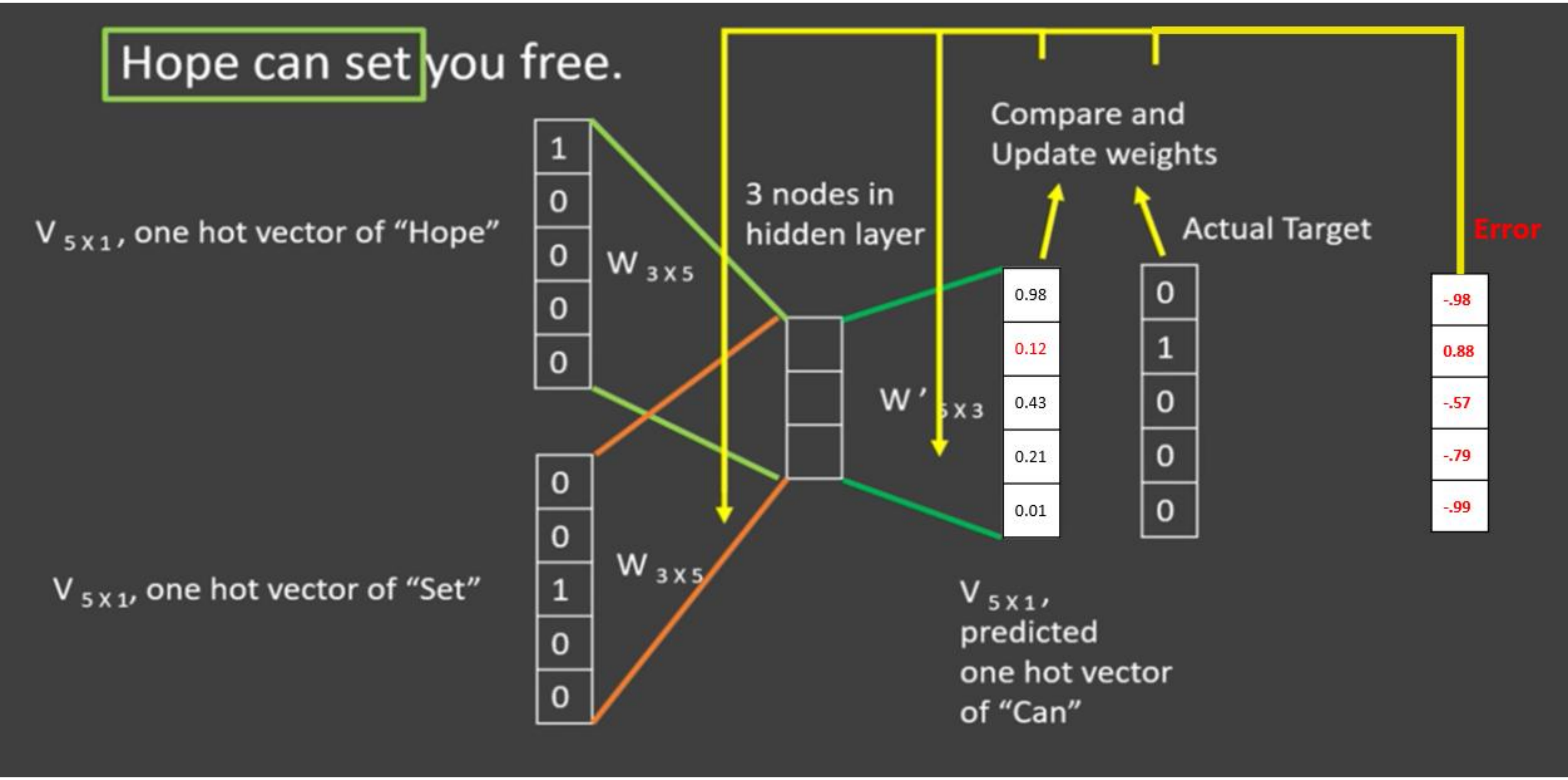
AI Bootcamp
NLP & LLMs

Text Classification

Dr Usman Zia
Asst Professor
SINES, NUST
usman.zia@sines.nust.edu.pk

Dense Embeddings



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Dense Embeddings

Getting word embeddings

Weights after training

W_{3x5}

w00	w01	w02	w03	w04
w10	w11	w12	w13	w14
w20	w21	w22	w23	w24

One Hot vector of words V_{5x1}

Hope	can	set	you	free
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Word Vector for hope $= W_{3x5} \times V_{5x1}$

$$\begin{bmatrix} w00 & w01 & w02 & w03 & w04 \\ w10 & w11 & w12 & w13 & w14 \\ w20 & w21 & w22 & w23 & w24 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} w00 \\ w10 \\ w20 \end{bmatrix}$$

V_{3x1}

Word Vector for Hope

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

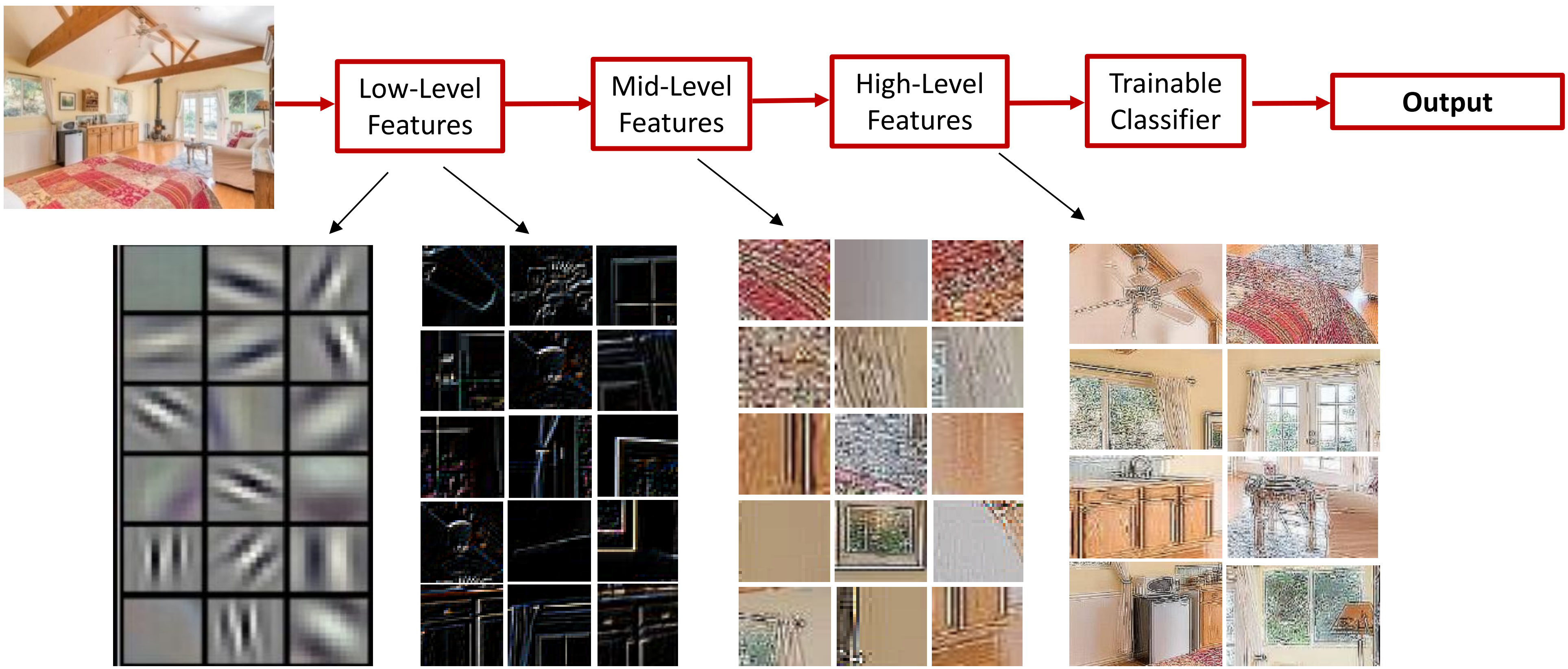
AI Bootcamp
NLP & LLMs

Text Classification using CNNs

Dr Usman Zia
Asst Professor
SINES, NUST
usman.zia@sines.nust.edu.pk

Convolutional Neural Networks (CNNs)

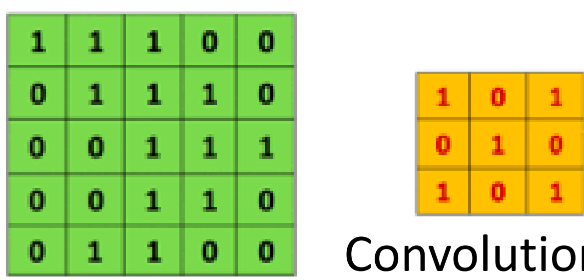
- DL applies a multi-layer process for learning rich hierarchical features (i.e., data representations)
  - Input image pixels → Edges → Textures → Parts → Objects



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

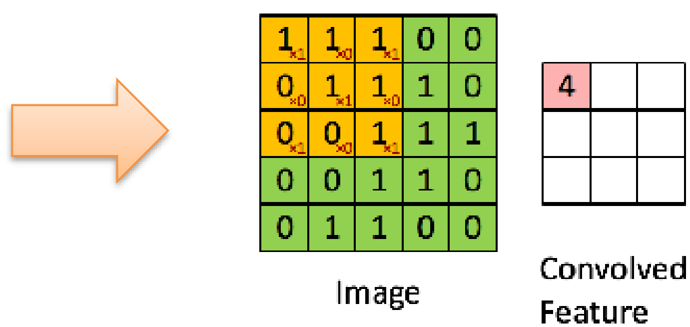
Convolutional Neural Networks (CNNs)

- *Convolutional neural networks* (CNNs) were primarily designed for image data
- CNNs use a convolutional operator for extracting data features
  - Allows parameter sharing
  - Efficient to train
  - Have less parameters than NNs with fully-connected layers
- CNNs are robust to spatial translations of objects in images
- A convolutional filter slides (i.e., convolves) across the image



Input matrix

Convolutional 3x3 filter



Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- Convolutional Layers
  - A convolution layer has several filters that perform the convolution operation.
  - Every image is considered as a matrix of pixel values.

Image pixels

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Filter

1	0	1
0	1	0
1	0	1

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- Convolutional Layers
  - When the convolutional filters are scanned over the image, they capture useful features
  - E.g., edge detection by convolutions
  - Convolution later is typically followed by an **activation layer** (ReLU being most used activation layer)

Filter 

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Input Image

Convoluted Image

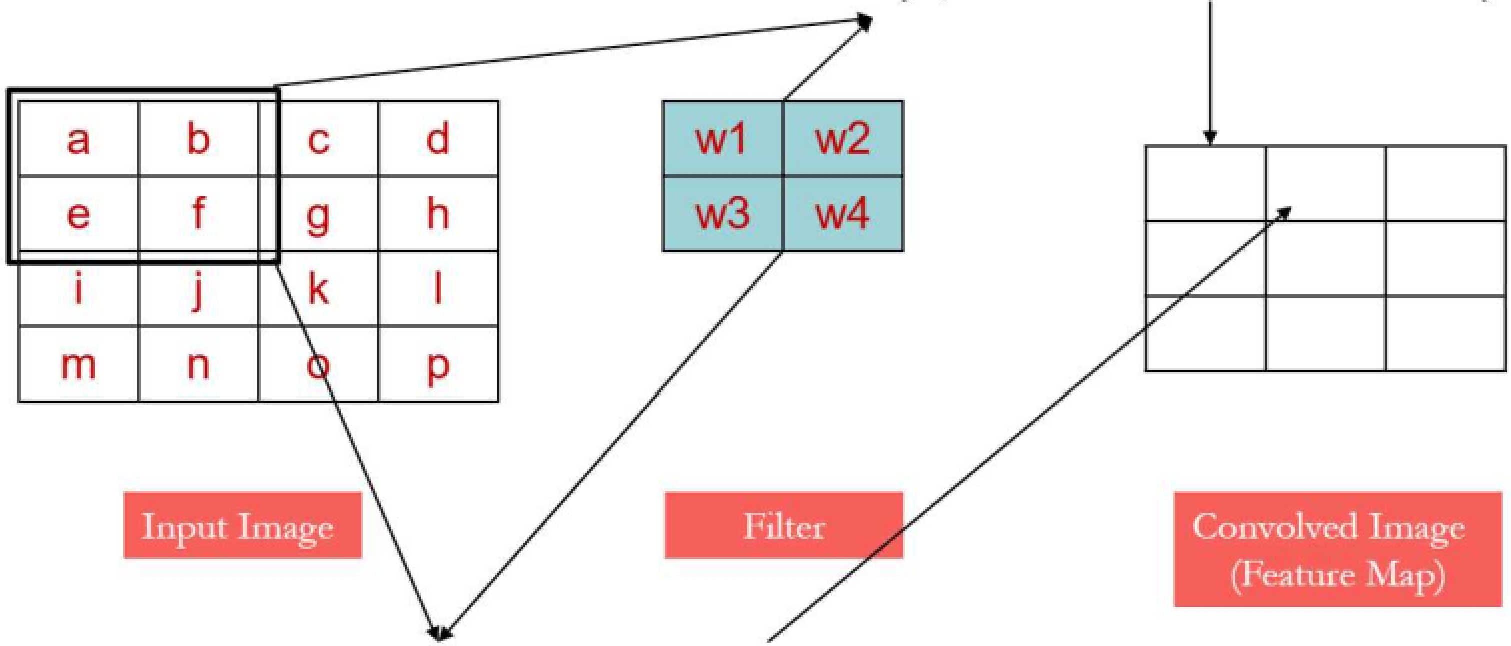
usman.zia@sines.nust.edu.pk
linkedin.com/in/usmanxia/

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- What is Convolution?

$$h1 = f(a * w1 + b * w2 + e * w3 + f * w4)$$



$$h2 = f(b * w1 + c * w2 + f * w3 + g * w4)$$

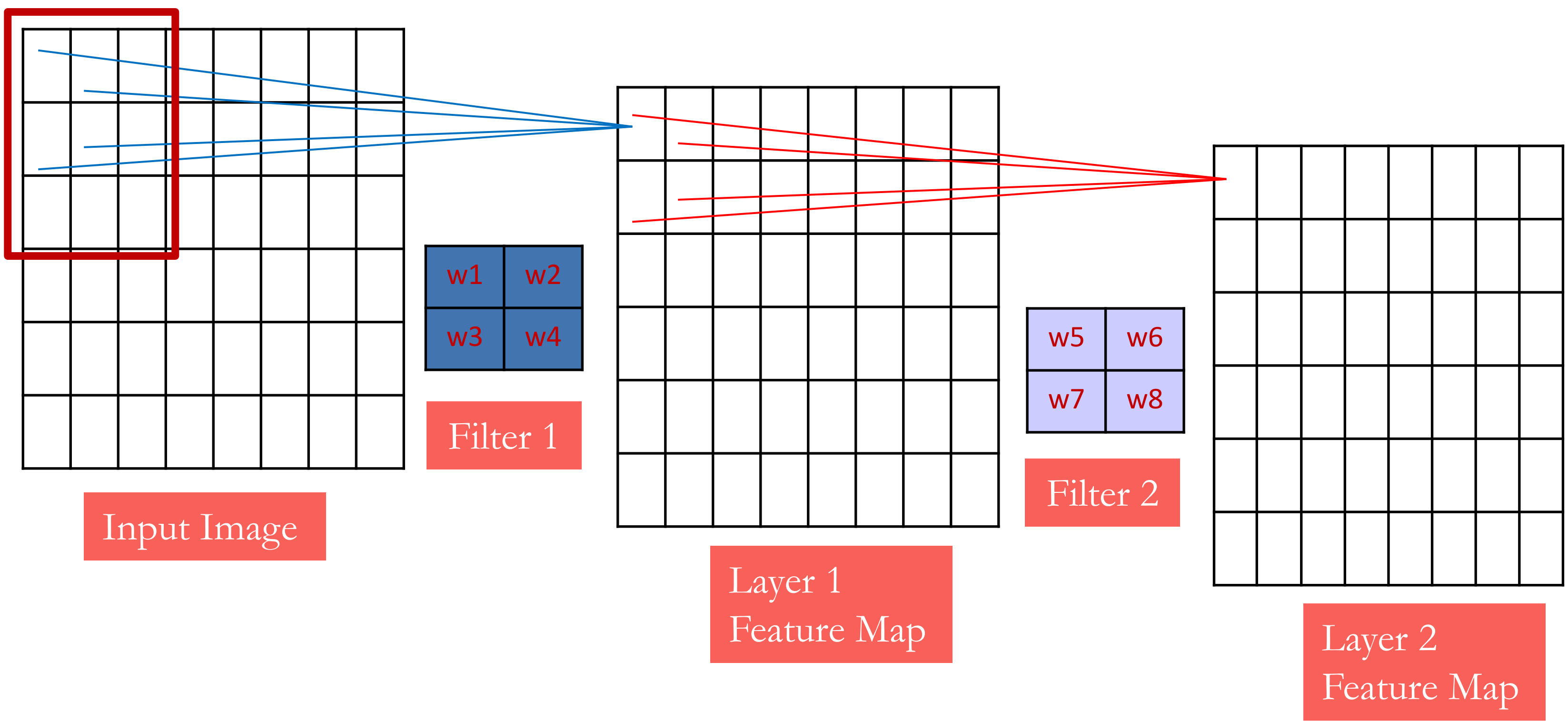
Number of Parameters for one feature map = 4

Number of Parameters for 100 feature map = 4*100

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- In CNNs, hidden units in a layer are only connected to a small region of the layer before it (called local receptive field)
  - The depth of each feature map corresponds to the number of convolutional filters used at each layer



Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- ***Pooling Layer*** is a down-sampling operation that reduces the dimensionality of the feature map. The rectified feature map now goes through a pooling layer to generate a pooled feature map.
- ***Type of Pooling Layers***
  - ***Max pooling***: reports the maximum output within a rectangular neighborhood
  - ***Average pooling***: reports the average output of a rectangular neighborhood
- Pooling layers reduce the spatial size of the feature maps
  - Reduce the number of parameters, prevent overfitting

MaxPool with a 2×2 filter with stride of 2

1	3	5	3
4	2	3	1
3	1	1	3
0	1	0	4

Input Matrix

4	5
3	4

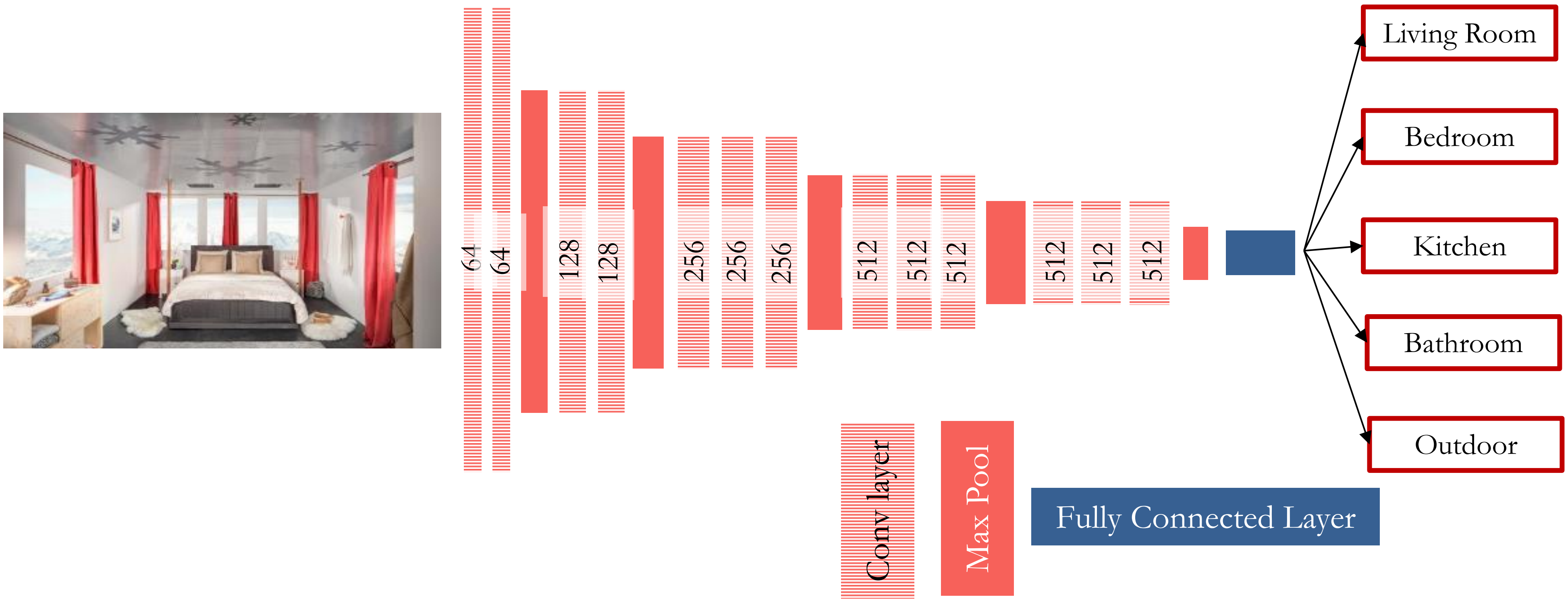
Output Matrix

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks

- Feature extraction architecture
  - After 2 convolutional layers, a max-pooling layer reduces the size of the feature maps (typically by 2)
  - A fully convolutional and a softmax layers are added last to perform classification



usman.zia@sines.nust.edu.pk
linkedin.com/in/usmanzia/

A 1D convolution for text

tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3

t,d,r	−1.0
d,r,t	−0.5
r,t,k	−3.6
t,k,g	−0.2
k,g,o	0.3

Apply a **filter** (or **kernel**) of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

1D convolution for text with padding

∅	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
∅	0.0	0.0	0.0	0.0

∅,t,d	−0.6
t,d,r	−1.0
d,r,t	−0.5
r,t,k	−3.6
t,k,g	−0.2
k,g,o	0.3
g,o,∅	−0.5

Apply a **filter** (or **kernel**) of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

3 channel 1D convolution with padding = 1

Ø	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
Ø	0.0	0.0	0.0	0.0

Ø,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,Ø	−0.5	−0.9	0.1

Apply 3 **filters** of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

1	0	0	1
1	0	−1	−1
0	1	0	1

1	−1	2	−1
1	0	−1	3
0	2	2	1

Could also use (zero) padding = 2

Also called "wide convolution"

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanzia/

conv1d, padded with max pooling over time

∅	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
∅	0.0	0.0	0.0	0.0

∅,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,∅	−0.5	−0.9	0.1

max p	0.3	1.6	1.4

Apply 3 **filters** of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

1	0	0	1
1	0	−1	−1
0	1	0	1

1	−1	2	−1
1	0	−1	3
0	2	2	1

conv1d, padded with ave pooling over time

∅	0.0	0.0	0.0	0.0
tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3
∅	0.0	0.0	0.0	0.0

∅,t,d	−0.6	0.2	1.4
t,d,r	−1.0	1.6	−1.0
d,r,t	−0.5	−0.1	0.8
r,t,k	−3.6	0.3	0.3
t,k,g	−0.2	0.1	1.2
k,g,o	0.3	0.6	0.9
g,o,∅	−0.5	−0.9	0.1

ave p	−0.87	0.26	0.53

Apply 3 **filters** of size 3

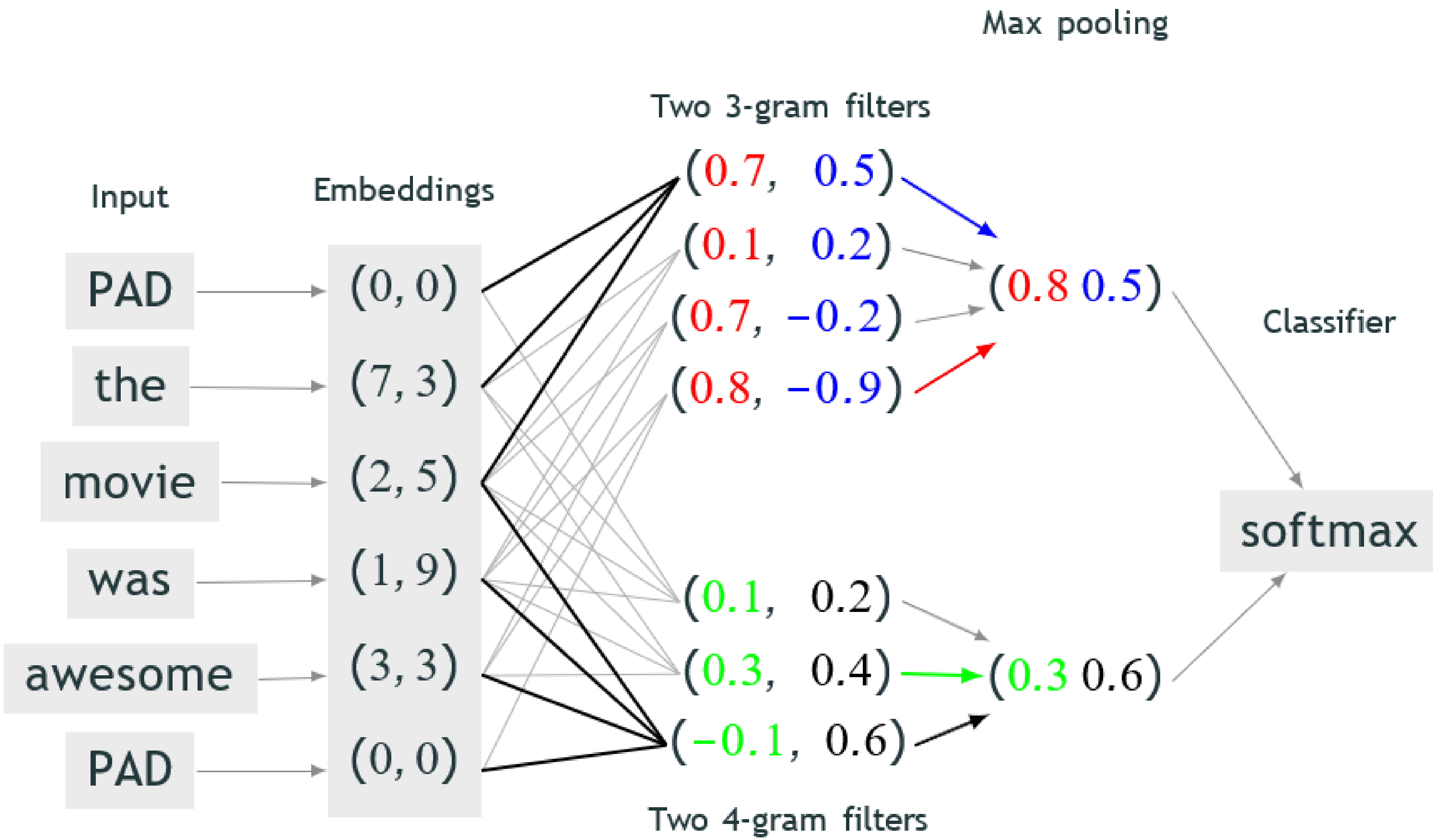
3	1	2	−3
−1	2	1	−3
1	1	−1	1

1	0	0	1
1	0	−1	−1
0	1	0	1

1	−1	2	−1
1	0	−1	3
0	2	2	1

Classification with convolution and pooling

Convolutional Neural Networks



Efficacy of CNNs in NLP

Convolutional Neural Networks

- In computer vision
  - Effectiveness of deep CNNs can be very well explained
  - Primary conv. layers detect the edges of an object
  - As we go deeper, more complex features of an image are learnt
- In NLP
  - Not much understanding

AI Bootcamp
NLP & LLMs

SEQUENCE MODELS

Dr Usman Zia
Asst Professor
SINES, NUST
usman.zia@sines.nust.edu.pk

Why Sequence Models?

to model sequences, we need:

1. to deal with variable-length sequences
2. to maintain sequence order
3. to keep track of long-term dependencies
4. to share parameters across the sequence

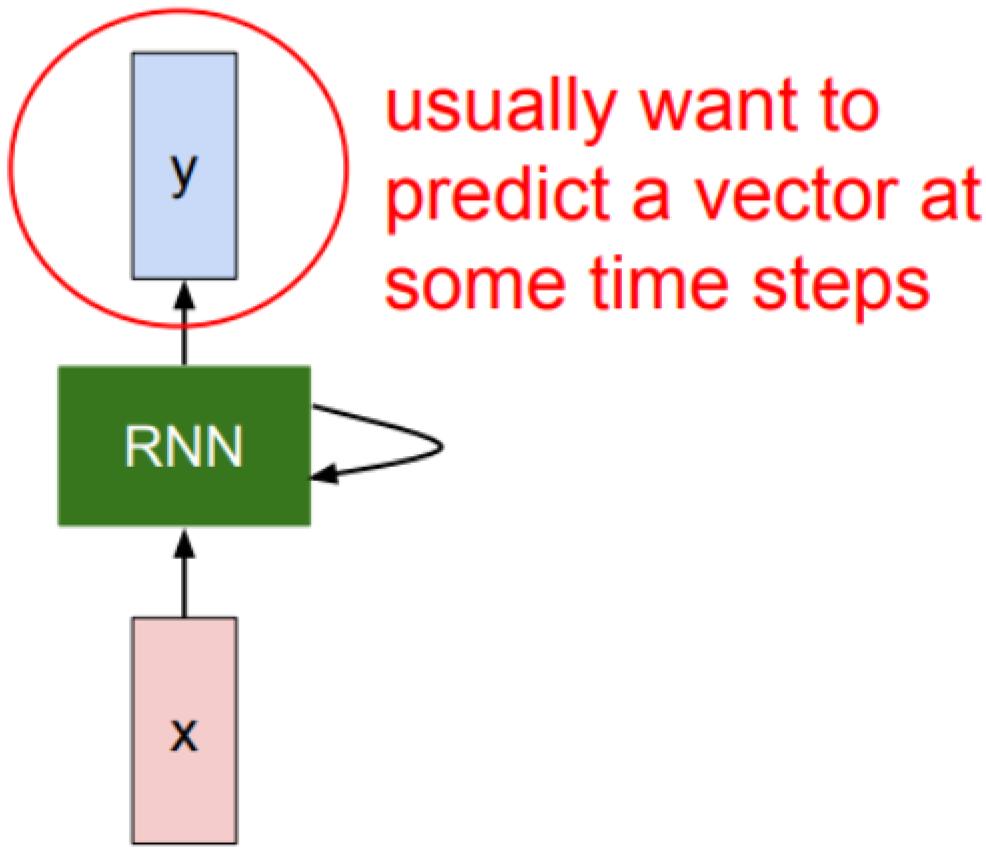
usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Recurrent Neural Networks (RNNs)

- **Rationale:**
  - Rather than fixing the amount of history our network can handle, we allow it to **accumulate a representation over time**.
  - This can work over arbitrary sequences.
  - Hopefully, it will also encode similar things in similar ways (less representational redundancy).
  - In the accumulated representation, it should also capture dependencies between elements at different (possibly distant) time-steps.

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Recurrent Neural Networks (RNNs)



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Recurrent Neural Networks (RNNs)

We can process a sequence of vectors **x** by applying a **recurrence formula** at every time step:

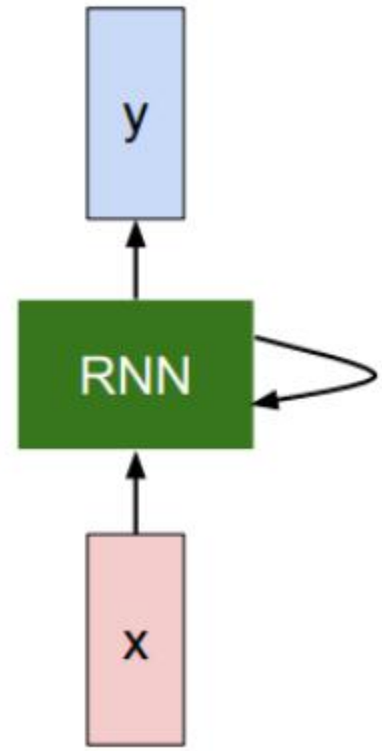
$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

some function
with parameters W

old state

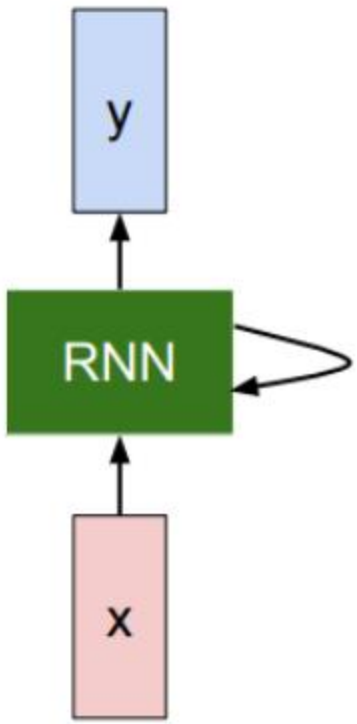
input vector at
some time step



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

Recurrent Neural Networks (RNNs)

The state consists of a single *"hidden"* vector **h**:



$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

RNN Computational Graph

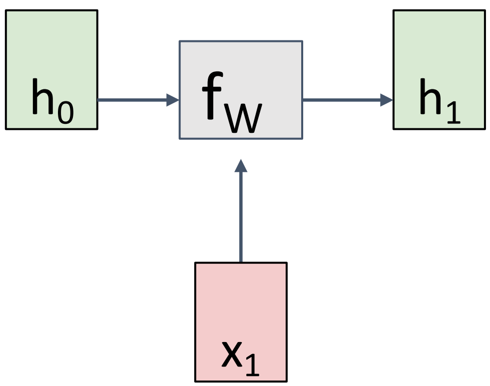
Initial hidden state
Either set to all 0,
Or learn it

h_0

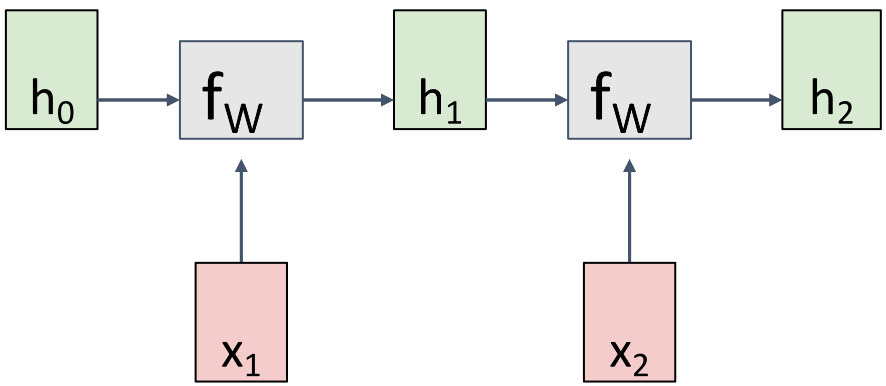
x_1

usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

RNN Computational Graph

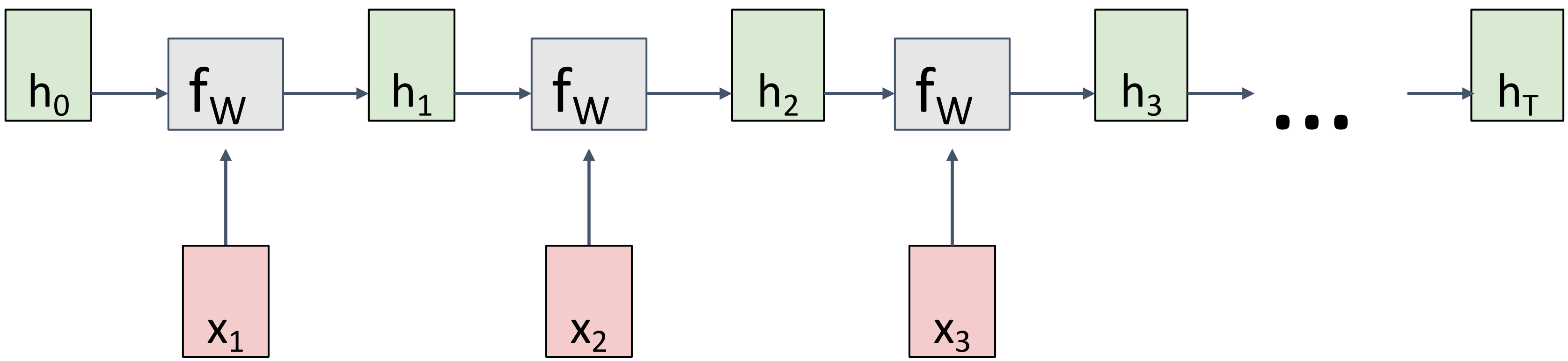


RNN Computational Graph



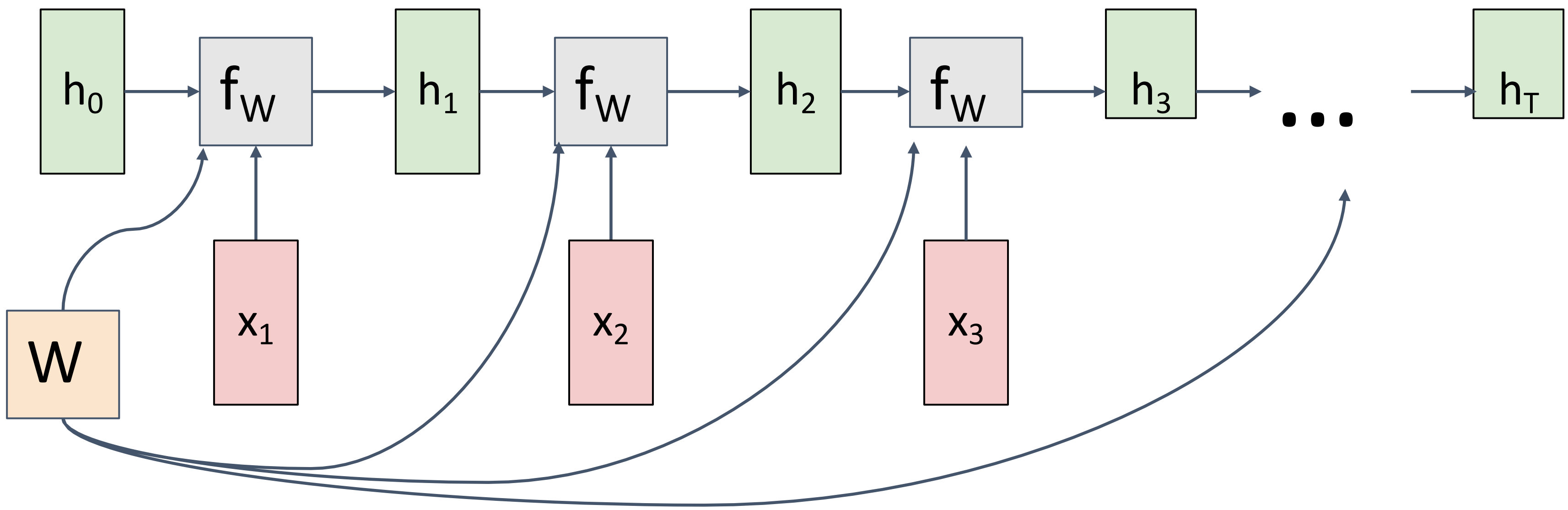
usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

RNN Computational Graph

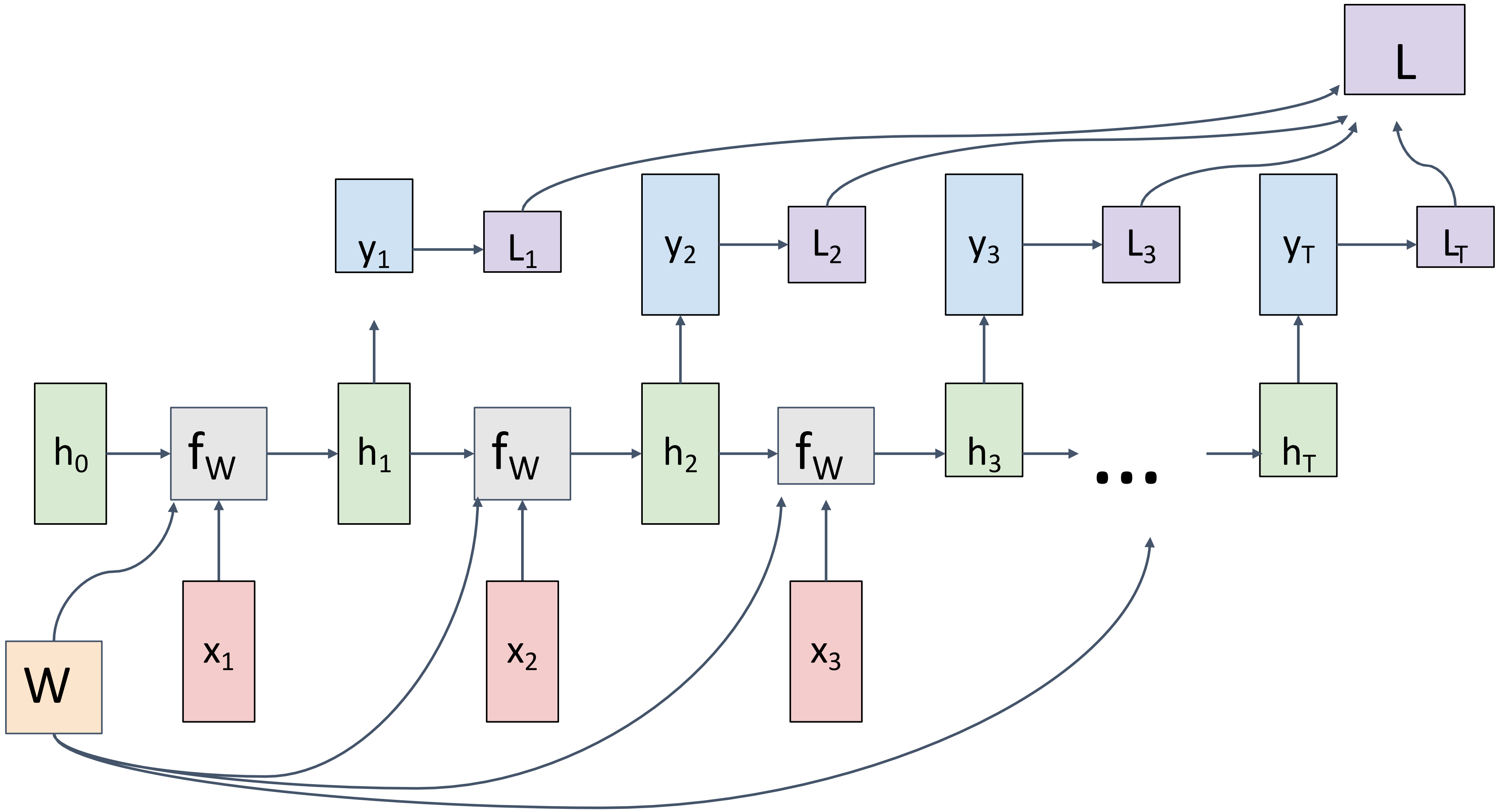


RNN Computational Graph

Re-use the same weight matrix at every time-step

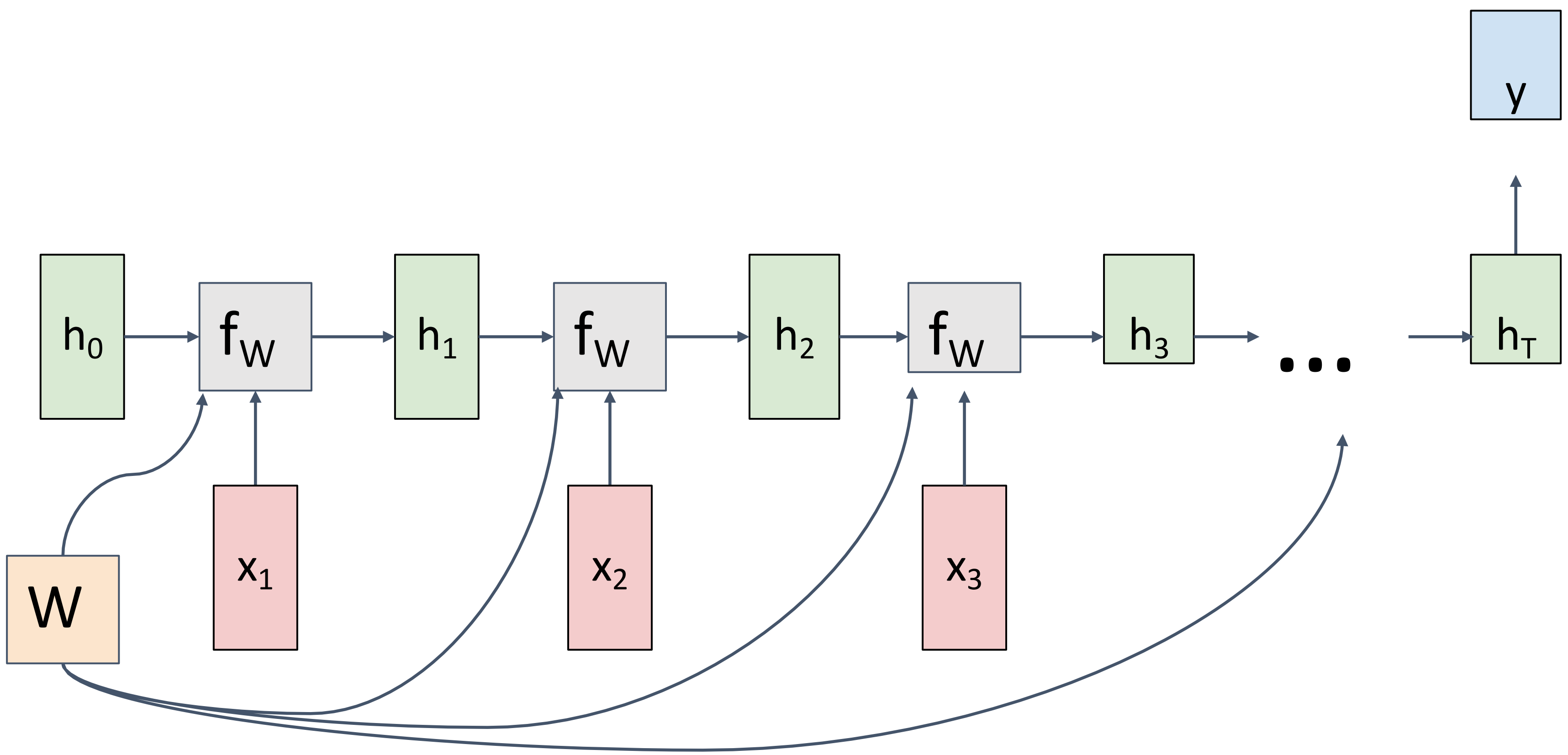


RNN Computational Graph (Many to Many)



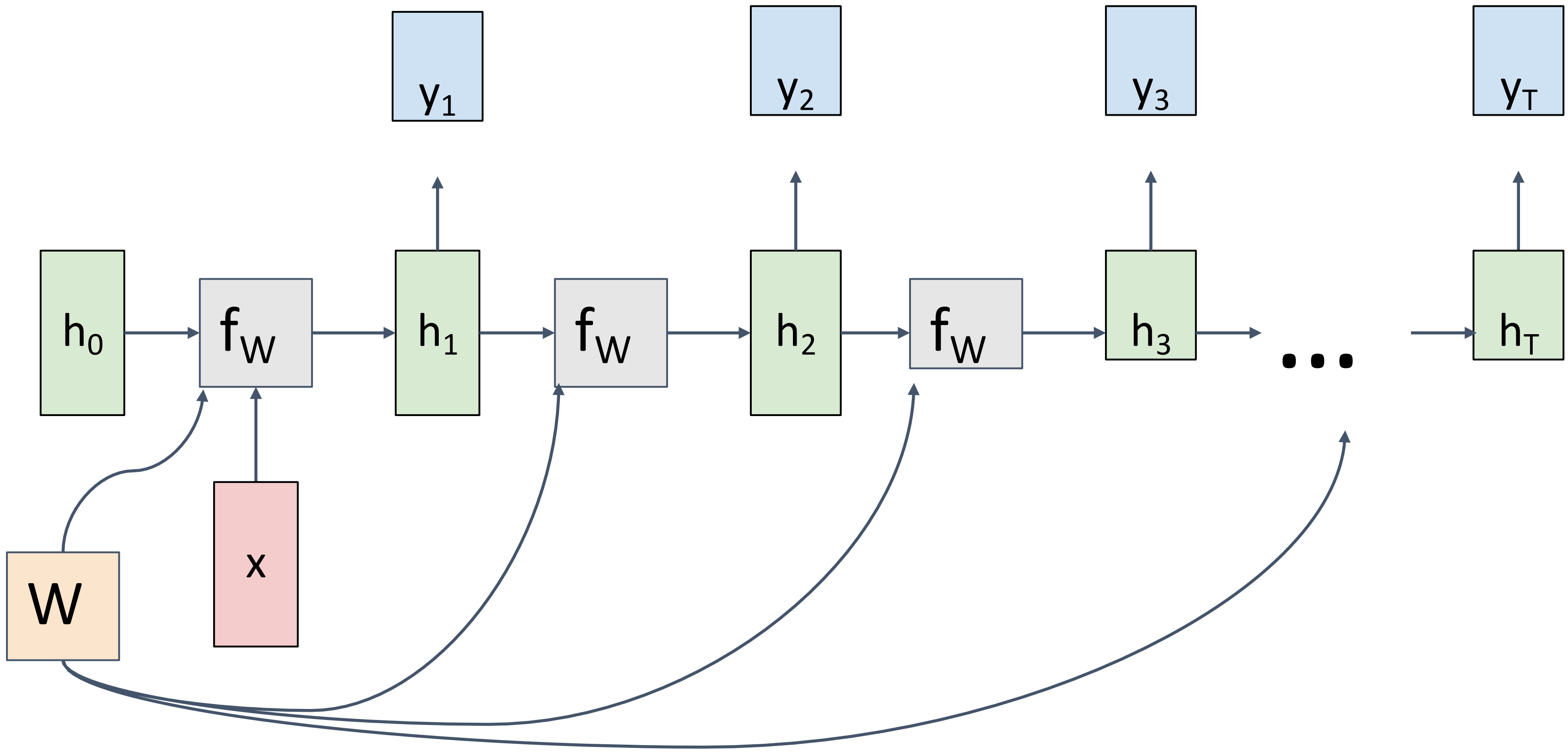
usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

RNN Computational Graph (Many to One)



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

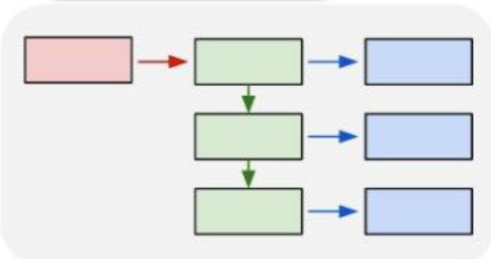
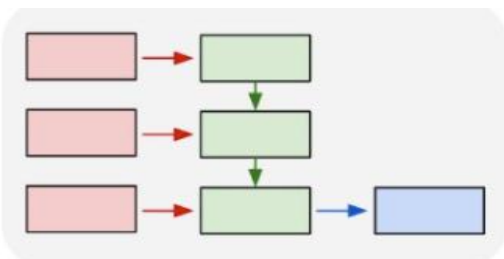
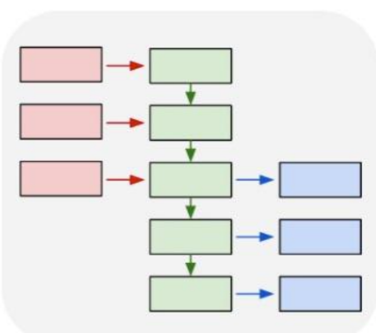
RNN Computational Graph (One to Many)



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Recurrent Neural Networks (RNNs)

- Use cases for RNNs

RNN	Application	Input	Output
 one to many	Image Captioning		A person riding a motorbike on dirt road
 many to one	Sentiment Analysis	Awesome movie. Highly recommended.	Positive
 many to many	Machine Translation Video Classification	Happy Eid	عيد مبارک

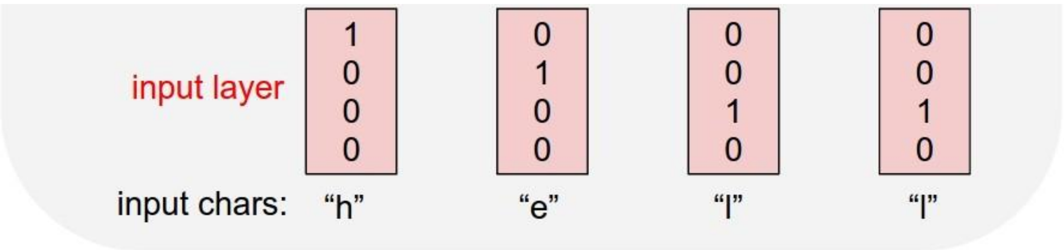
usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Example: Language Modeling

Given characters 1, 2, …, t,
model predicts character t

Training sequence: ”hello”

Vocabulary: [h, e, l, o]



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

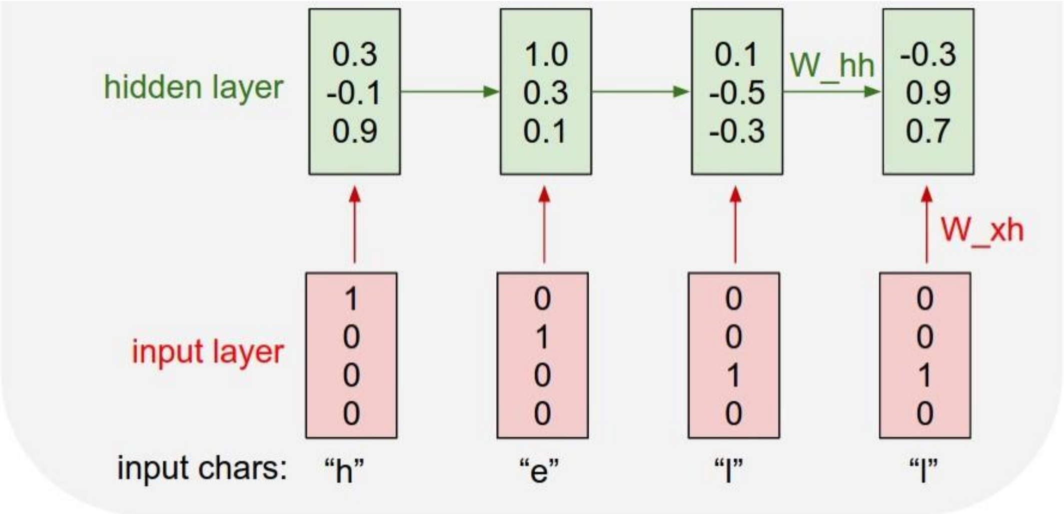
Example: Language Modeling

Given characters 1, 2, ..., t,
model predicts character t

$$\boxed{h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)}$$

Training sequence: ”hello”

Vocabulary: [h, e, l, o]



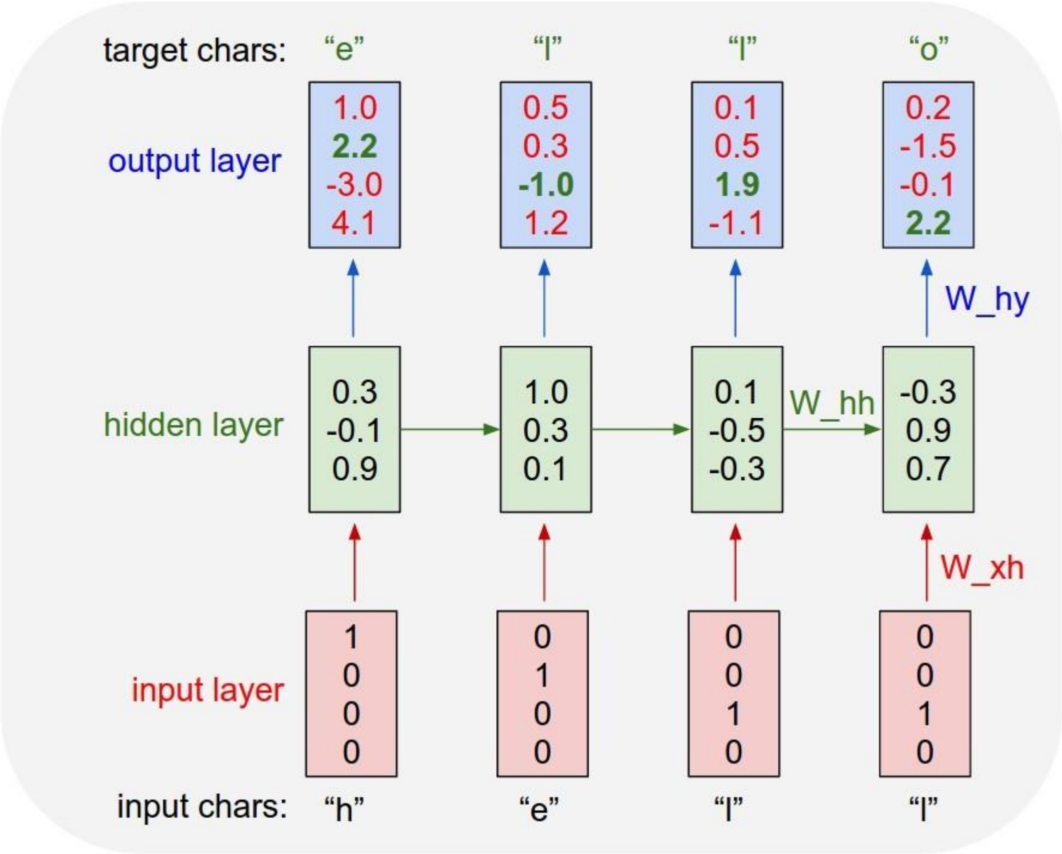
Example: Language Modeling

Given characters 1, 2, ..., t, model predicts character t

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

Training sequence: ”hello”

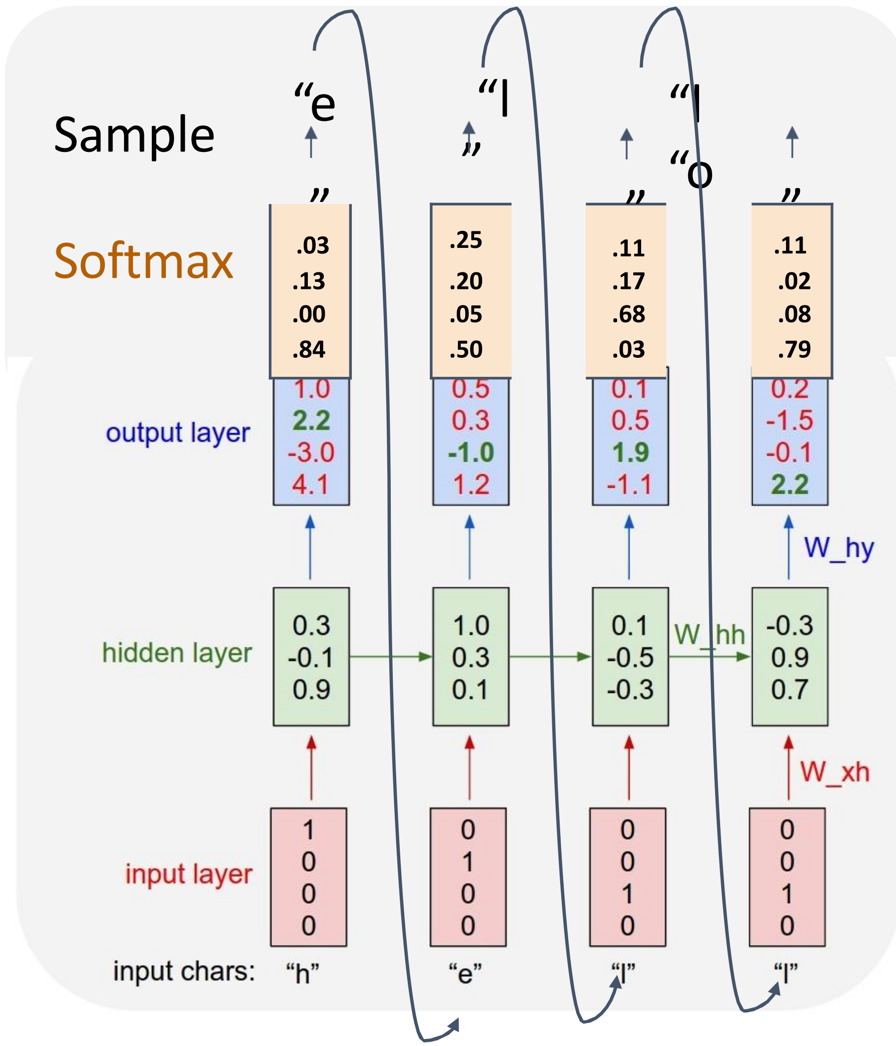
Vocabulary: [h, e, l, o]



usman.zia@sines.nust.edu.pk linkedin.com/in/usmanxia/

Example: Language Modeling

At test-time, **generate** new text: sample characters one at a time, feed back to model

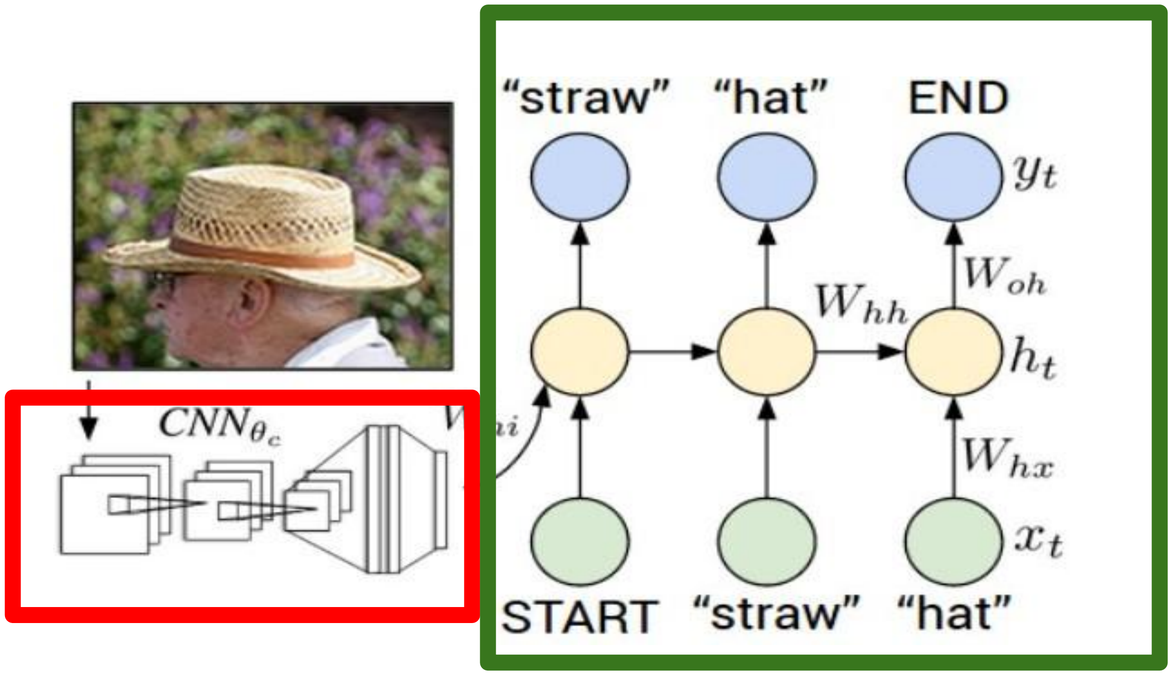


Training sequence: "hello"

Vocabulary: [h, e, l, o]

usman.zia@sines.nust.edu.pk
linkedin.com/in/usmanxia/

Example: Image Captioning



Recurrent Neural Network

Convolutional Neural Network

Example: Image Captioning

before:

$h = \tanh(W_{xh}*x + W_{hh}*h)$

now:

$h = \tanh(W_{xh}*x + W_{hh}*h + W_{ih}*v)$

W_{ih}

Stop after sampling <END> token