HERIOT WATT UNIVERSITY

Applied Text Analytics (F20/F21AA)

Heriot Watt University , Dubai Campus

Dr. Radu Mihailescu

Associate Professor

Today's Lecture Lectures

- Introduction to Sequence-to-Sequence models (RNN)

- Word ordering **(sequence)** conveys semantics that cannot be inferred from the bag-of-words representation:

The cat chased the mouse. **The mouse chased the cat.**

Today's Lecture Recurrent Neural Networks

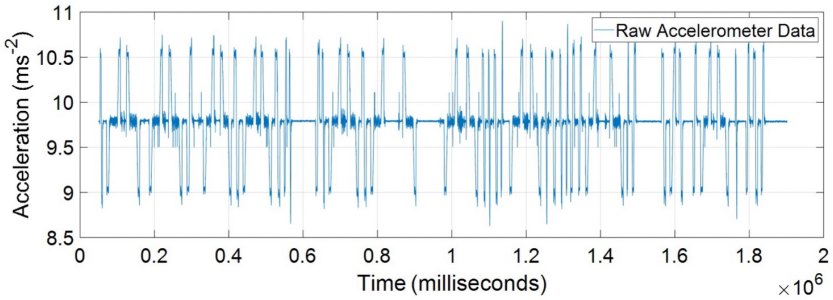
Family of neural networks for temporal/sequential data:

- Timeseries data

(stock market prices, tourist arrivals,…)

- Sequence of text

Audio



RNN popular model that has been used in NLP.

RNN is a sequential model that can perceive the text as a sequence of characters & words.

RNN (Recurrent Neural Network)

RNN (Recurrent Neural Network)

- They can take sentences, documents, or audio samples as input, making them extremely useful for natural language processing (NLP) systems such as automatic translation, speech-to-text, or sentiment analysis.

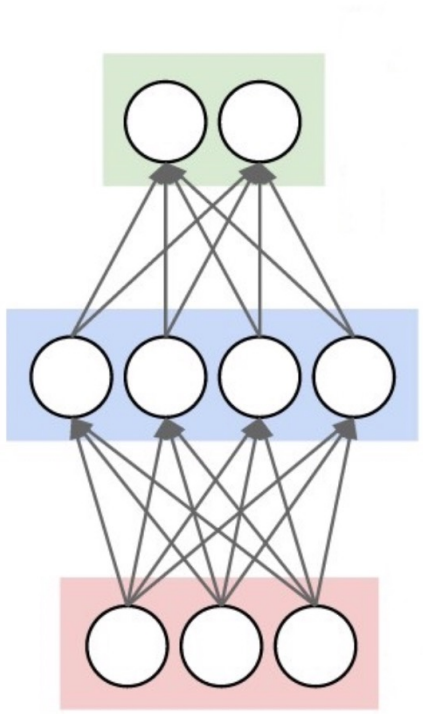
- (Unlike other NN models):
  - Take sequential input of any length
  - Share parameters over time

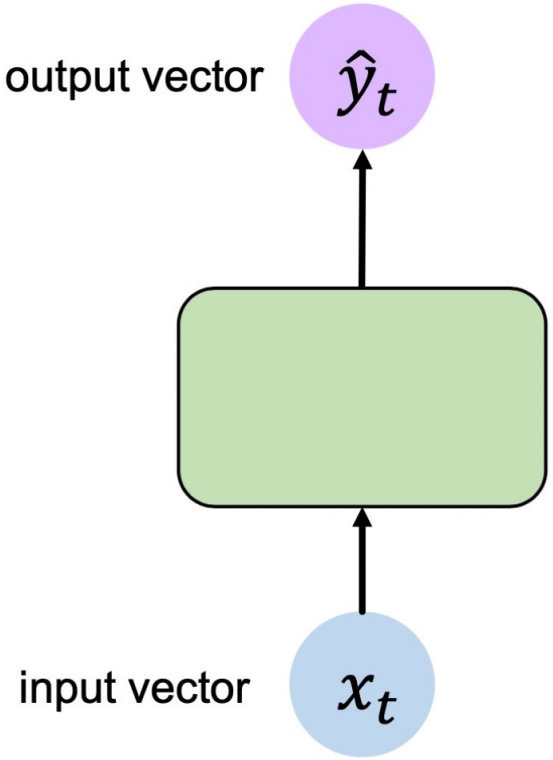
- *RNNs' ability to anticipate also makes them capable of surprising creativity.
  - compose a melody such as the one produced by Google's Magenta project.
  - Generate sentences, image captions, and much more

Franz Schubert - Symphony No.8 in B minor, D.759 ("Unfinished") finalized by artificial intelligence

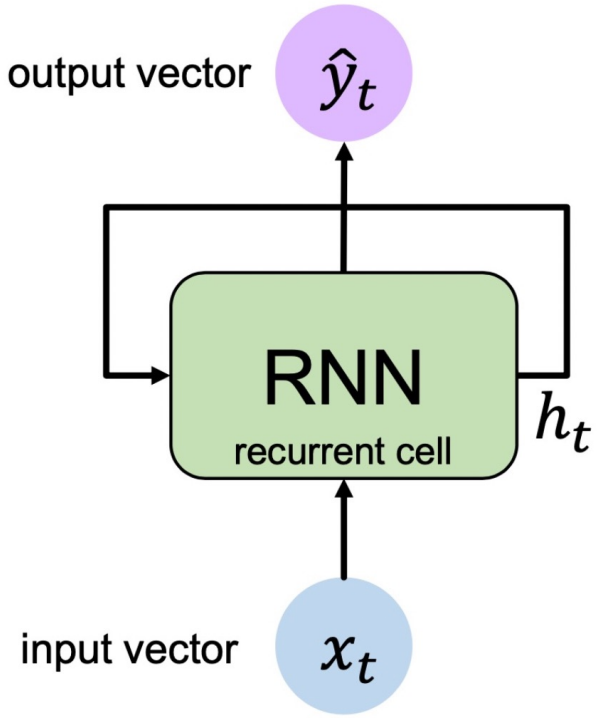
https://www.youtube.com/watch?v=_6OUGRsslJY

Standard Neural Network





RNN (Recurrent Neural Network)



Apply a recurrence relation at every time step to process a sequence:

$$\boxed{h_t} = \boxed{f_W}\left(\boxed{h_{t-1}}, \boxed{x_t}\right)$$

cell state

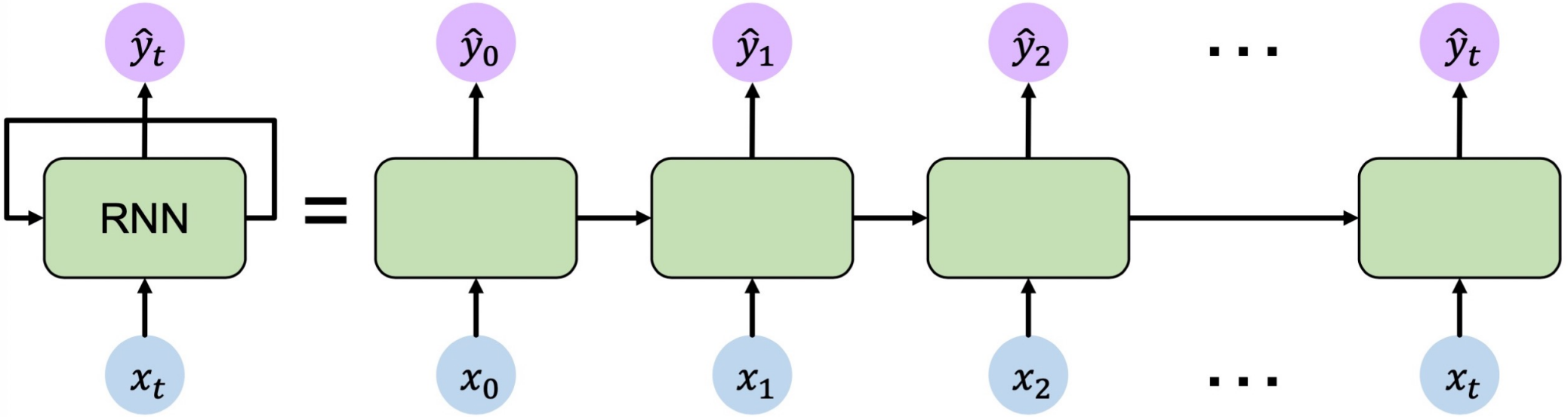
function parameterized by W

old state

input vector at time step *t*

the same function and set of parameters are used at every time step

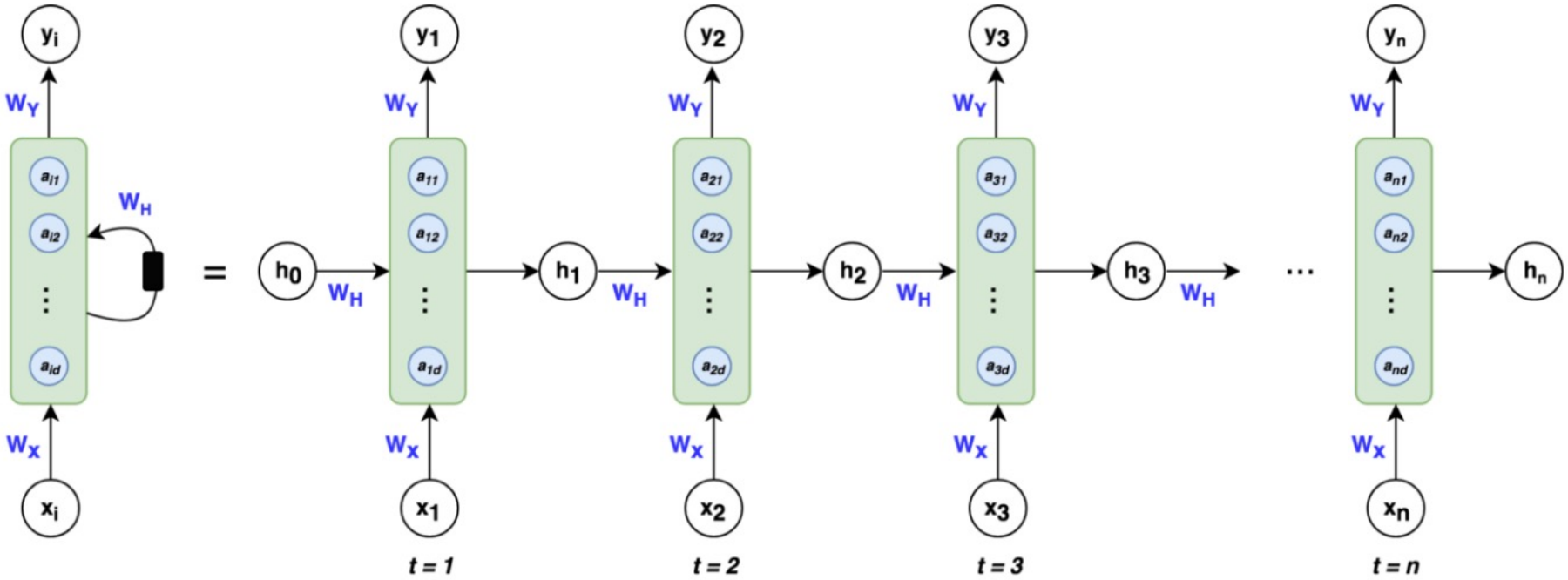
Unrolling a RNN through time



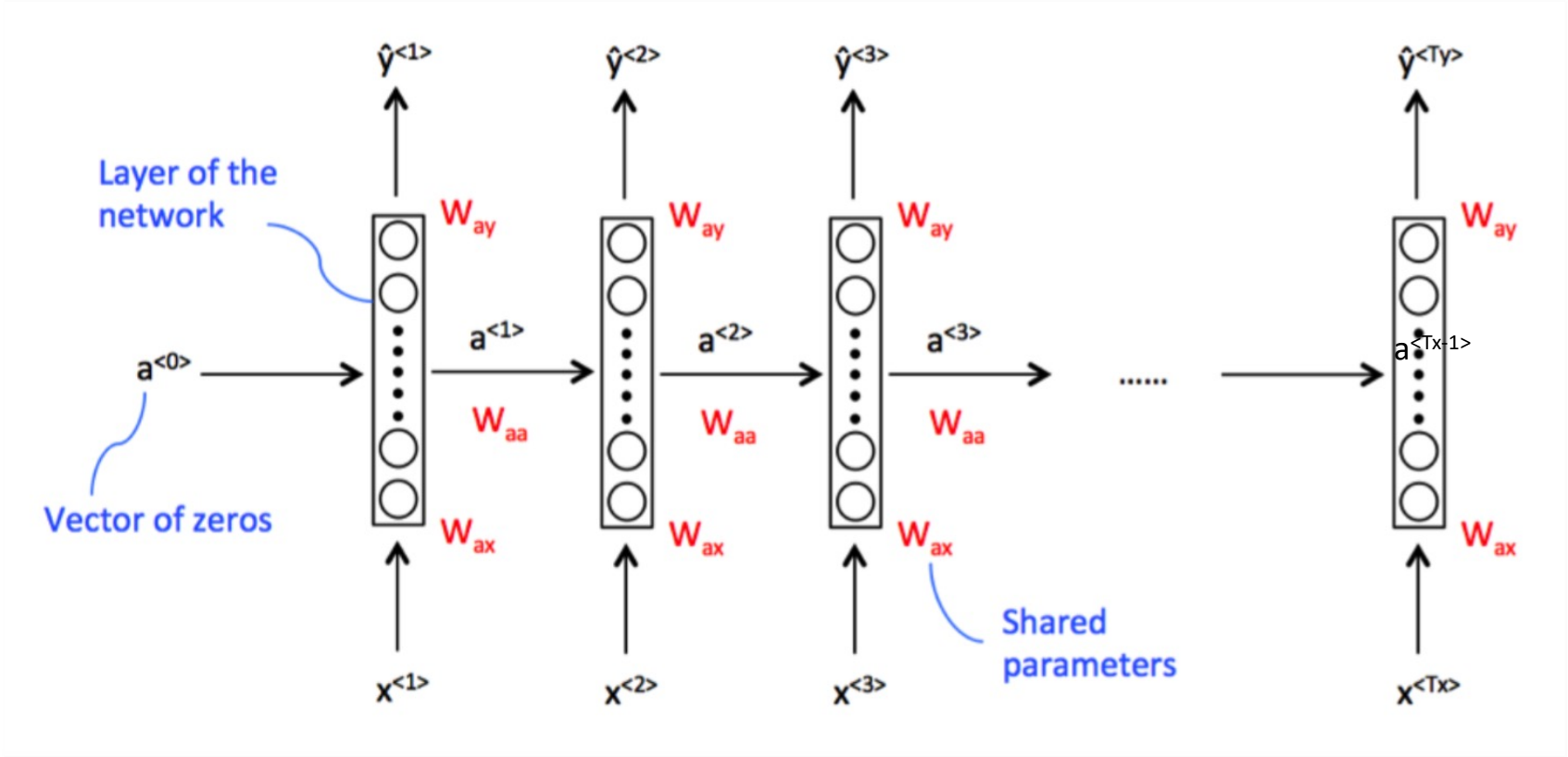
Parameters are shared across inputs

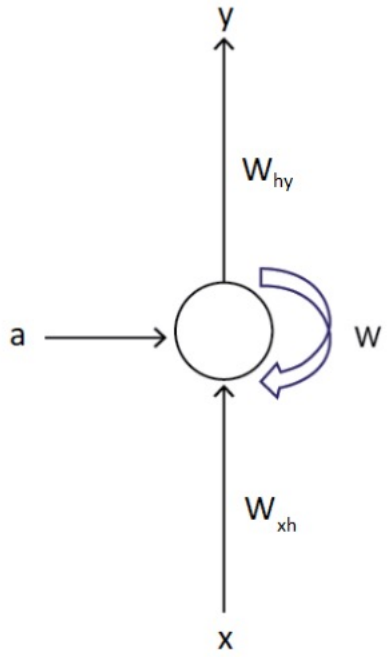
RNN UNFOLDED

Parameters are shared across inputs: W_X, W_H, W_Y



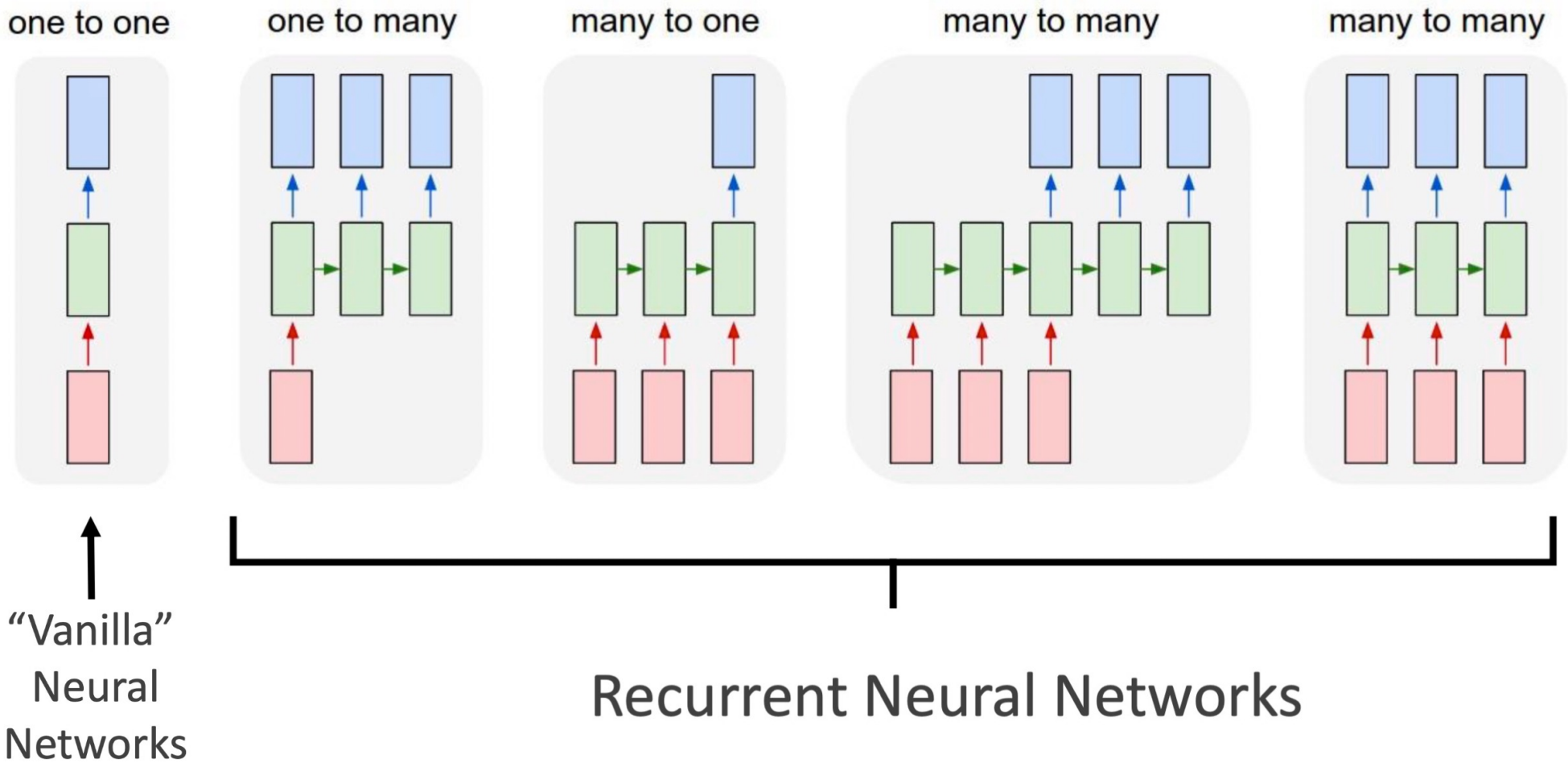
RNN model



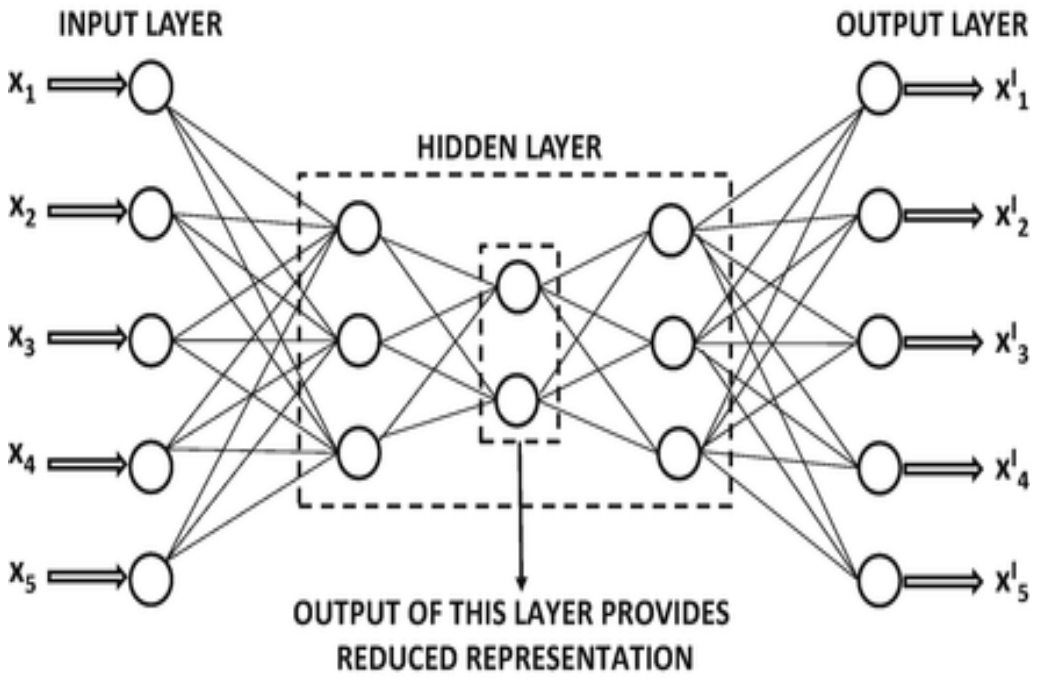


Parameters are shared across inputs: W_{ax}, W_{aa}, W_{ay}

RNN model


one to one
one to many
many to one
many to many
many to many
"Vanilla" Neural Networks
Recurrent Neural Networks

Feed forward NN vs. RNN



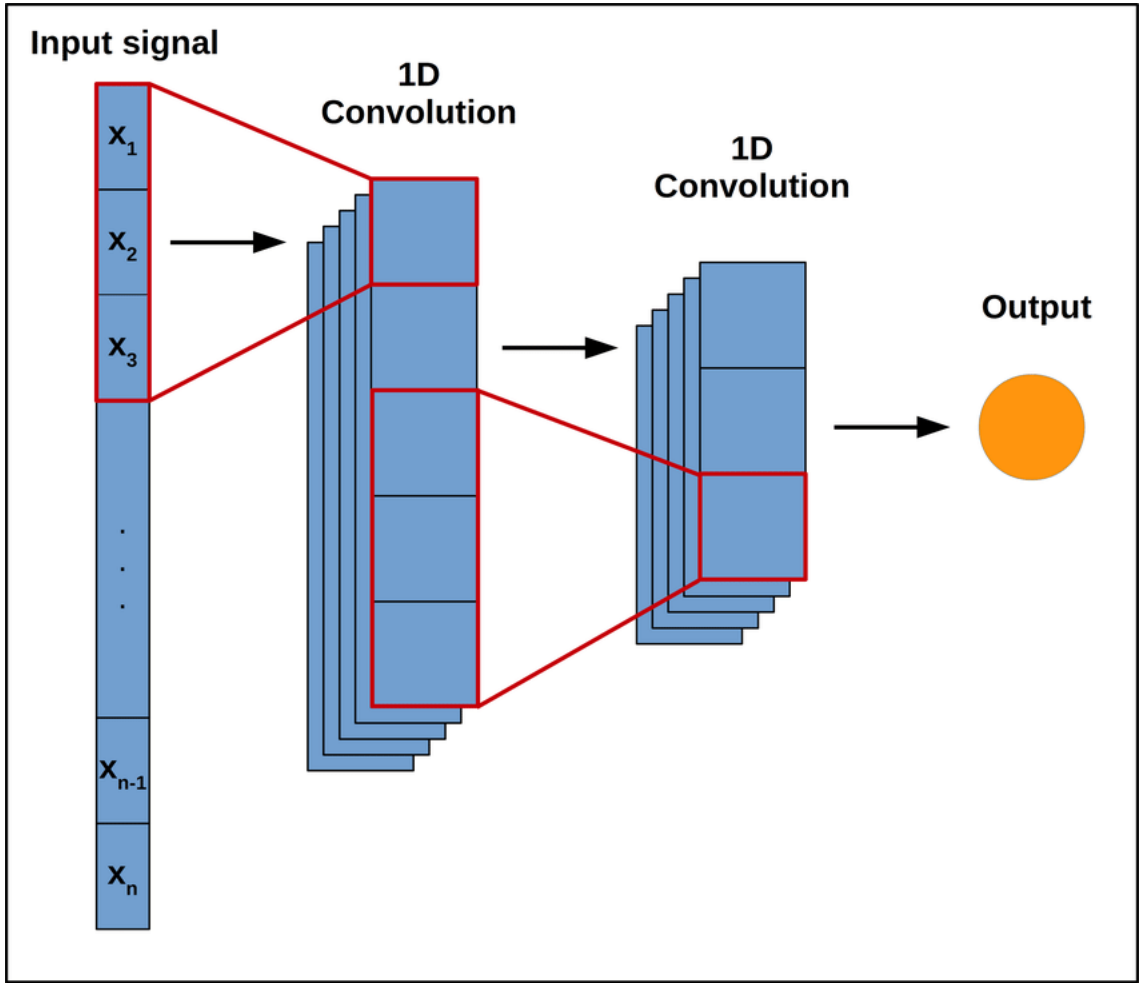
Feed forward NN

Why just not use a feed forward NN ?

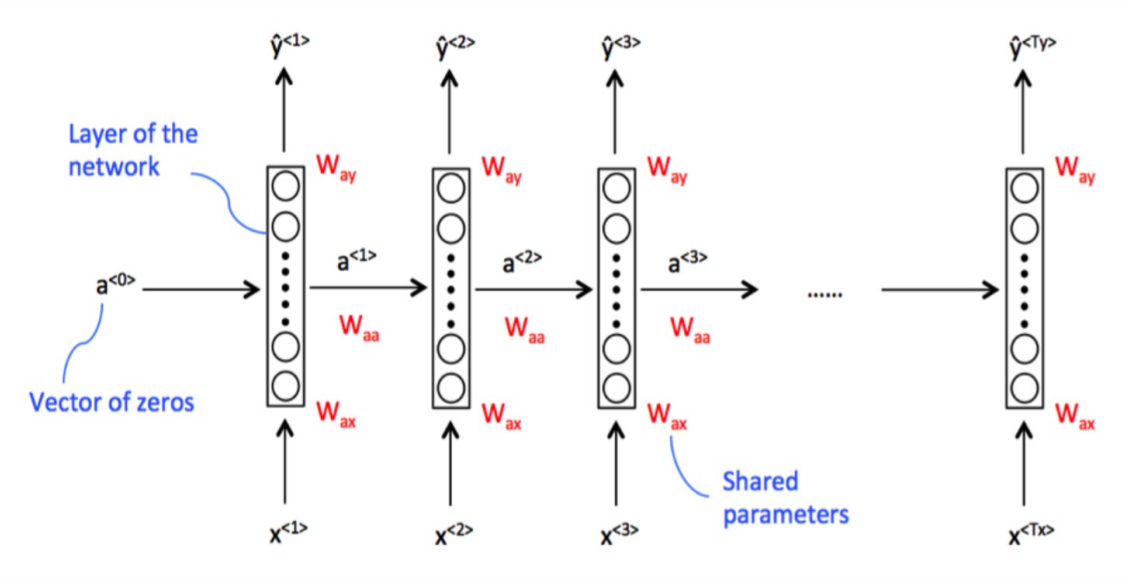
- Inputs and outputs for the different examples can be of different lengths.
- Features learned from different positions of the text do not generalize for other positions

Side Note: CNN

- Convolutional network can also leverage locality (learn local patterns)
- Often competitive with RNNs and much faster cheaper
- Works well for simple tasks like classification and forecasting



Notations



$x^{<t>}$: t-th element in the input sequence

$x^{(i)<t>}$: t-th element in the input sequence of training examples i

T_x : length of the input sequence

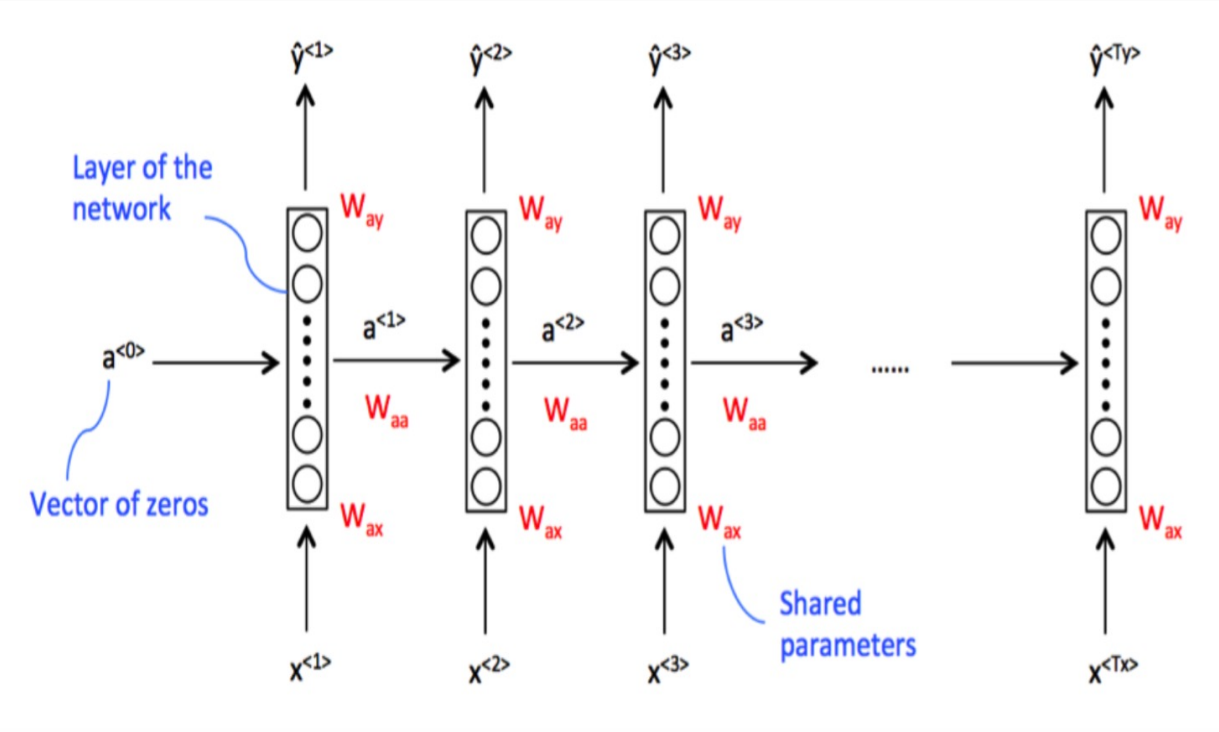
T_y : length of the output sequence

$T_x^{(i)}$: input sequence lenght for training example i

$y^{<t>}$: t-th element in the output sequence

$y^{(i)<t>}$: t-th element in the output sequence of training examples i

Forward Propagation

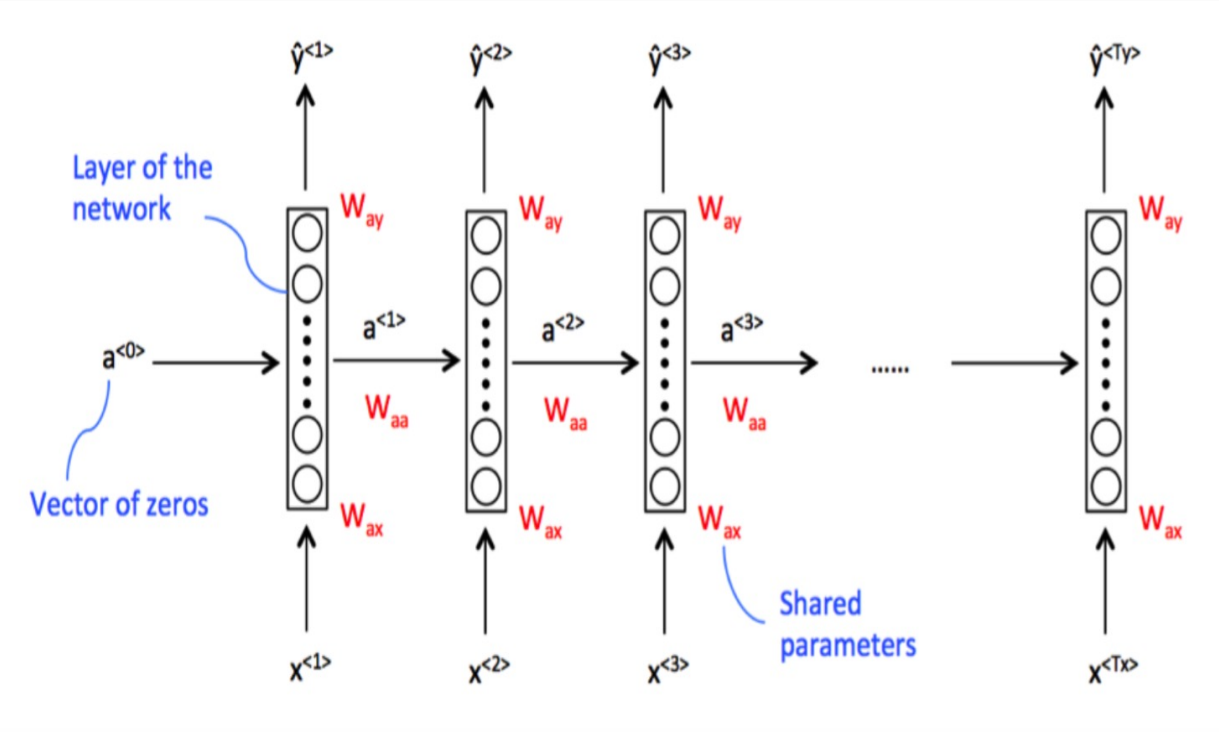


$$a^{<0>} = \vec{0} \text{ (a vector of zeros)}$$

$$a^{<t>} = g\left(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a\right)$$ tanh/Relu

$$\hat{y}^{<t>} = g\left(W_{ya}a^{<t>} + b_y\right)$$ sigmoid/Softmax

Forward Propagation



Matrix W_a as: $[W_{aa} \mid W_{ax}]$

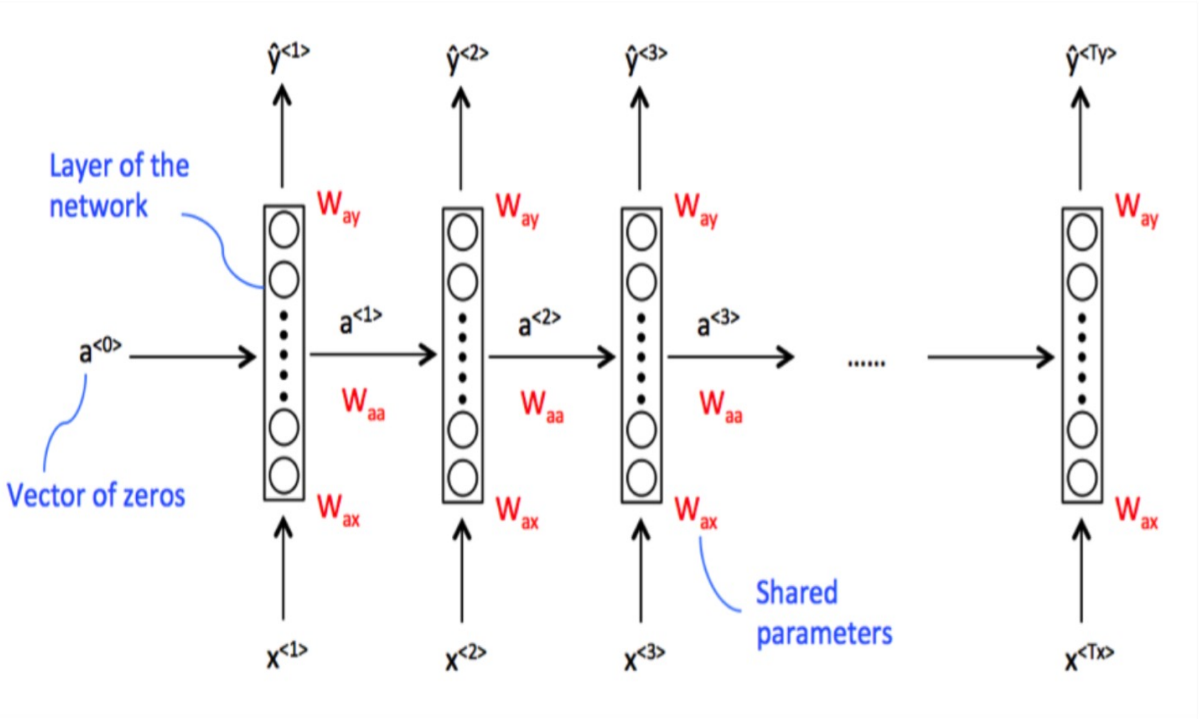
$$a^{<t>} = g\,(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

$$\hat{y}^{<t>} = g\,(W_y a^{<t>} + b_y)$$

Vectors stacked together as: $\begin{bmatrix} a^{<t-1>} \\ - - - \\ x^{<t>} \end{bmatrix}$

Backward propagation through time (BPTT)

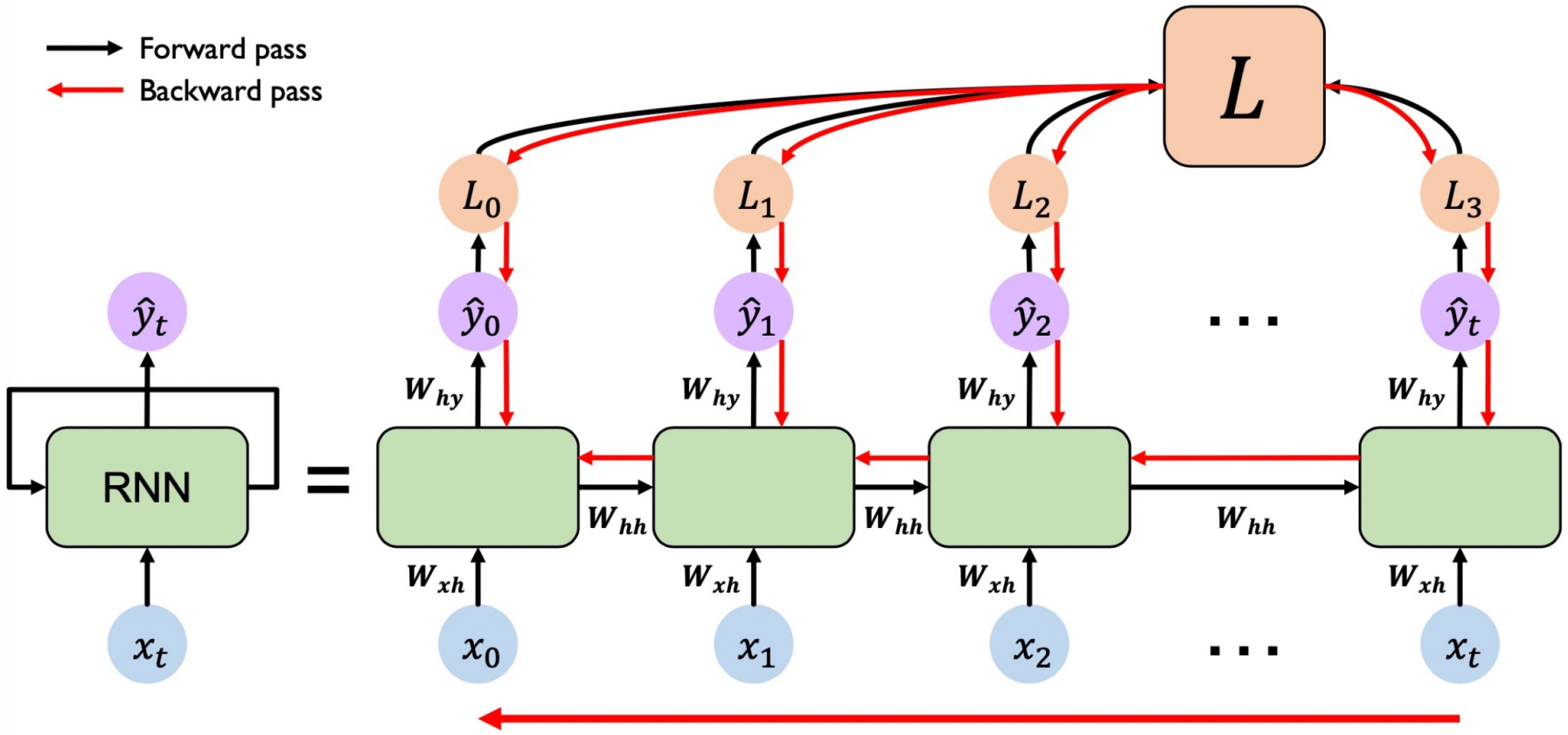
For each output y<> will calculate the loss, and then sum the losses up to obtain the entire loss for the sequence.



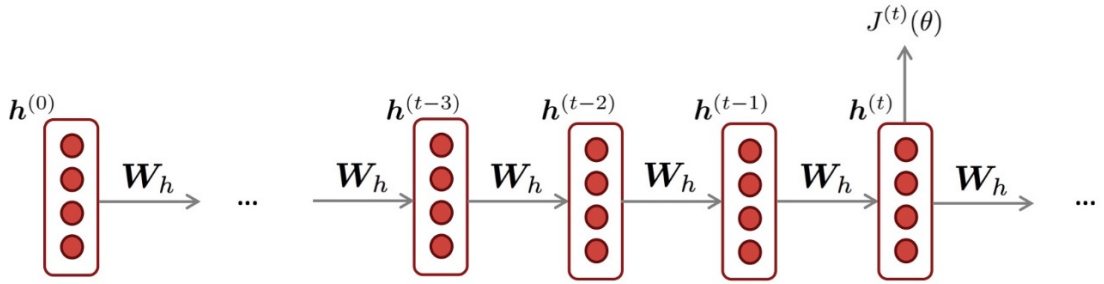
$$\mathcal{L}(\hat{y}^{<t>}, y^{<t>}) = -\sum_i y_i^{<t>} \log \hat{y}_i^{<t>}$$

$$\mathcal{L} = \sum_t \mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

Backward propagation through time (BPTT)



Backward propagation through time (BPTT)



What's the derivative of $J^{(t)}$ w.r.t the repeated weight matrix W_h ?

$$\frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} = \sum_{i=1}^{t} \frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}}\bigg|_{(i)}$$

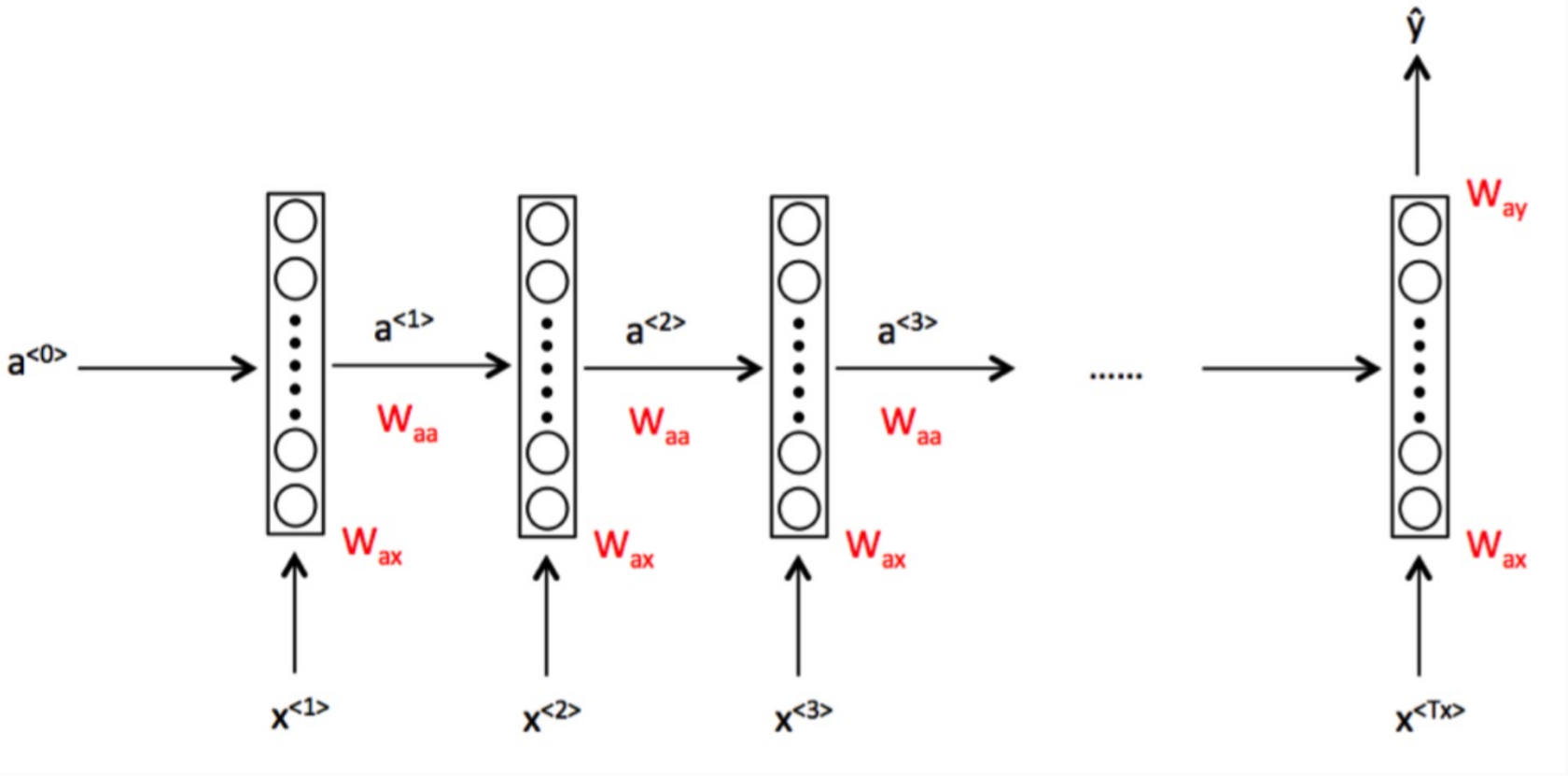
Compute the Loss (J) for the entire sequence:

$$J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta)$$

The gradient w.r.t. a repeated weight is the sum of the gradient w.r.t. each time it appears

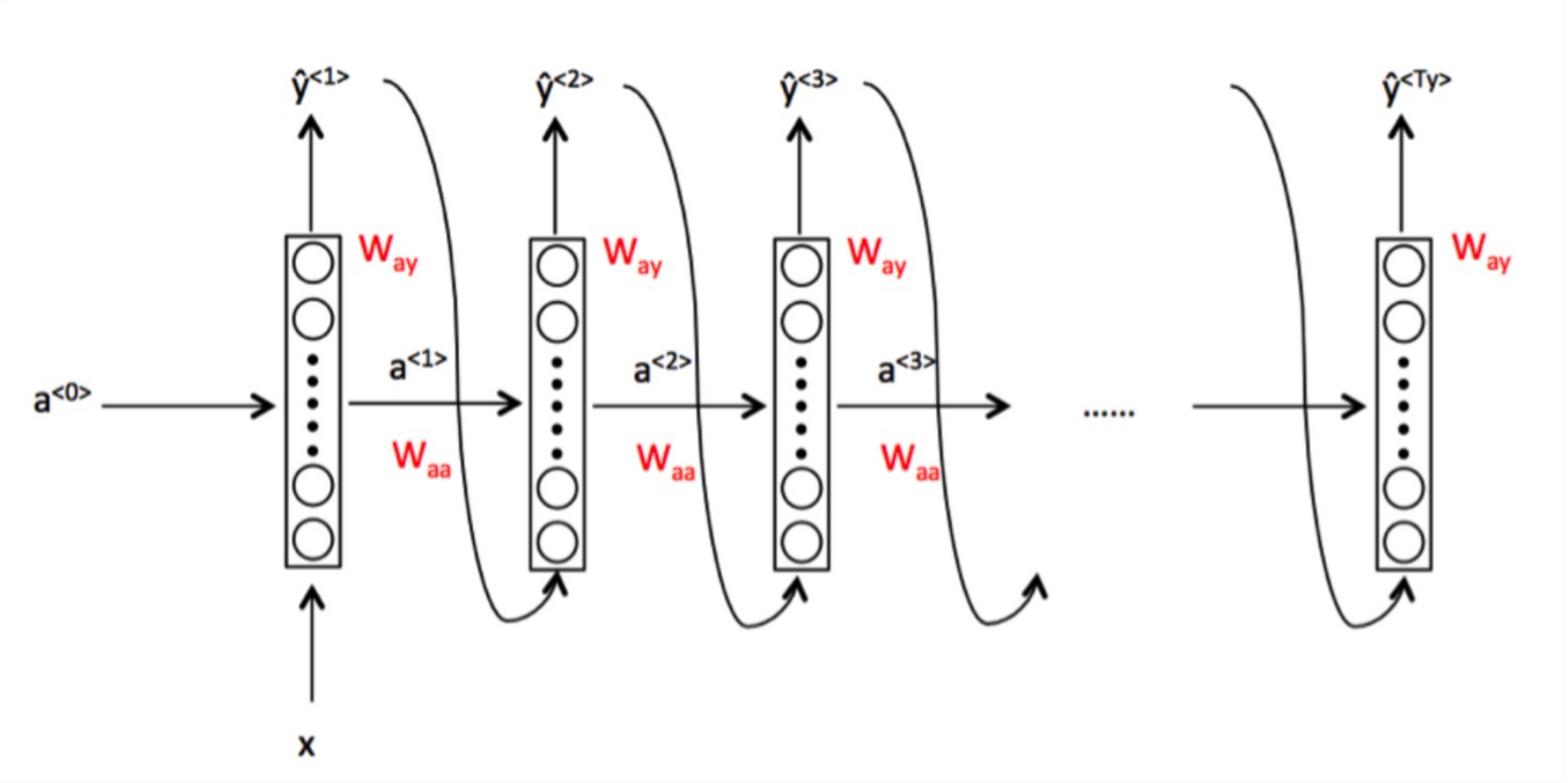
We introduce dummy variables $W_h^{(i)}$ that are defined as copies of W_h but with each $W_h^{(i)}$ used only at timestep (i)

Many to one RNN (Sentiment Classification)



Ty=1

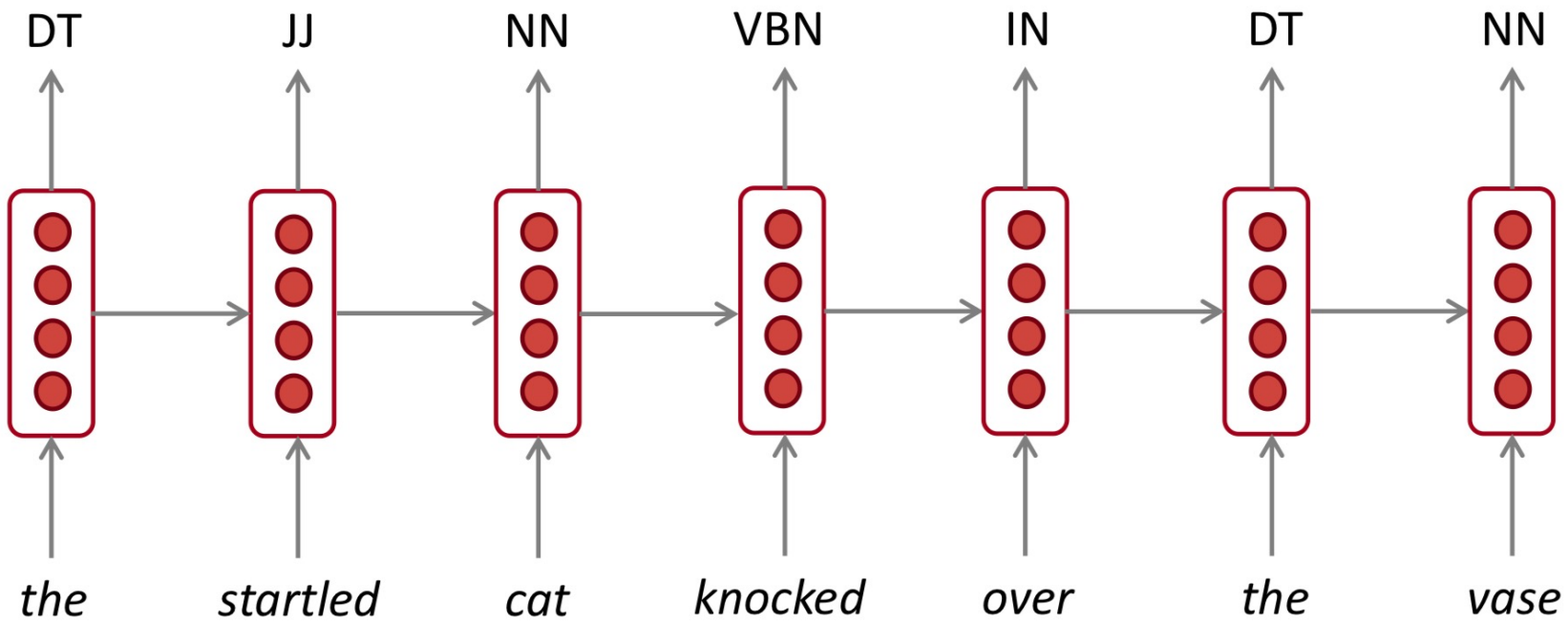
One to many RNN (Music Generation)



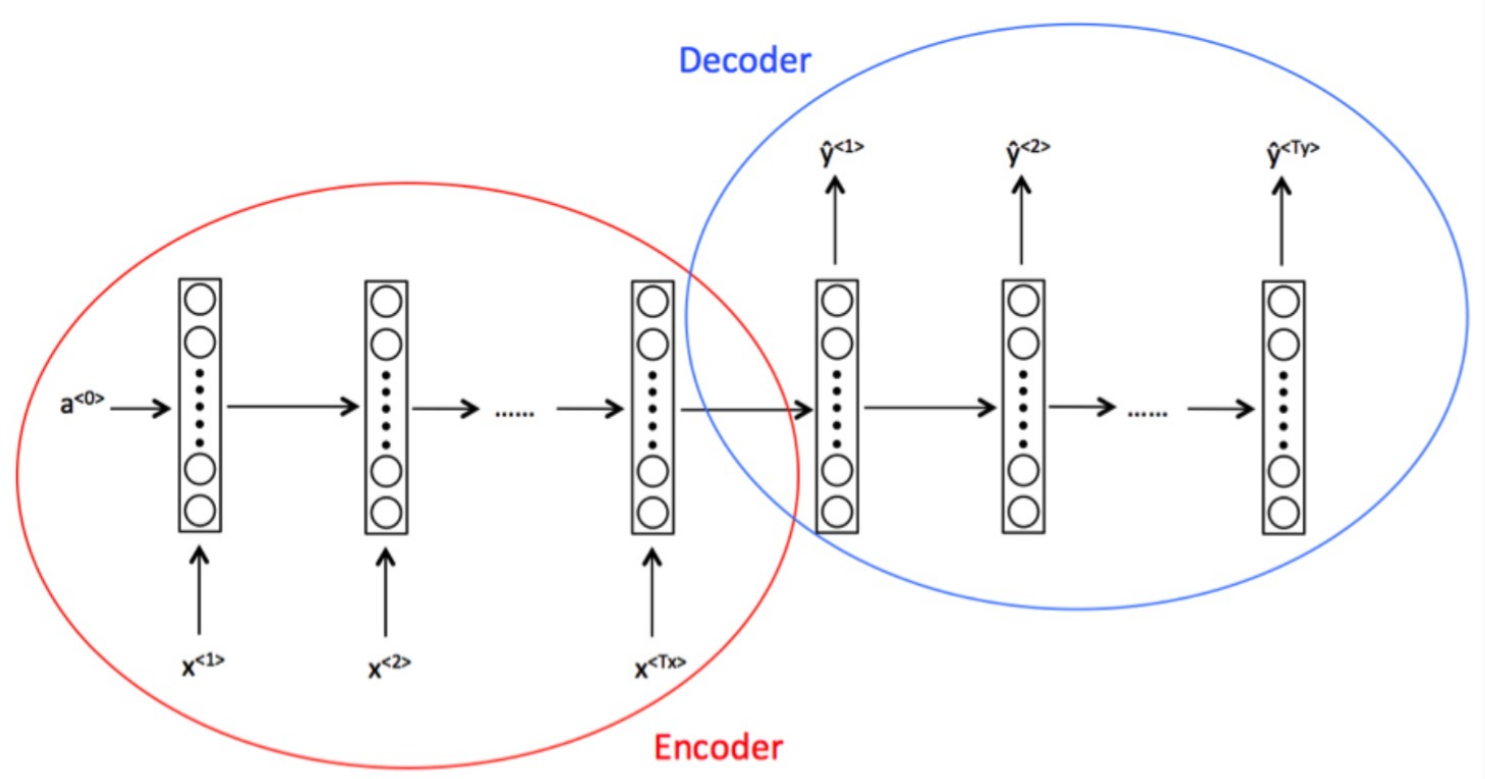
Tx=1

Many to many RNN (Part-of-speech tagging, named entity recognition)

$T_x = T_y$



Many to many RNN (Machine Translation)



$Tx \neq Ty$

Summary of RNN types

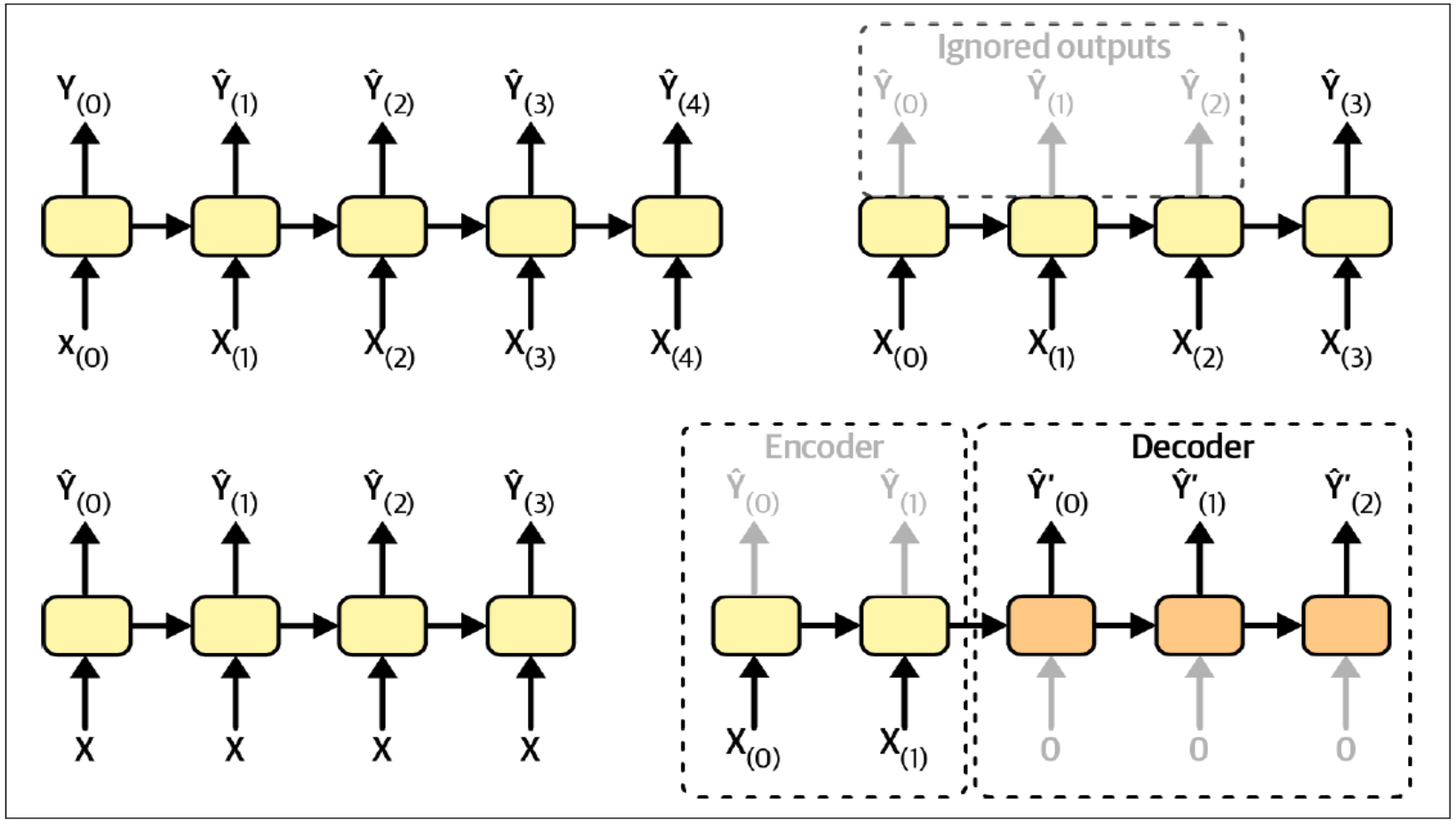
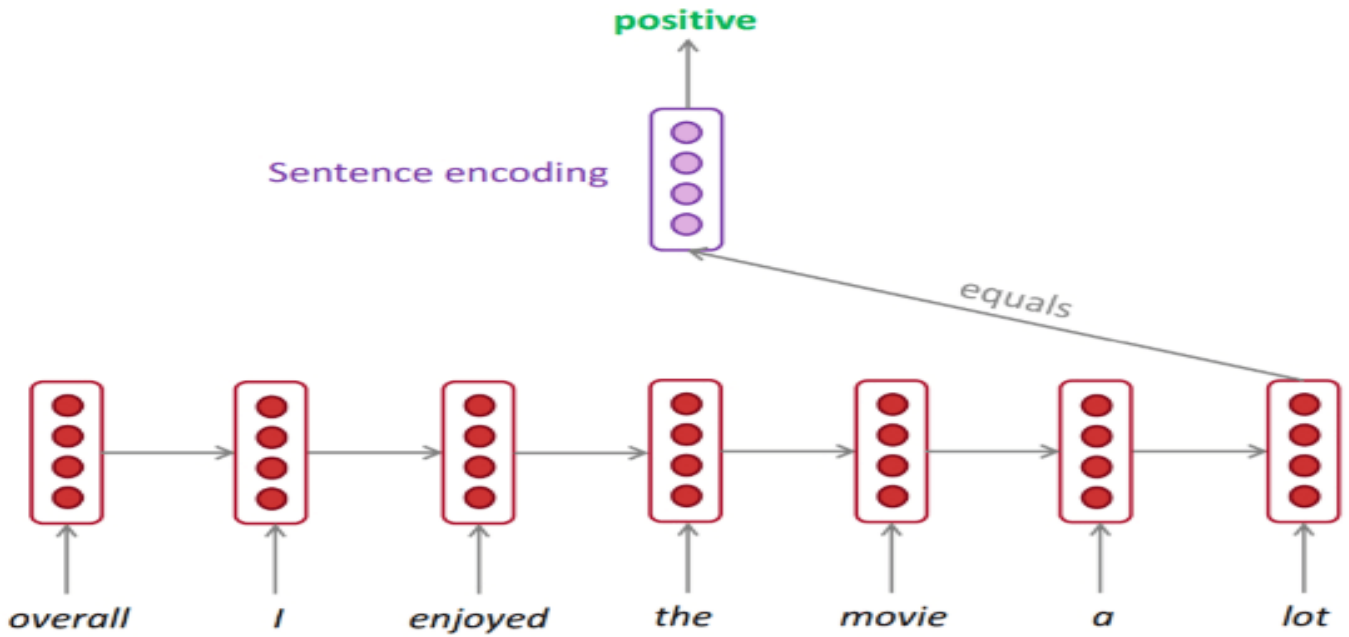
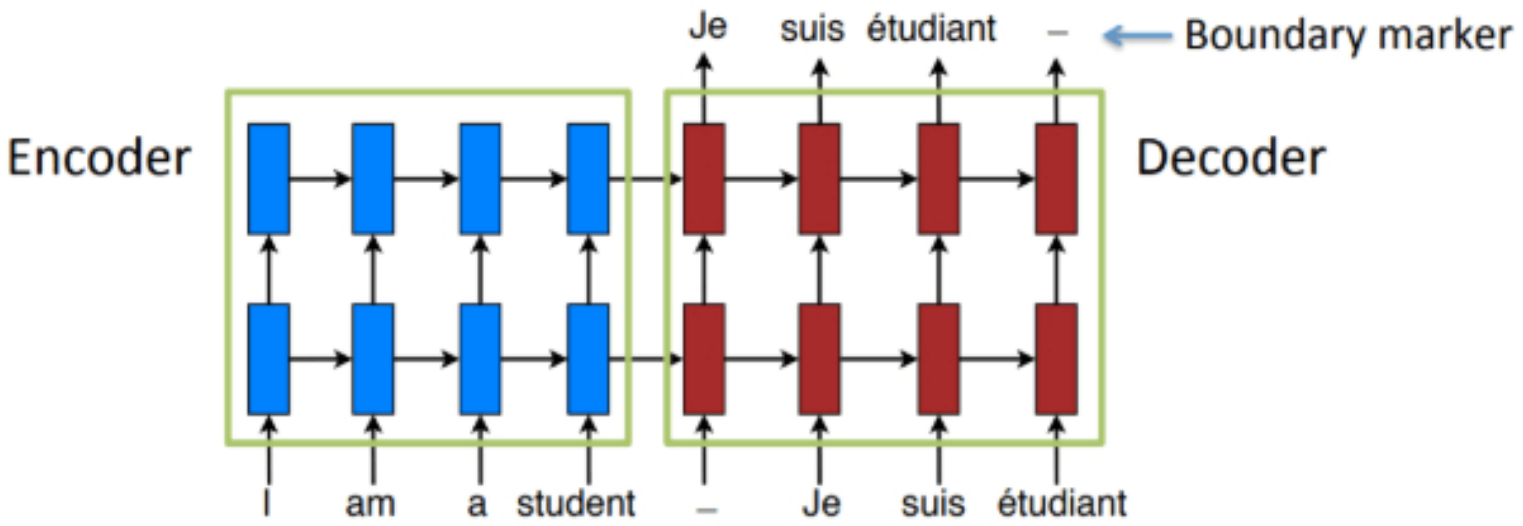


Figure 15-4. Sequence-to-sequence (top left), sequence-to-vector (top right), vector-to-sequence (bottom left), and encoder–decoder (bottom right) networks

What type of an RNN?



What type of an RNN?



Questions?

Thank you!