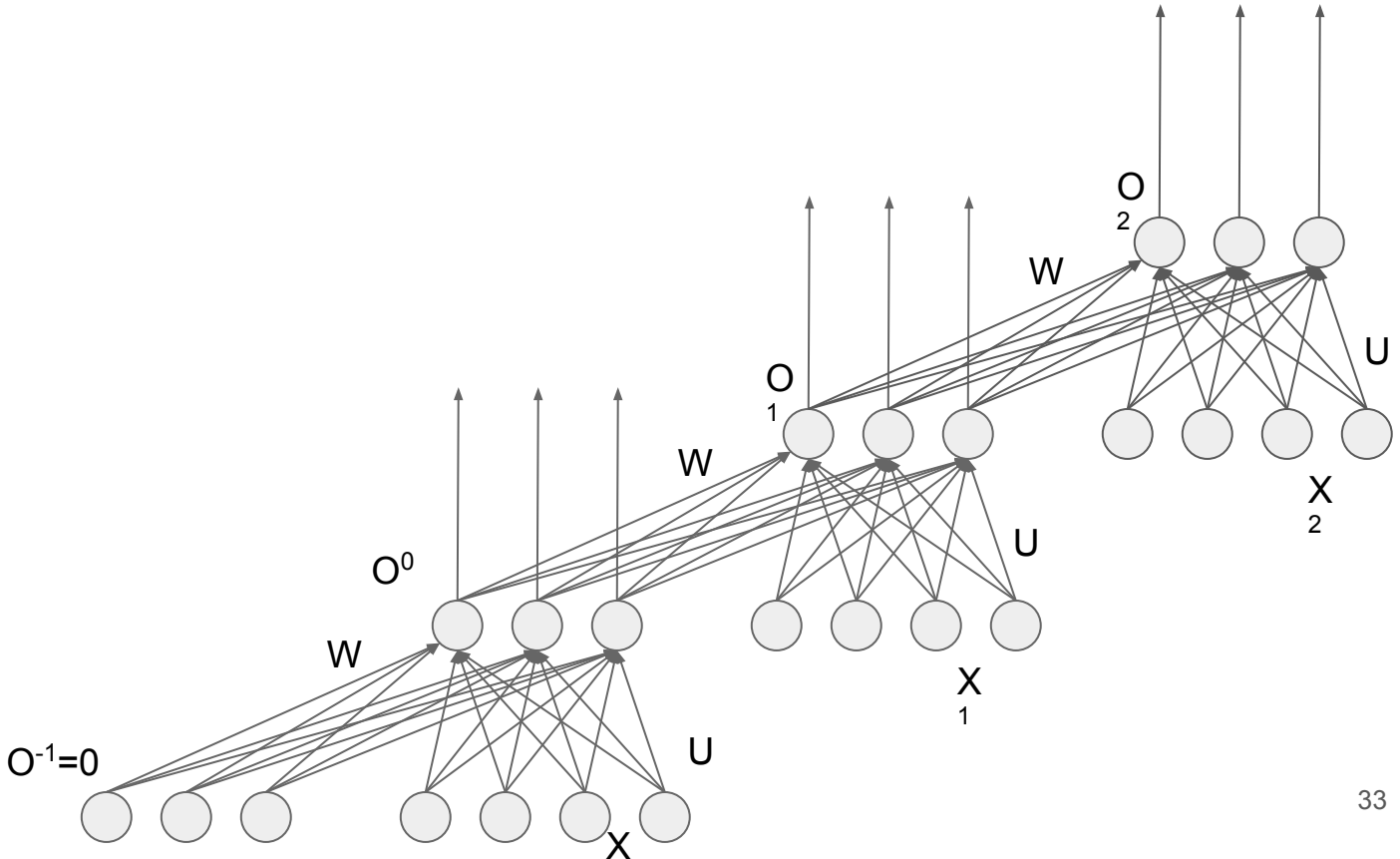
0

$O^2 = f(W.O^1 + U.X^2)$

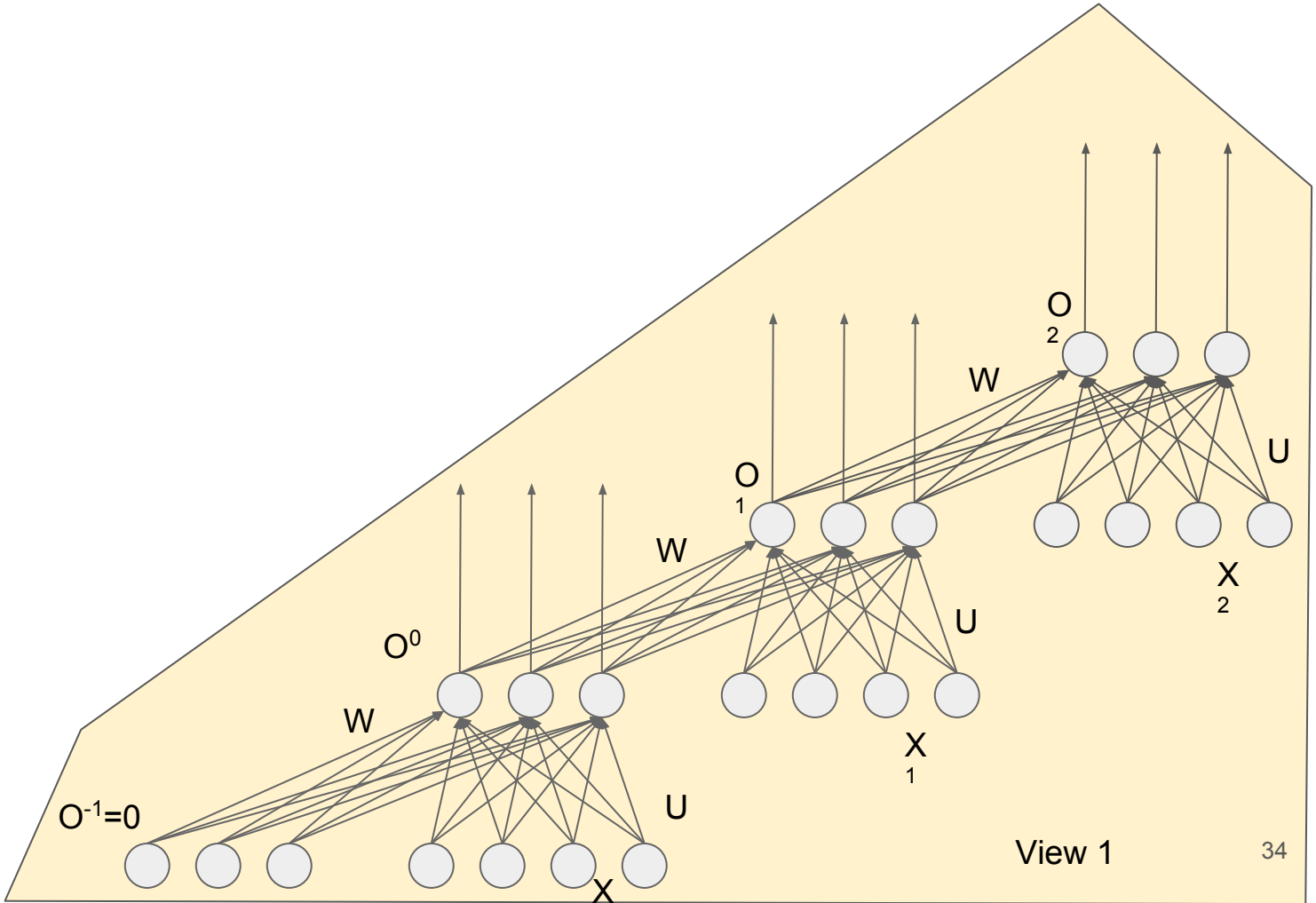
$= f([W, U] . [O^1, x^2])$



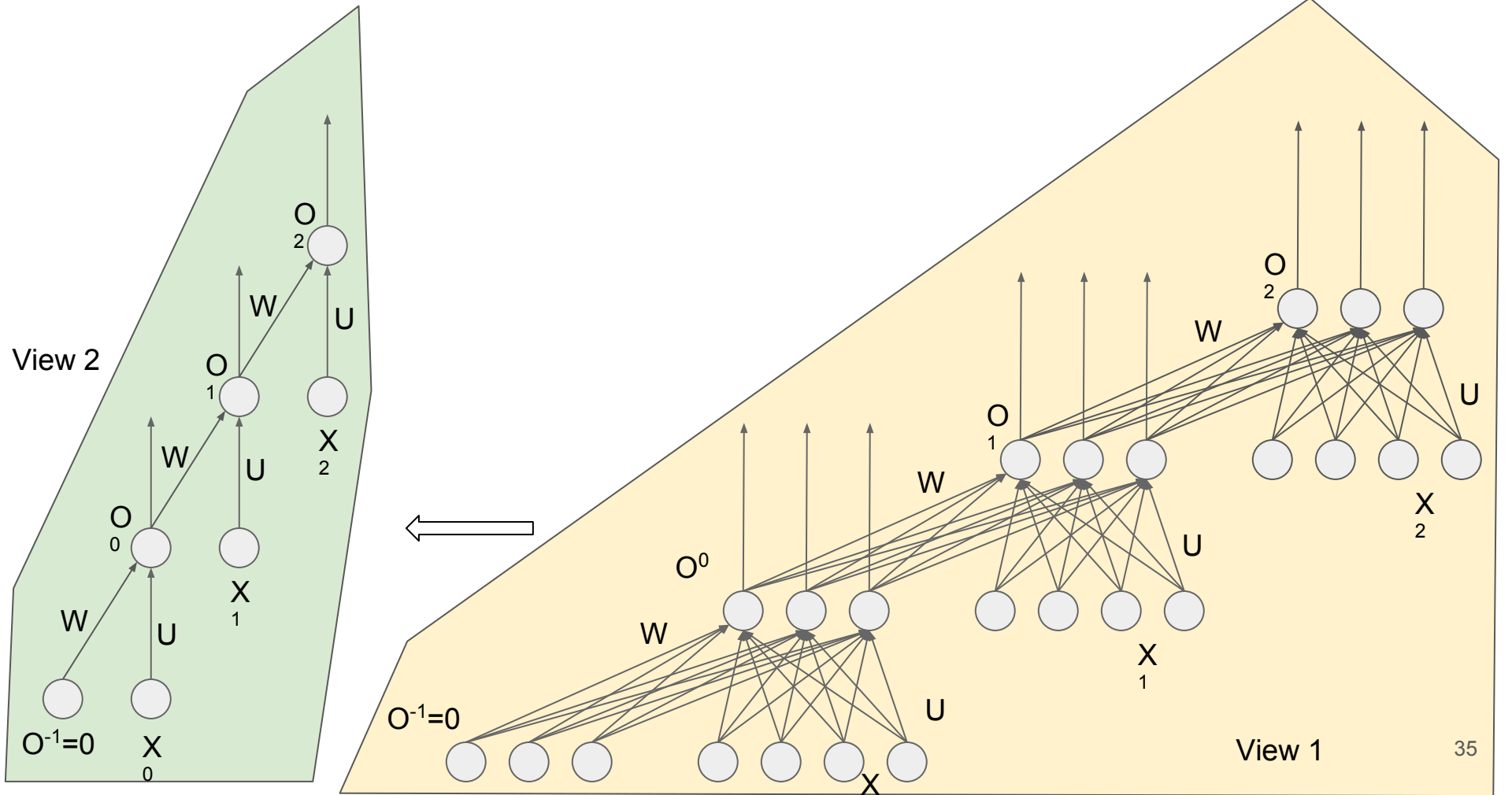
Complete Network



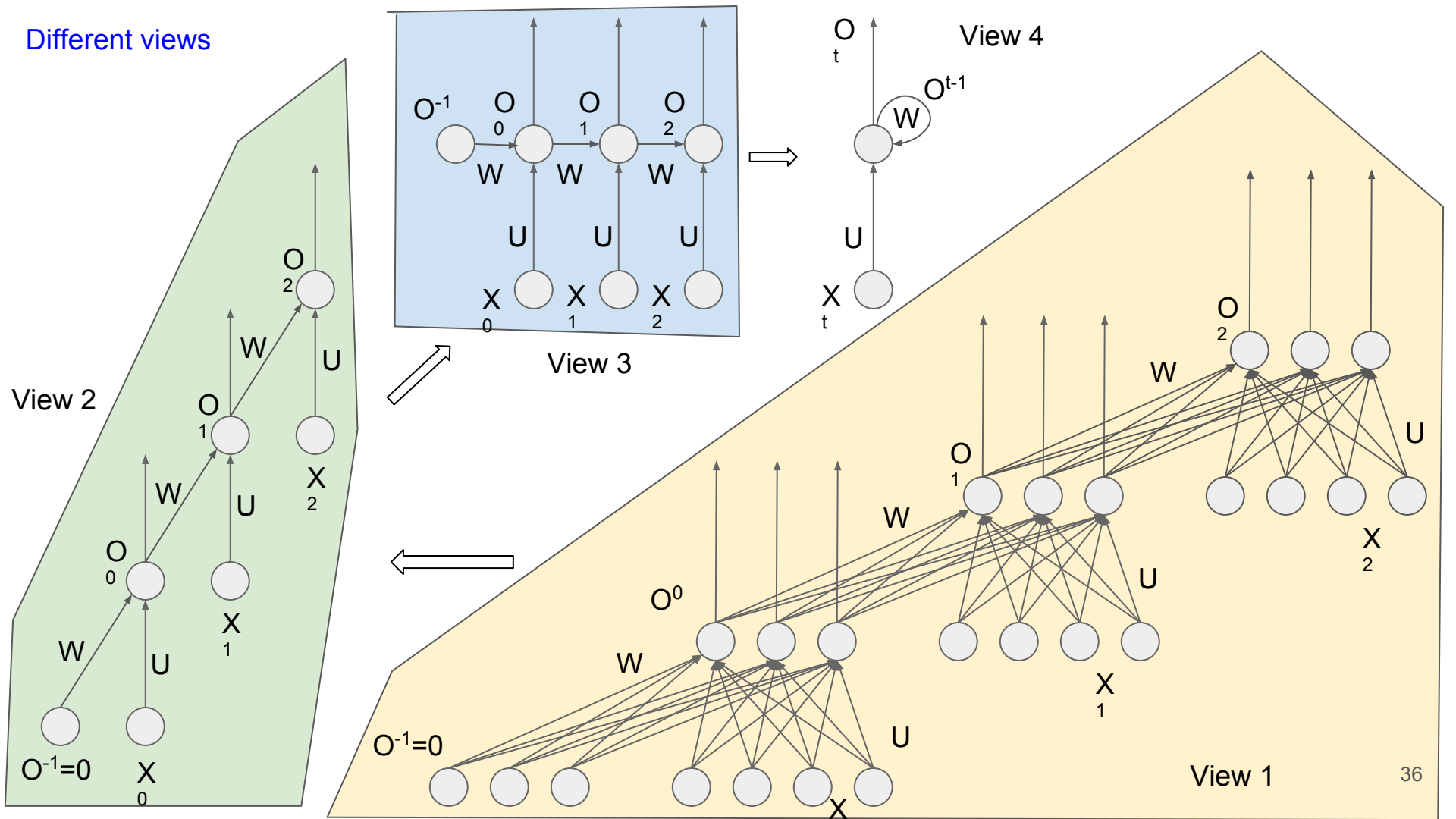
Different views



Different views



Different views



Different configurations of RNNs

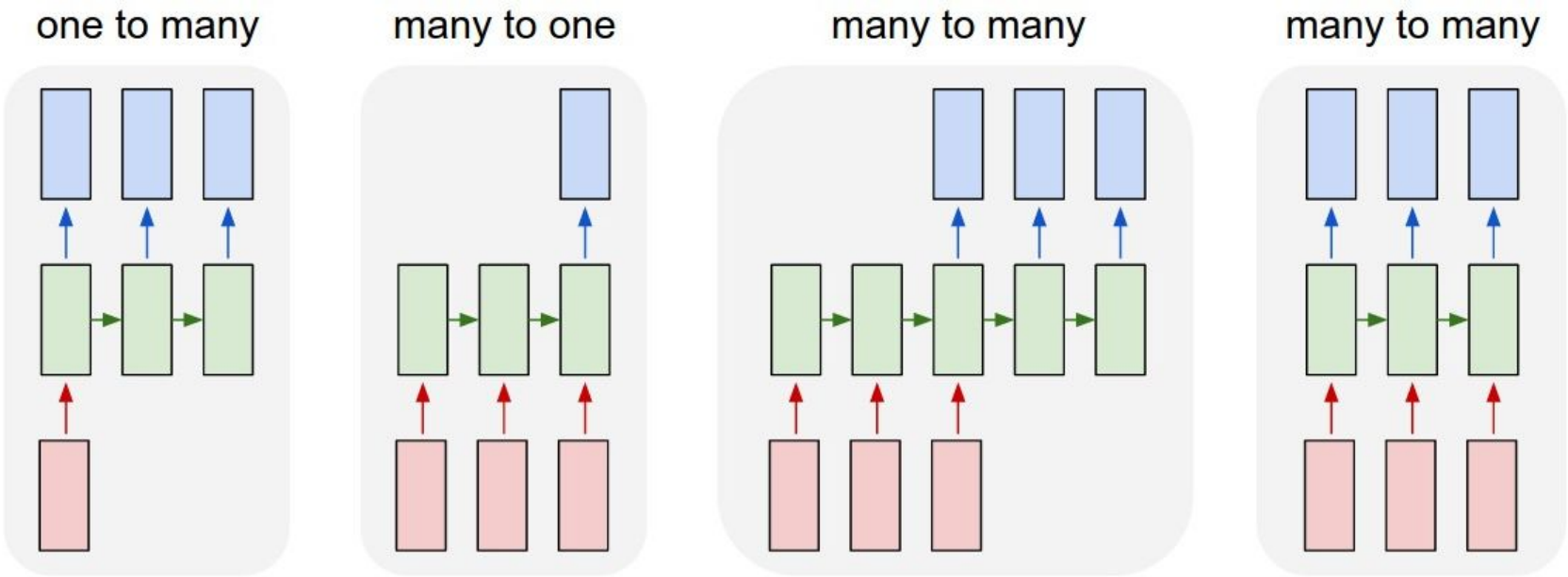


Image Captioning

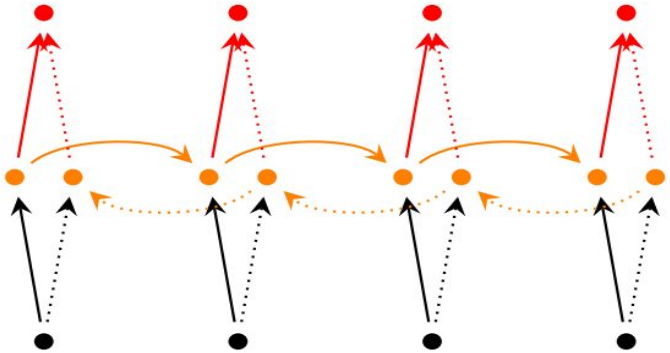
Sentiment Analysis

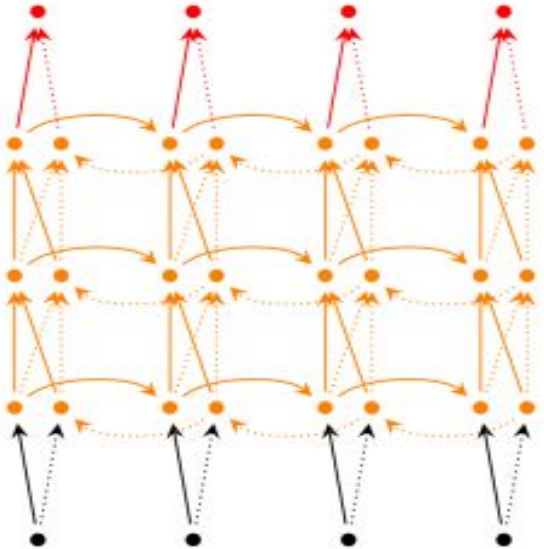
Machine Translation

Language modelling

RNN extensions

- Bidirectional RNN
- Deep (Bi-directional) RNN

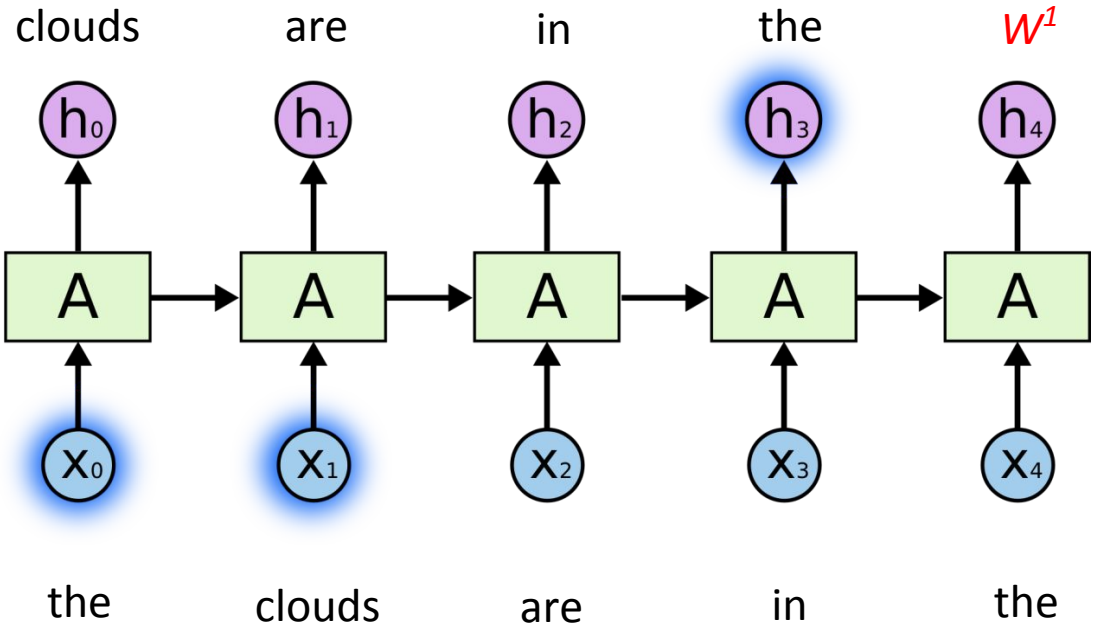




Problems with RNNs

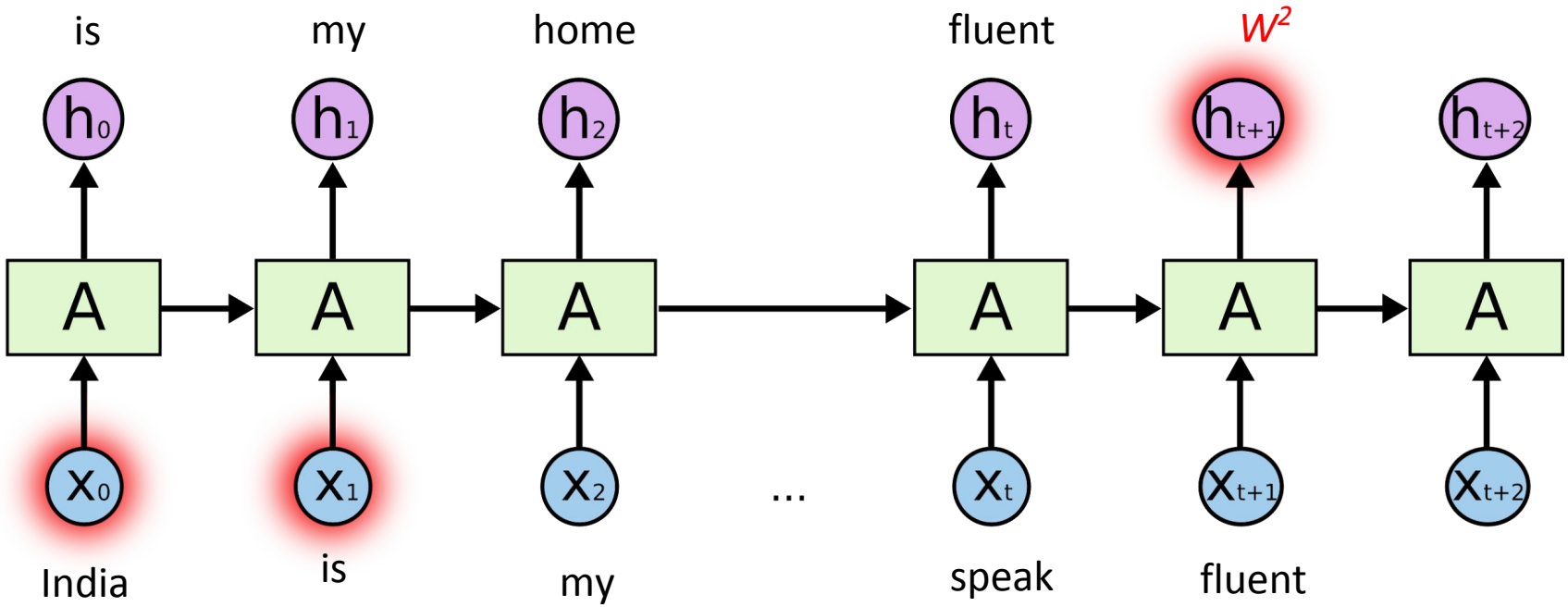
Language modelling: Example - 1

- “the clouds are in the *sky*”



Language modelling: Example - 2

- "India is my home country. I can speak fluent *Hindi*."



Vanishing/Exploding gradients

- Cue word for the prediction
  - Example 1: ***sky*** → ***clouds*** [3 units apart]
  - Example 2: ***hindi*** → ***India*** [9 units apart]

- As the sequence length increases, it becomes hard for RNNs to learn "long-term dependencies."
  - **Vanishing gradients:** If weights are small, gradient shrinks exponentially. Network stops learning.
  - **Exploding gradients:** If weights are large, gradient grows exponentially. Weights fluctuate and become unstable.

Long Short Term Memory (LSTM) [Hochreiter & Schmidhuber (1997)]

- A variant of simple RNN (Vanilla RNN)
- Capable of learning long dependencies.
- Regulates information flow from recurrent units.

An LSTM cell

- Cell state c_t (blue arrow), hidden state h_t (green arrow) and input x_t (red arrow)
- Three gates
  - Forget (Red-dotted box)

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t])$$

$$c_t = f_t \otimes c_{t-1}$$

  - Input (Green-dotted box)

$$z_t = tanh(W_z \cdot [h_{t-1}, x_t])$$

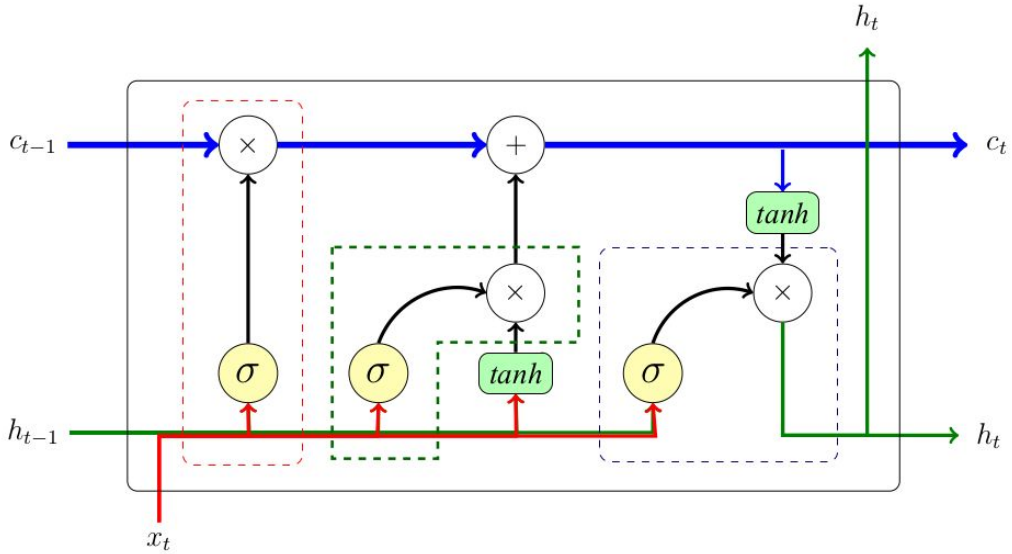
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t])$$

$$c_t = c_t + (i_t \otimes z_t)$$

  - Output (Blue-dotted box)

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t])$$

$$h_t = o_t \otimes tanh(c_t)$$

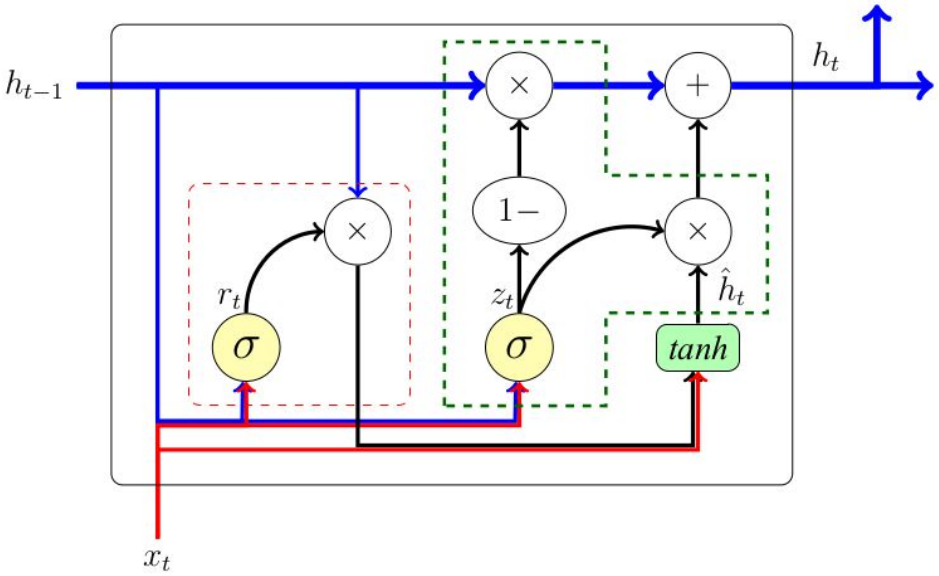


Gated Recurrent Unit (GRU) [Cho et al. (2014)]

- A variant of simple RNN (Vanilla RNN)
- Similar to LSTM
  - Whatever LSTM can do GRU can also do.
- Differences
  - Cell state and hidden are merged together
  - Two gates
    - Reset gate - similar to forget
    - Update gate - similar to input gate
  - No output gate
  - Cell/Hidden state is completely exposed to subsequent units.
- GRU needs fewer parameters to learn and is relatively efficient *w.r.t.* computation.

A GRU cell

$$
\begin{aligned}
z_t &= \sigma(W_z \cdot [h_{t-1}, x_t]) \\
r_t &= \sigma(W_r \cdot [h_{t-1}, x_t]) \\
\hat{h}_t &= tanh(W \cdot [r_t \otimes h_{t-1}, x_t]) \\
h_t &= (1 - z_t) \otimes h_{t-1} + z_t \otimes \hat{h}_t
\end{aligned}
$$



Application of DL methods for NLP tasks

Natural Language Processing (NLP)

- A subfield of artificial intelligence (AI) that deals with the human language processing in the textual form.
- Objective is to provide a platform for the human-machine interaction.
  a. The machine should be capable enough to understand the human language (e.g., English, Hindi etc.)
  b. Process it.
  c. Generate a response in human understandable language.

Problem	Query	Response
SA	I had a great day today!	Positive
QA	Who won the cricket WC in 2011?	India
MT	Be the change you want to see in the world.	वह बनें, जो आप दुनिया में बदलाव देखना चाहते हैं।

NLP hierarchy

- Like deep learning, NLP happens in layers!
- Each task receives features from its previous (lower-level) task, and process them to produce its own output and so on.

Pragmatics & Discourse	*Study of semantics in context.*
Semantics	*Meaning of the sentence.*
Parsing	*Syntactic structure of the sentence.*
Chunking	*Grouping of meaningful phrases.*
Part of speech tagging	*Grammatical classes.*
Morphology	*Study of word structure.*

Increasing
Complexity Of
Processing

NLP problems

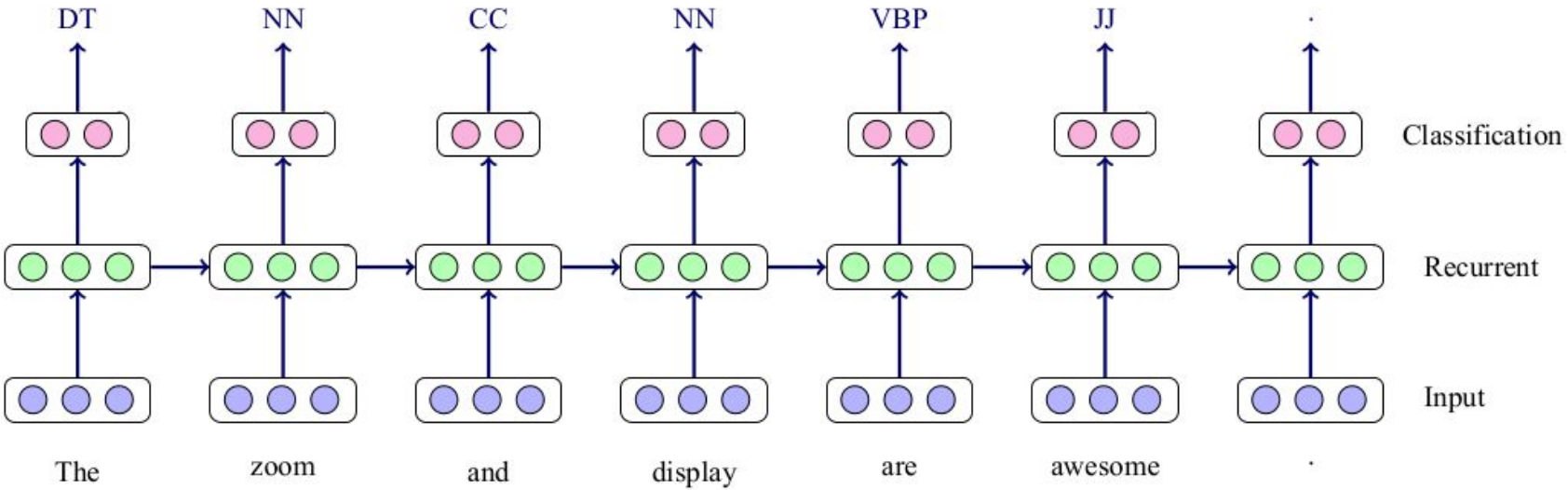
Problems	Paradigm
POS Tagging	Sequence Labelling
Named Entity Recognition	
Sentiment Analysis	Classification
Machine Translation	Sequence Transformation
Question Answering	
Summarization	

Sequence Labelling

RNN/LSTM/GRU for Sequence Labelling

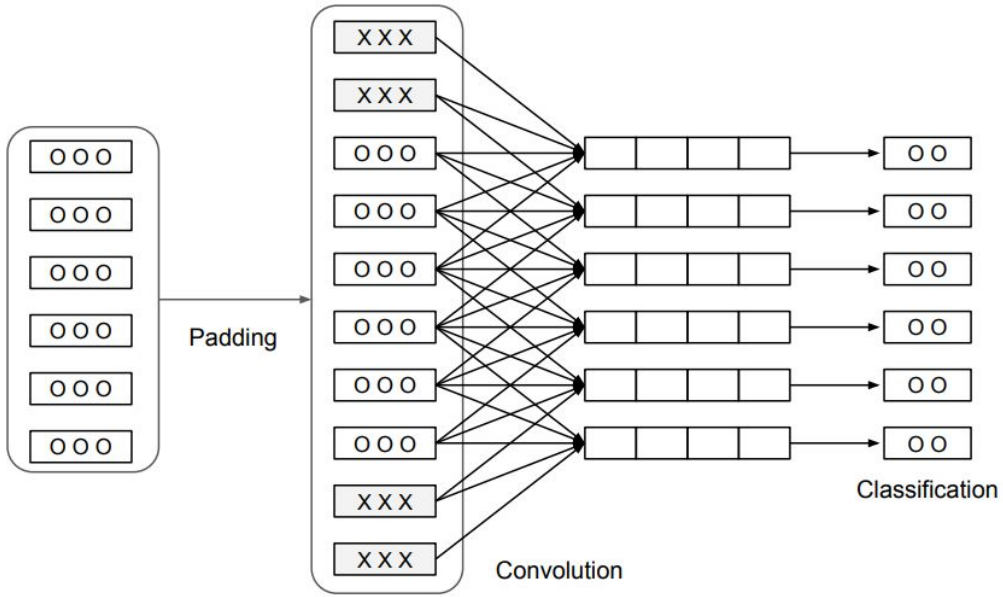
Part-of-speech tagging:

- Given a sentence X, tag each word its corresponding grammatical class.



CNN for Sequence Labelling

- Sentence matrix
- Pad sentence to ensure the sequence length
  - Pad length = filter_size - 1
  - Evenly distribute padding at the start and end of the sequence.
- Apply Convolution filters
- Classification

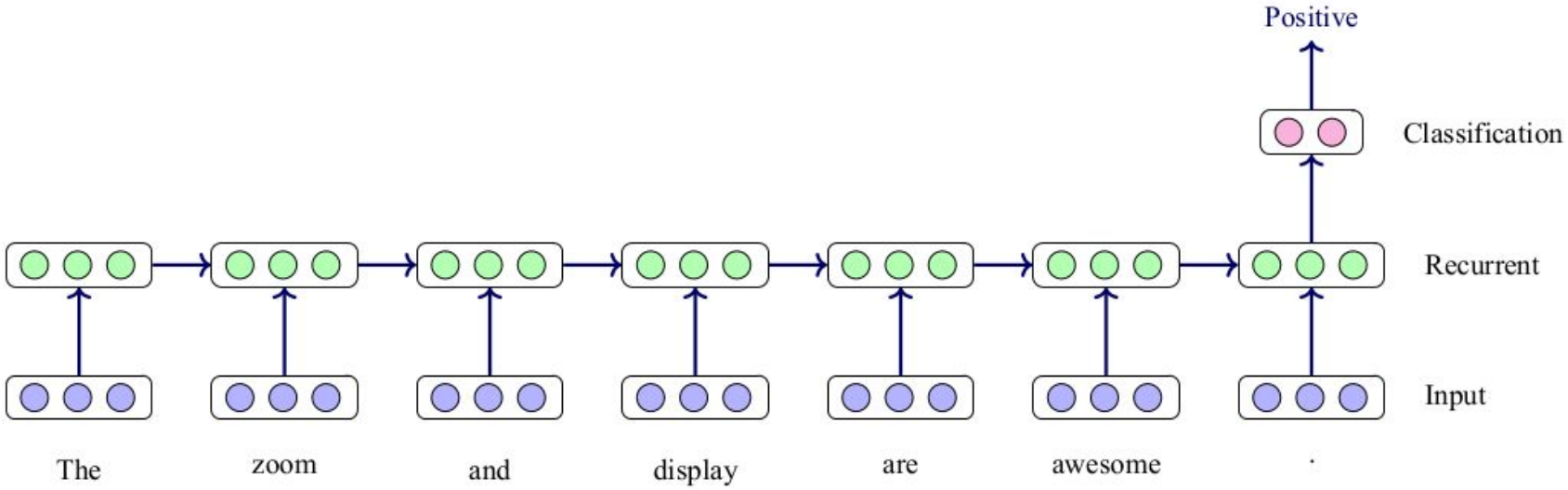


Classification

RNN/LSTM/GRU for Sentence Classification

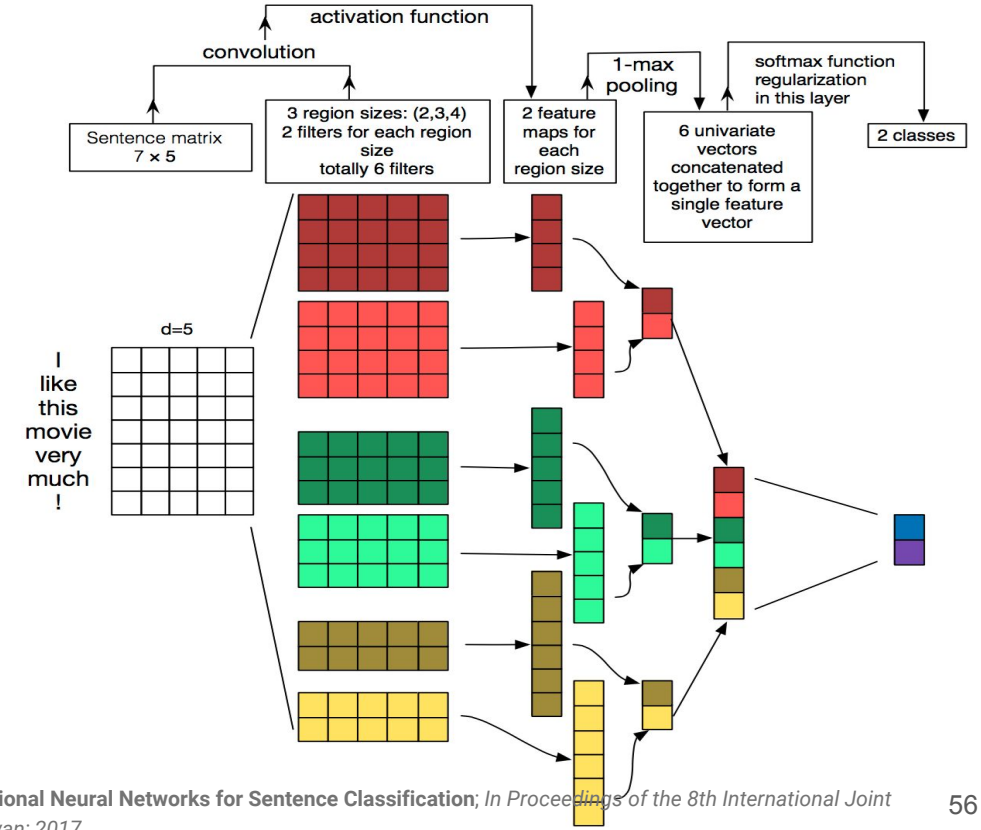
Sentiment Classification:

- Given a sentence X, identify the expressed sentiment.



CNN for Sentence Classification

1. Sentence matrix
   a. embeddings of words
2. Convolution filters
   a. Total 6 filters; Two each of size 2, 3 & 4.
   b. 1 feature maps for each filter
3. Pooling
   a. 1-max pooling
4. Concatenate the max-pooled vector
5. Classification
   a. Softmax

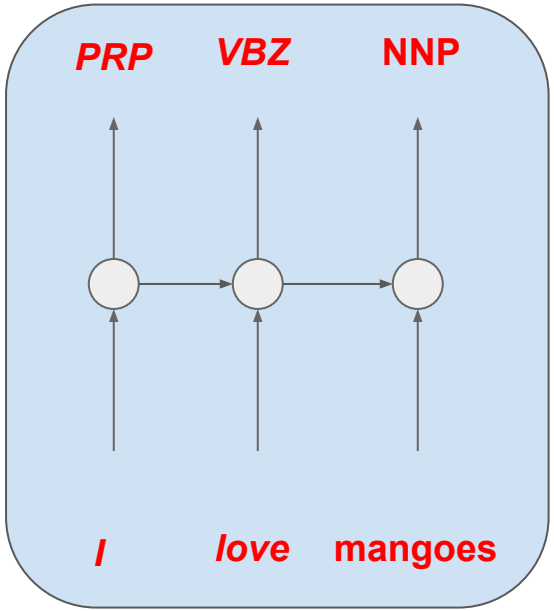


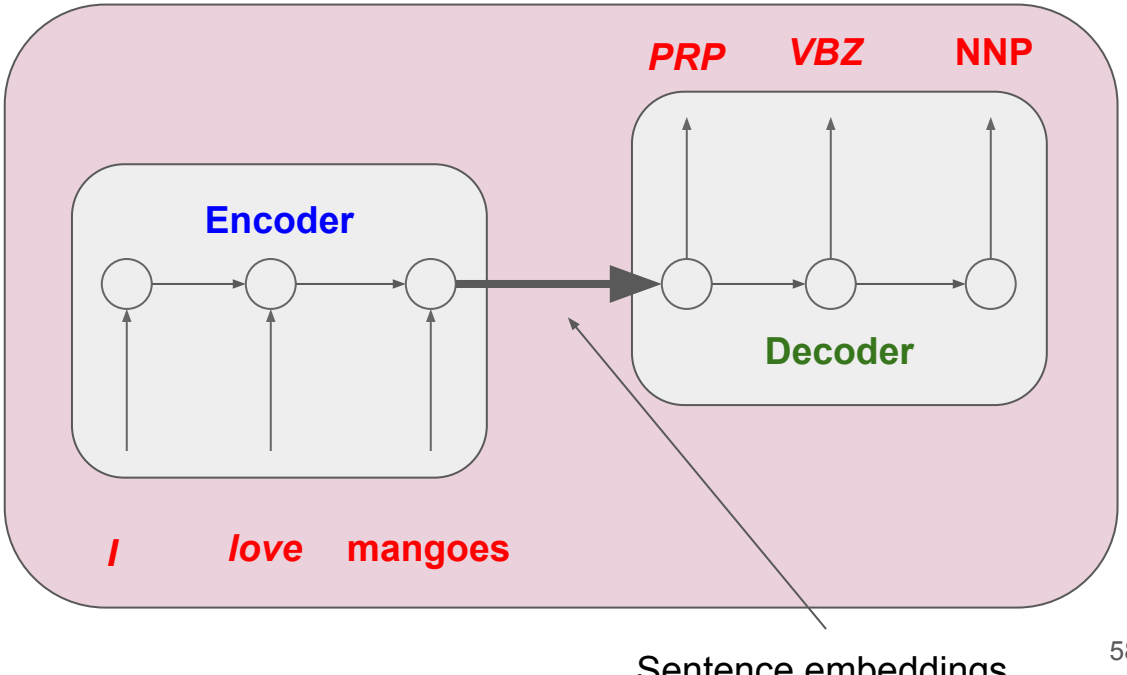
Zhang, Y., Wallace, B. ; A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification; *In Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP-2017); pages 253-263; Taipie, Taiwan; 2017.*

Sequence Transformation

Sequence labeling v/s Sequence transformation

- PoS Tagging



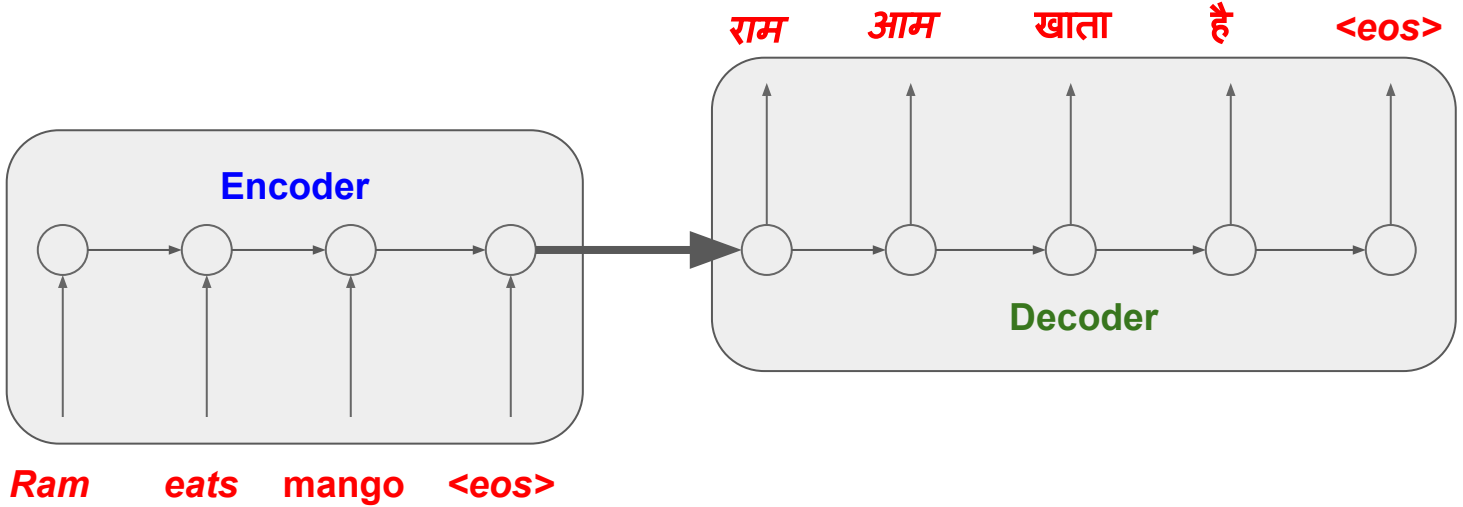


Why sequence transformation is required?

- For many application length of I/p and O/p are not necessarily same.
  - E.g. Machine Translation, Summarization, Question Answering etc.
- For many application length of O/p is not known.
- Non-monotone mapping
  - Reordering of words.
- Applications like PoS tagging, Named Entity Recognition does not require these capabilities.

Encode-Decode paradigm

- English-Hindi Machine Translation
  - Source sentence: 3 words
  - Target sentence: 4 words
  - Second word of the source sentence maps to 3rd & 4th words of the target sentence.
  - Third word of the source sentence maps to 2nd word of the target sentence



Problems with Encode-Decode paradigm

- Encoding transforms the entire sentence into a single vector.
- Decoding process uses this sentence representation for predicting the output.
  - Quality of prediction depends upon the quality of sentence embeddings.
- After few time steps decoding process may not properly use the sentence representation due to long-term dependency.

Solutions

- To improve the quality of predictions we can
  - Improve the quality of sentence embeddings

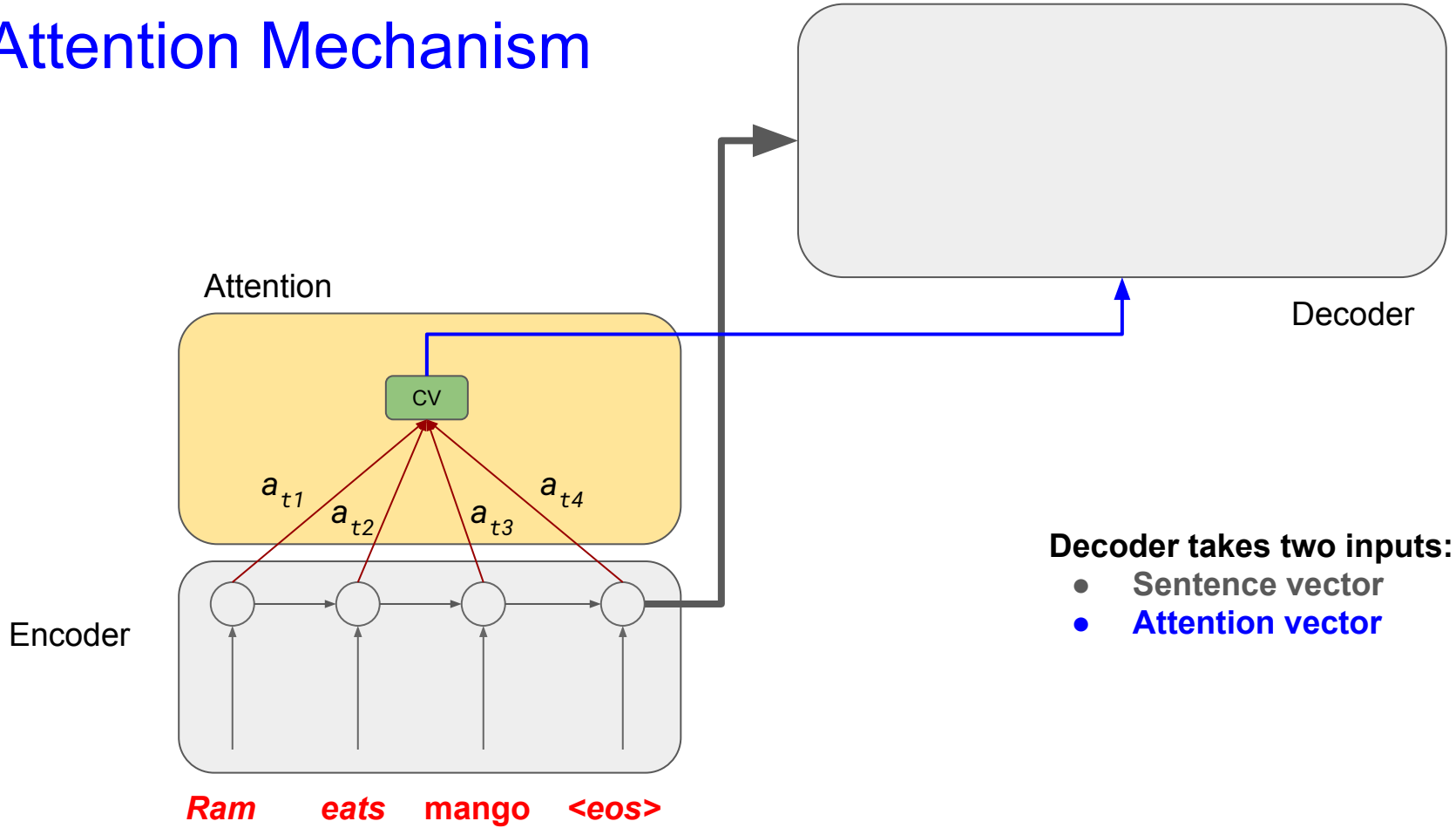
    ‘OR’

  - Present the source sentence representation for prediction at each time step.

    ‘OR’

  - Present the RELEVANT source sentence representation for prediction at each time step.
    - *Encode* - *Attend* - *Decode* (Attention mechanism)

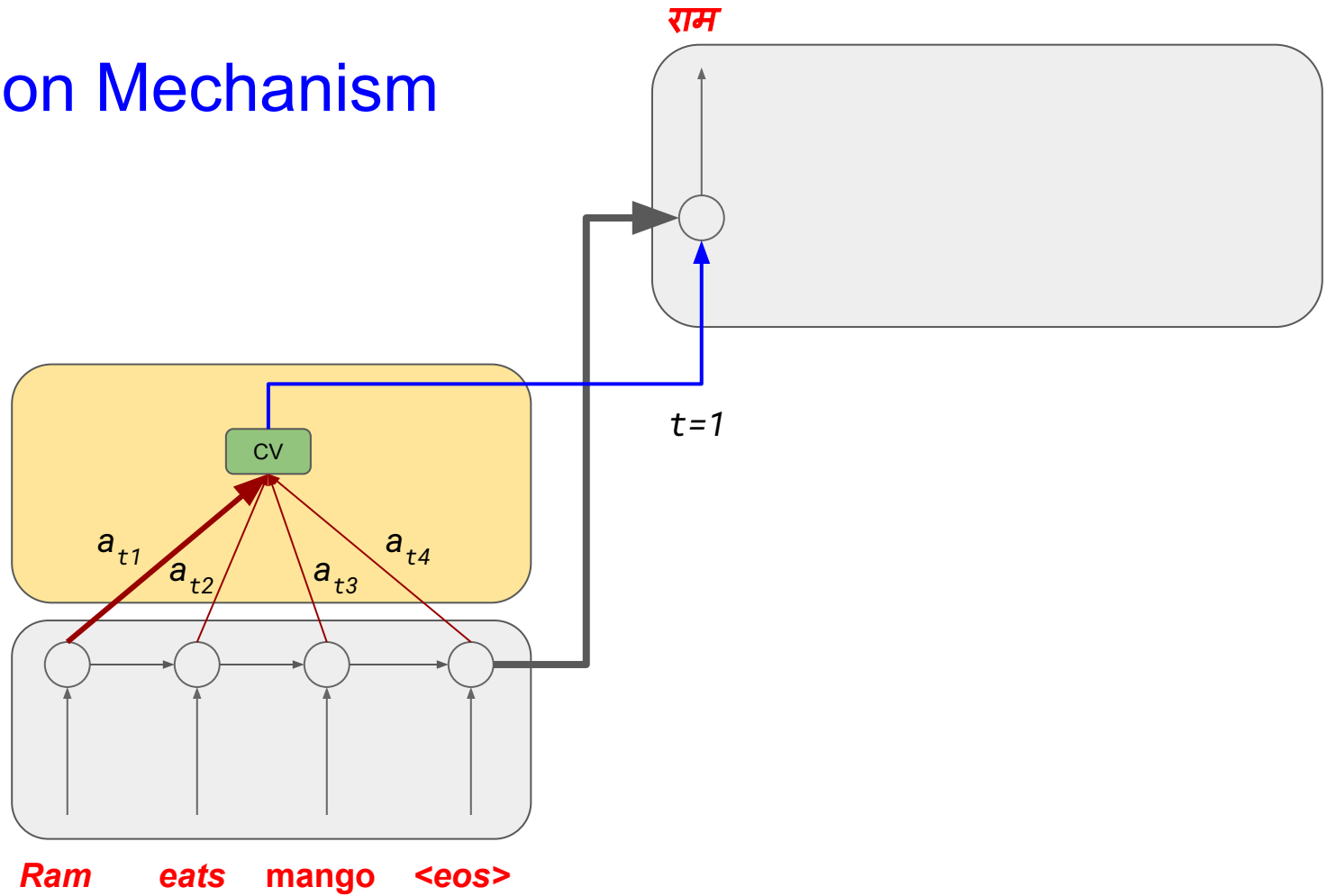
Attention Mechanism



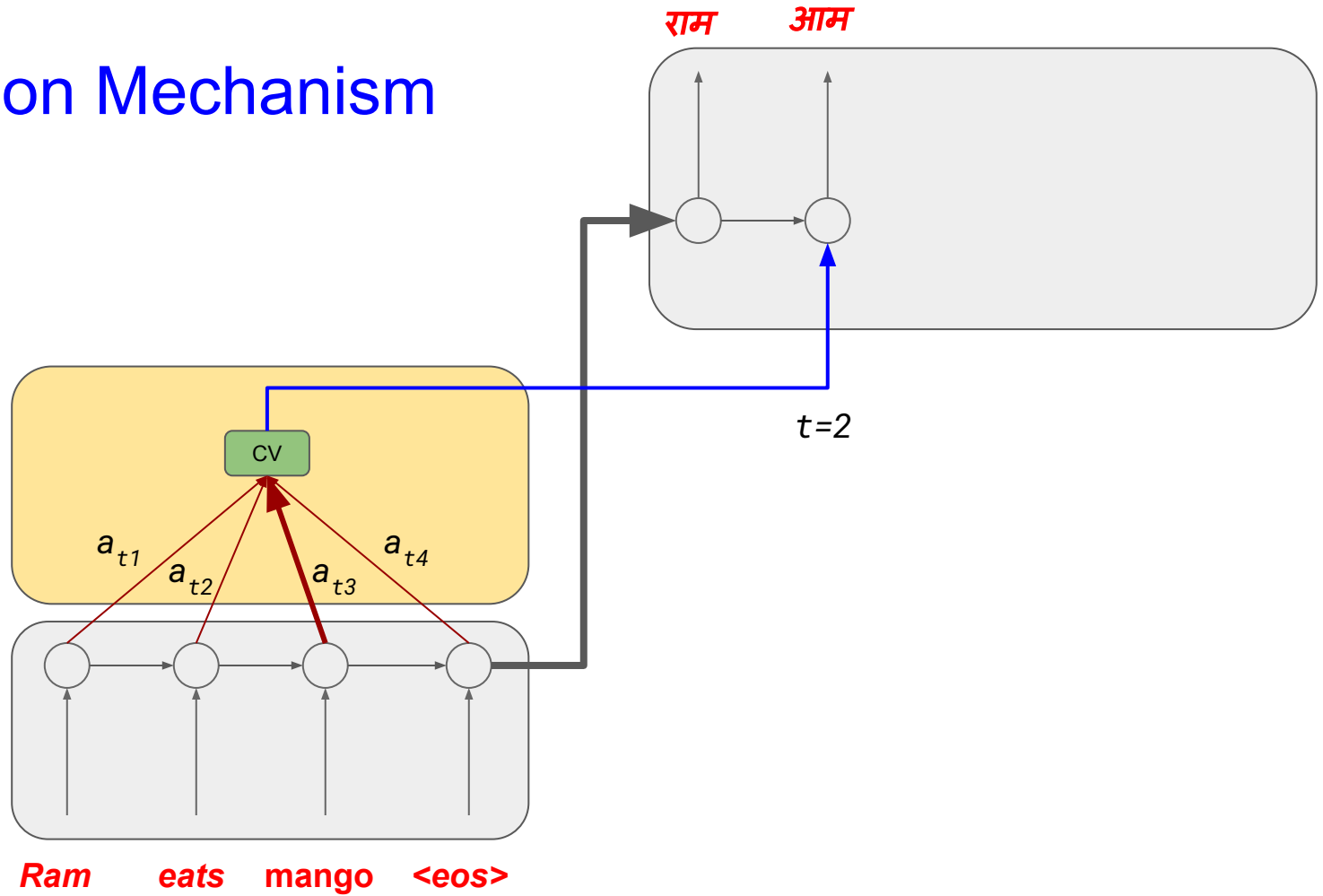
Decoder takes two inputs:

- Sentence vector
- **Attention vector**

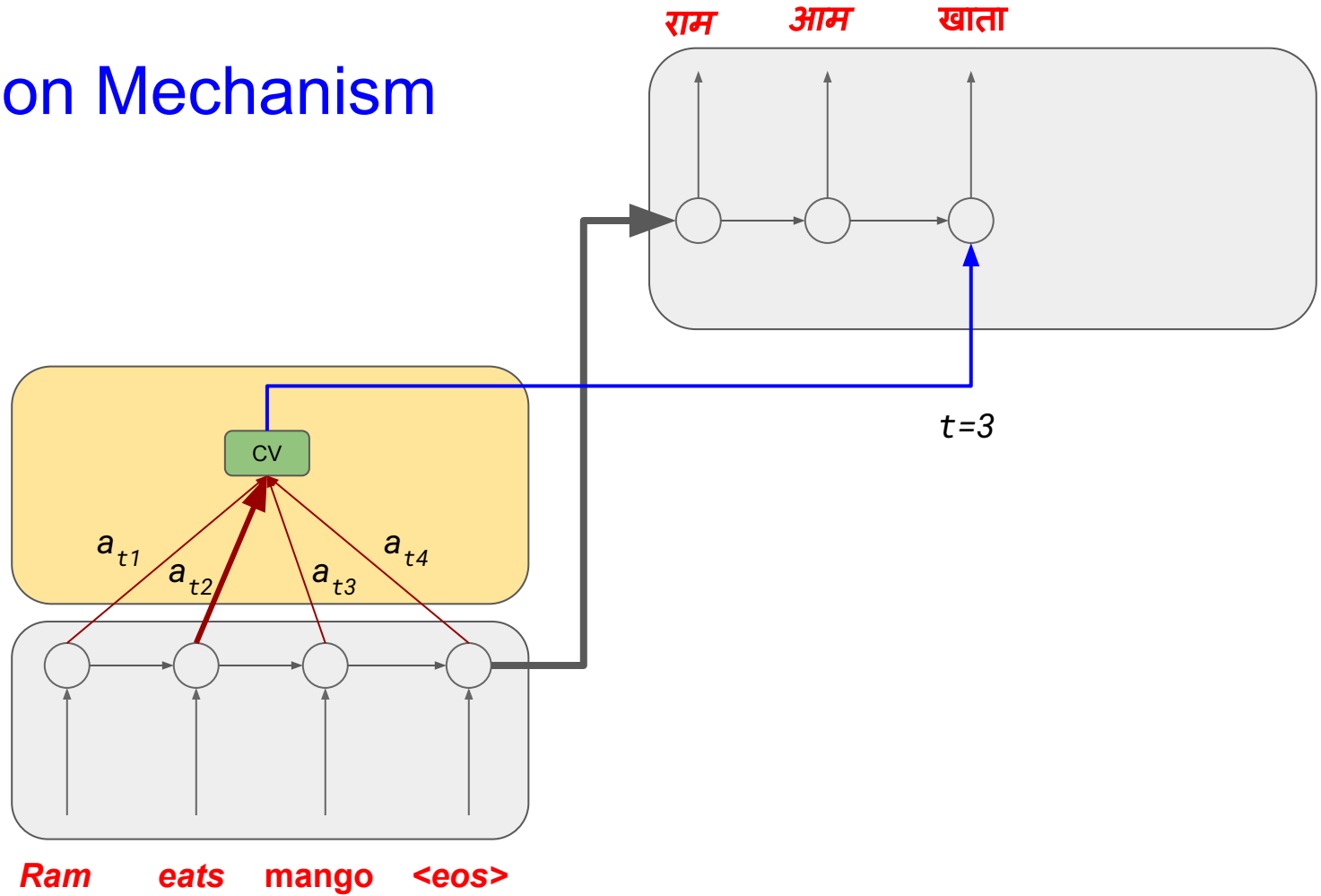
Attention Mechanism



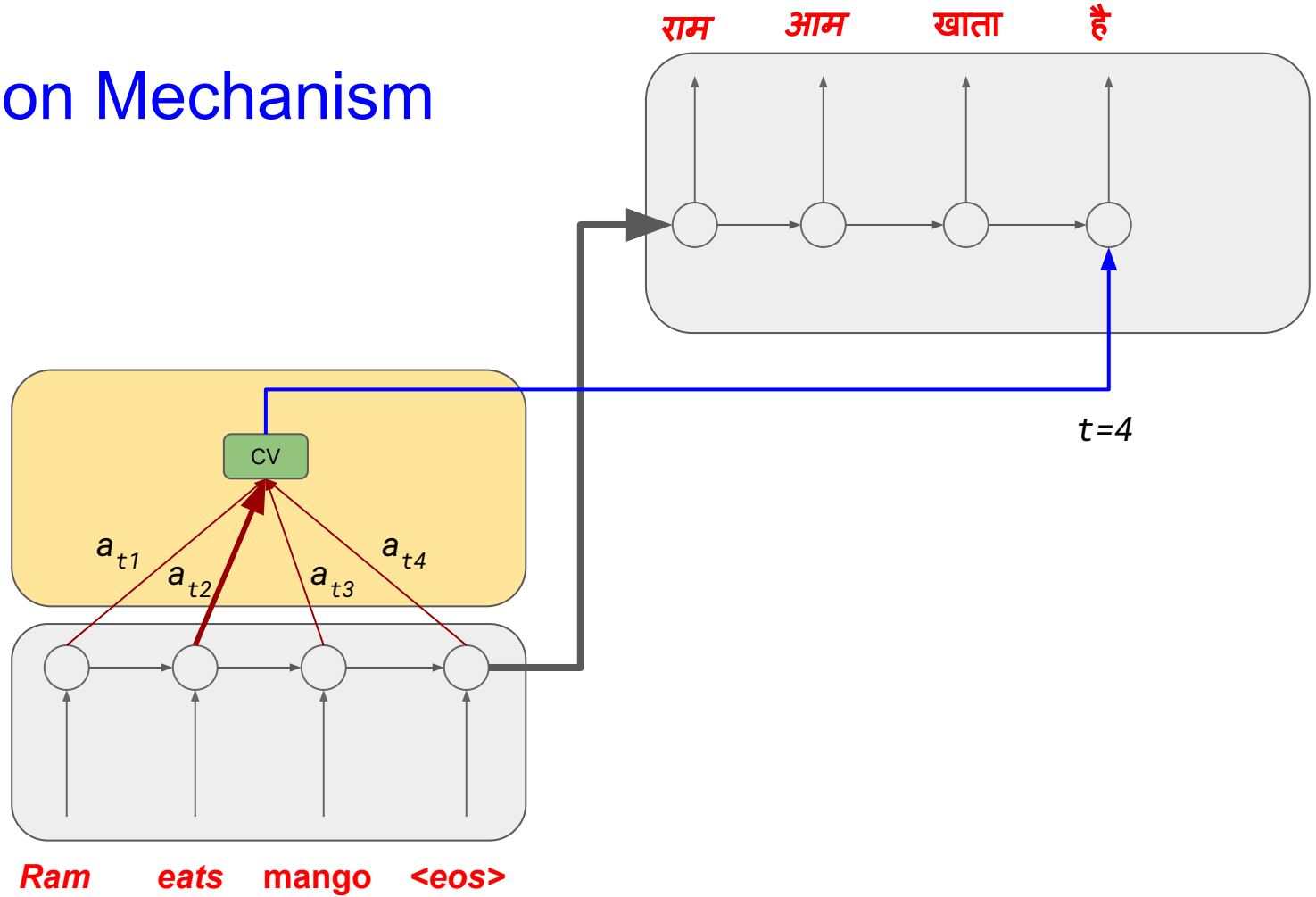
Attention Mechanism



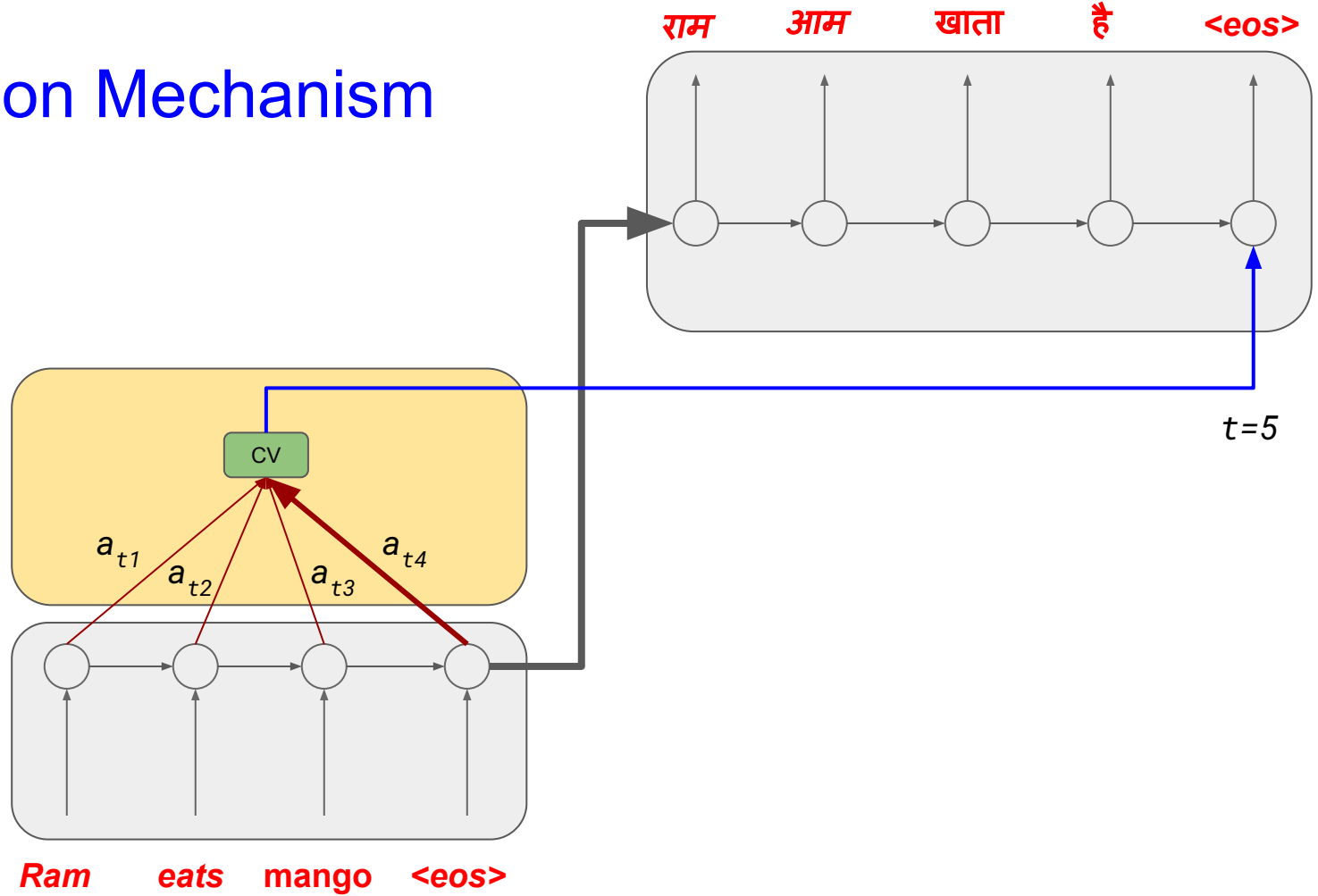
Attention Mechanism



Attention Mechanism



Attention Mechanism



Word Representation

Why do we need word representation?

- Many Machine Learning algorithms does not understand text data, they require input to be numeric. E.g. SVM, NN etc.
- Two types of representations
  - Local Representation
    - One hot
      - Cat = [0,0,0,0,1,0,0,0,0,0]
      - Sparse
      - No semantics
      - Curse of Dimensionality
  - Distributed Representation
    - Word embeddings
      - Cat = [2.4, 1.0,3.1,5.3]
      - Dense
      - Very good at capturing semantic relations.

Available word representation models

- Word2vec [Mikolov et al., 2013]
  - Contextual model
  - Two variants
    - Skip-gram
    - Continuous Bag-of-word
- GloVe [Pennington et al., 2014]
  - Co-occurrence matrix
  - Matrix Factorization
- FastText [Bojanowski et al., 2016]
  - Similar to word2vec
  - Works on sub-word level
- Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018]
  - Based on Transformer model
- Embeddings from Language Models (ELMo) [Peters et al., 2018]
  - Contextual
    - The representation for each word depends on the entire context in which it is used.

Recent Trends in DL-NLP

Recent trends

- Multi-task Learning
- Transfer Learning
- Multi-linguality
- Multi-modality
- Reinforcement Learning
- Interpretability/Explainability of Neural Net

Top AI-NLP-ML Venues

- NLP AI venues: ACL, EMNLP, NAACL, COLING, CoNLL, TACL
- AI venues: AAAI, IJCAI
- ML venues: NIPS, ICML etc.

Turing Award - 2018

- Nobel Prize of computing
- Godfathers of AI
  - **Yann LeCun:** Professor at New York University and Facebook's chief AI scientist
  - **Geoffrey Hinton:** Professor at University of Toronto and associated with the Google Brain
  - **Yoshua Bengio:** Professor at the University of Montreal and AI advisor to Microsoft

Few good reads..

- Denny Britz; Recurrent Neural Networks Tutorial, Part 1-4
  http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/
- Andrej Karpathy; The Unreasonable Effectiveness of Recurrent Neural Networks
  http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Chris Olah; Understanding LSTM Networks
  http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Yoon Kim; Convolutional Neural Networks for Sentence Classification
  https://www.aclweb.org/anthology/D14-1181.pdf

Thank You!