TURKU AMK

Deep Learning

What is a Neural Network?

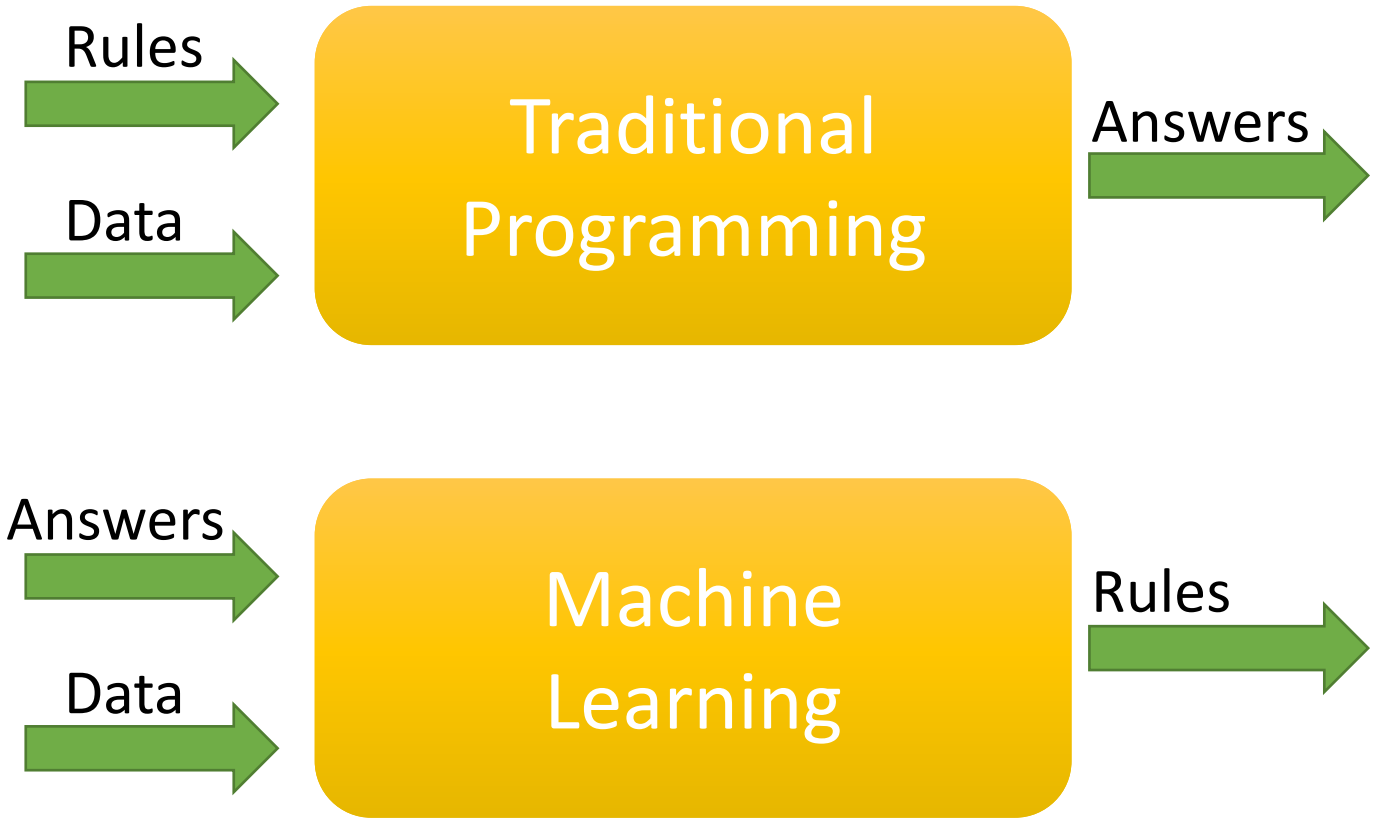
- Lecturer:

Amin Majd (DSc and PhD)

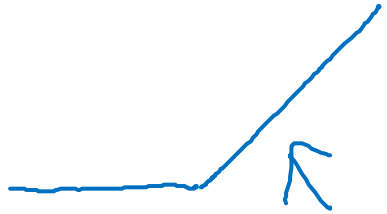
amin.majd@turkuamk.fi

- References: deeplearning.ai

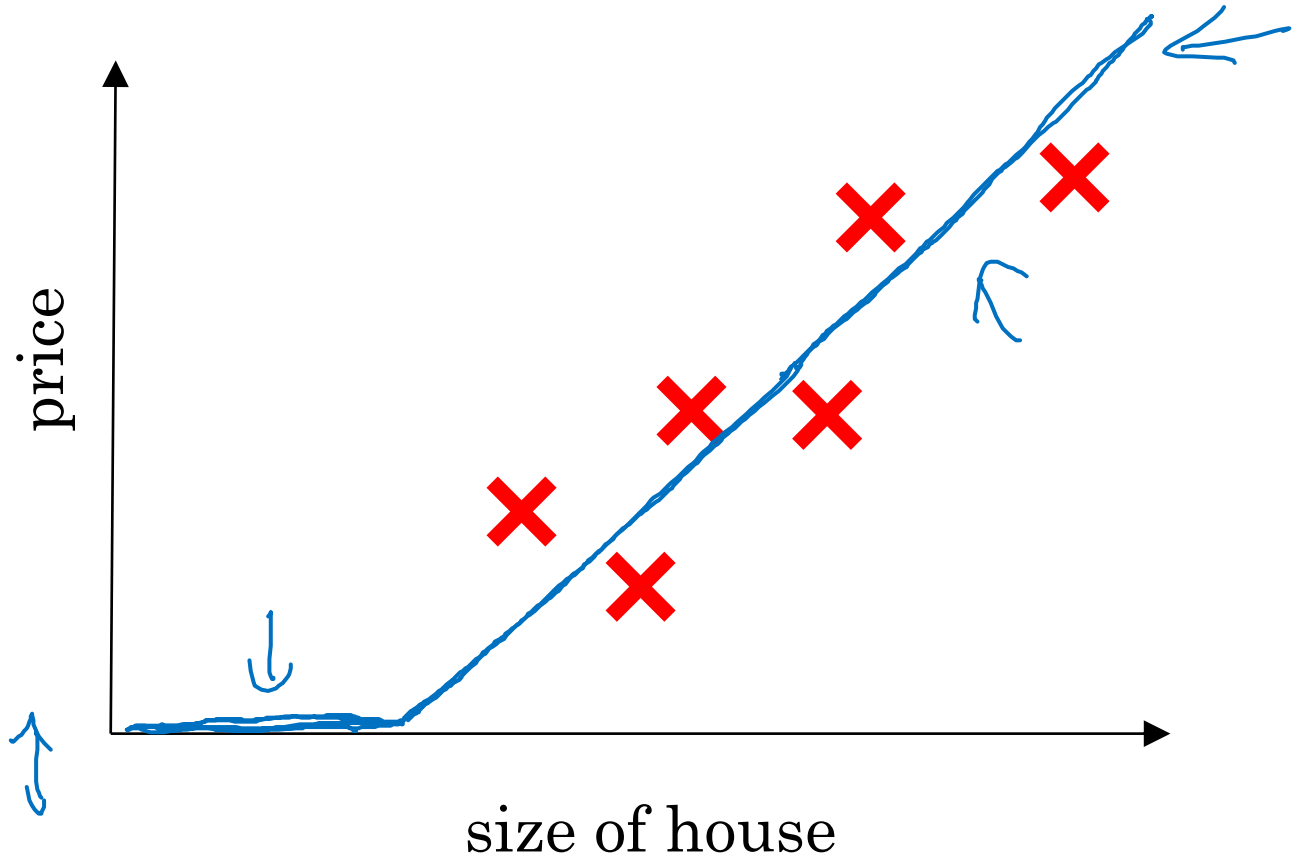
Traditional Programming vs Machine Learning

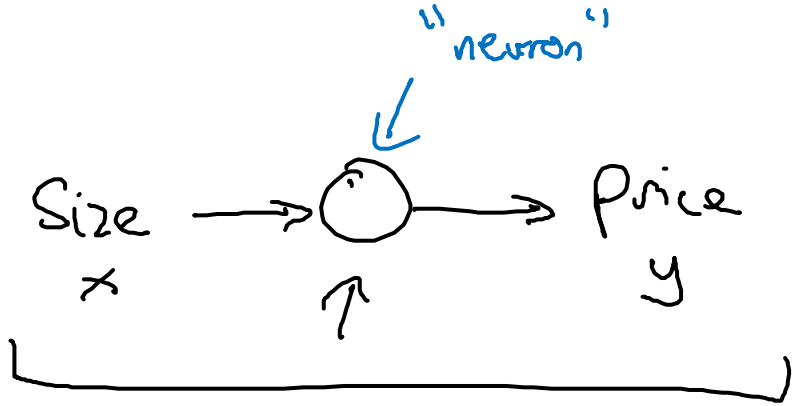


Housing Price Prediction

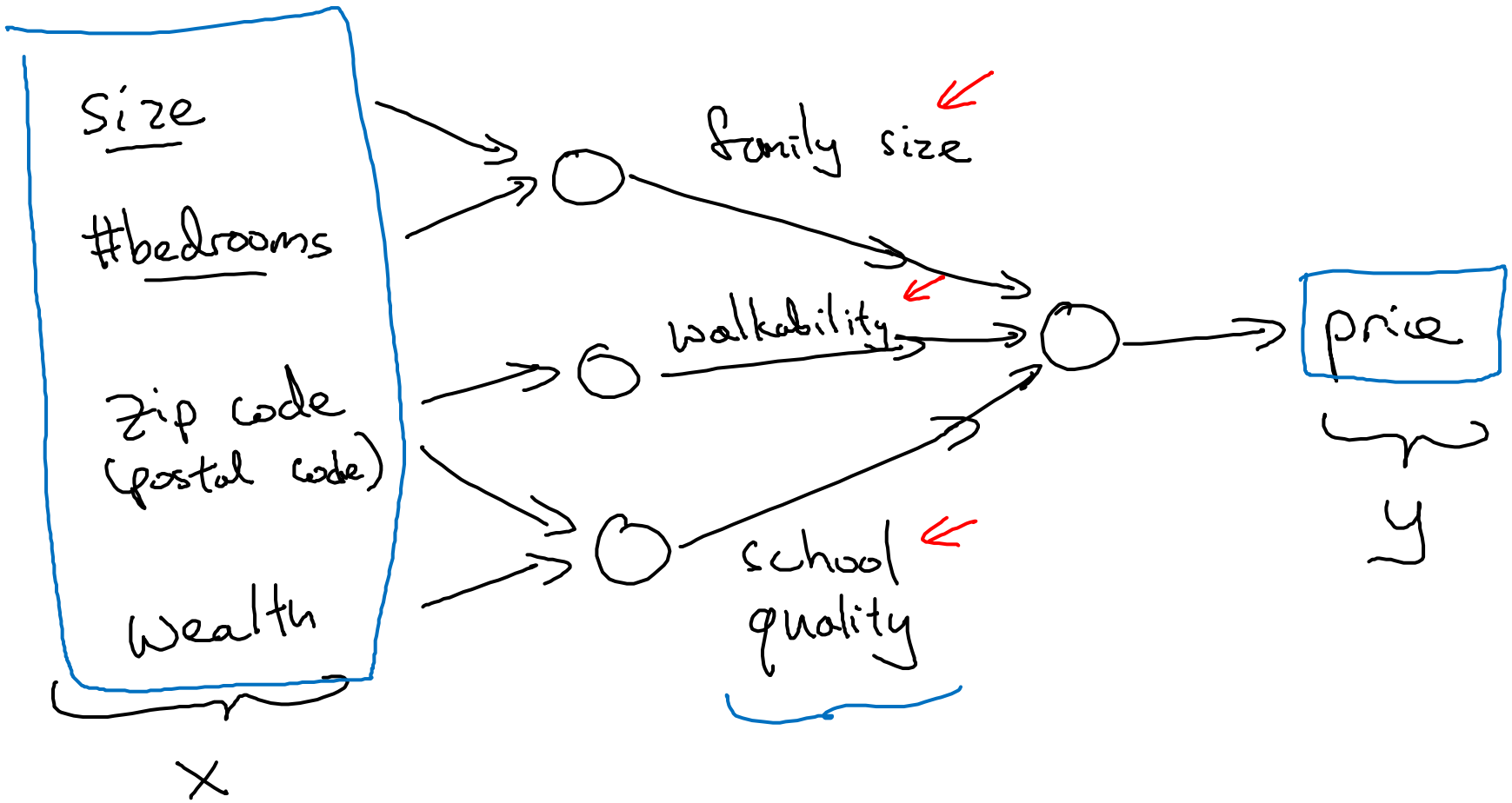


ReLU
Rectified
Linear
Unit

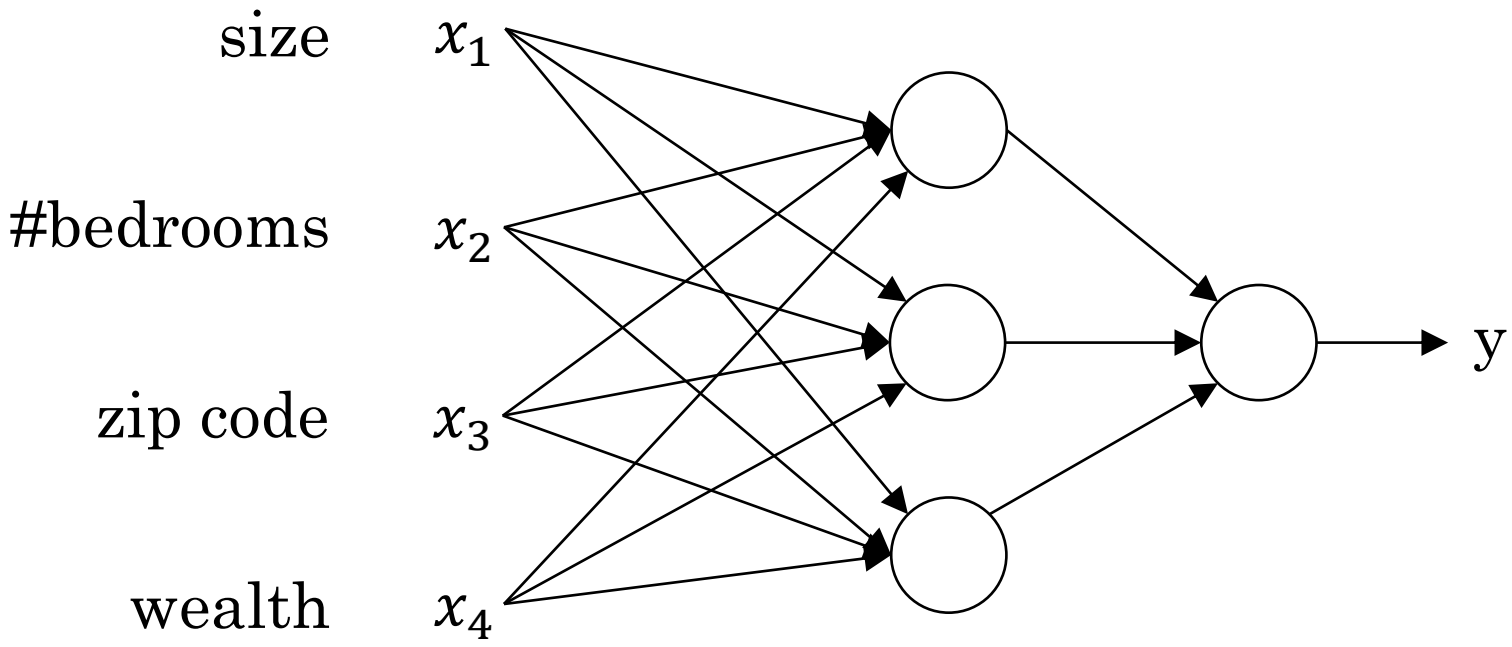




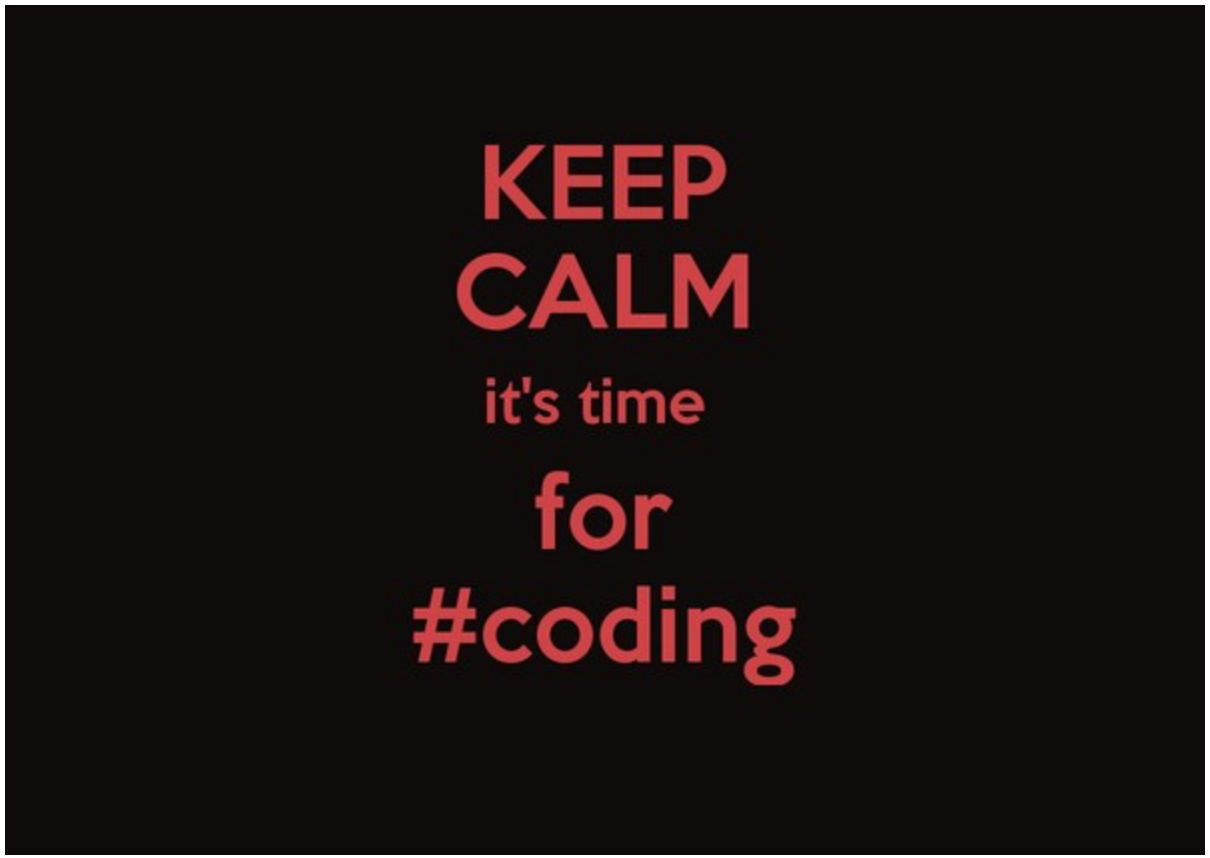
Housing Price Prediction



Housing Price Prediction



Drawing of previous Image


KEEP
CALM
it's time
for
#coding

Introduction to Deep Learning

Supervised Learning with Neural Networks

Supervised Learning

Input(x)	Output (y)	Application
Home features	Price	Real Estate
Ad, user info	Click on ad? (0/1)	Online Advertising
Image	Object (1,…,1000)	Photo tagging
Audio	Text transcript	Speech recognition
English	Chinese	Machine translation
Image, Radar info	Position of other cars	Autonomous driving

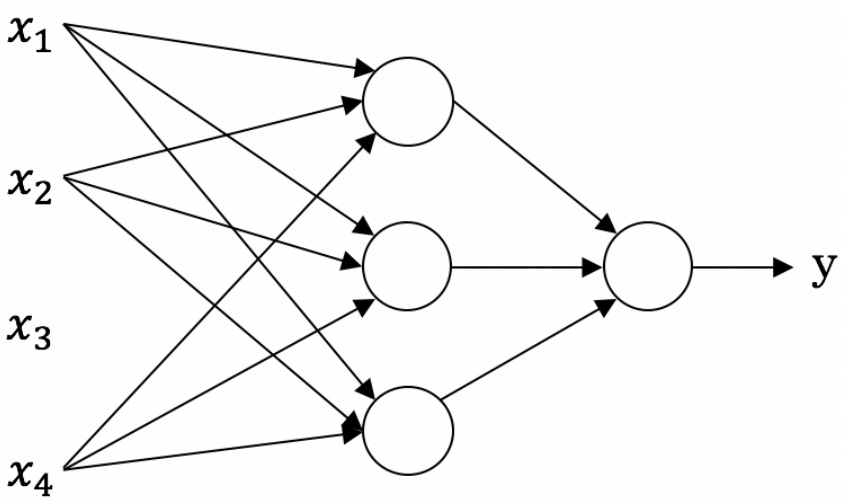
Standard NN

CNN

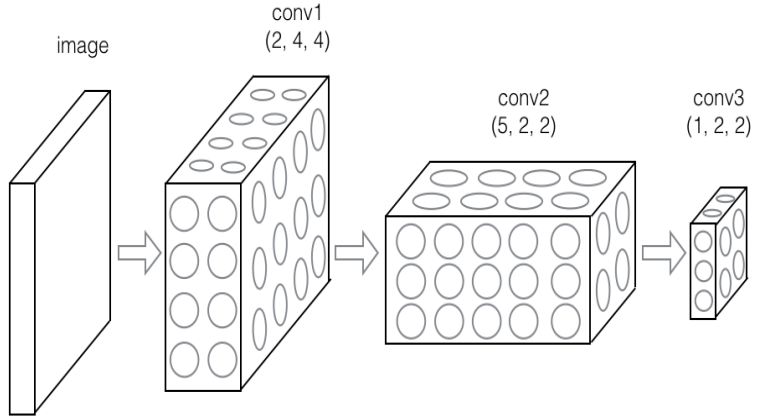
RNN

Custom/ Hybrid

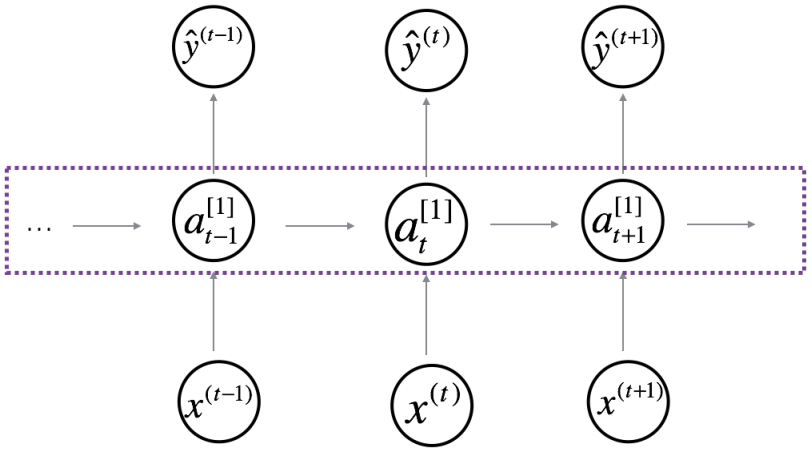
Neural Network examples



Standard NN



Convolutional NN



Recurrent NN

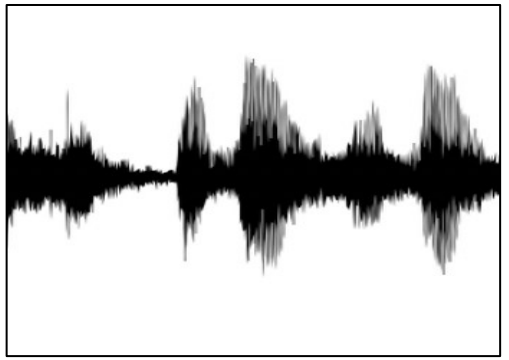
Supervised Learning

Structured Data

Size	#bedrooms	⋯	Price (1000$s)
2104	3		400
1600	3		330
2400	3		369
⋮	⋮		⋮
3000	4		540

User Age	Ad Id	⋯	Click
41	93242		1
80	93287		0
18	87312		1
⋮	⋮		⋮
27	71244		1

Unstructured Data



Audio

Image

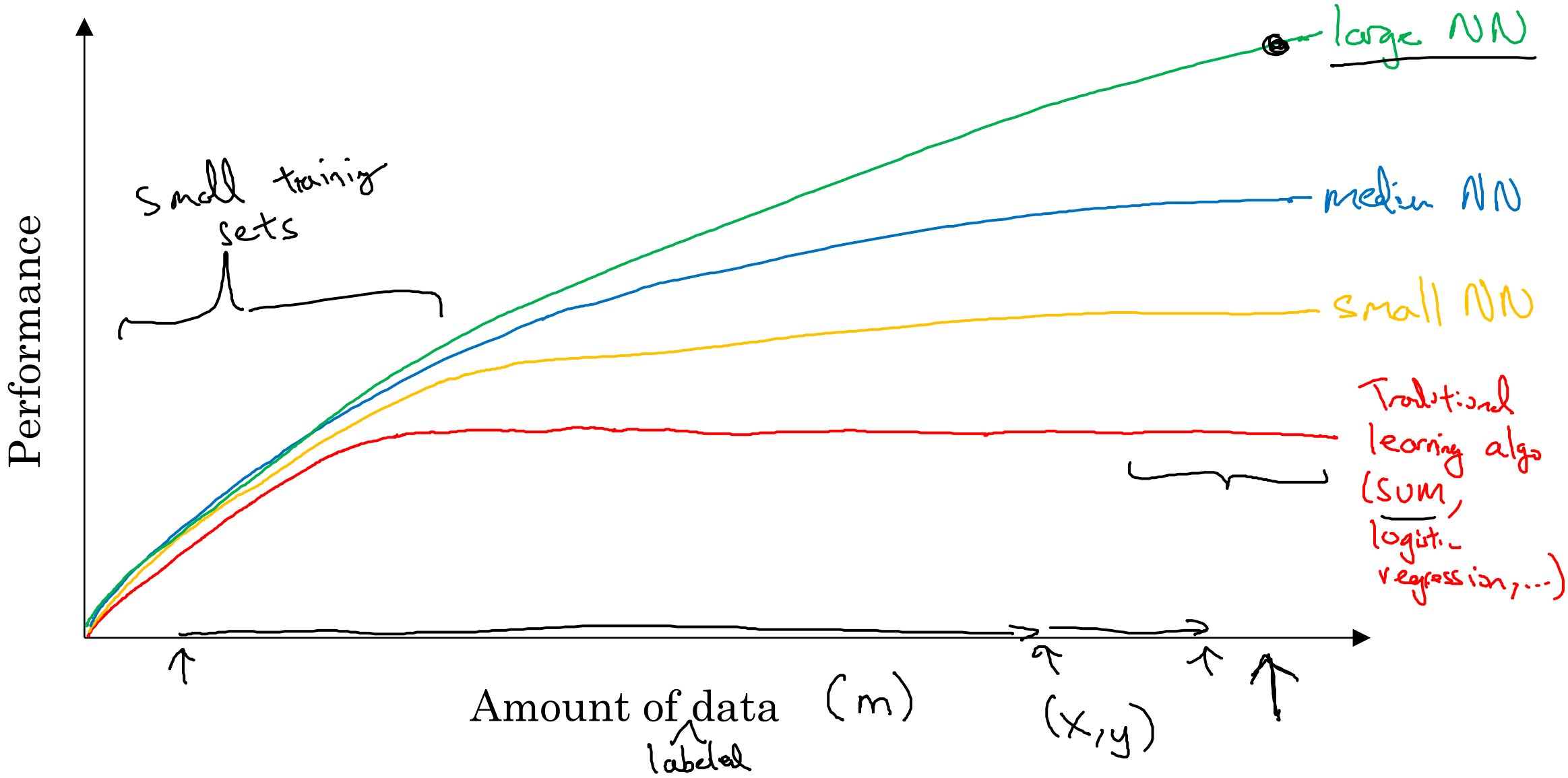
Four scores and seven years ago…

Text

- Using Tensorflow and Keras.

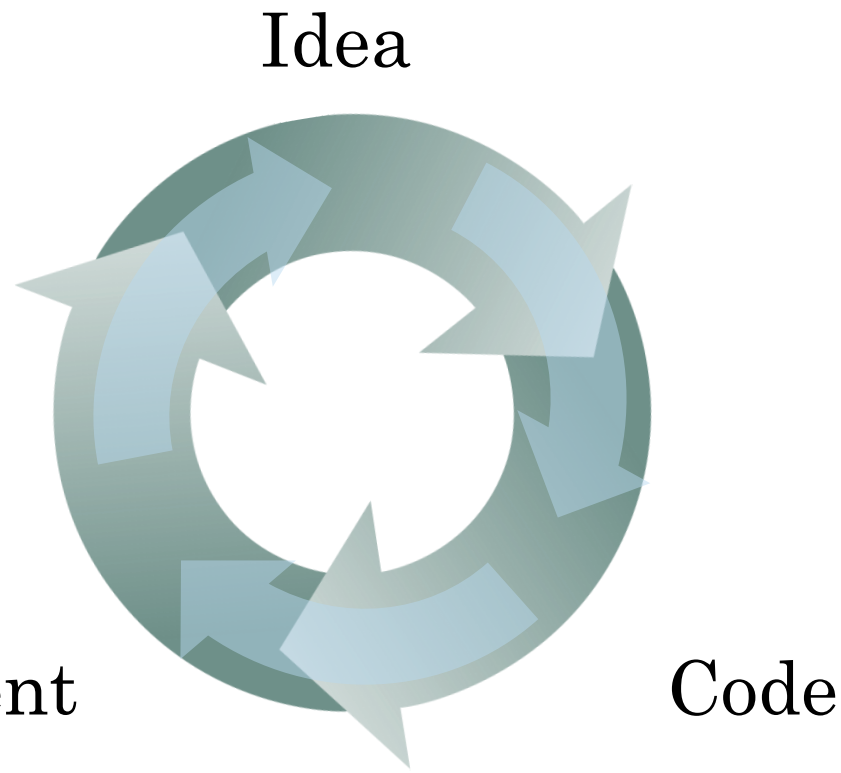
- Using CSC, Google Colab, etc.

Scale drives deep learning progress



Scale drives deep learning progress

- Data
- Computation
- Algorithms



Basics of Neural Network Programming

Binary Classification

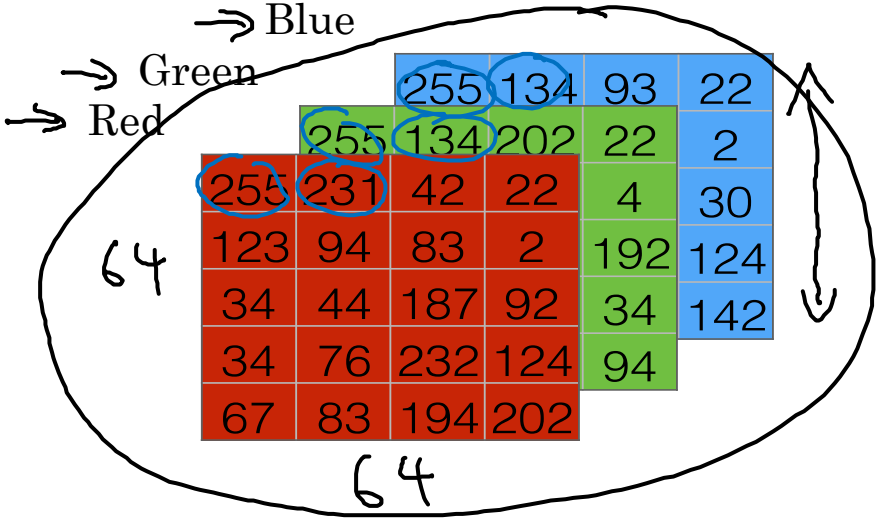
Binary Classification

64

64

1 (cat) vs 0 (non cat)

y



$$x = \begin{bmatrix} 255 \\ 231 \\ \vdots \\ \vdots \\ 255 \\ 134 \\ \vdots \end{bmatrix}$$

$64 \times 64 \times 3 = 12288$

$n = n_x = 12288$

$x \longrightarrow y$

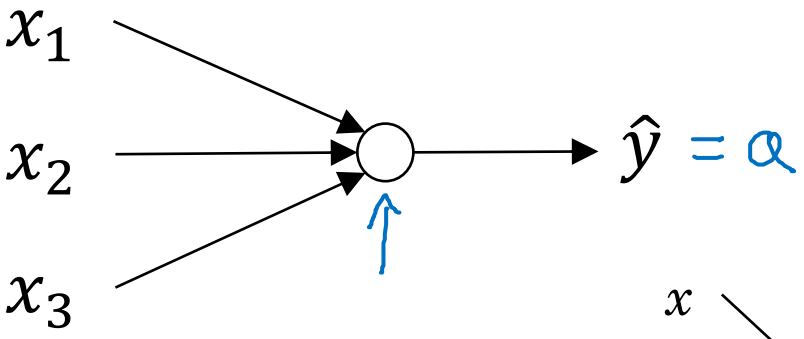
Notation

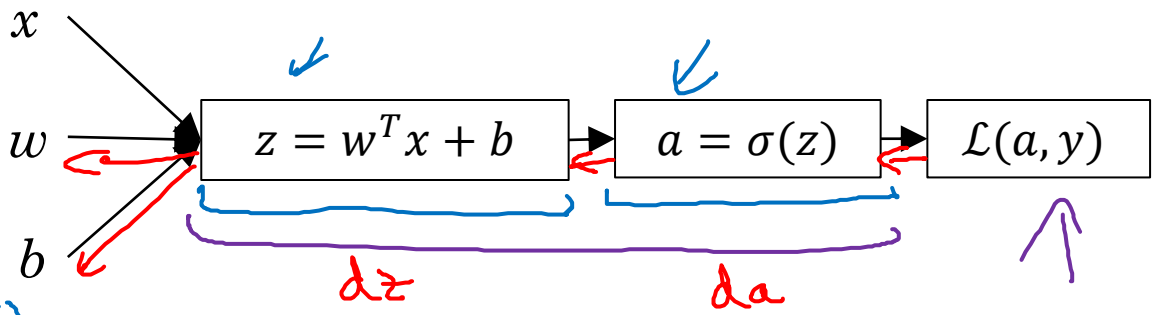
(x, y) $\quad x \in \mathbb{R}^{n_x}$, $y \in \{0, 1\}$

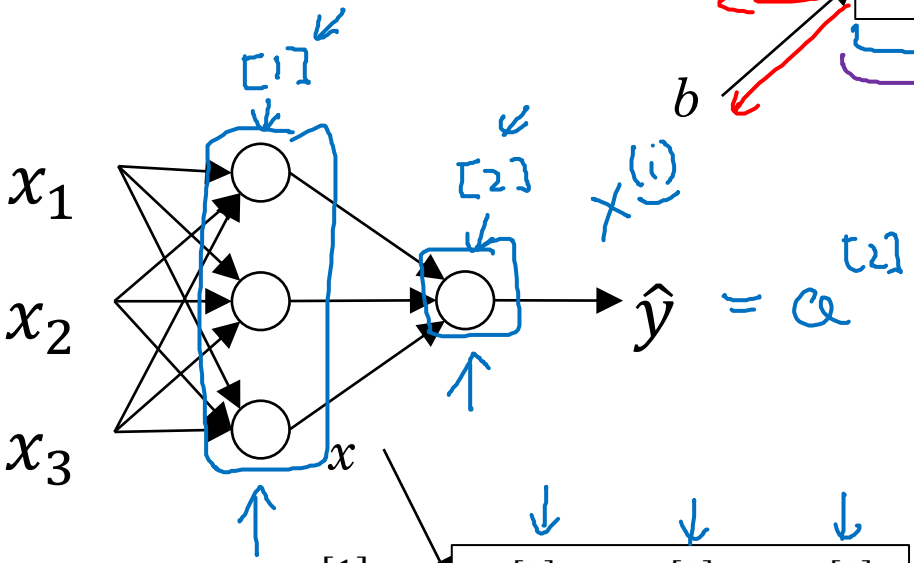
m training examples: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$

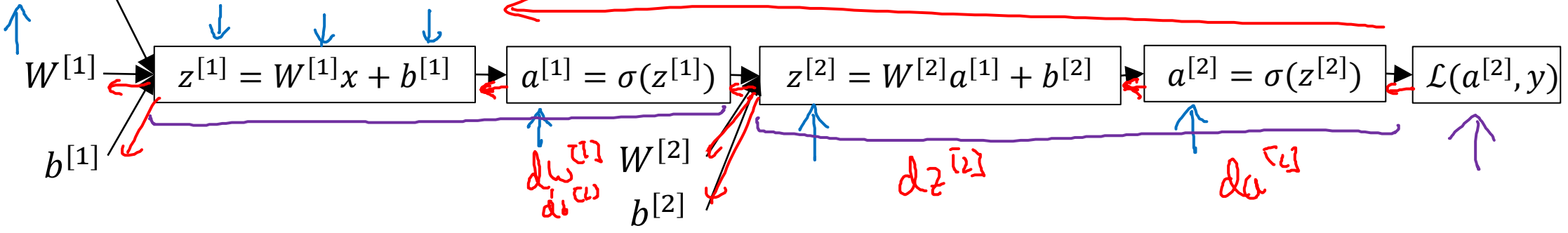
$m = m_{train}$ $\qquad$ m_{test} = #test examples.

What is a Neural Network?





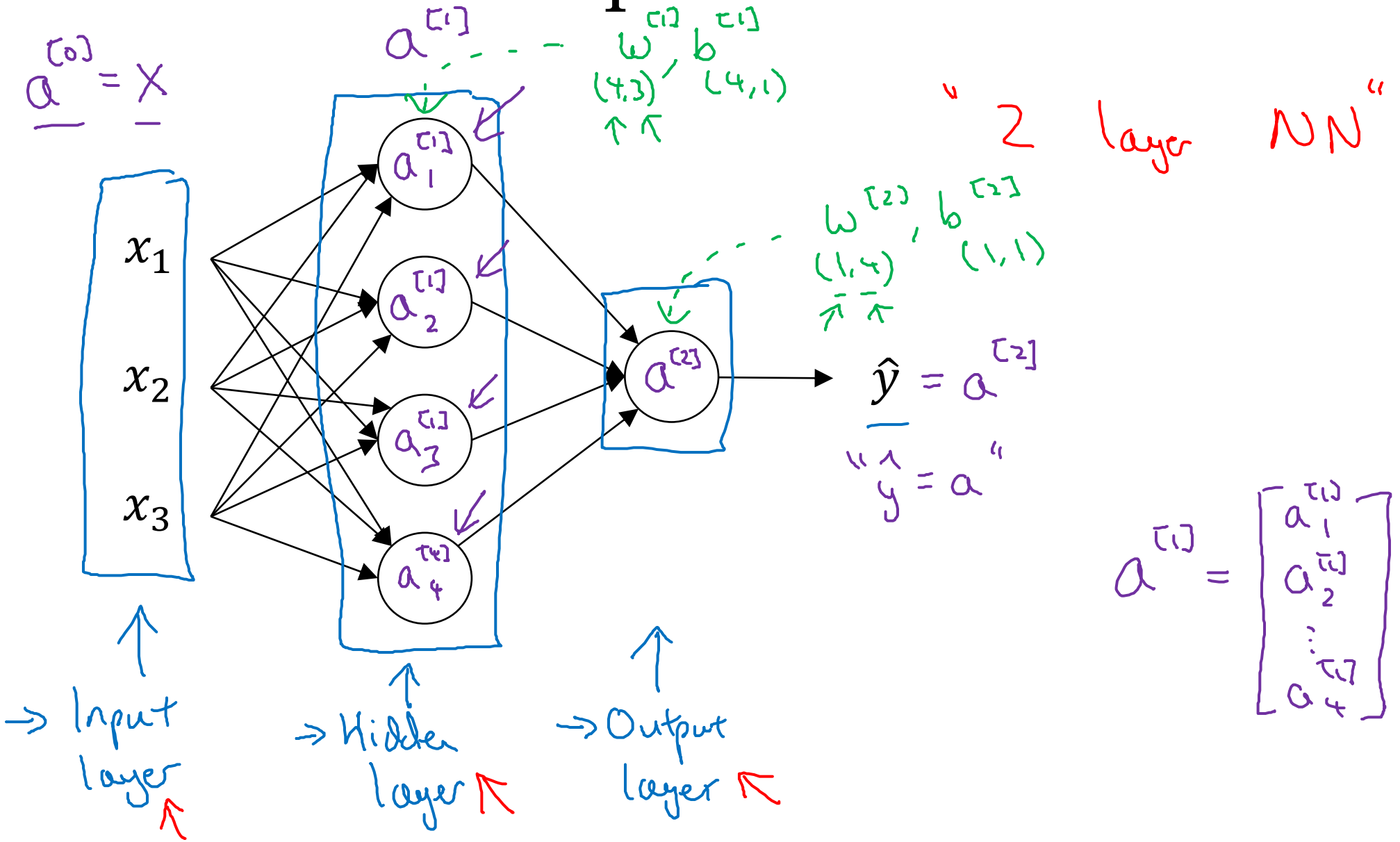




One hidden layer Neural Network

Neural Network Representation

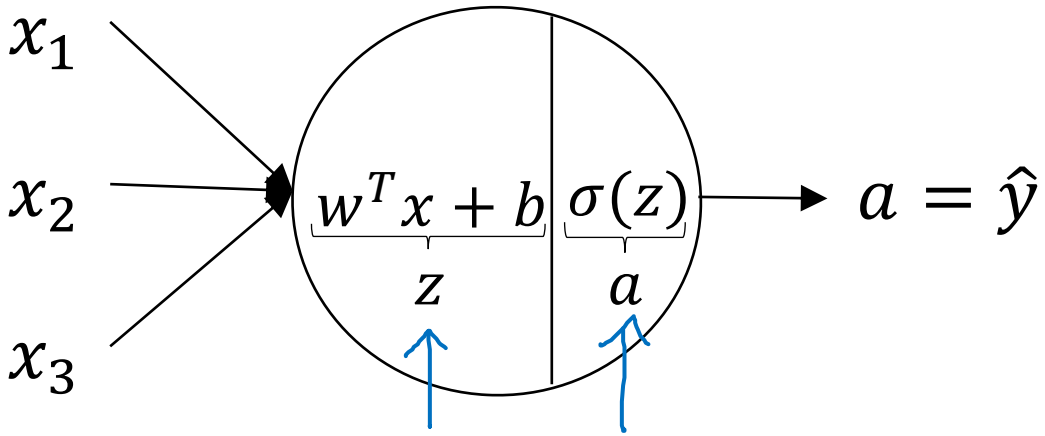
Neural Network Representation



One hidden layer Neural Network

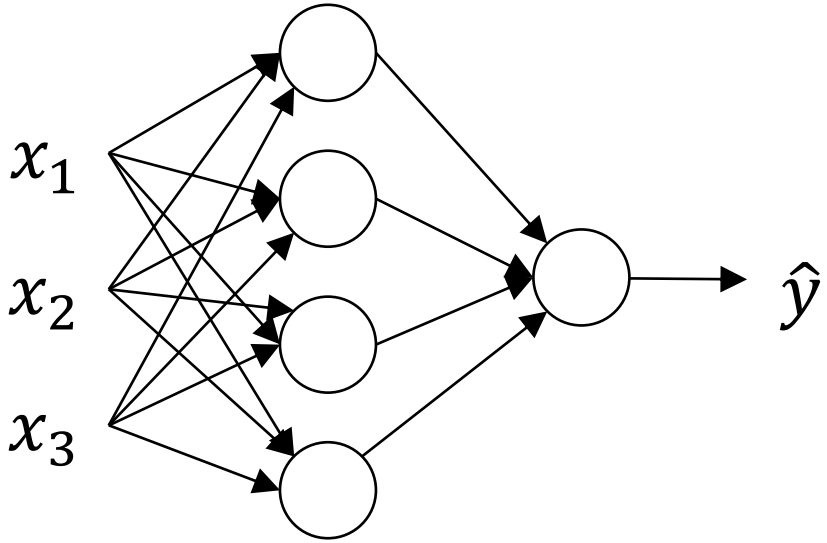
Computing a Neural Network's Output

Neural Network Representation

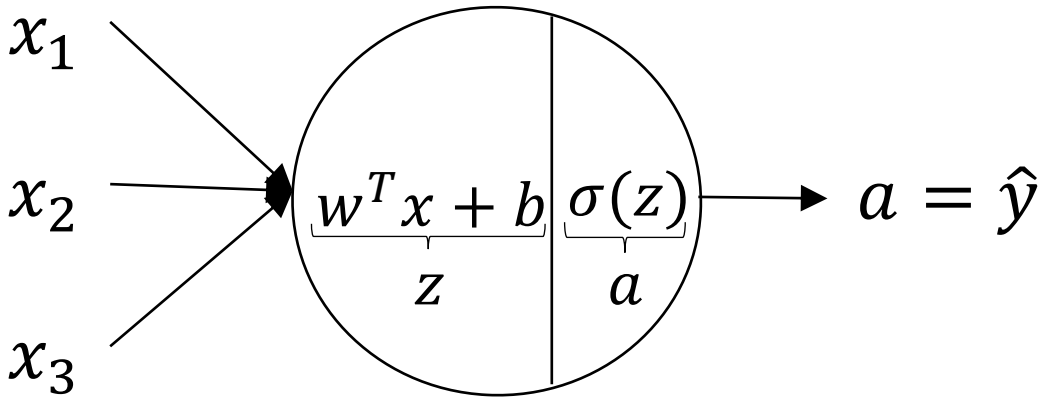


$$z = w^T x + b$$

$$a = \sigma(z)$$

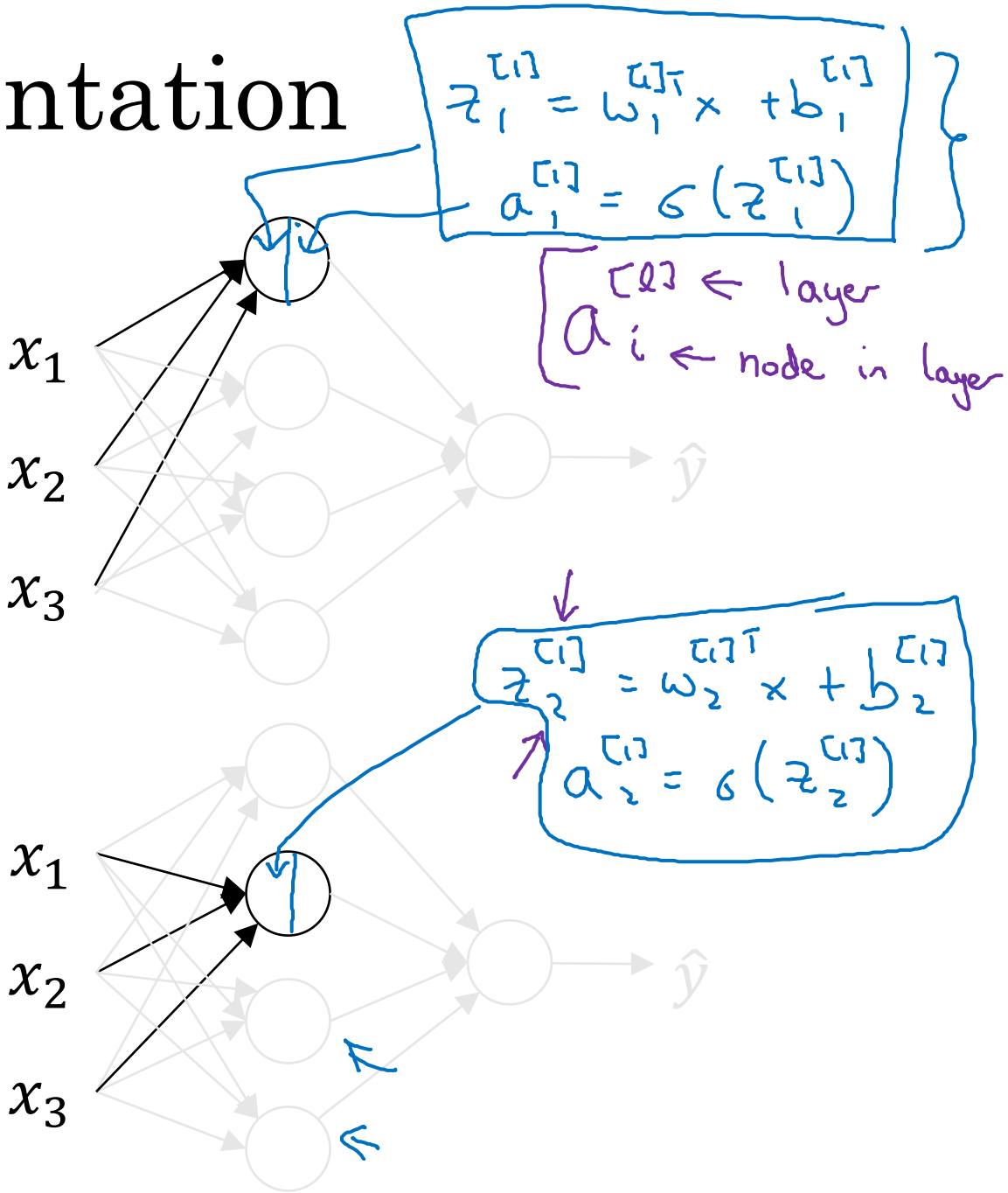


Neural Network Representation

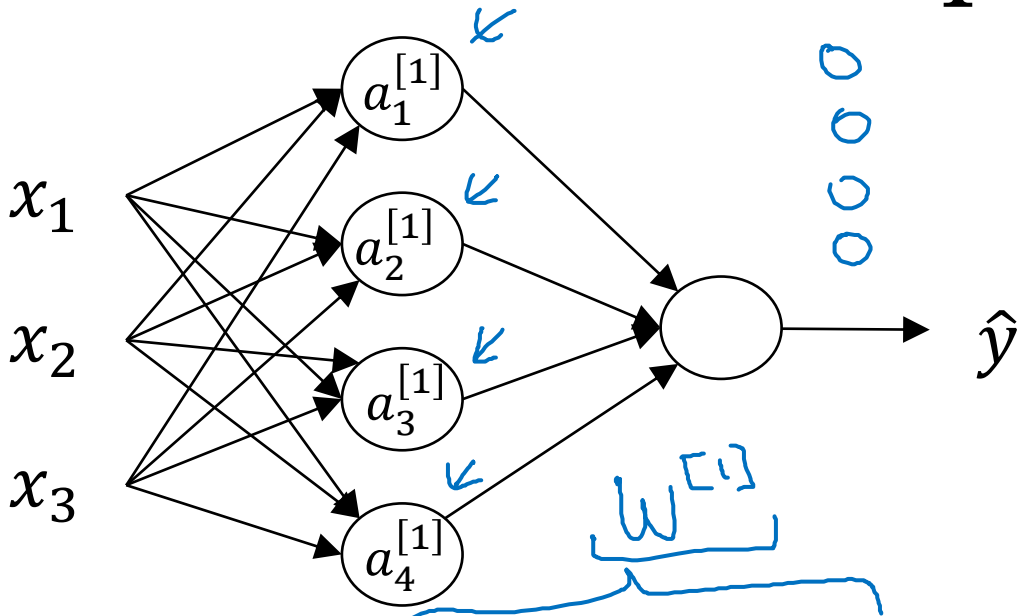


$$z = w^T x + b$$

$$a = \sigma(z)$$



Neural Network Representation



$$z_1^{[1]} = w_1^{[1]T} x + b_1^{[1]}, \quad a_1^{[1]} = \sigma(z_1^{[1]})$$
$$z_2^{[1]} = w_2^{[1]T} x + b_2^{[1]}, \quad a_2^{[1]} = \sigma(z_2^{[1]})$$
$$z_3^{[1]} = w_3^{[1]T} x + b_3^{[1]}, \quad a_3^{[1]} = \sigma(z_3^{[1]})$$
$$z_4^{[1]} = w_4^{[1]T} x + b_4^{[1]}, \quad a_4^{[1]} = \sigma(z_4^{[1]})$$

$$z^{[1]} = \begin{bmatrix} — w_1^{[1]T} — \\ — w_2^{[1]T} — \\ — w_3^{[1]T} — \\ — w_4^{[1]T} — \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ b_4^{[1]} \end{bmatrix} = \begin{bmatrix} w_1^{[1]T} x + b_1^{[1]} \\ w_2^{[1]T} x + b_2^{[1]} \\ w_3^{[1]T} x + b_3^{[1]} \\ w_4^{[1]T} x + b_4^{[1]} \end{bmatrix} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \\ z_4^{[1]} \end{bmatrix}$$

$W^{[1]}$ $(4,3)$ $\quad b^{[1]}$ $(4,1)$

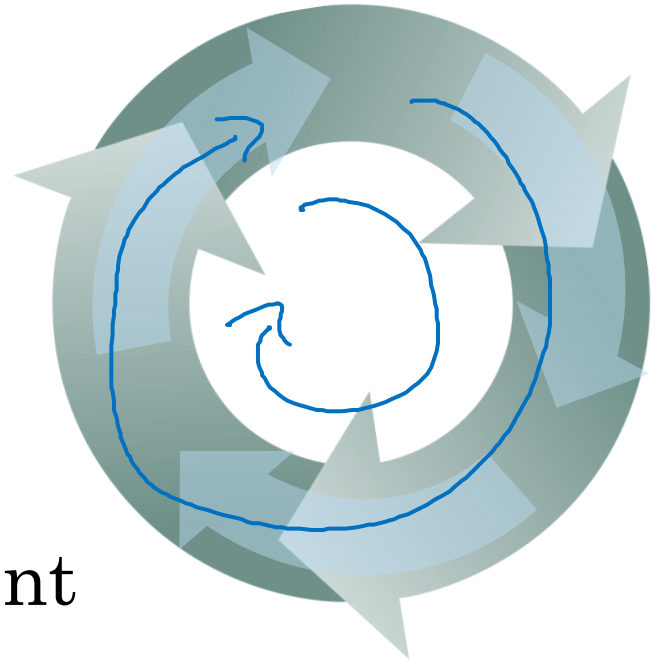
$$a^{[1]} = \begin{bmatrix} a_1^{[1]} \\ \vdots \\ a_4^{[1]} \end{bmatrix} = \sigma(z^{[1]})$$

Setting up your ML application

Train/dev/test sets

Applied ML is a highly iterative process

- # layers
- # hidden units
- learning rates
- activation functions
- …



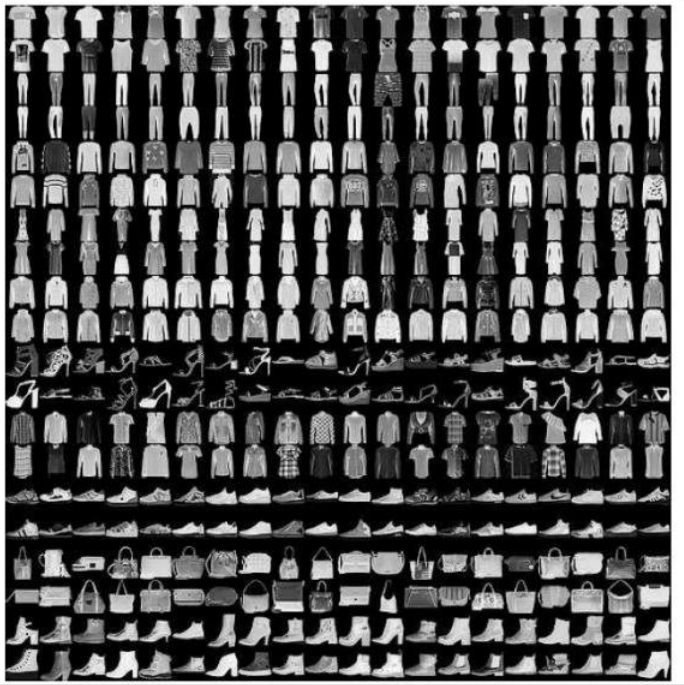
NLP, Vision, Speech, Structured data

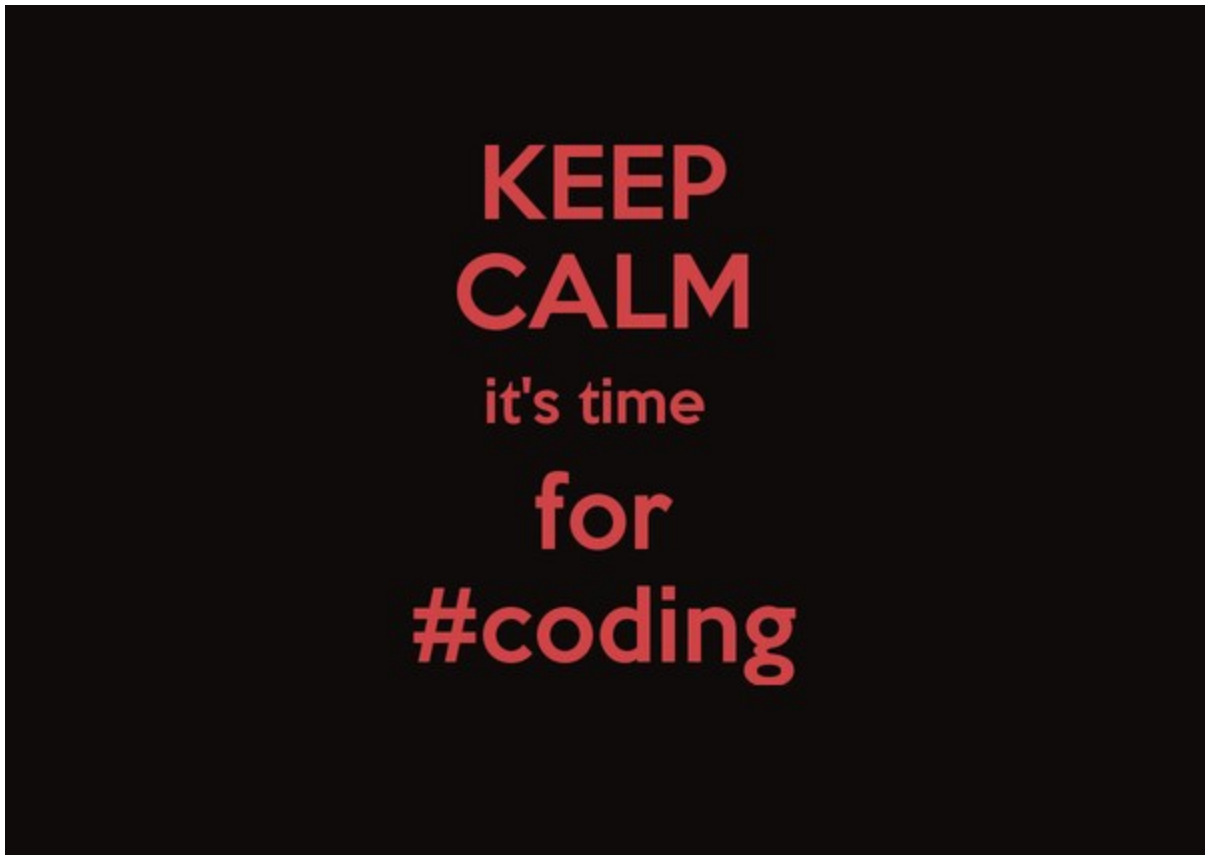
Ads Search Security logistic ....

Fashion MNIST

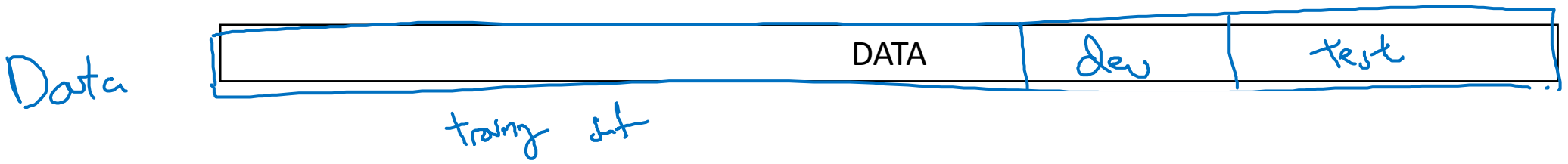
Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples.

Each example is a 28x28 grayscale image, associated with a label from 10 classes.

Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	


KEEP
CALM
it's time
for
#coding

Train/dev/test sets



Normal Size Data

Big Size Data

Mismatched train/test distribution

Cats

Training set:
Cat pictures from webpages

Dev/test sets:
Cat pictures from users using your app

→ Make sure dev and test come from same distribution.

"test"

train / dev

train / test

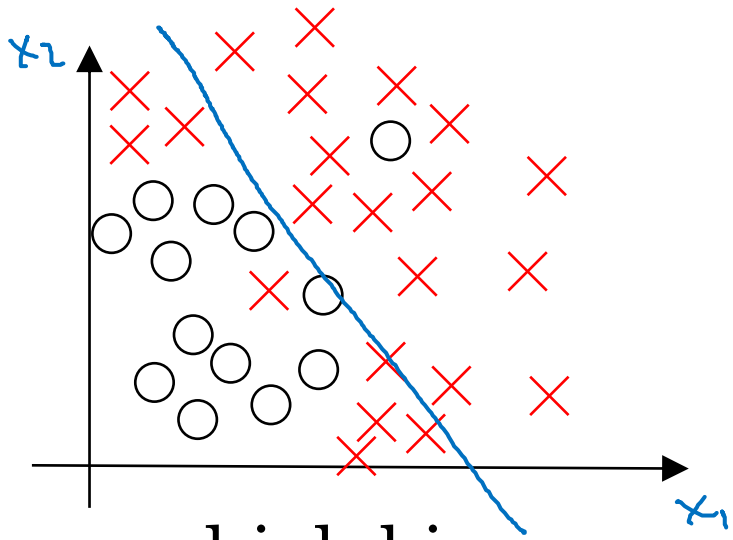
→ train / dev

Not having a test set might be okay. (Only dev set.)

Setting up your ML application

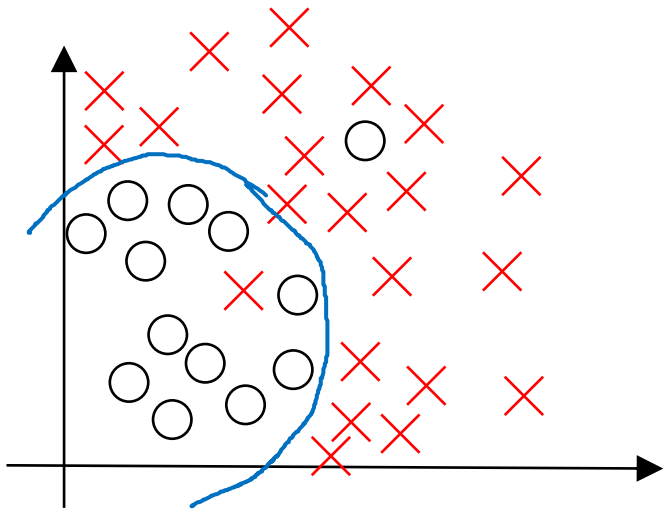
Bias/Variance

Bias and Variance

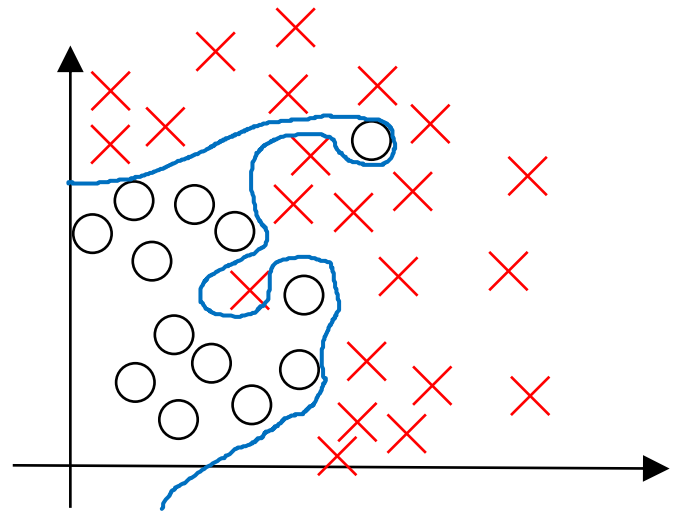


high bias

Underfitting



→ "just right"



high variance

Overfitting

Bias and Variance

Cat classification

y=1

y=0

Train set error:

Dev set error:

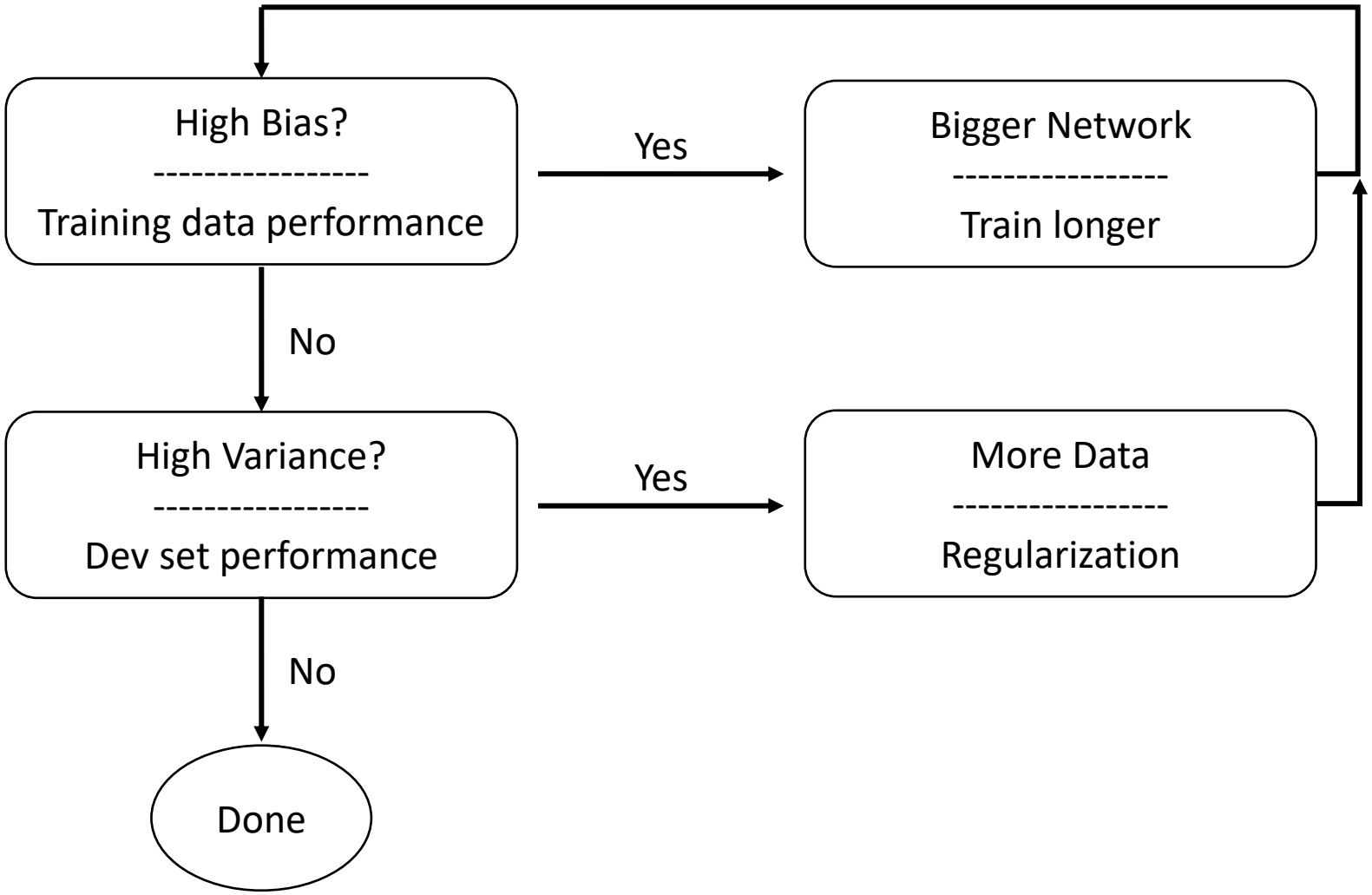
Human: ≈0%

Optimal (

Setting up your ML application

Basic “recipe” for machine learning

Basic “recipe” for machine learning



Setting up your goal

Satisficing and optimizing metrics

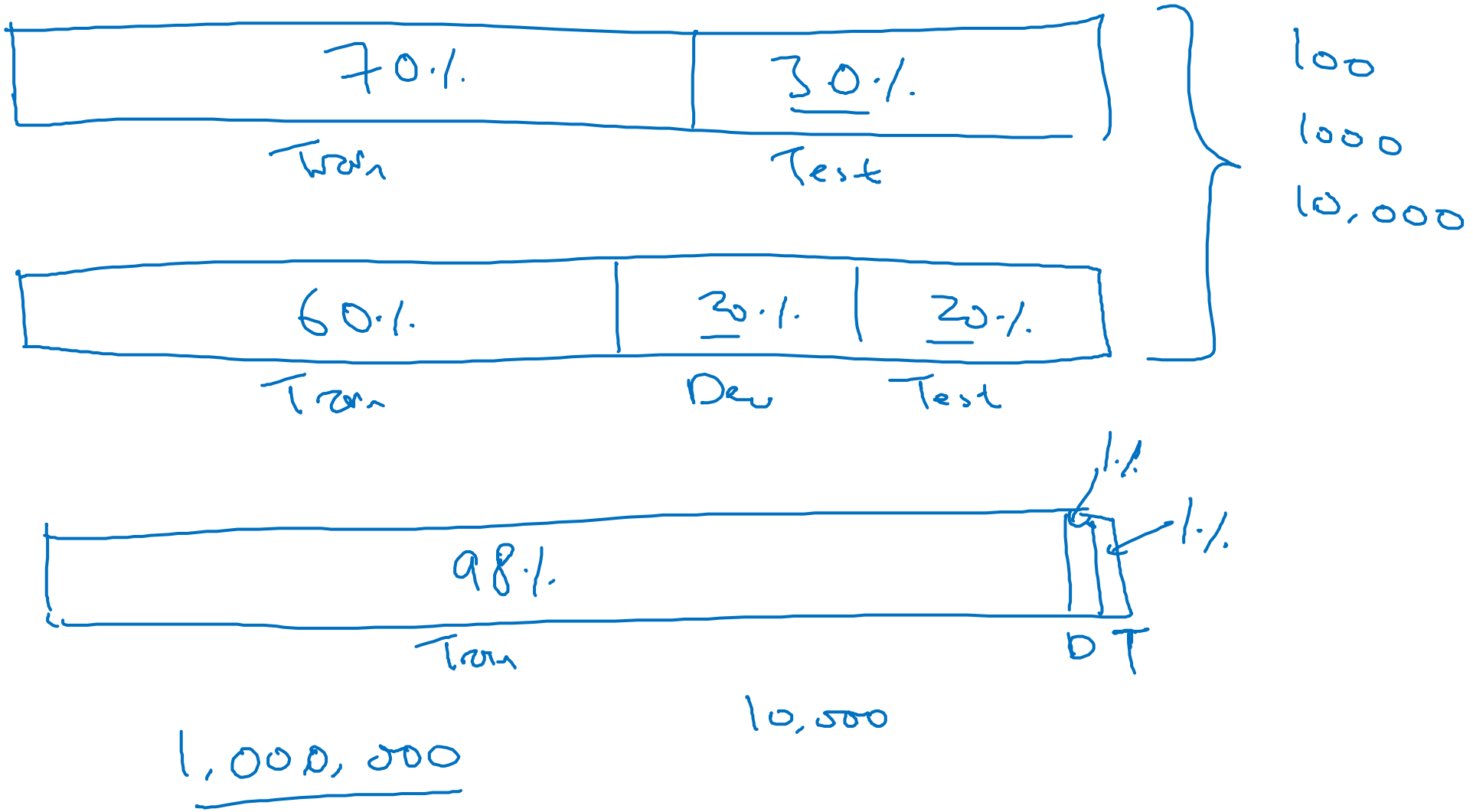
Another cat classification example

Classifier	Accuracy	Running time
A	90%	80ms
B	92%	95ms
C	95%	1,500ms

Setting up
your goal

Size of dev and test sets

Old way of splitting data



Size of dev set

Set your dev set to be big enough to detect differences in algorithm/models you're trying out.

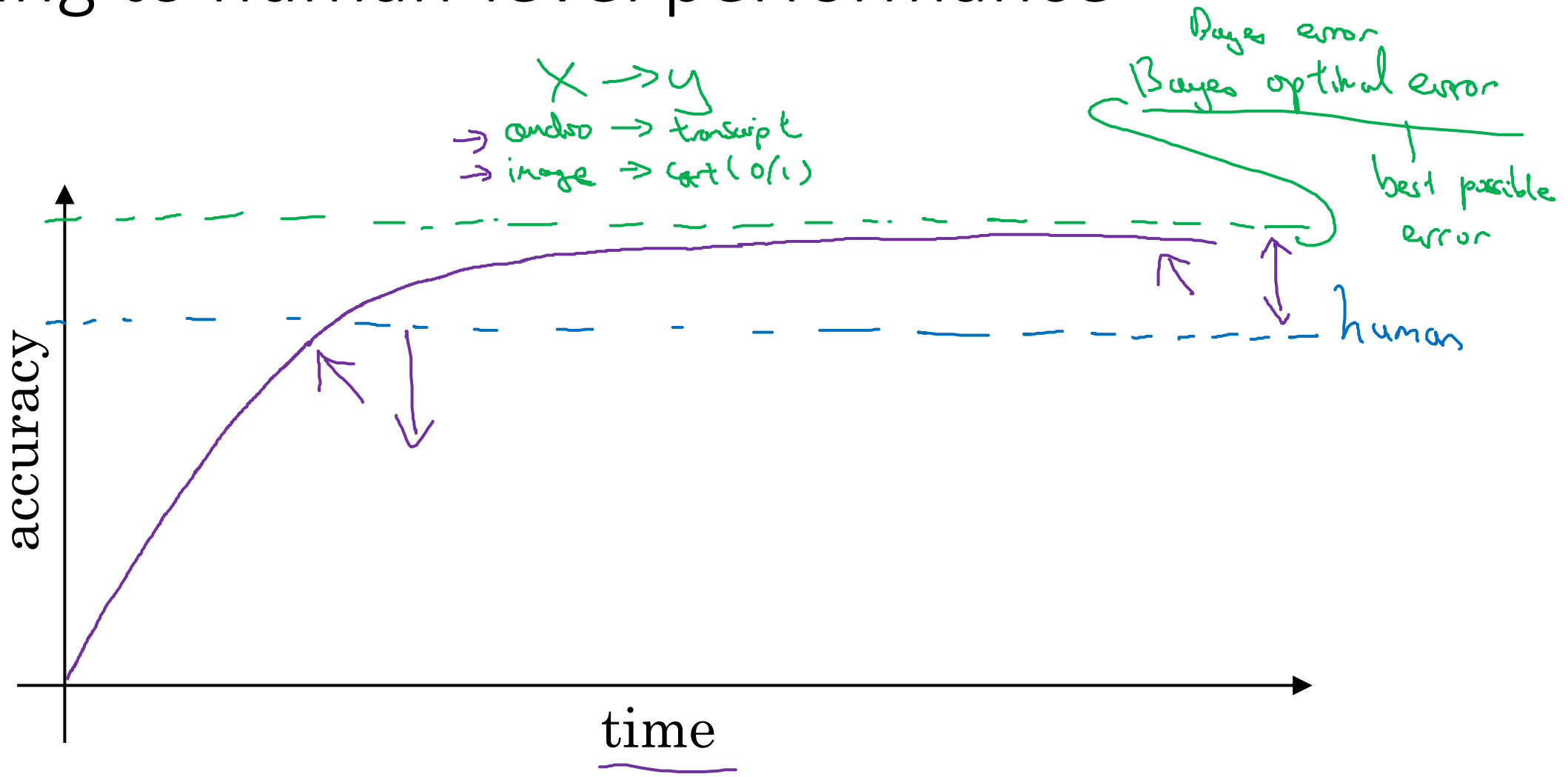
Size of test set

Set your test set to be big enough to give high confidence in the overall performance of your system.

Comparing to human-level performance

Why human-level performance?

Comparing to human-level performance



Why compare to human-level performance

Humans are quite good at a lot of tasks. So long as ML is worse than humans, you can:

- Get labeled data from humans.
- Gain insight from manual error analysis: Why did a person get this right?
- Better analysis of bias/variance.

Bias and Variance

Cat classification

Training set error:

Dev set error:

Cat classification example

Human error	1%	7.5%
Training error	8%	8 %
Dev error	10%	10 %

Focus on Bias

Focus on Variance

Human-level error as a proxy for Bayes error

Medical image classification example:

Suppose:

(a) Typical human ..... 3 % error

(b) Typical doctor ..... 1 % error

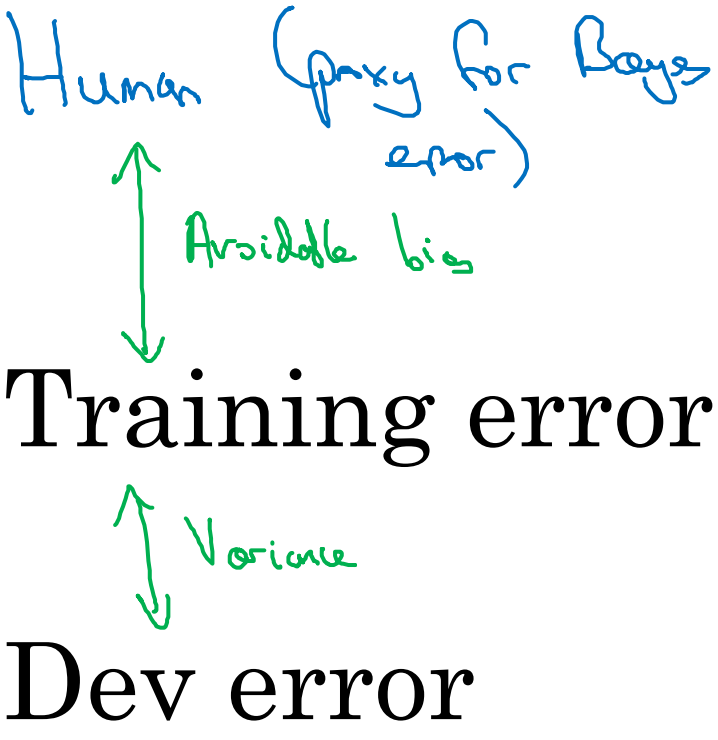
(c) Experienced doctor ..... 0.7 % error

(d) Team of experienced doctors .. 0.5 % error

Bayes error $\leq$ 0.5%

What is "human-level" error?

Error analysis example



myCallBack

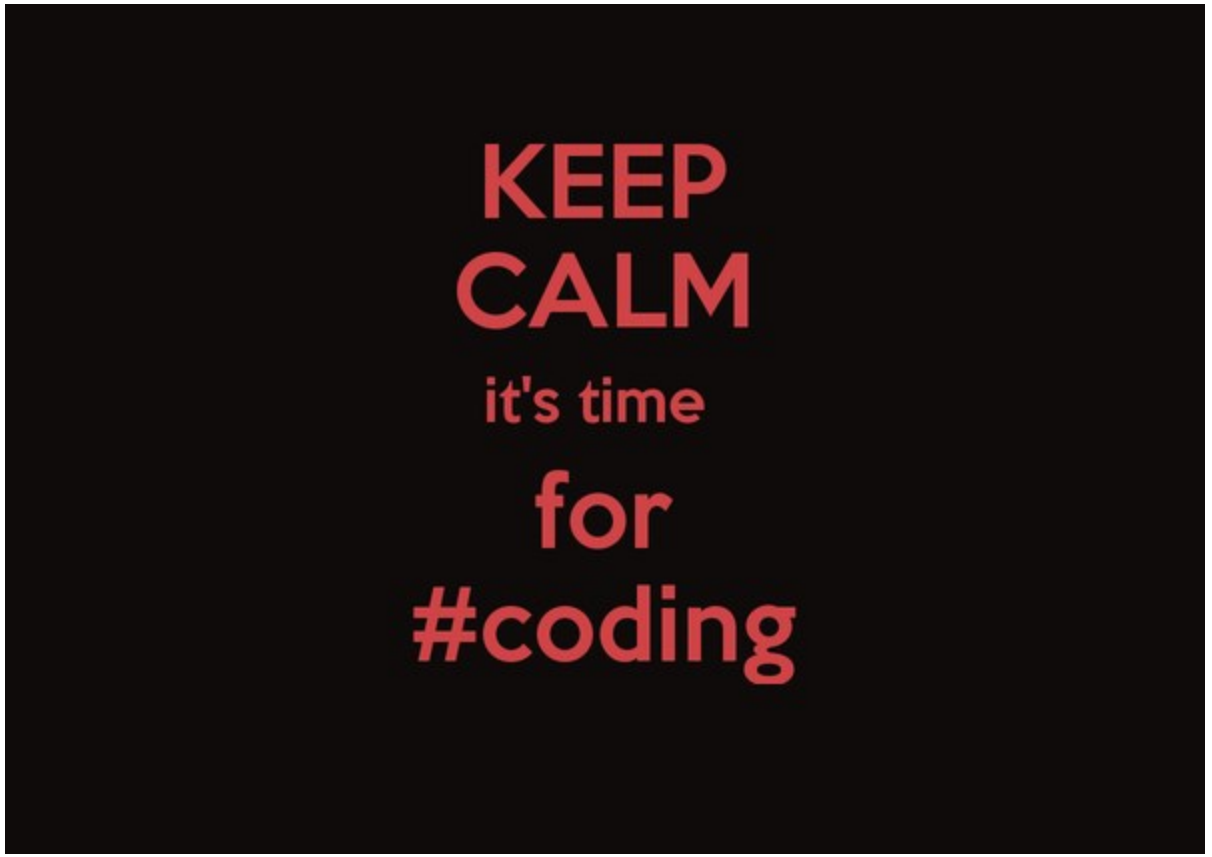
- **class myCallback**(tf.keras.callbacks.Callback):

  def on_epoch_end(self, epoch, logs={}):

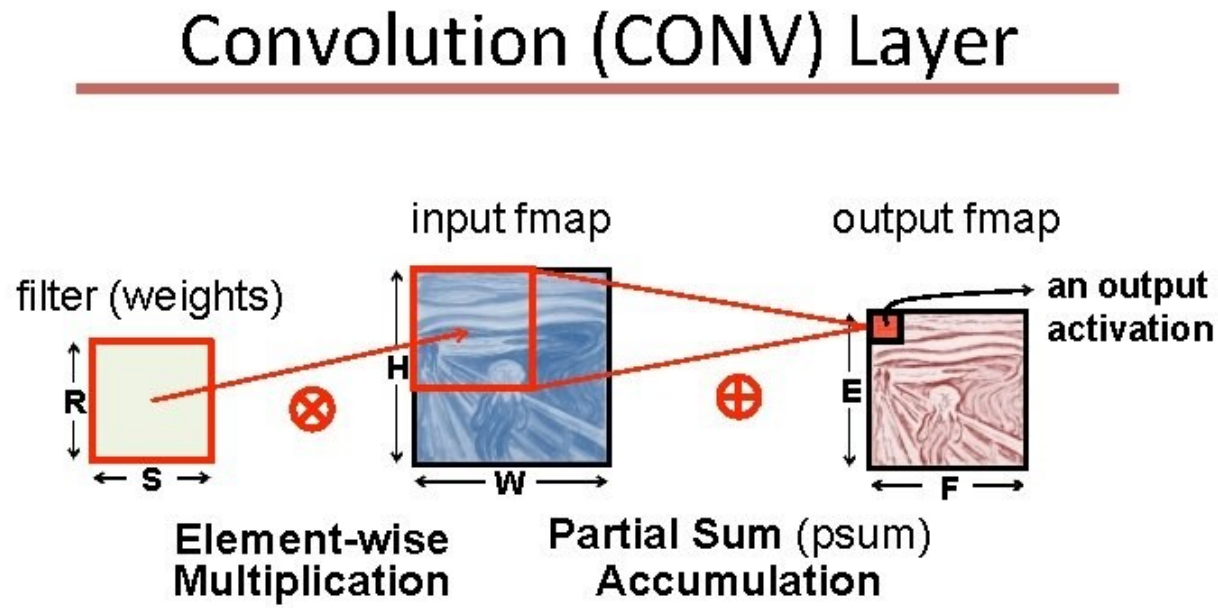
    if(logs.get('acc')>0.6):

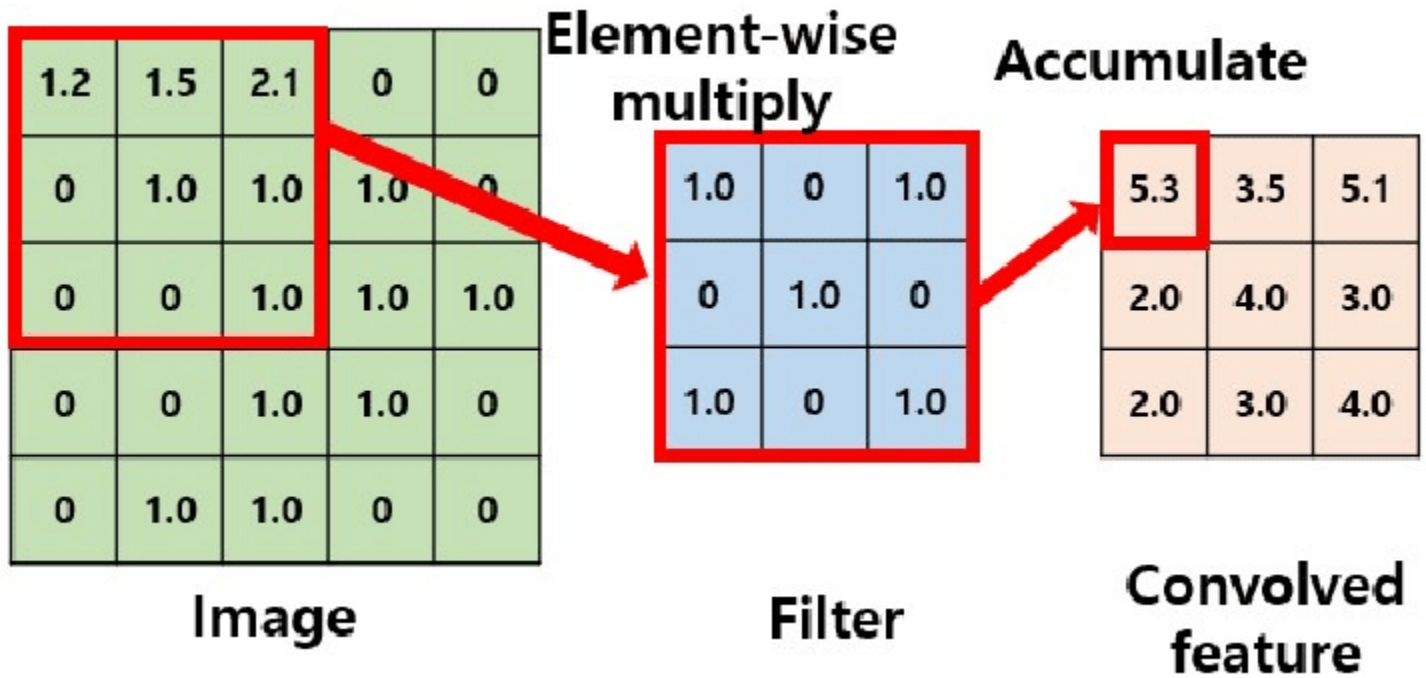
      print("**\n**Reached 60% accuracy so cancelling training!")

      self.model.stop_training = **True**

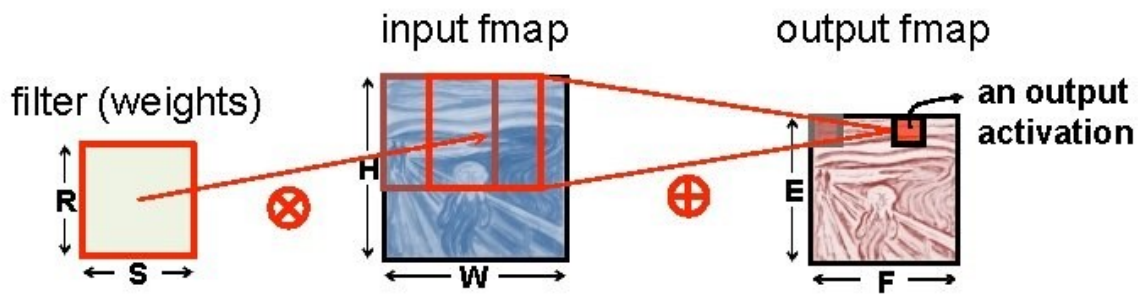

KEEP
CALM
it's time
for
#coding

How a Conv works?

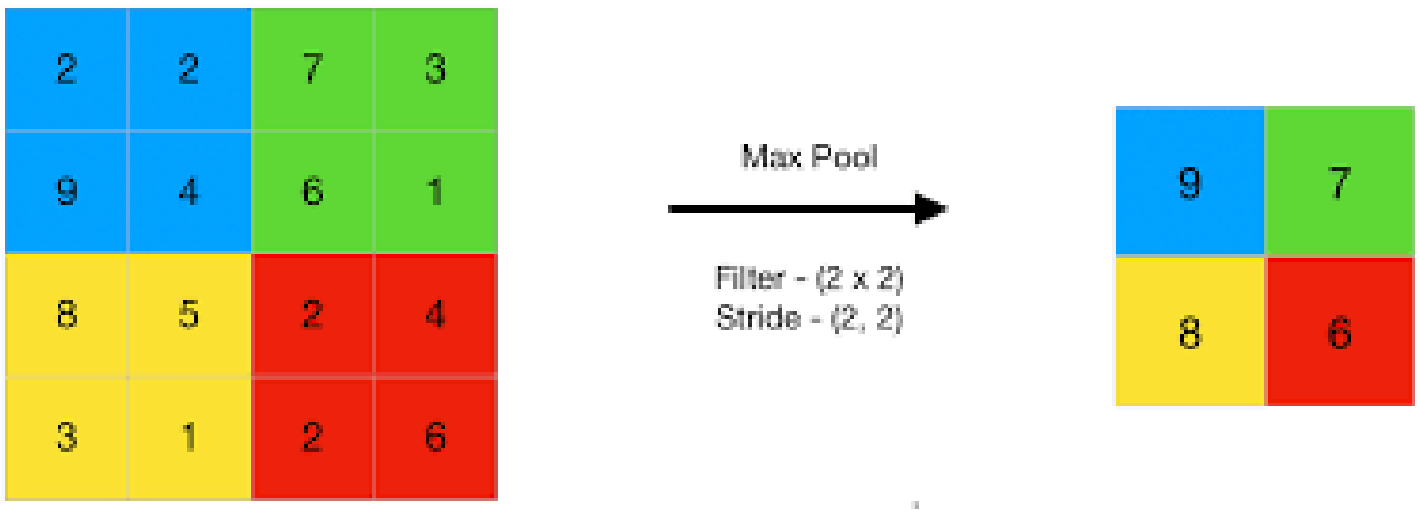


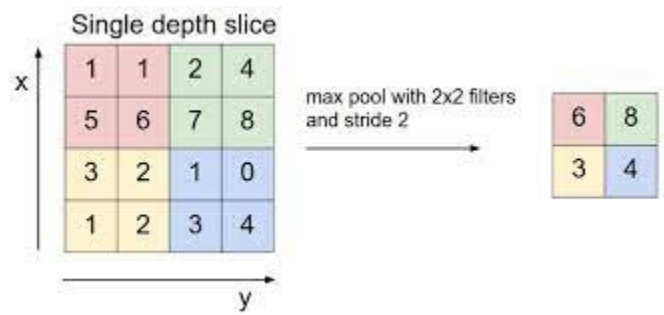

Element-wise multiply
Accumulate
1.2 1.5 2.1 0 0
0 1.0 1.0 1.0 0
0 0 1.0 1.0 1.0
0 0 1.0 1.0 0
0 1.0 1.0 0 0
1.0 0 1.0
0 1.0 0
1.0 0 1.0
5.3 3.5 5.1
2.0 4.0 3.0
2.0 3.0 4.0
Image
Filter
Convolved feature

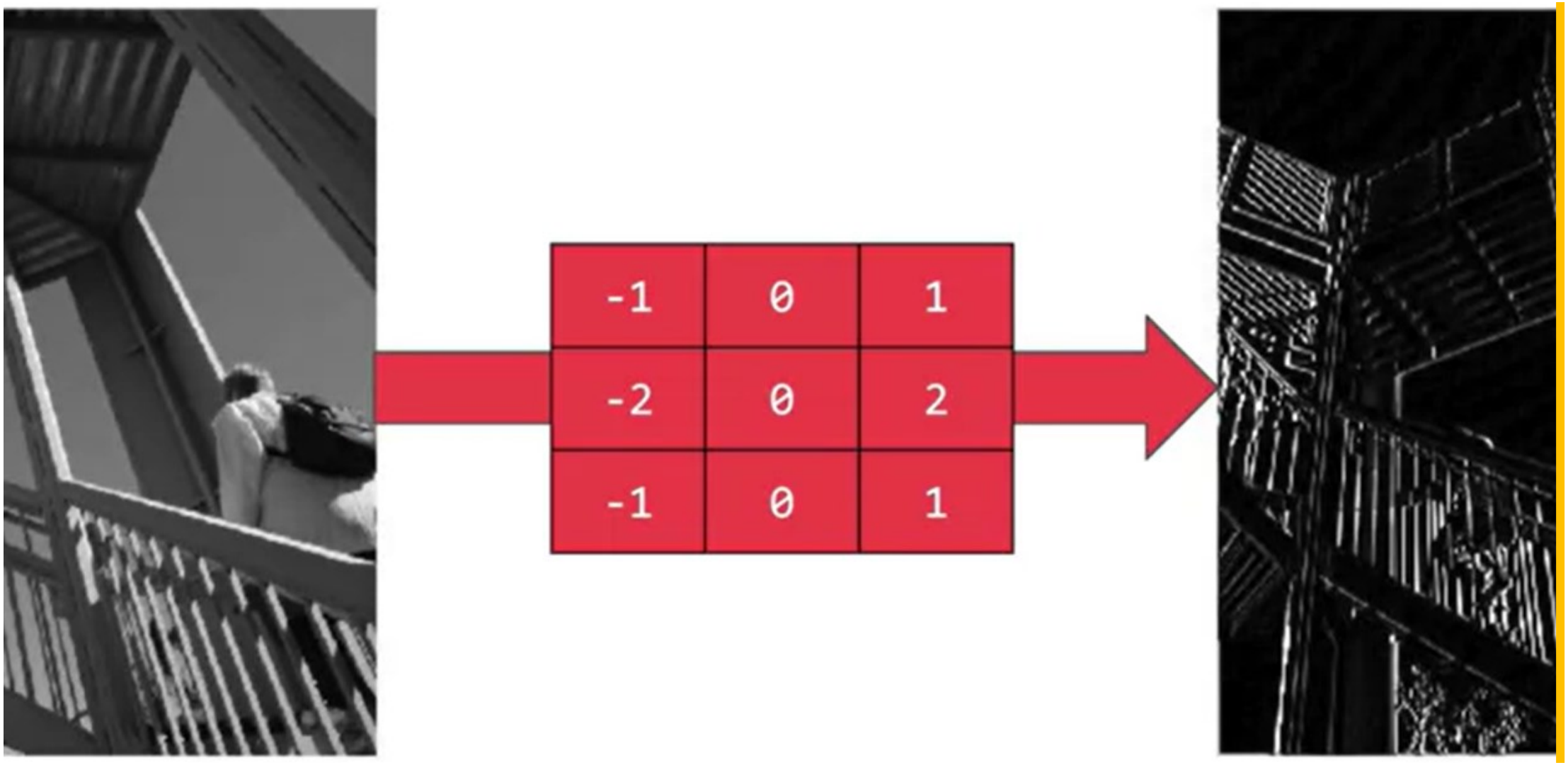
Convolution (CONV) Layer

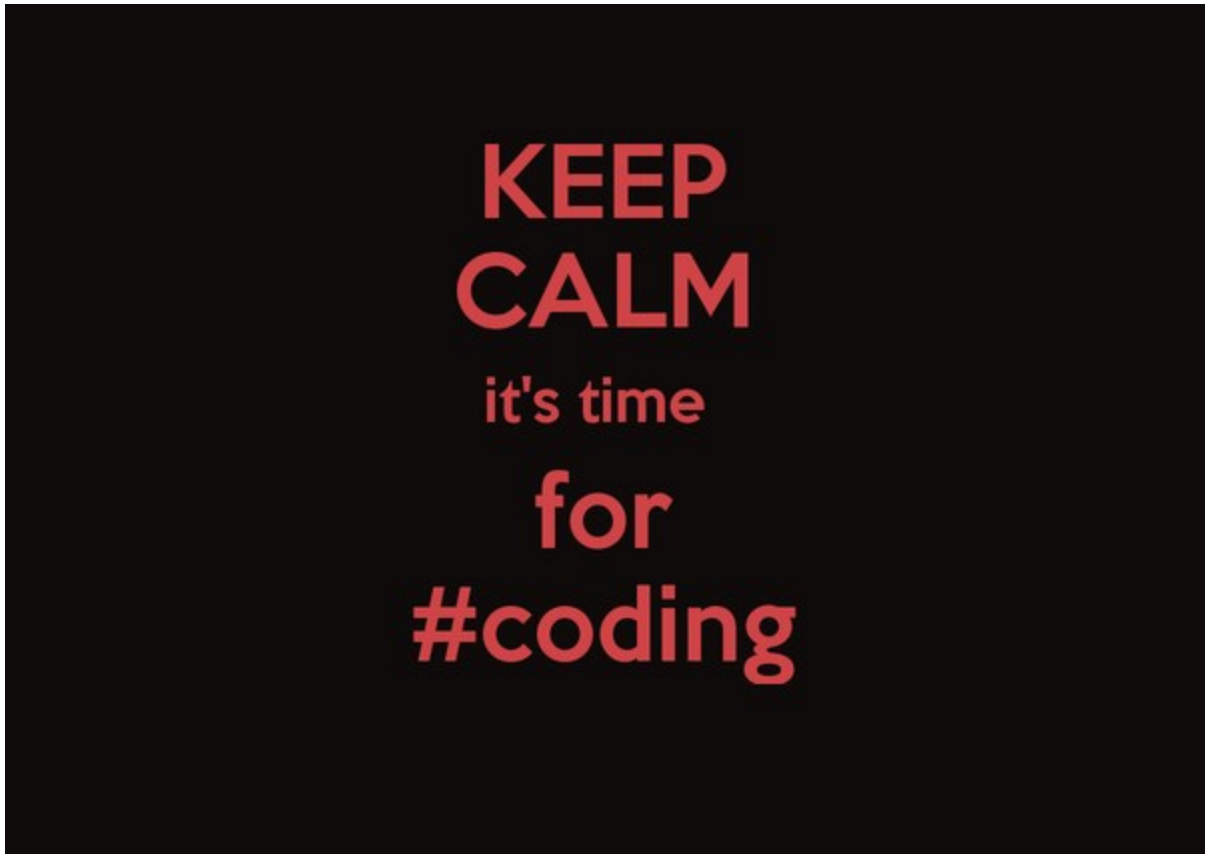


Sliding Window Processing


2
2
7
3
9
4
6
1
8
5
2
4
3
1
2
6
Max Pool
Filter - (2 x 2)
Stride - (2, 2)
9
7
8
6


Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

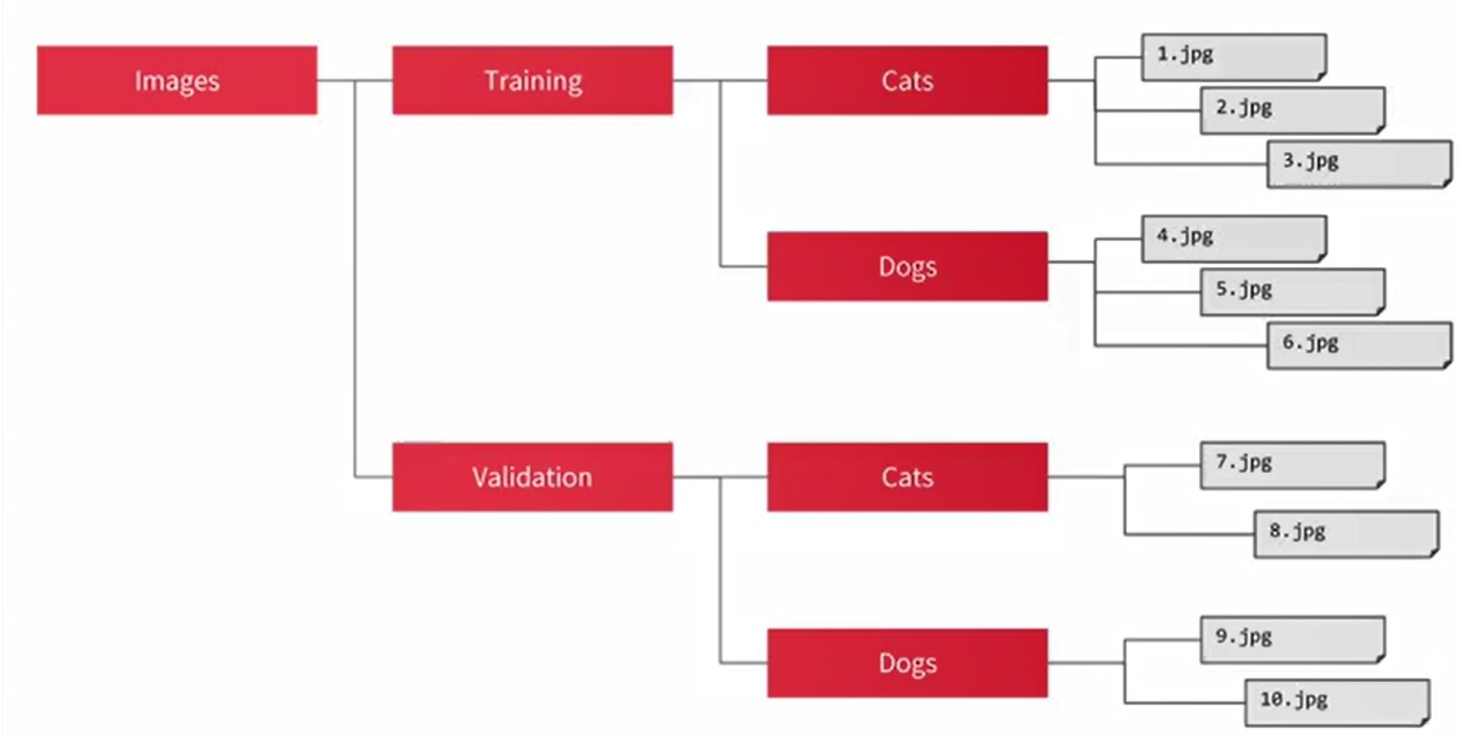

-1 0 1
-2 0 2
-1 0 1



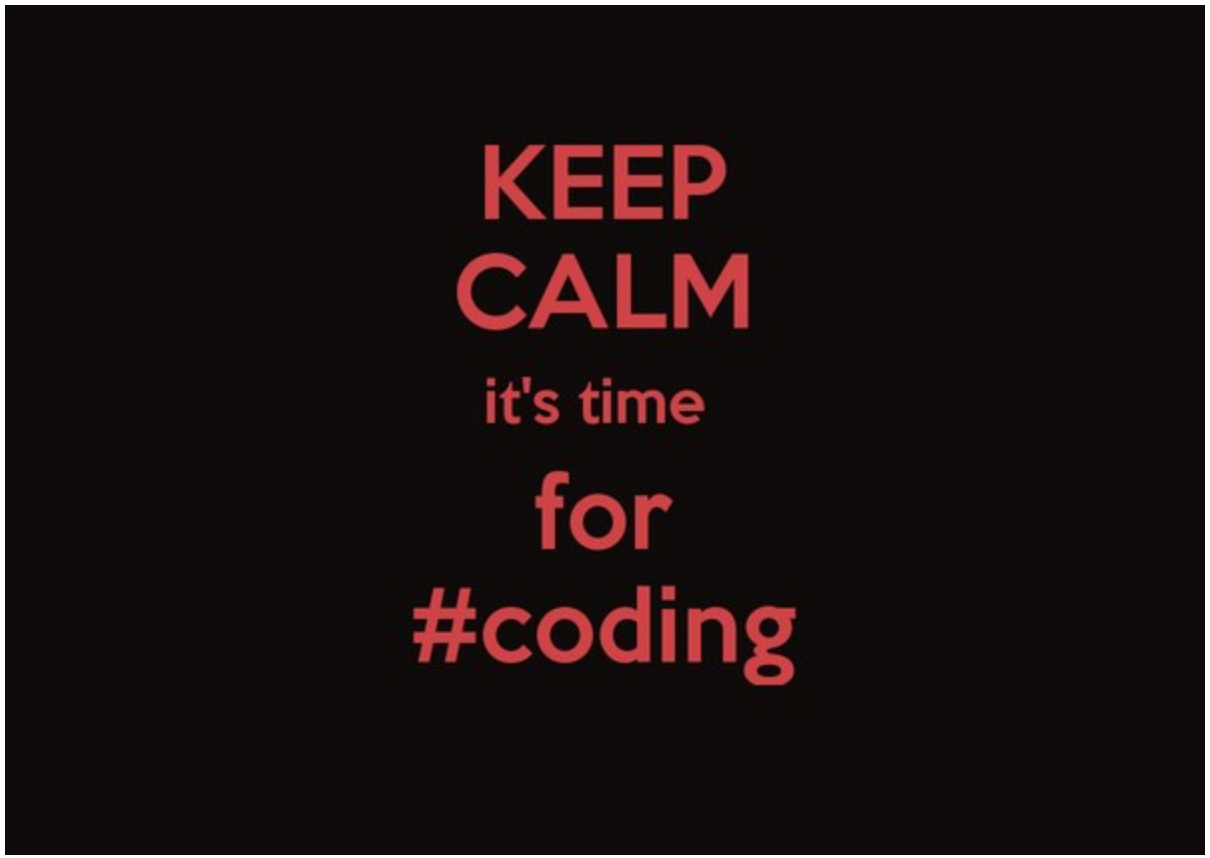
2_FahionMNIS_NN, 2_1_CallBack, 3_Basic_of_Conv, 4_Convolutions Sidebar,

Image Generator

- If you put your images in to name subdirectory ImageGenerator will auto lable them for you



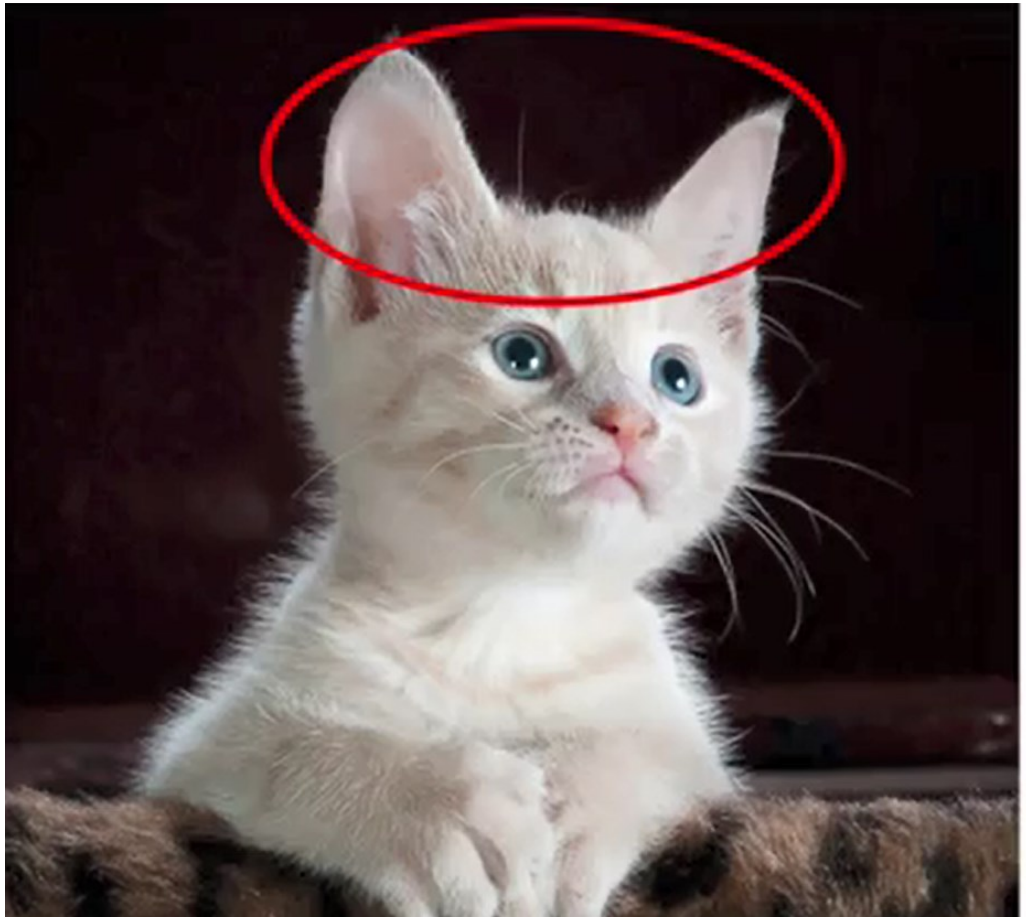
```
from tensorflow.keras.preprocessing.image
import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(150, 150),
        batch_size=20,
        class_mode='binary')
```


KEEP
CALM
it's time
for
#coding

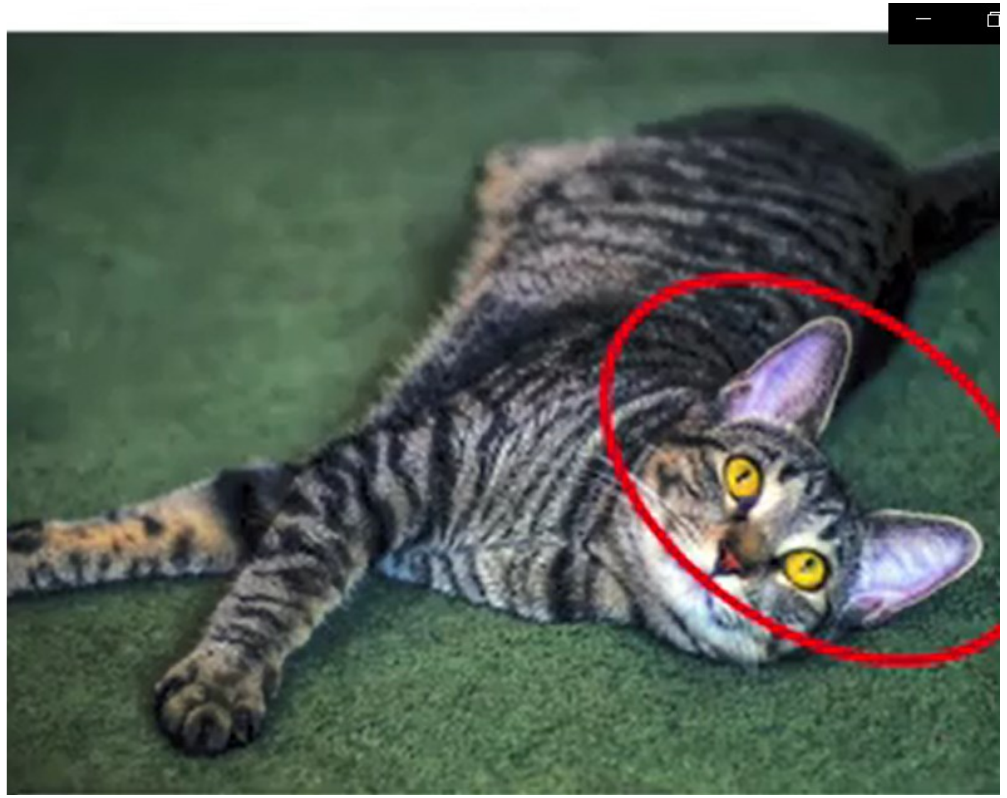
Data Augmentation

- Increase data image size
- A solution for overfitting problem

Data Augmentation

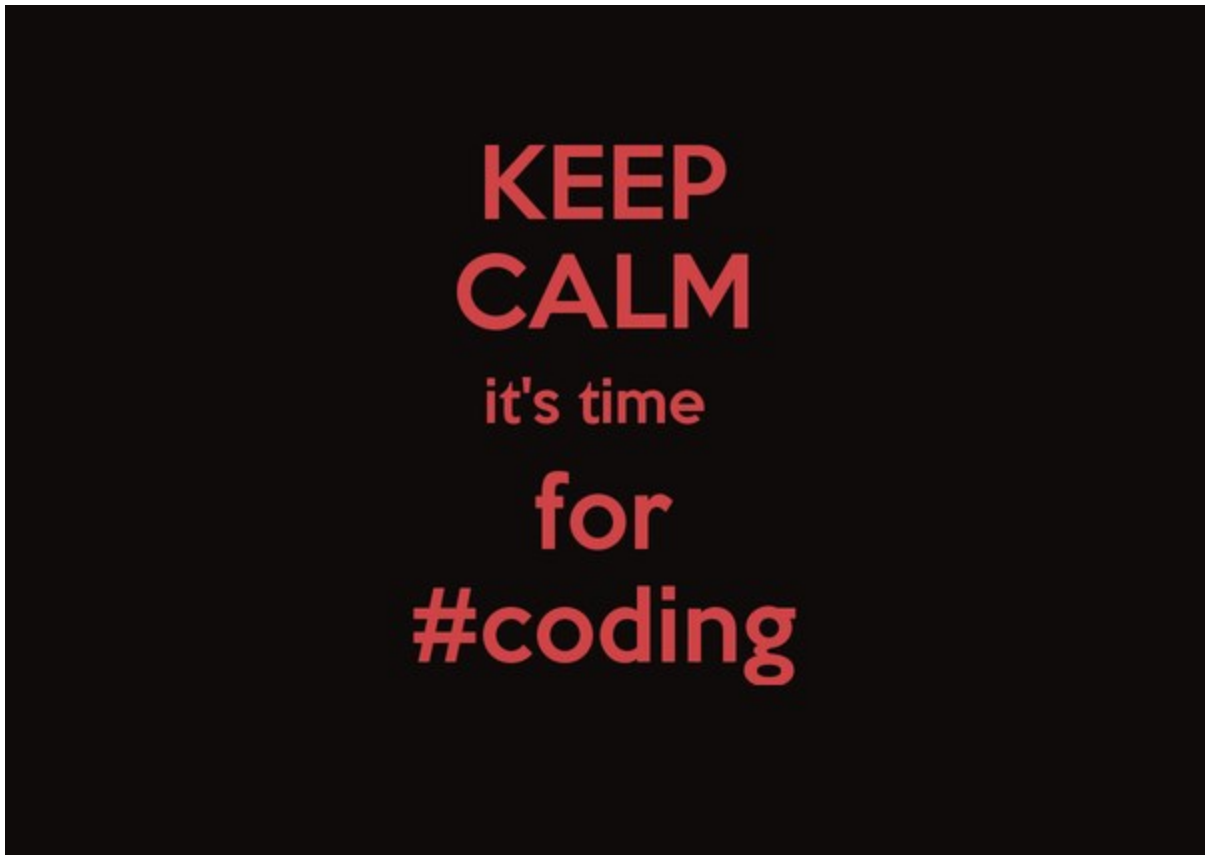


Data Augmentation



Data Augmentation

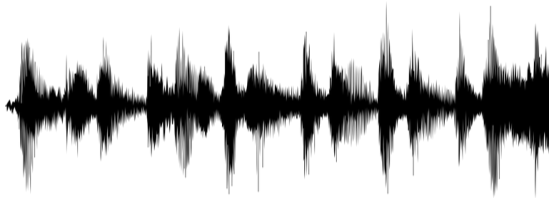
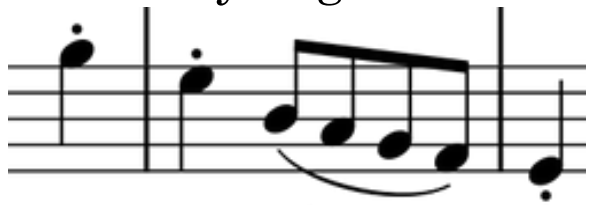
```
# Updated to do image augmentation
train_datagen = ImageDataGenerator(
      rescale=1./255,
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
      fill_mode='nearest')
```


KEEP
CALM
it's time
for
#coding

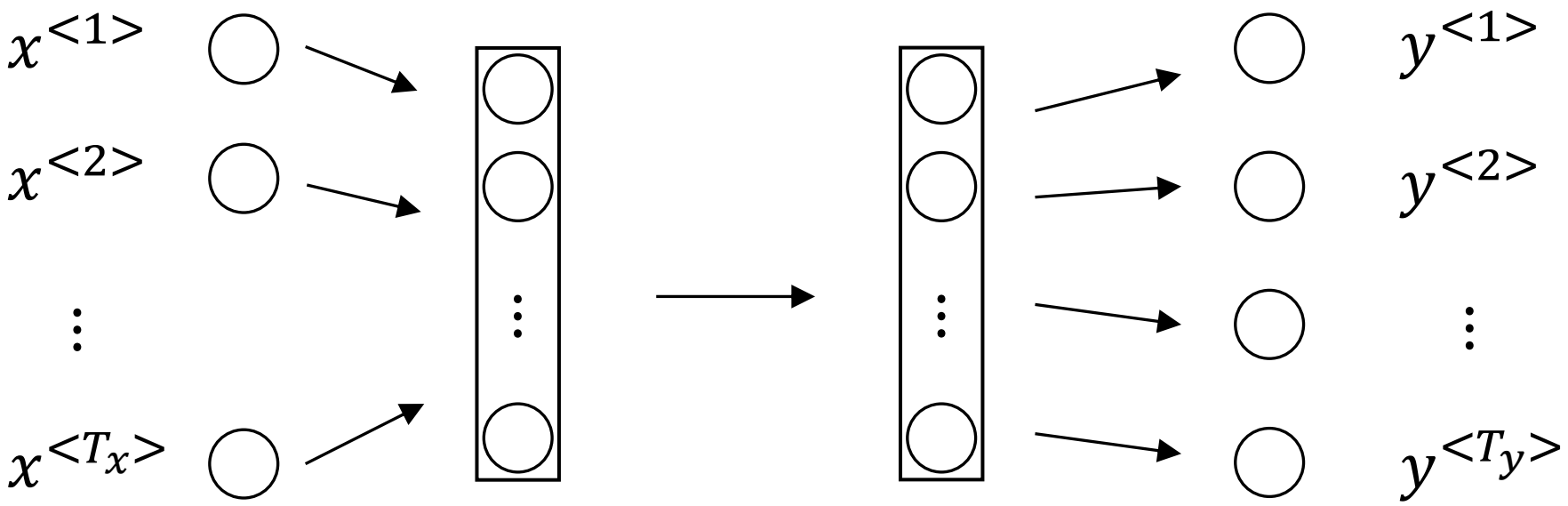
Recurrent Neural Networks

Why sequence models?

Examples of sequence data

Speech recognition		→	"The quick brown fox jumped over the lazy dog."
Music generation	∅	→	
Sentiment classification	"There is nothing to like in this movie."	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

Why not a standard network?

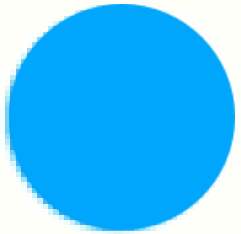


Problems:

- Inputs, outputs can be different lengths in different examples.
- Doesn't share features learned across different positions of text.

What is a Sequence Data

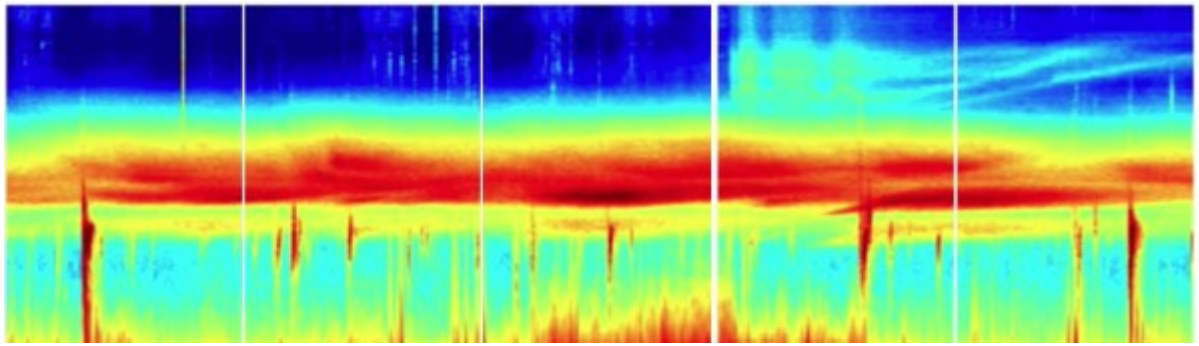
- Take a still snapshot of a ball moving in time.
- Predict the direction that the ball was moving?
- If you record many snapshots of the ball's position in succession, you will have enough information to make a better prediction.



This is a sequence, a particular order in which one thing follows another.

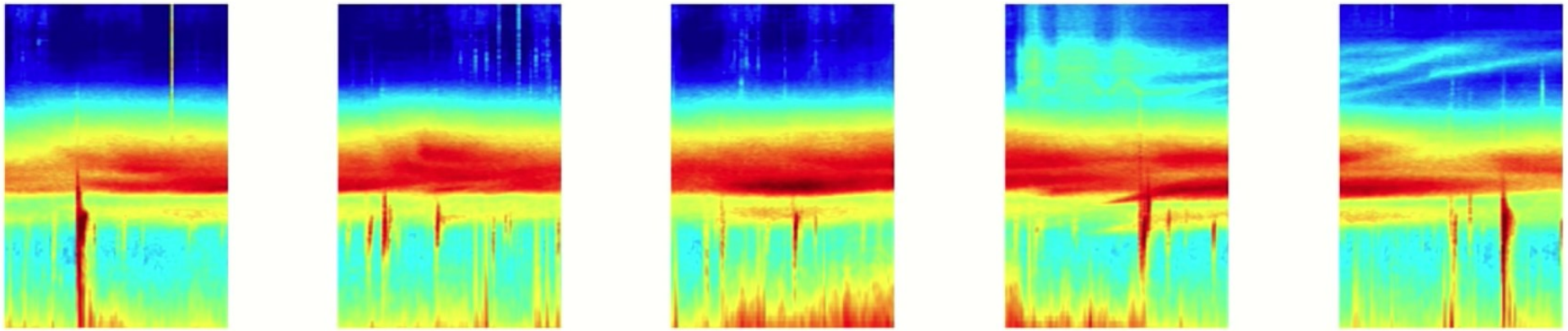
Sequential Data

- Sequence data comes in many forms. Audio is a natural sequence.



Sequential Data

- You can chop up an audio spectrogram into chunks and feed that into RNN’s



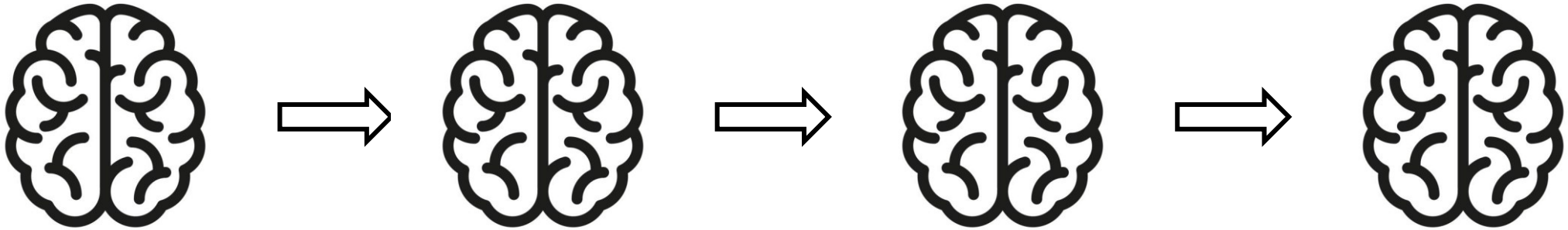
Sequential Data

T e x t

Text can be sequence of data

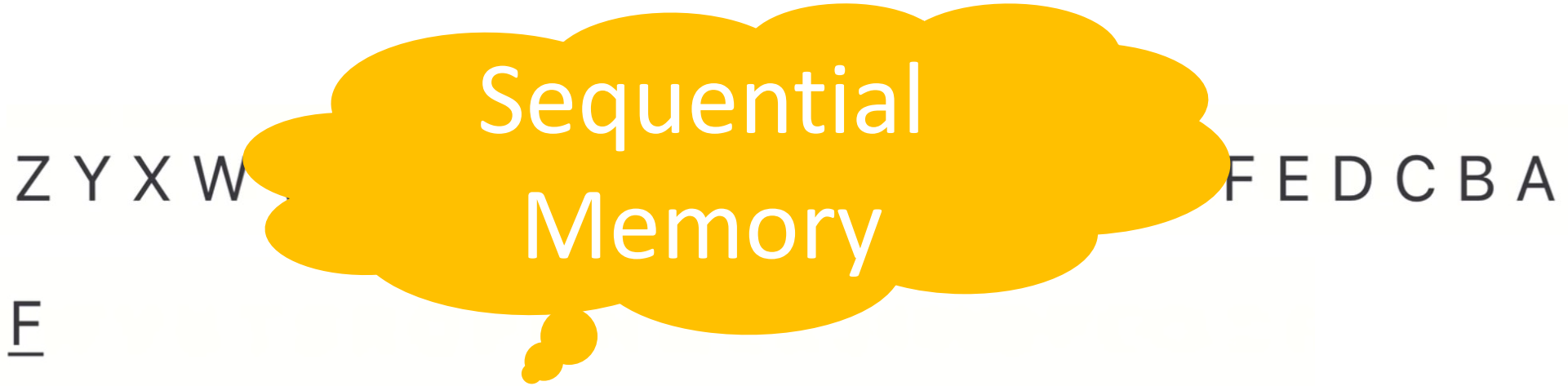
Sequential Memory

- RNN’s are good at processing sequence data for predictions. But how??
- They do that by having a concept I like to call sequential memory.



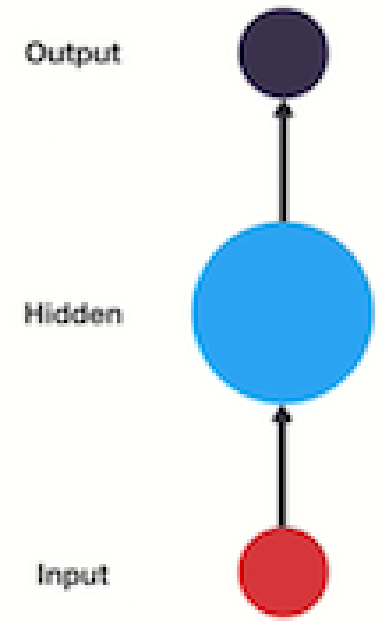
Sequential Memory

- I want to invite you to say the alphabet in your head.



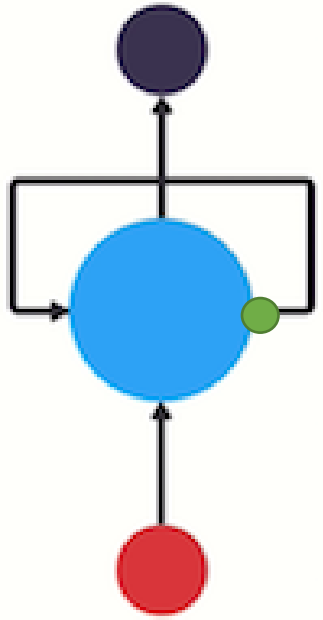
NNs

Look at a traditional neural network



RNNs

An RNN has a looping mechanism that acts as a highway to allow information to flow from one step to the next.



Classifying intents from users inputs

- Let’s say we want to build a chatbot

Classifying intents from users inputs

- First, we are going to encode the sequence of text using an RNN.
- Then, we are going to feed the RNN output into a feed-forward neural network which will classify the intents

- I lived in Iran, so at school they made me learn how to speak Persian (Farsi)

An Example

- A user types in… ***what time is it?***.
- To start, we break up the sentence into individual words.

An Example

- The first step is to feed “What” into the RNN. The RNN encodes “What” and produces an output

An Example

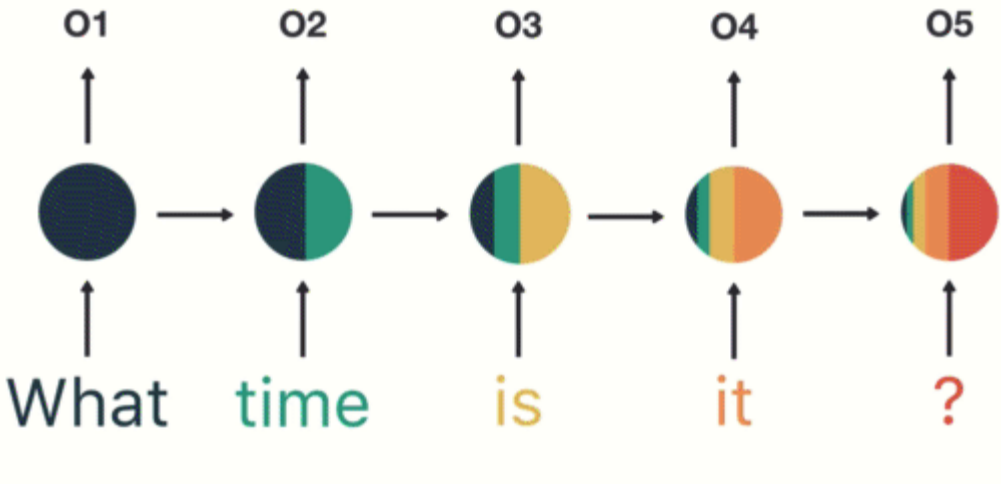
- For the next step, we feed the word “time” and the hidden state from the previous step.
- The RNN now has information on both the word “What” and “time.”

An Example

- We repeat this process, until the final step.
- You can see by the final step the RNN has encoded information from all the words in previous steps.

An Example

- Since the final output was created from the rest of the sequence, we should be able to take the final output and pass it to the feed-forward layer to classify an intent.



Short Term Memory

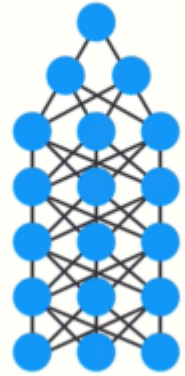
- You may have noticed the odd distribution of colors in the hidden states.
- That is to illustrate an issue with RNN's known as short-term memory.

Final hidden state of the RNN

Training a normal NN

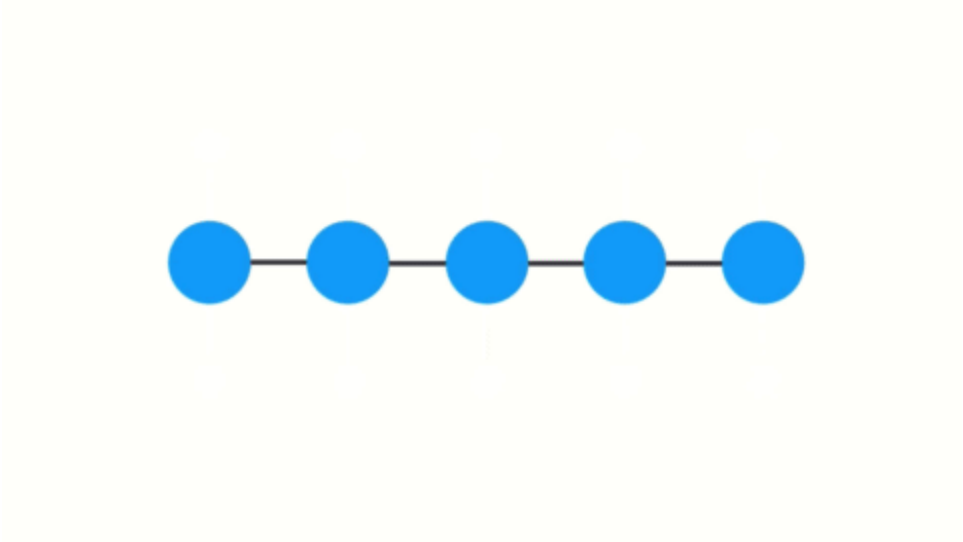
Training a neural network has three major steps:

- First, it does a forward pass and makes a prediction.
- Second, it compares the prediction to the ground truth using a loss function.
- Last, it uses that error value to do back propagation which calculates the gradients for each node in the network.



How about the RNNs

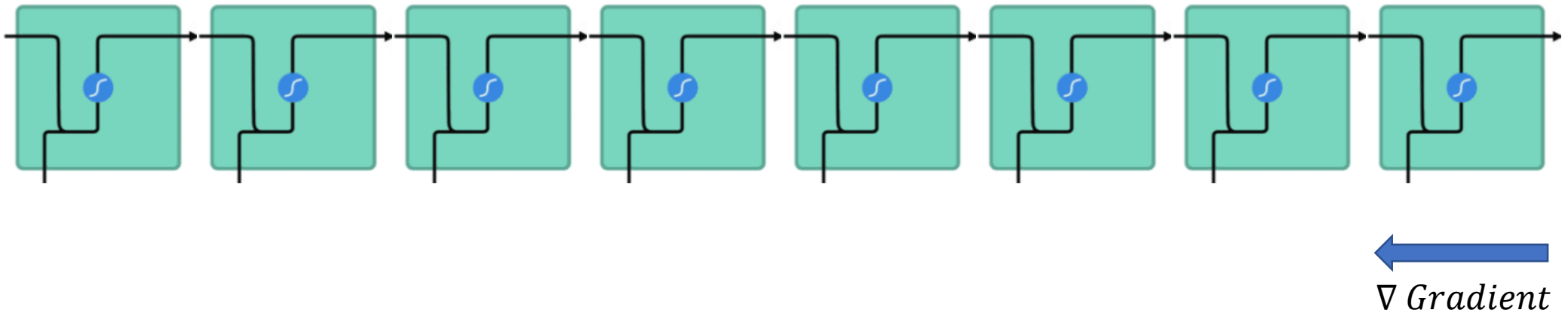
How about the RNNs



Gradients shrink as it back-propagates through time

Gradient and RNN

- Gradients are values used to update a neural networks weights.



Gradient Update Rule

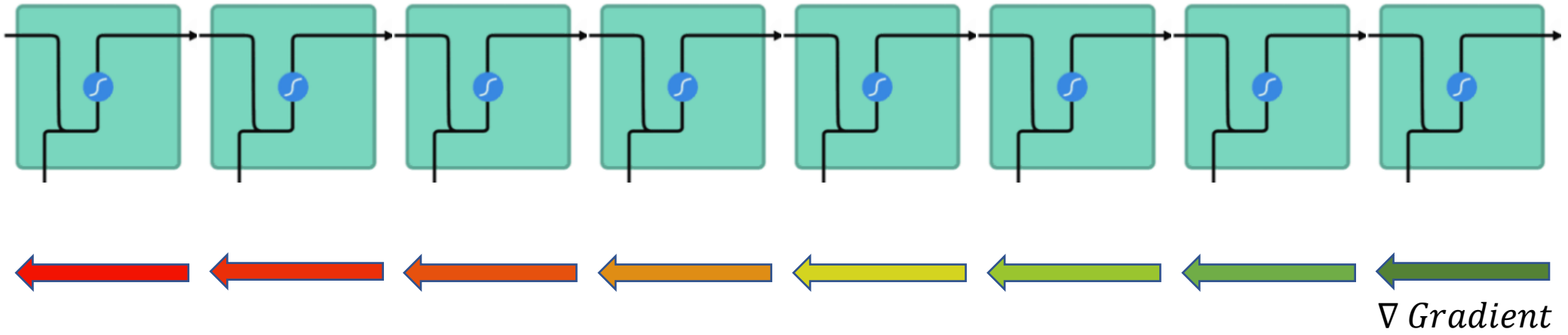
new weight = weight - learning rate*gradient

2.0999 = 2.1 - 0.001

Not much of a difference **update value**

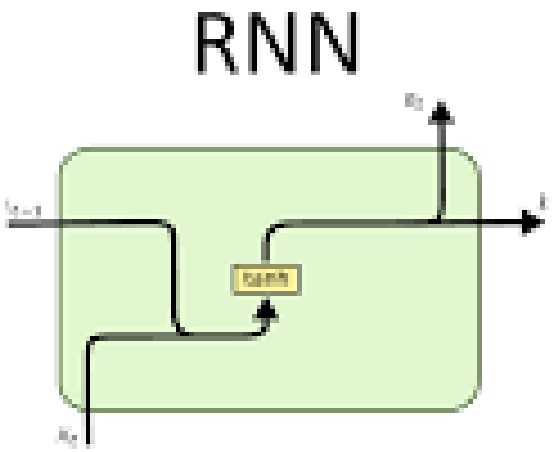
Back Propagation

- During back propagation, recurrent neural networks suffer from the vanishing gradient problem



Short Term Memory

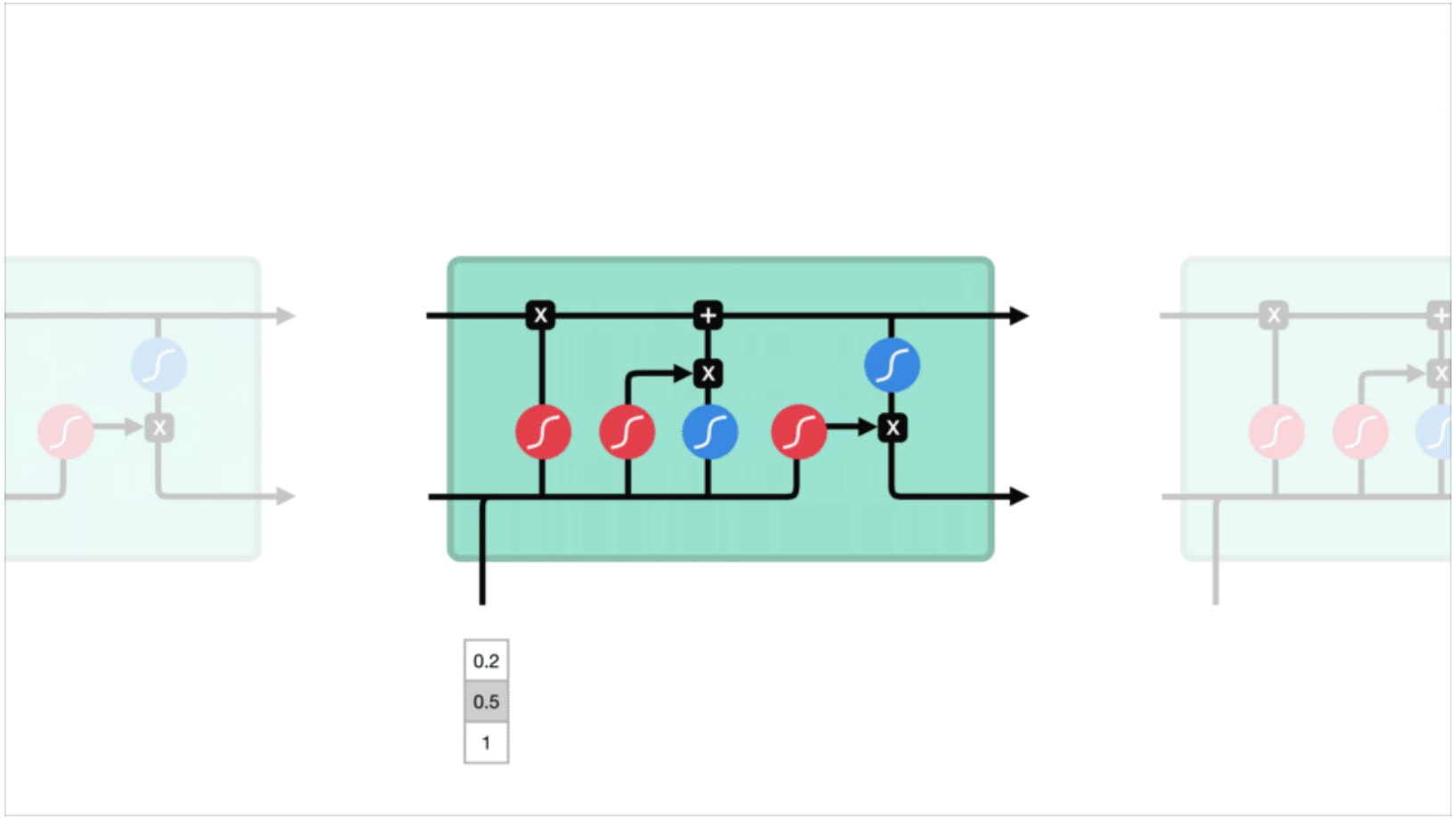
What is the solution



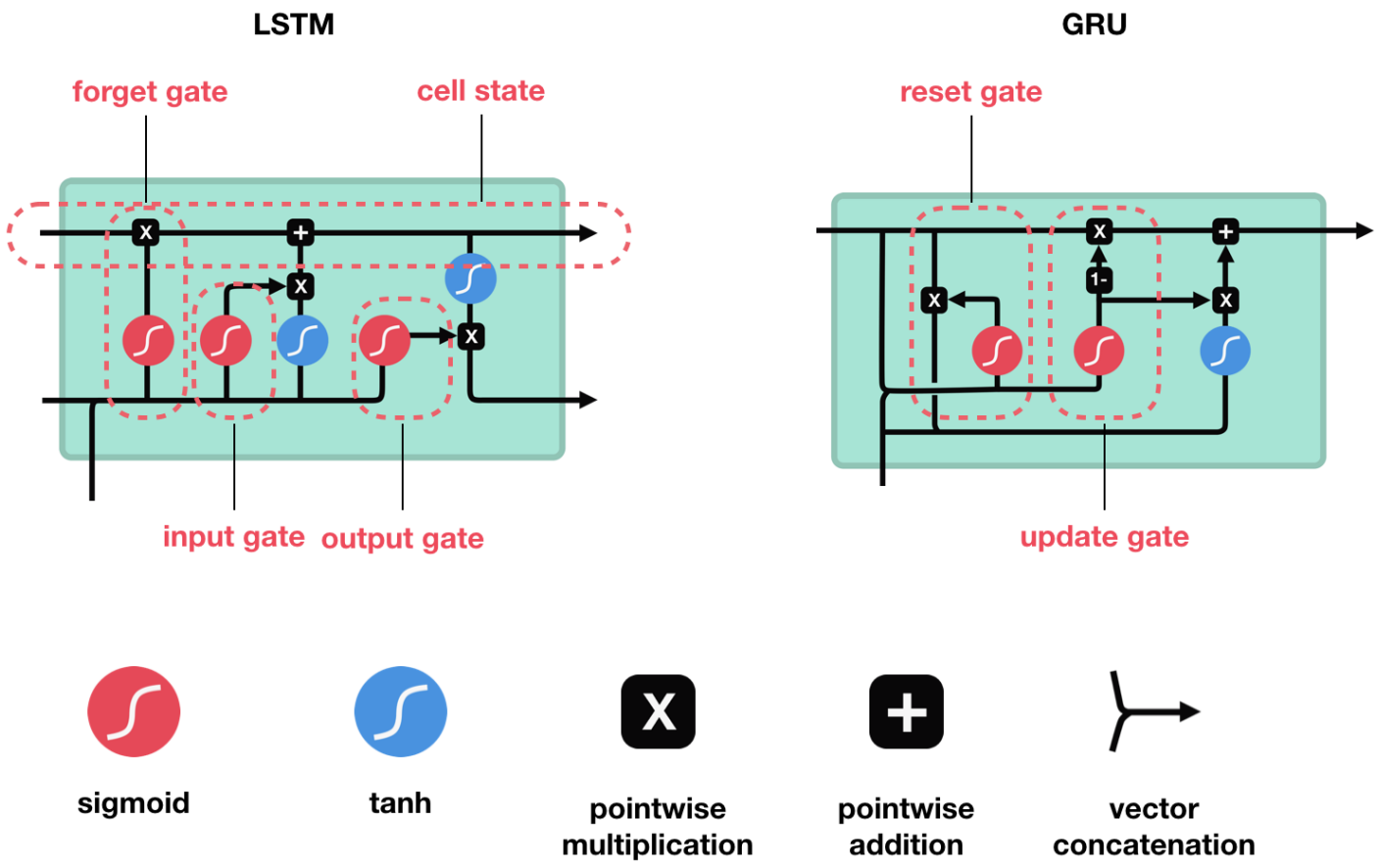
- See this link:

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

LSTM and GRU



Solving the Problem



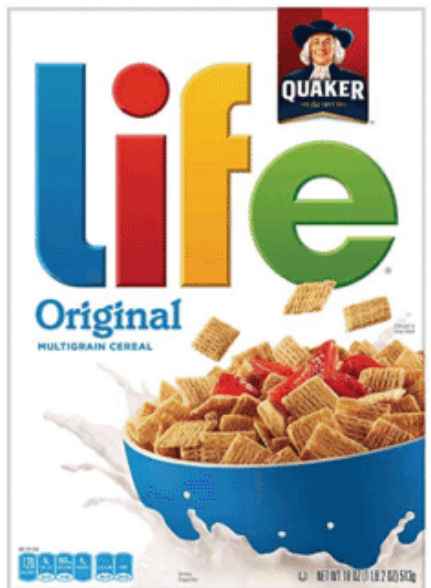
Customers Review 2,491

Thanos

September 2018

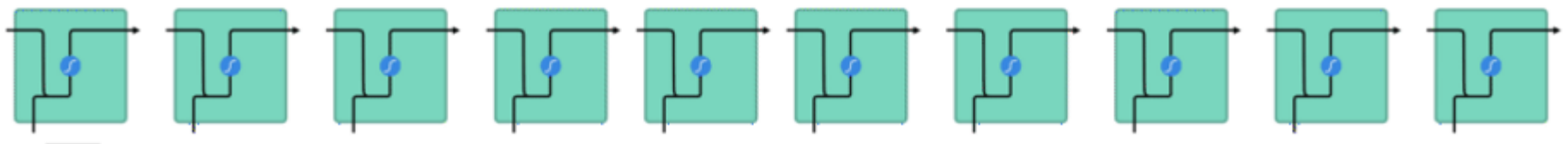
Verified Purchase

Amazing! This box of cereal gave me a perfectly balanced breakfast, as all things should be. I only ate half of it but will definitely be buying again!



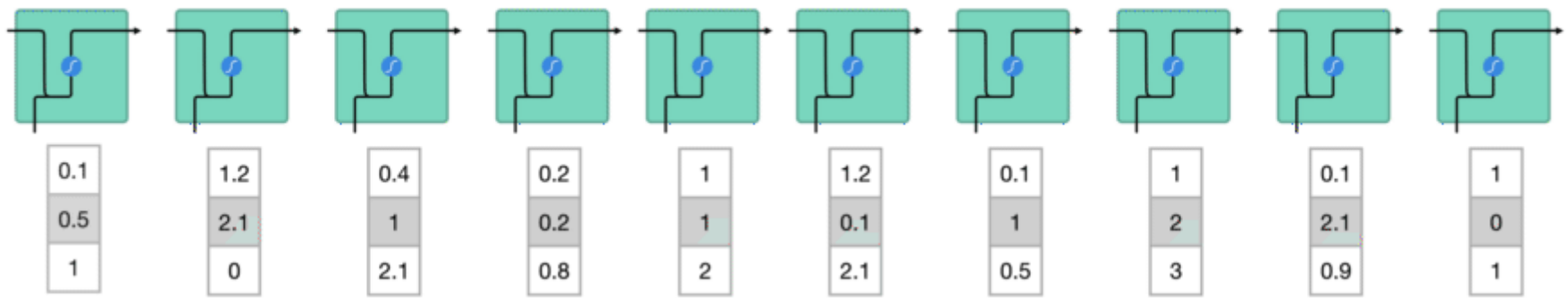
A Box of Cereal

$3.99

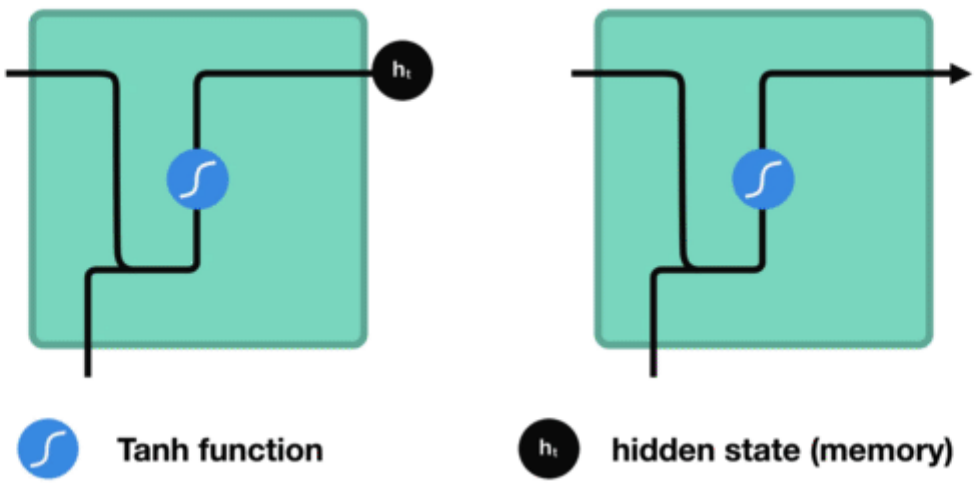


This box of cereal gave me a perfectly balanced breakfast

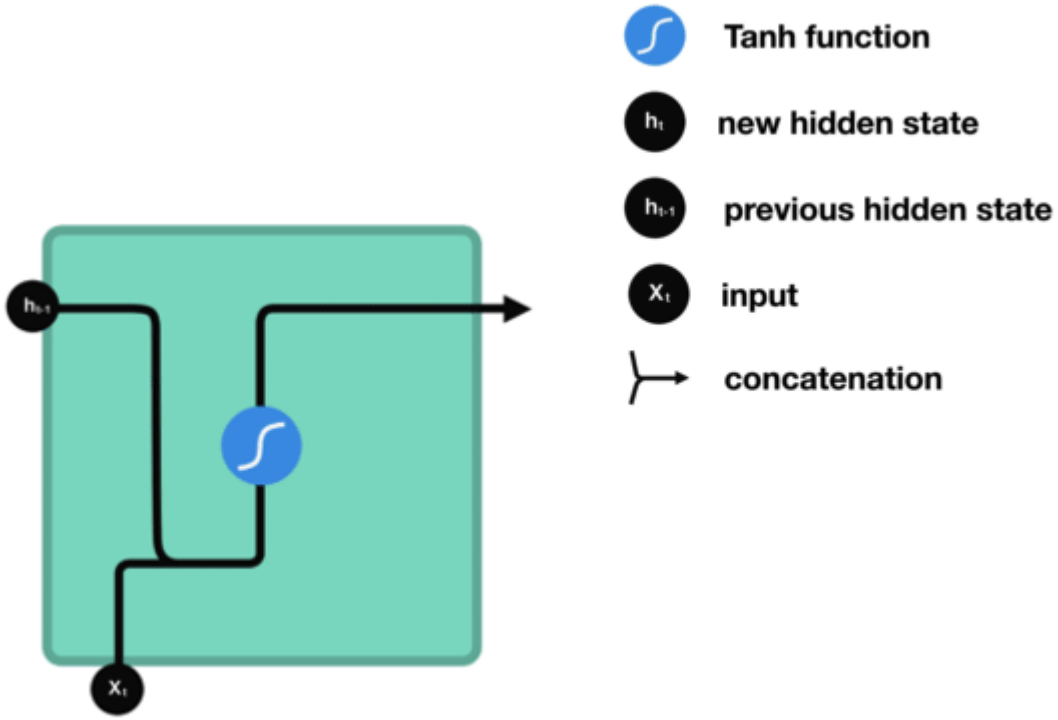
- Processing sequence one by one



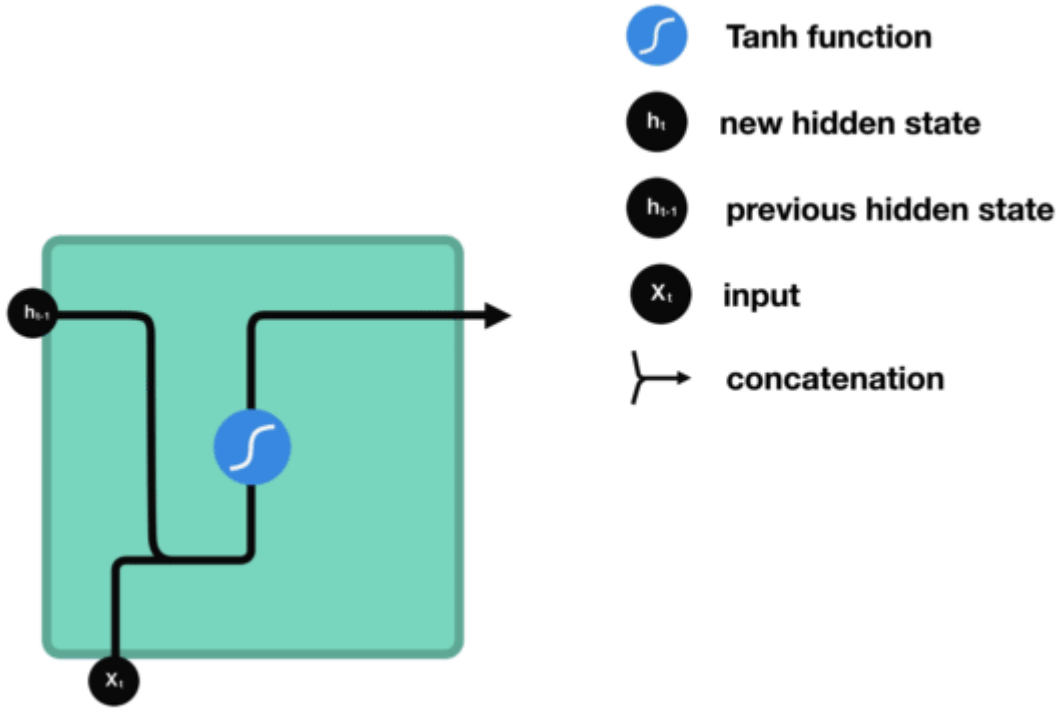
- Passing hidden state to next time step



RNN Cell


Tanh function
new hidden state
previous hidden state
input
concatenation

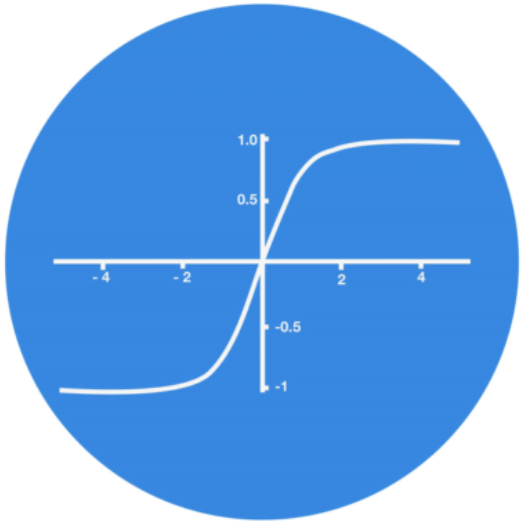
RNN Cell


Tanh function
new hidden state
previous hidden state
input
concatenation

Tanh activation

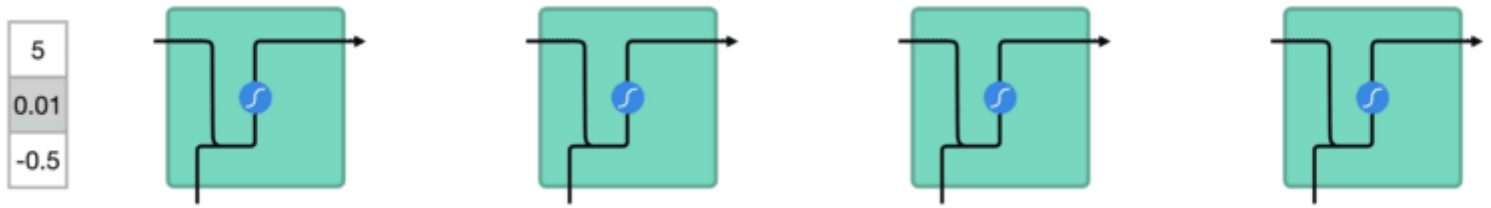
- The tanh activation is used to help regulate the values flowing through the network. The tanh function squishes values to always be between -1 and 1.

5
0.1
-0.5



Vector transformations with\ without tanh

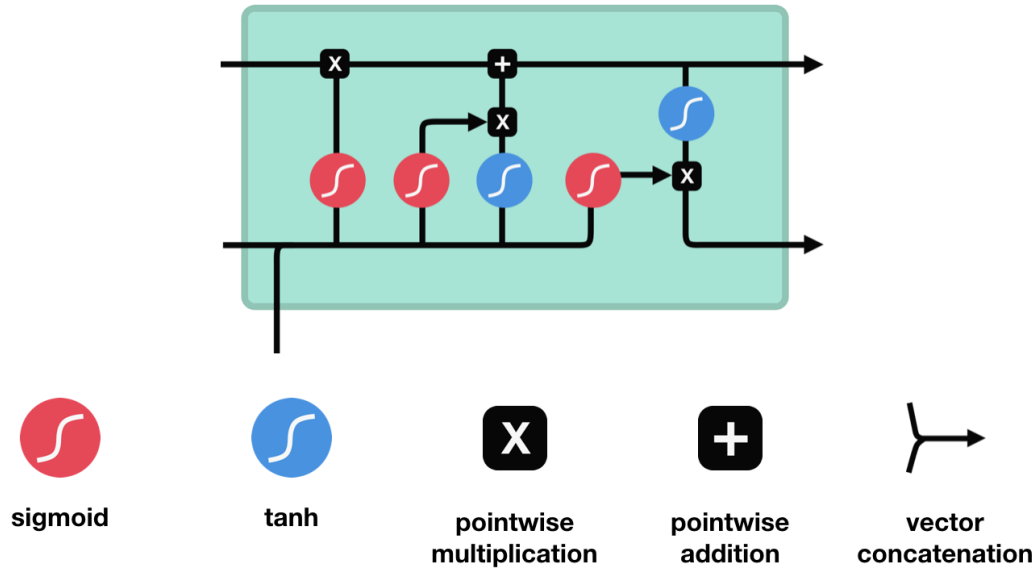
Vector transformations without tanh



Vector transformations with tanh

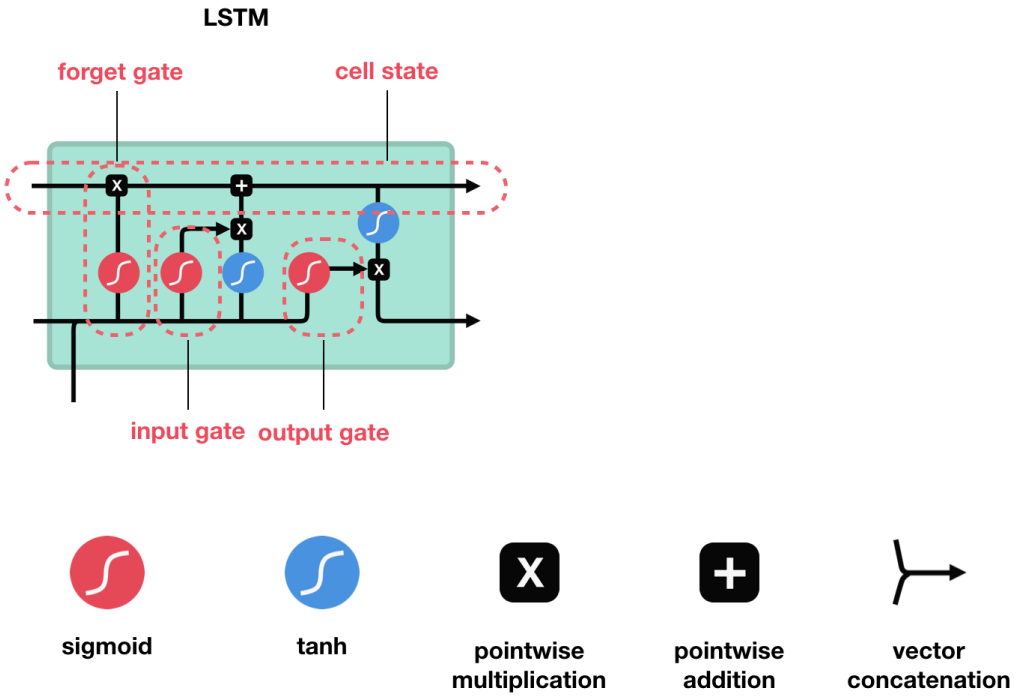
LSTM

- An LSTM has a similar control flow as a recurrent neural network.
- It processes data passing on information as it propagates forward.
- The differences are the operations within the LSTM's cells.
- These operations are used to allow the LSTM to keep or forget information.



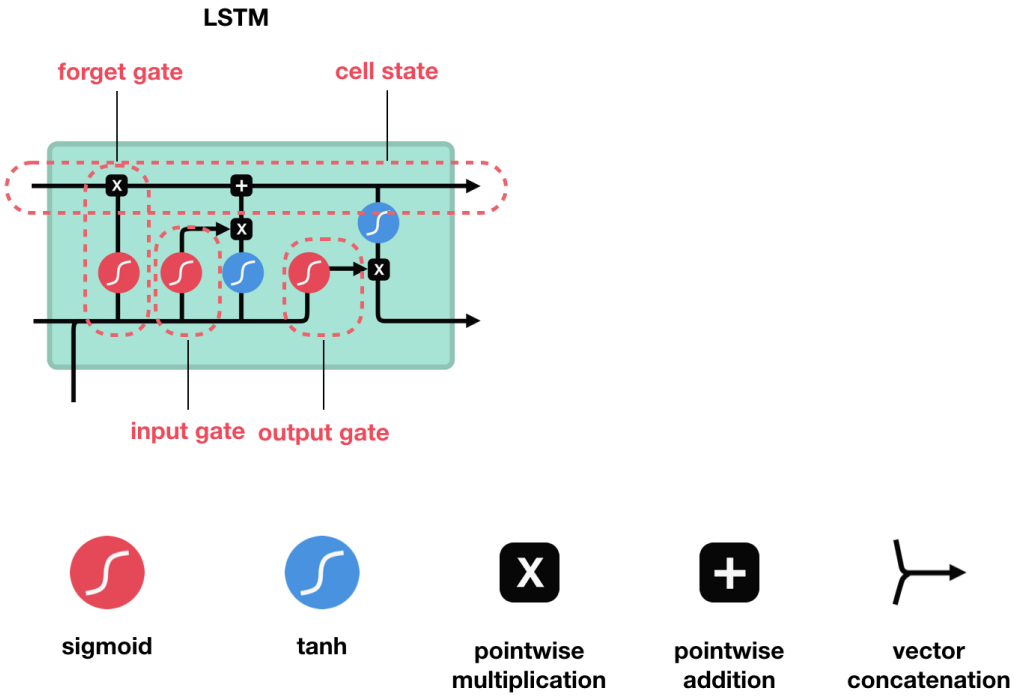
LSTM

- The core concept of LSTM's are the cell state, and it's various gates.
- The cell state act as a transport highway that transfers relative information all the way down the sequence chain.
- You can think of it as the "memory" of the network.



LSTM

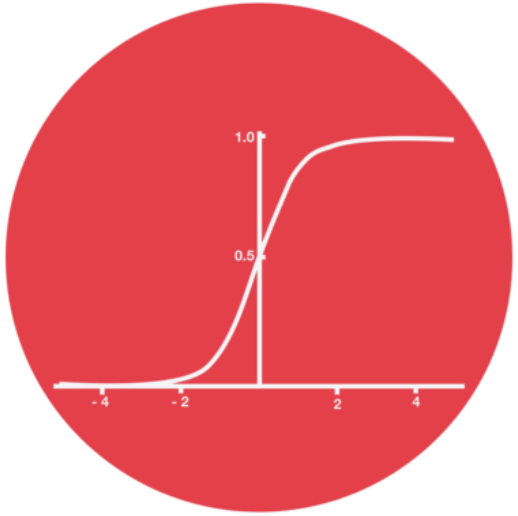
- So even information from the earlier time steps can make it's way to later time steps, reducing the effects of short-term memory.
- As the cell state goes on its journey, information get's added or removed to the cell state via gates.
- The gates are different NNS that decide which information is allowed on the cell state



Sigmoid

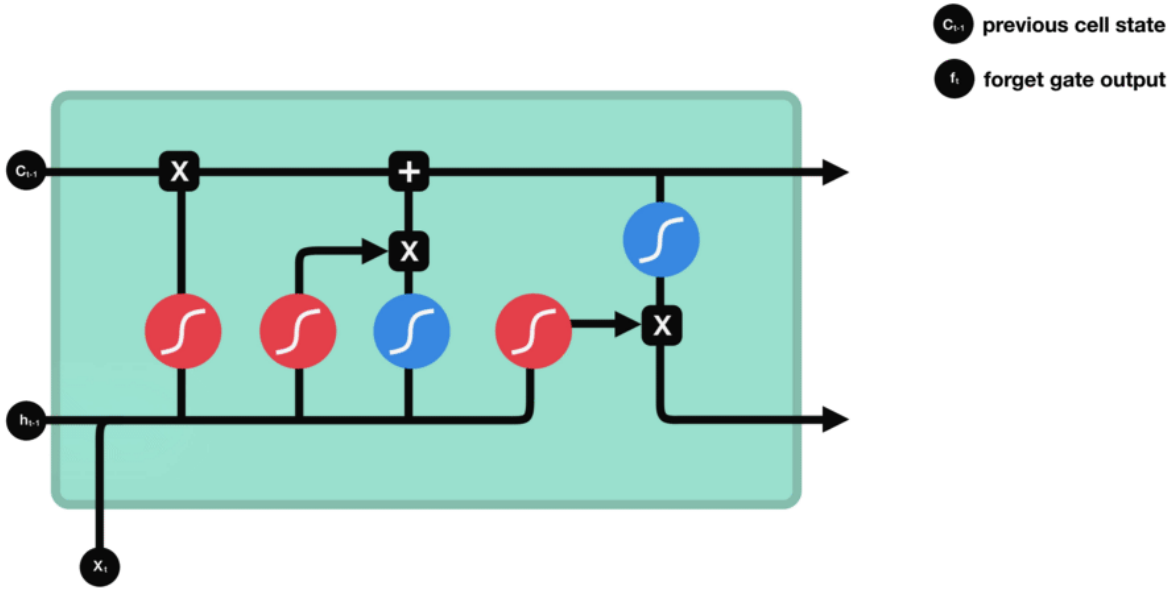
- A sigmoid activation is similar to the tanh activation. Instead of squishing values between -1 and 1
- Helpful to update or forget data
- Any number getting multiplied by 0 is 0, causing values to disappears or be “forgotten.”
- Any number multiplied by 1 is the same value therefore that value stay’s the same or is “kept.”
- The network can learn which data is not important therefore can be forgotten or which data is important to keep.

5
0.1
-0.5



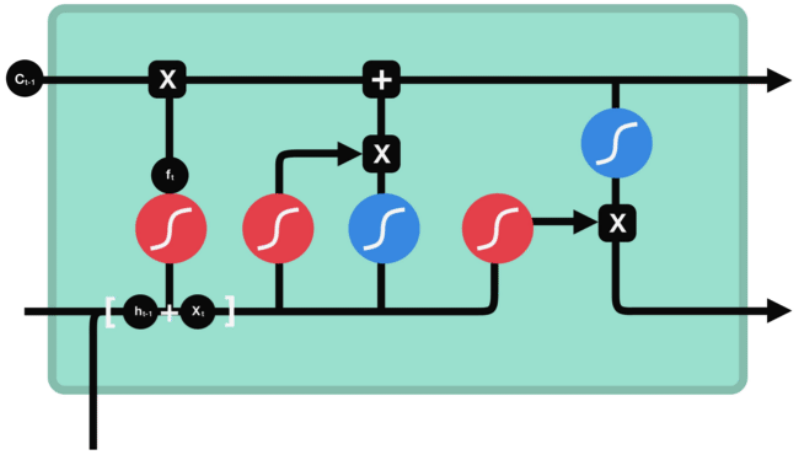
Forget gate operations

- This gate decides what information should be thrown away or kept.
- Information from the previous hidden state and information from the current input is passed through the sigmoid function.



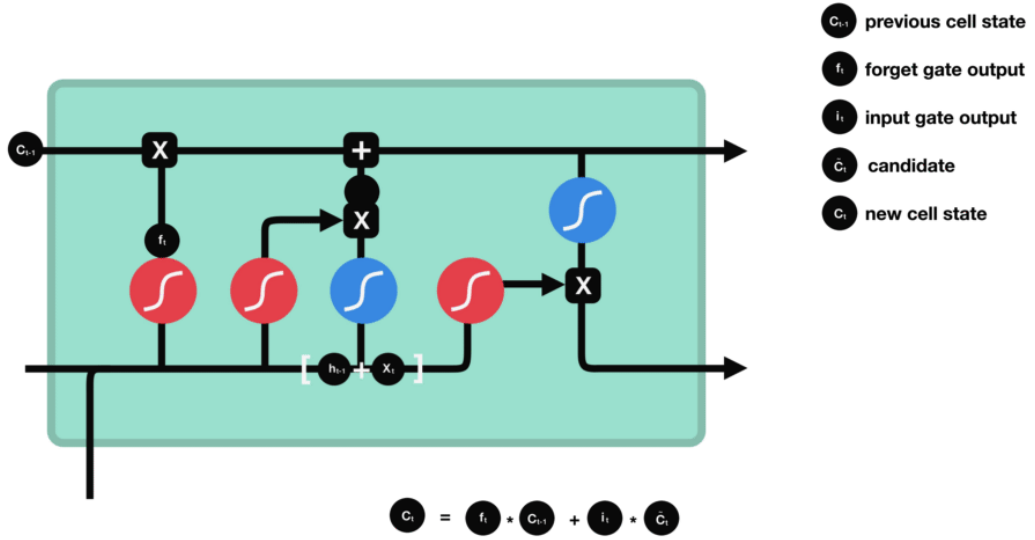
Input gate operations

- To update the cell state, we have the input gate.
- First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated.
- You also pass the hidden state and current input to help regulate the network.
- Then you multiply the tanh output with the sigmoid output.
- The sigmoid output will decide which information is important to keep from the tanh output.



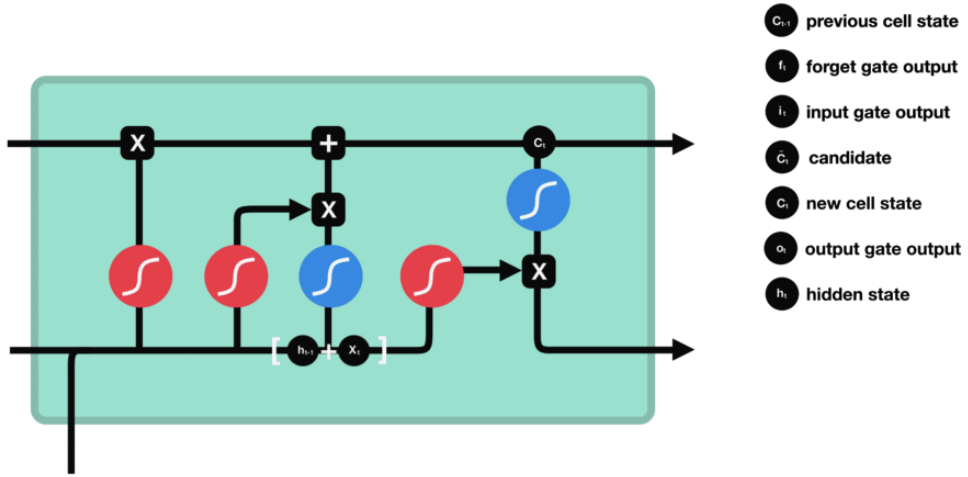
Cell state

- First, the cell state gets pointwise multiplied by the forget vector.
- This has a possibility of dropping values in the cell state if it gets multiplied by values near 0.
- Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant.
- That gives us our new cell state.



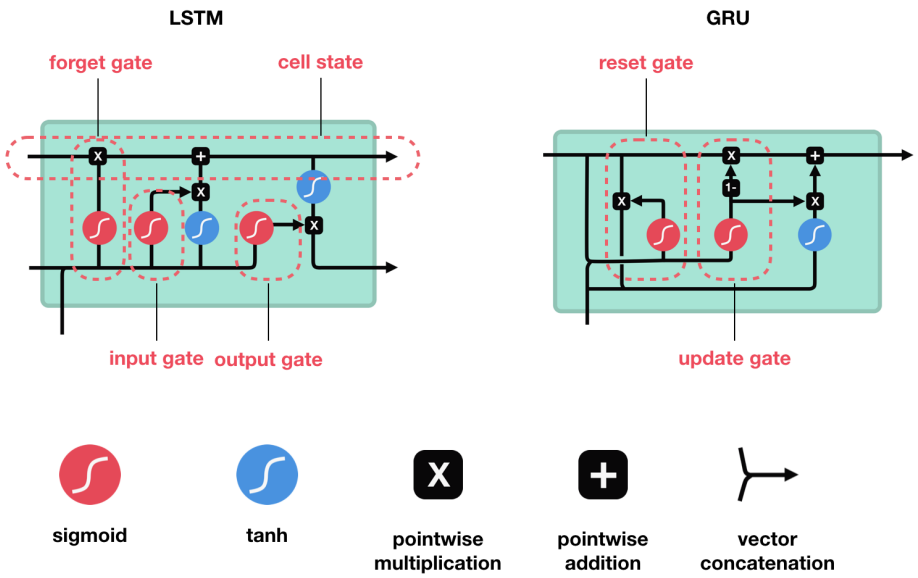
Output gate operations

- The output gate decides what the next hidden state should be.
- First, we pass the previous hidden state and the current input into a sigmoid function.
- Then we pass the newly modified cell state to the tanh function.
- We multiply the tanh output with the sigmoid output to decide what information the hidden state should carry. The output is the hidden state.



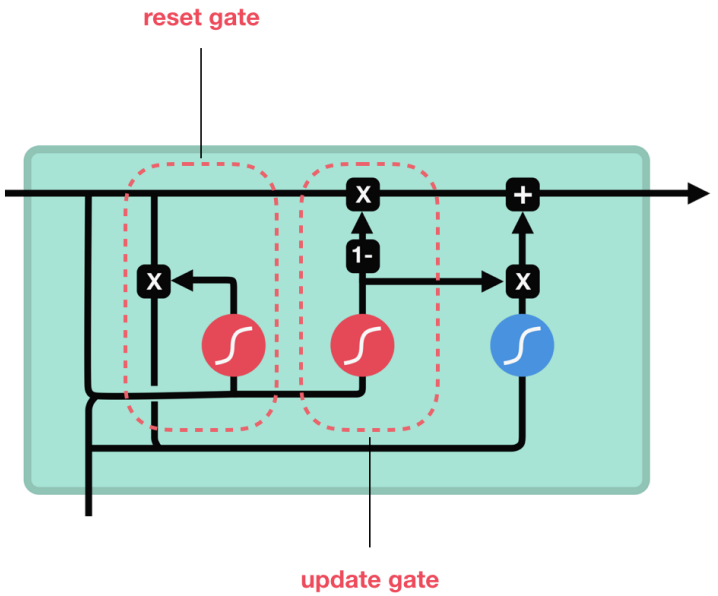
To Review:

- The Forget gate decides what is relevant to keep from prior steps.
- The input gate decides what information is relevant to add from the current step.
- The output gate determines what the next hidden state should be.



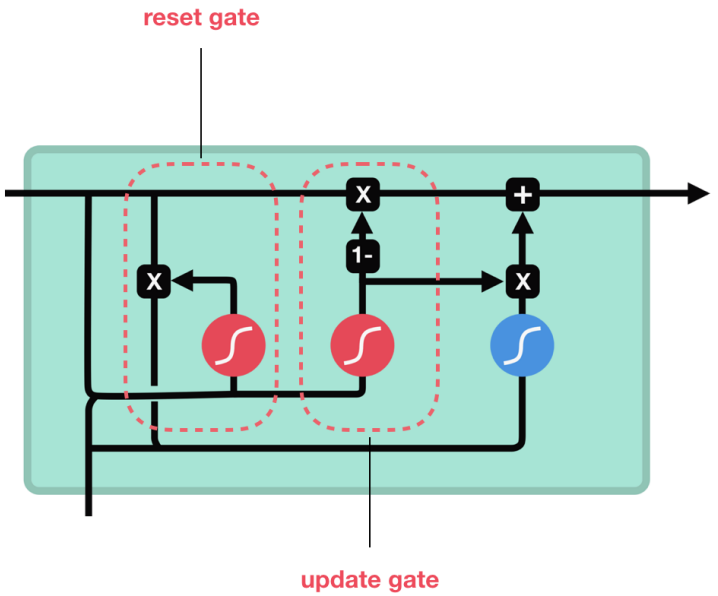
GRU

- The GRU is the newer generation of RNNs.
- GRU's got rid of the cell state and used the hidden state to transfer information.
- It also only has two gates, a reset gate and update gate.



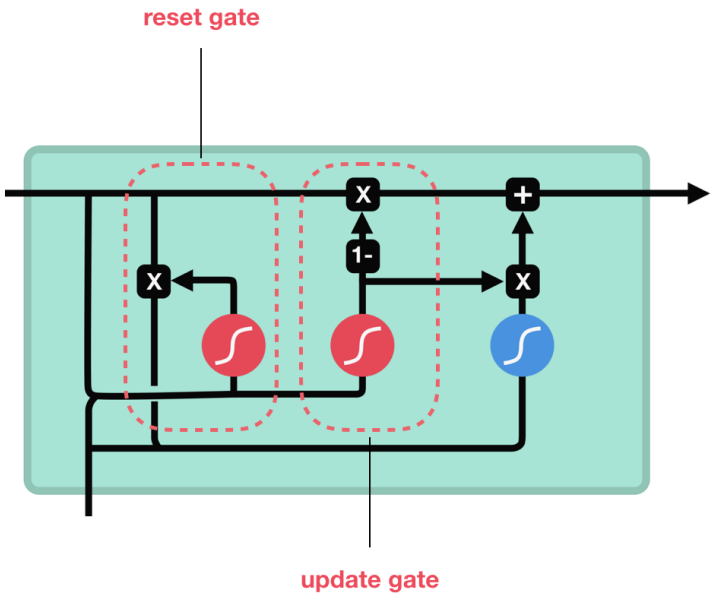
Update Gate

- The update gate acts similar to the forget and input gate of an LSTM.
- It decides what information to throw away and what new information to add.



Reset Gate

- The reset gate is another gate is used to decide how much past information to forget.



Recurrent Neural Networks

Recurrent Neural Network Model

Motivating example

NLP

x: Harry Potter and Hermione Granger invented a new spell.

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ ..... $x^{<t>}$ ..... $x^{<9>}$

$T_x = 9$

y: 1 1 0 1 1 0 0 0 0

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ ..... $y^{<9>}$

$T_y = 9$

$X^{(i)<t>}$

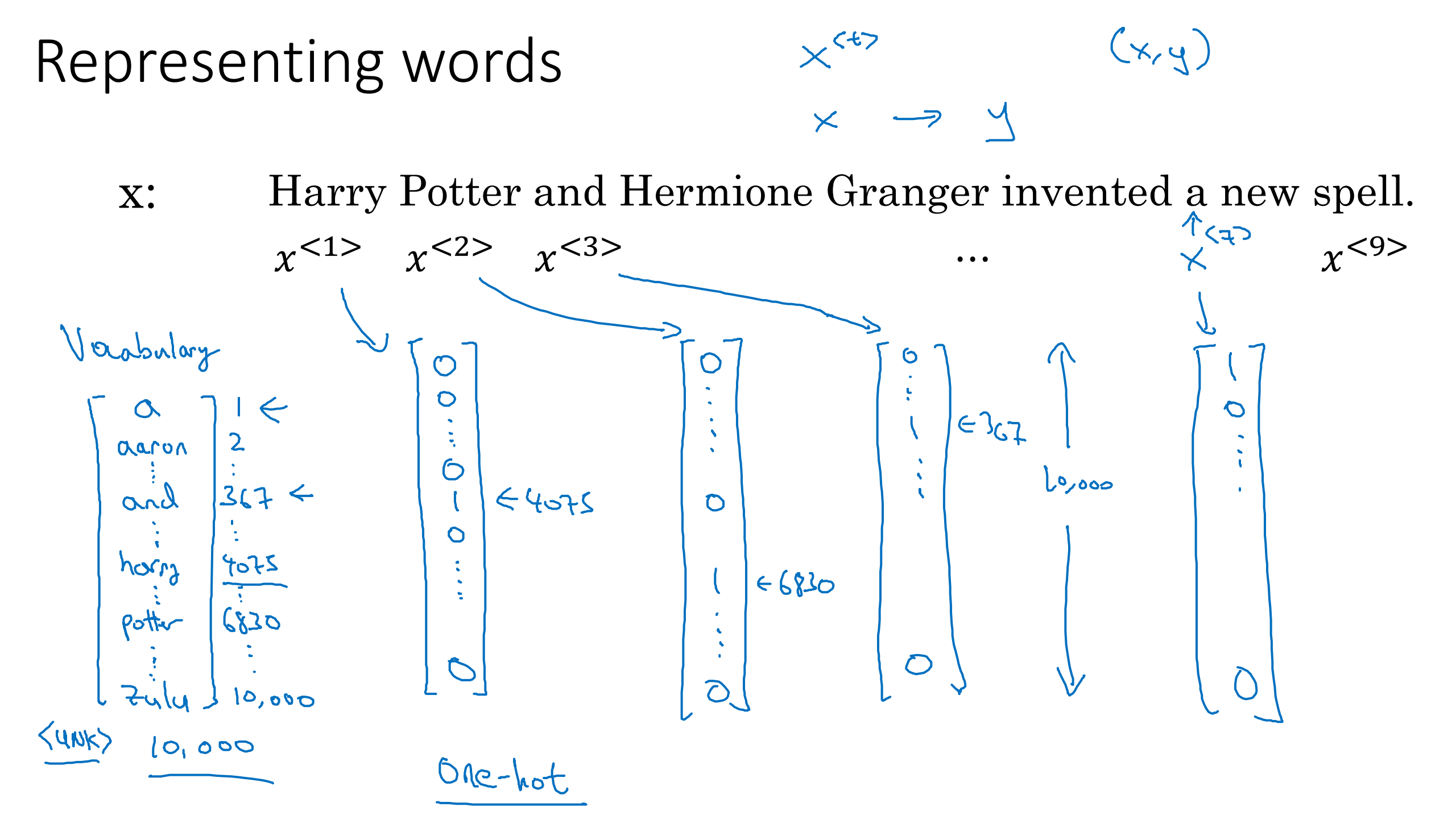
$y^{(i)<t>}$

$T_x^{(i)} = 9$

$T_y^{(i)}$

15

Representing words



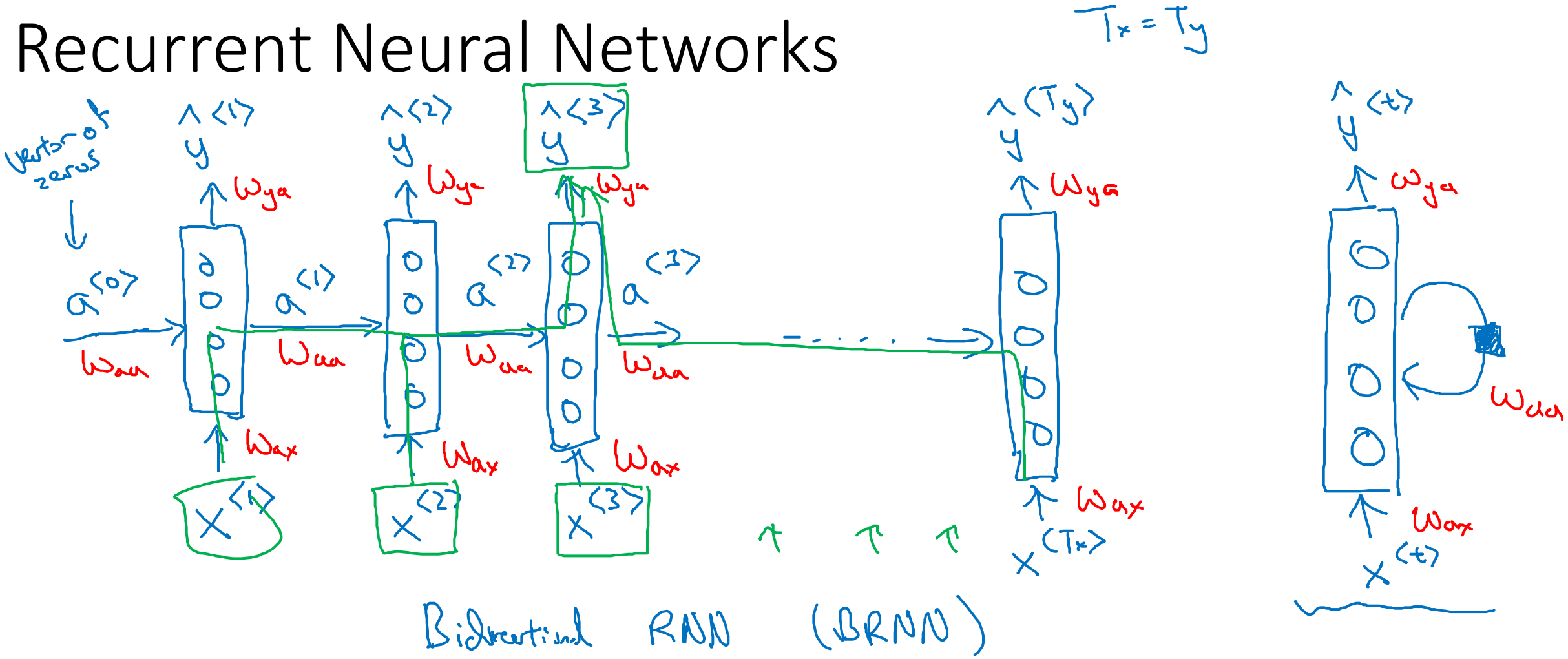
Representing words

x: Harry Potter and Hermione Granger invented a new spell.

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ … $x^{<9>}$

And = 367
Invented = 4700
A = 1
New = 5976
Spell = 8376
Harry = 4075
Potter = 6830
Hermione = 4200
Gran… = 4000

Recurrent Neural Networks

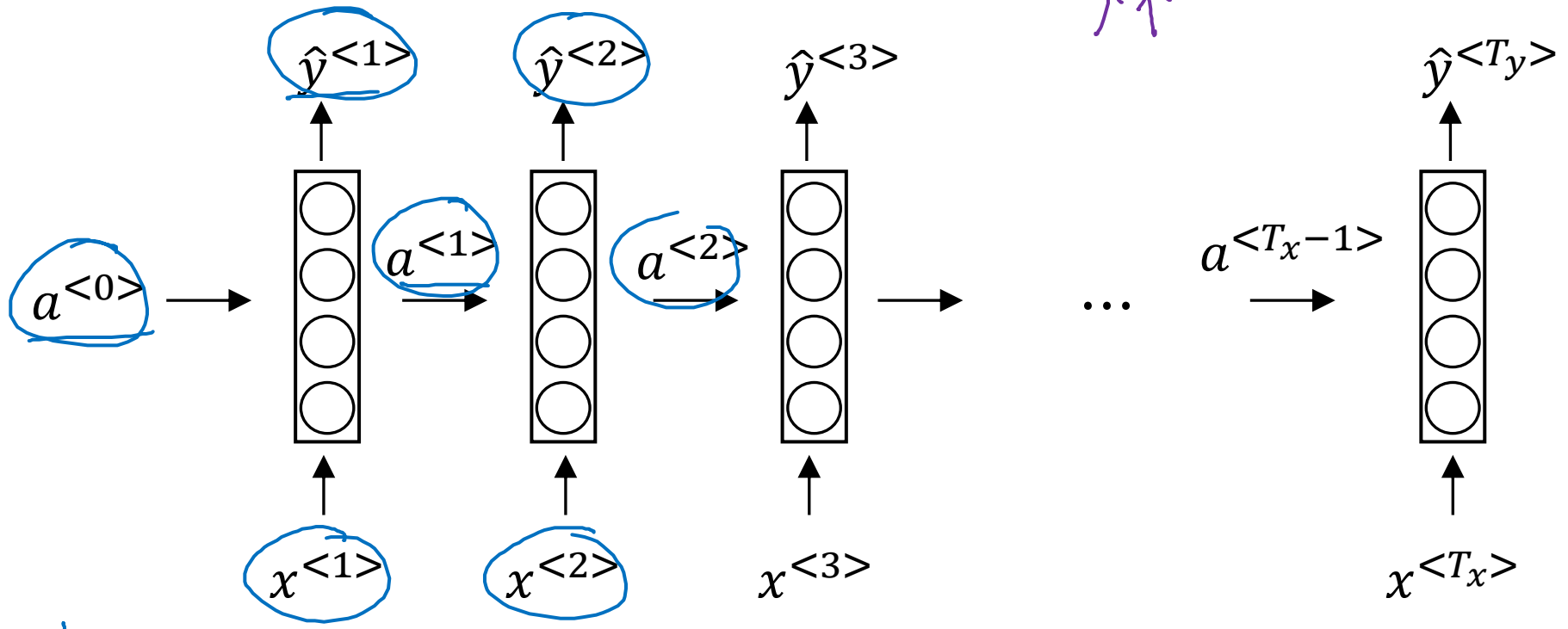


He said, “Teddy Roosevelt was a great President.”

He said, “Teddy bears are on sale!”

Forward Propagation

$a \leftarrow W_{ax} x^{\langle 1 \rangle}$



$a^{\langle 0 \rangle} = \vec{0}$.

$a^{\langle 1 \rangle} = g_1(W_{aa} a^{\langle 0 \rangle} + W_{ax} x^{\langle 1 \rangle} + b_a)$ ← tanh / ReLU

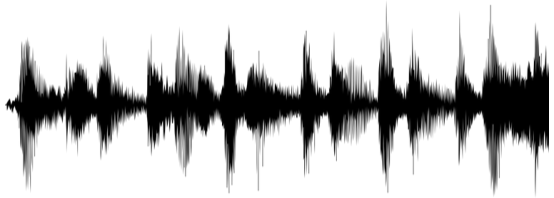
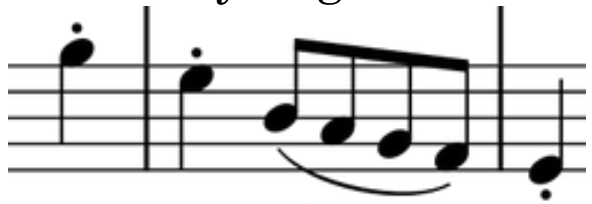
$\hat{y}^{\langle 1 \rangle} = g_2(W_{ya} a^{\langle 1 \rangle} + b_y)$ ← sigmoid

$$a^{\langle t \rangle} = g(W_{aa} a^{\langle t-1 \rangle} + W_{ax} x^{\langle t \rangle} + b_a)$$
$$\hat{y}^{\langle t \rangle} = g(W_{ya} a^{\langle t \rangle} + b_y)$$

Recurrent Neural Networks

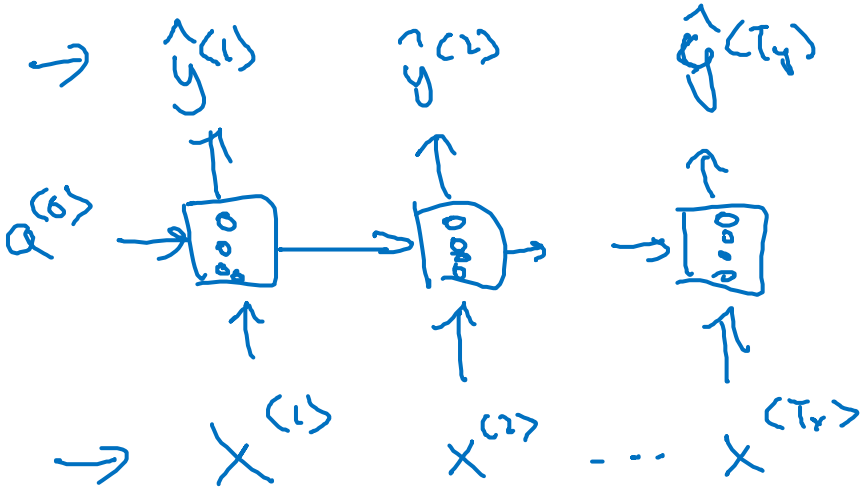
Different types of RNNs

Examples of sequence data

Speech recognition		→	"The quick brown fox jumped over the lazy dog."
Music generation	∅	→	
Sentiment classification	"There is nothing to like in this movie."	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

Examples of RNN architectures

$T_x = T_y$

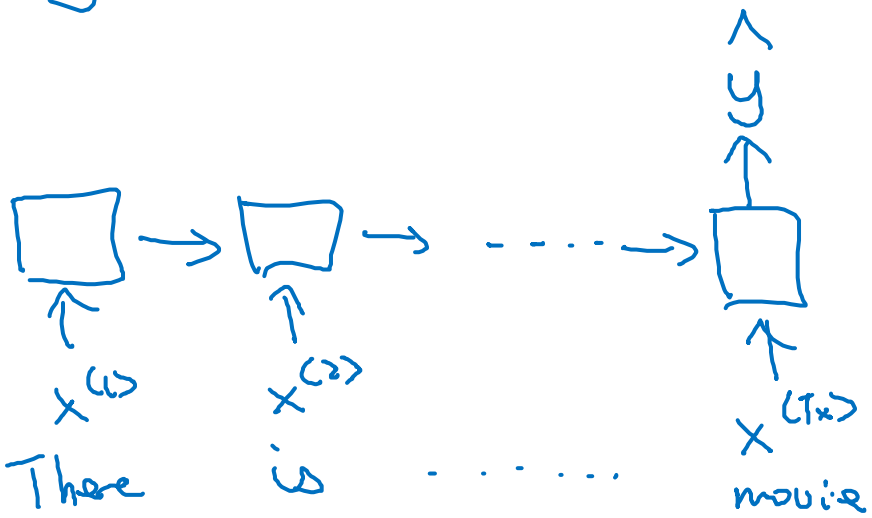


Many-to-many

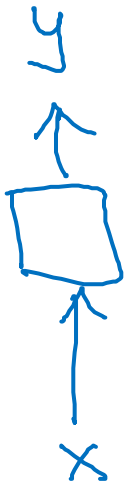
Sentiment classification

x = text

y = 0/1 ... 1...5

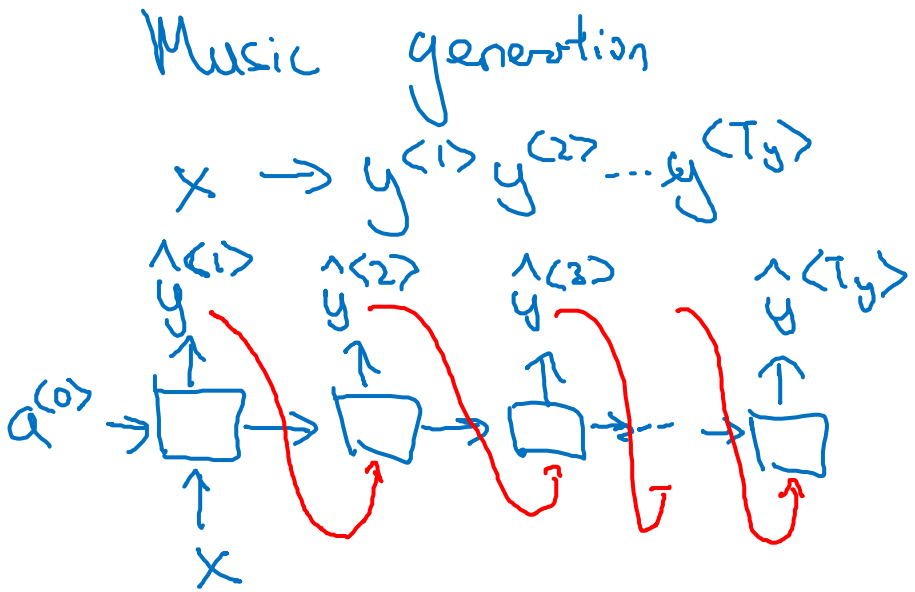


Many-to-one



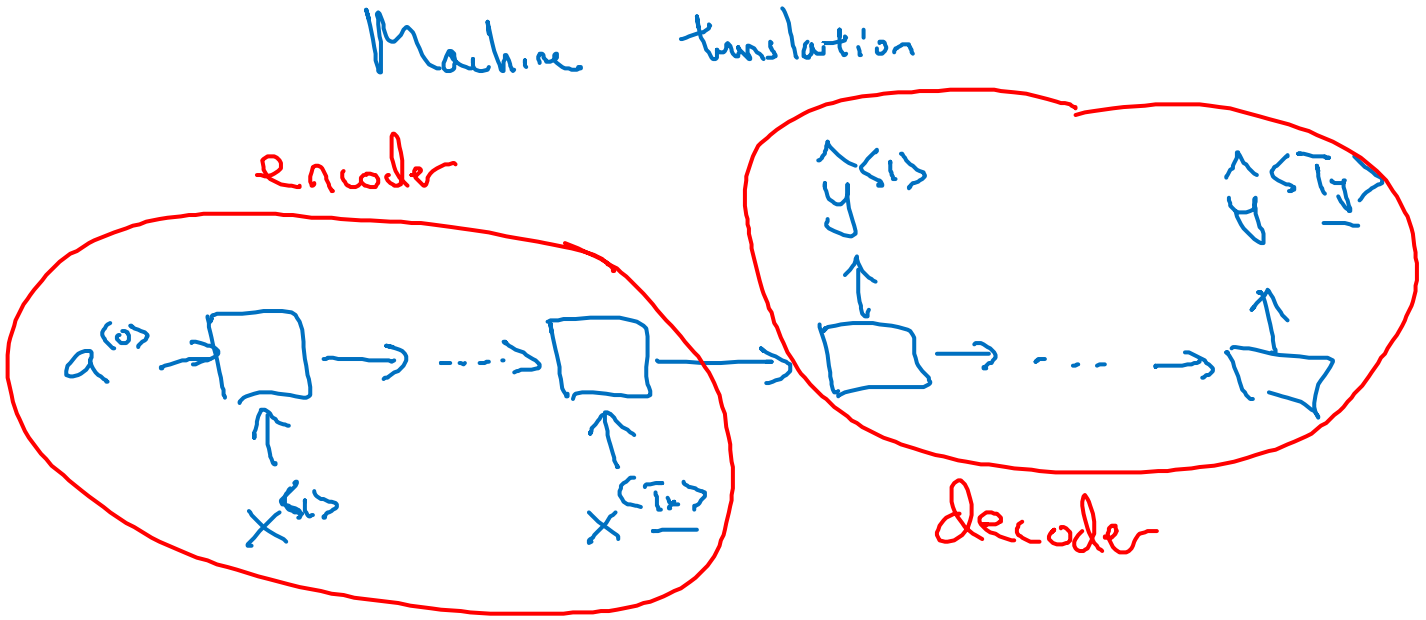
One-to-one

Examples of RNN architectures



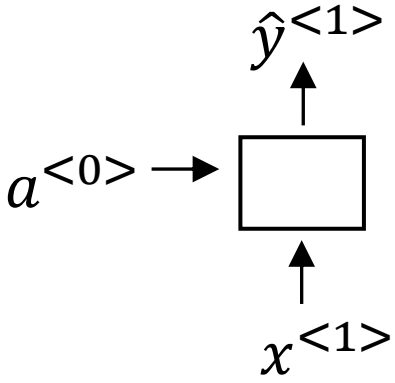
One-to-many

$x = \phi$

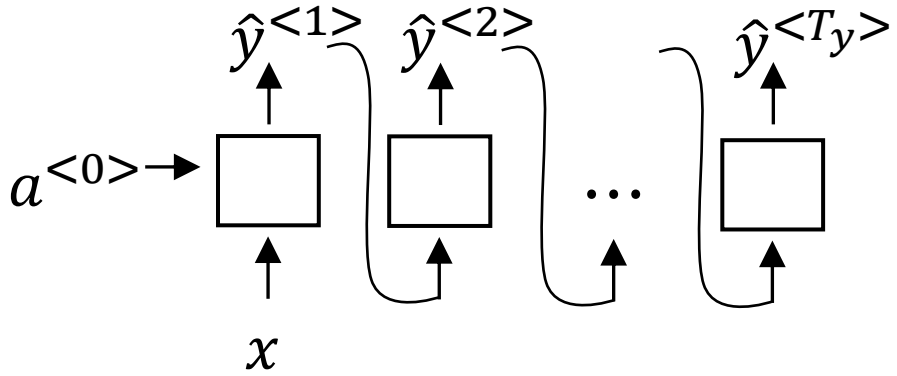


Many-to-many

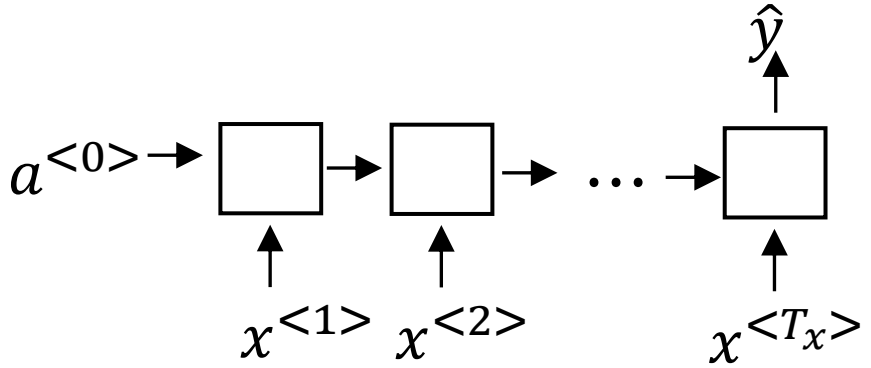
Summary of RNN types



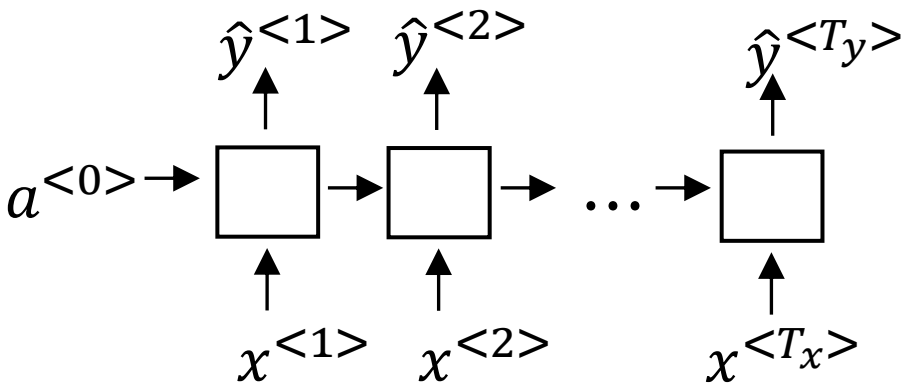
One to one



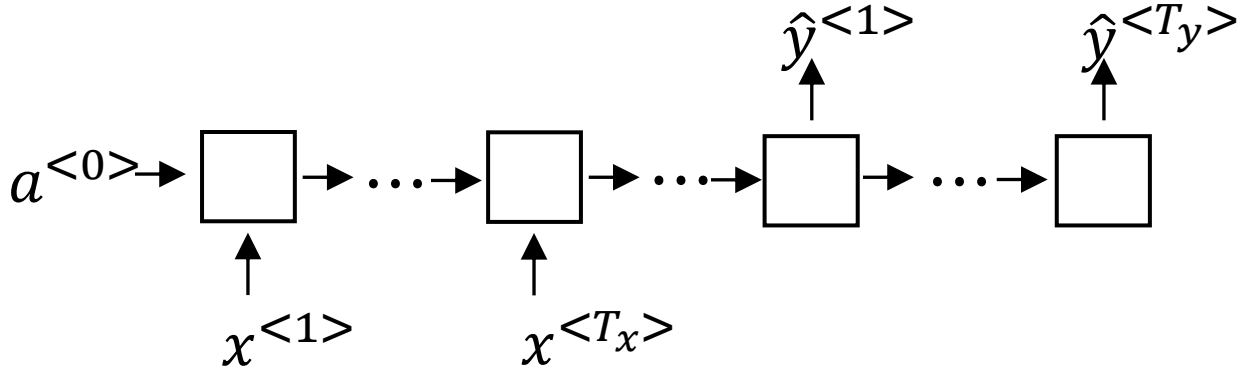
One to many



Many to one



Many to many



Many to many