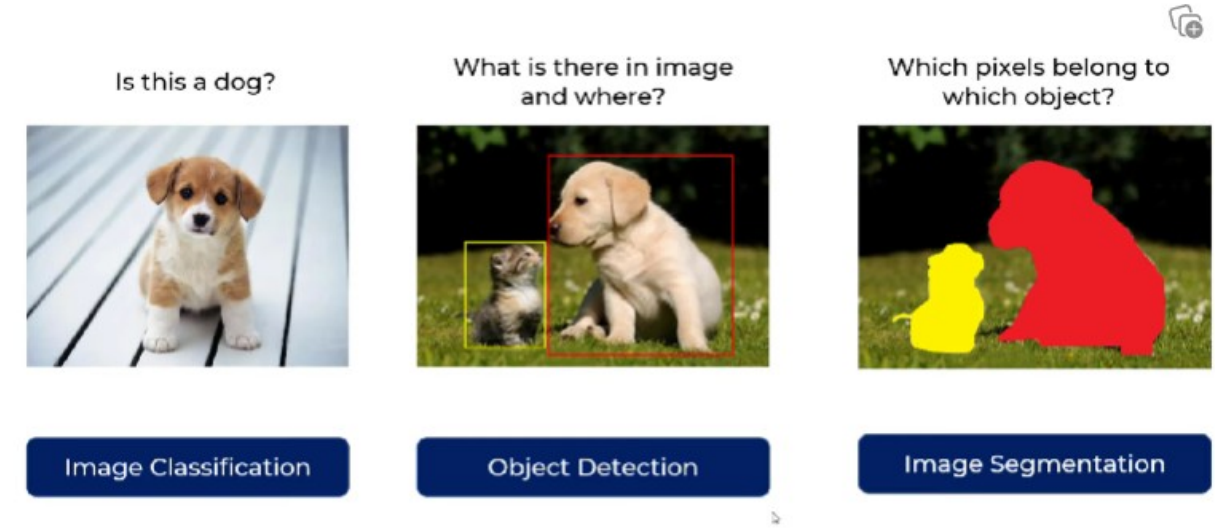
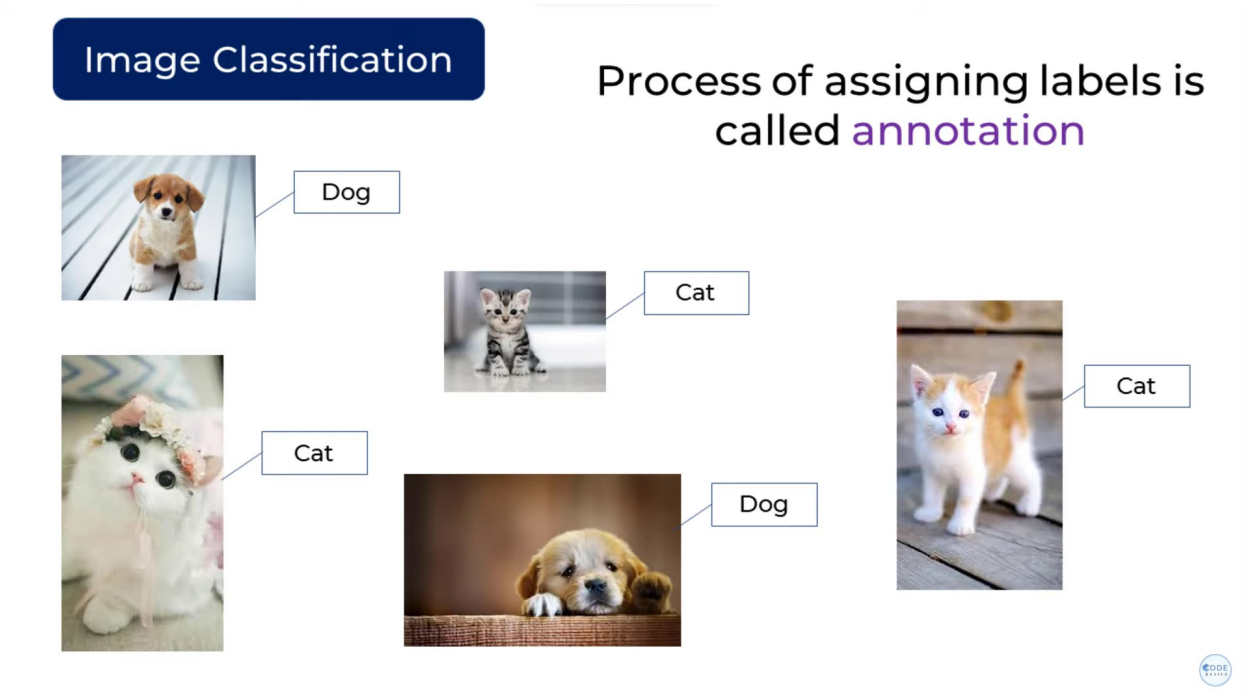
Computer Vision

Imagenet, Coco and google open images datasets are 3 most popular image datasets for computer vision. These datasets provides millions of hand annotated images with classification labels, bounding boxes for object detections and image segmentation masks. Data collection is the most difficult part in any supervised machine learning problem and availability of such datasets makes training CNNs very easy.





#

Object Detection

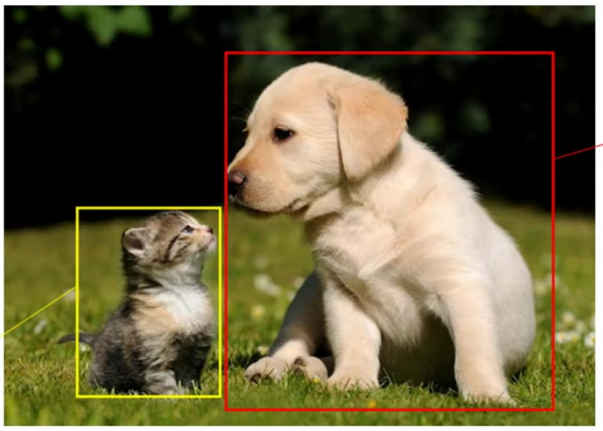
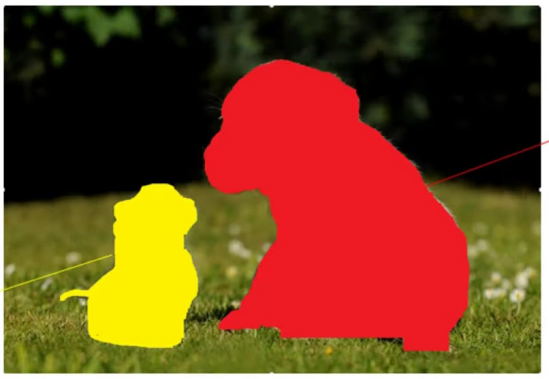


Image Segmentation



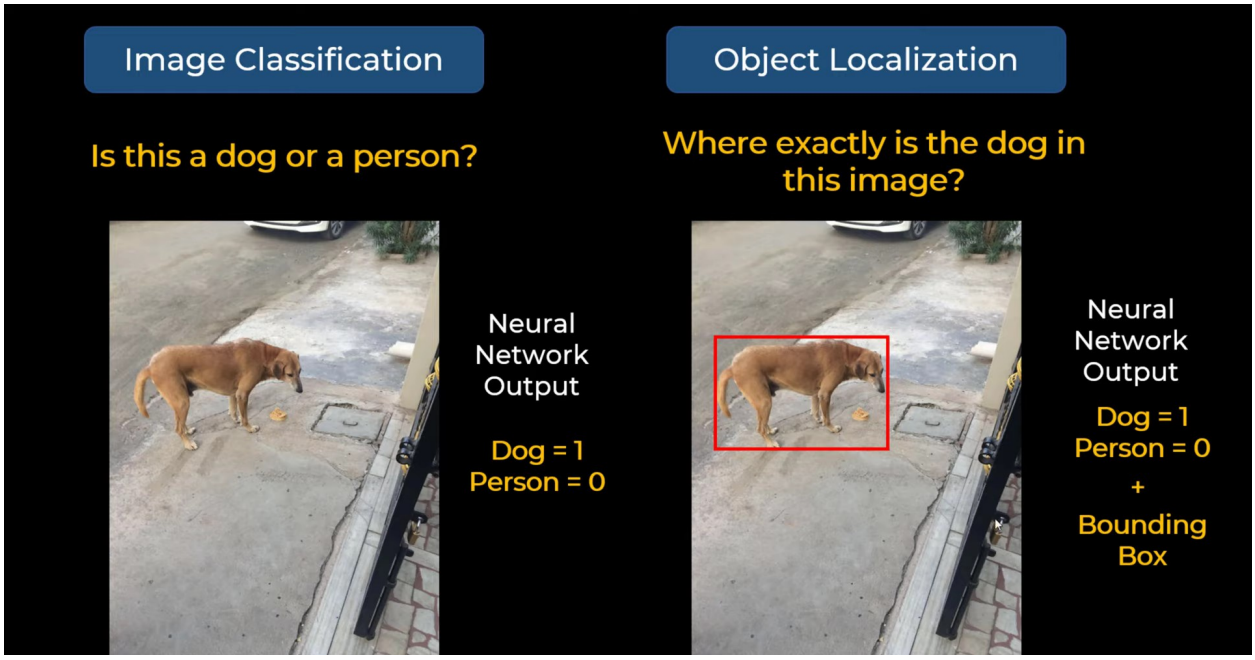
Sliding window object detection is a technique that allows you to detect objects in a picture. This technique is not very efficient as it is very compute intensive. Recently new techniques has been discovered that tried to improve performance such as R CNN, Fast R CNN, Faster R CNN etc. YOLO (You only look once) is a state of the art most modern technique that outperforms all other previous techniques such as sliding window object detection, R CNN, Fast and Faster R CNN etc. We will cover YOLO in future videos.

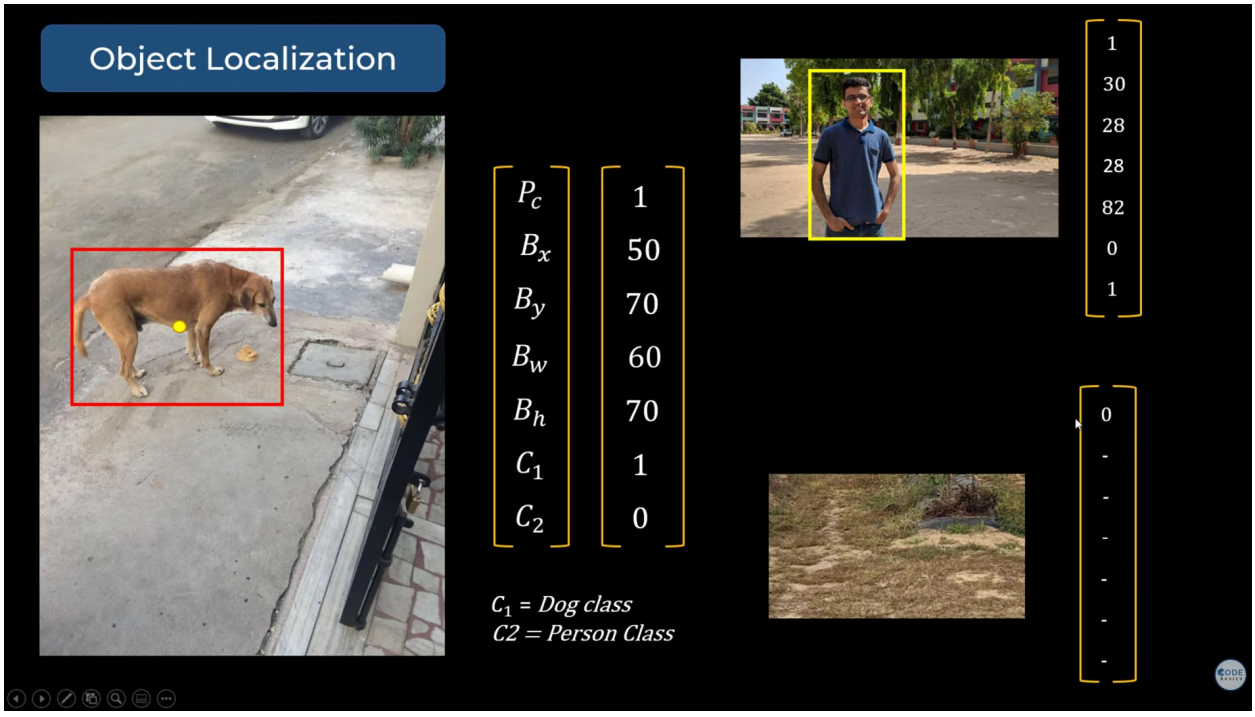
Try different sizes iteratively

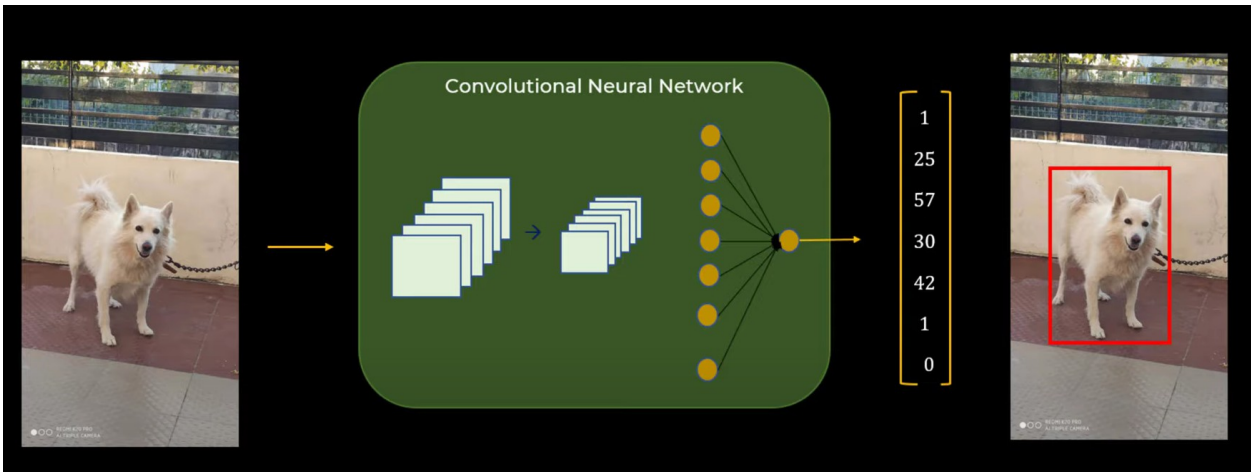
Disadvantages

Sliding Window Object Detection → R CNN → Fast R CNN → Faster R CNN → YOLO

YOLO(You Only Look Once)







This works ok only for single object. What about multiple objects in an image?

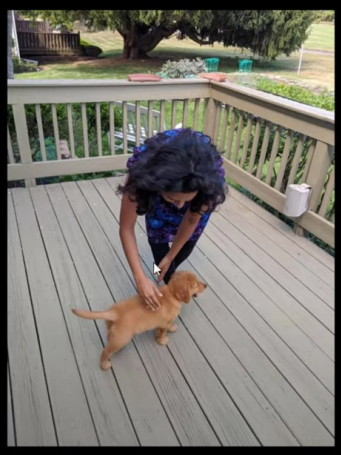


Image Classification

Is this a dog or a person?

Neural Network Output

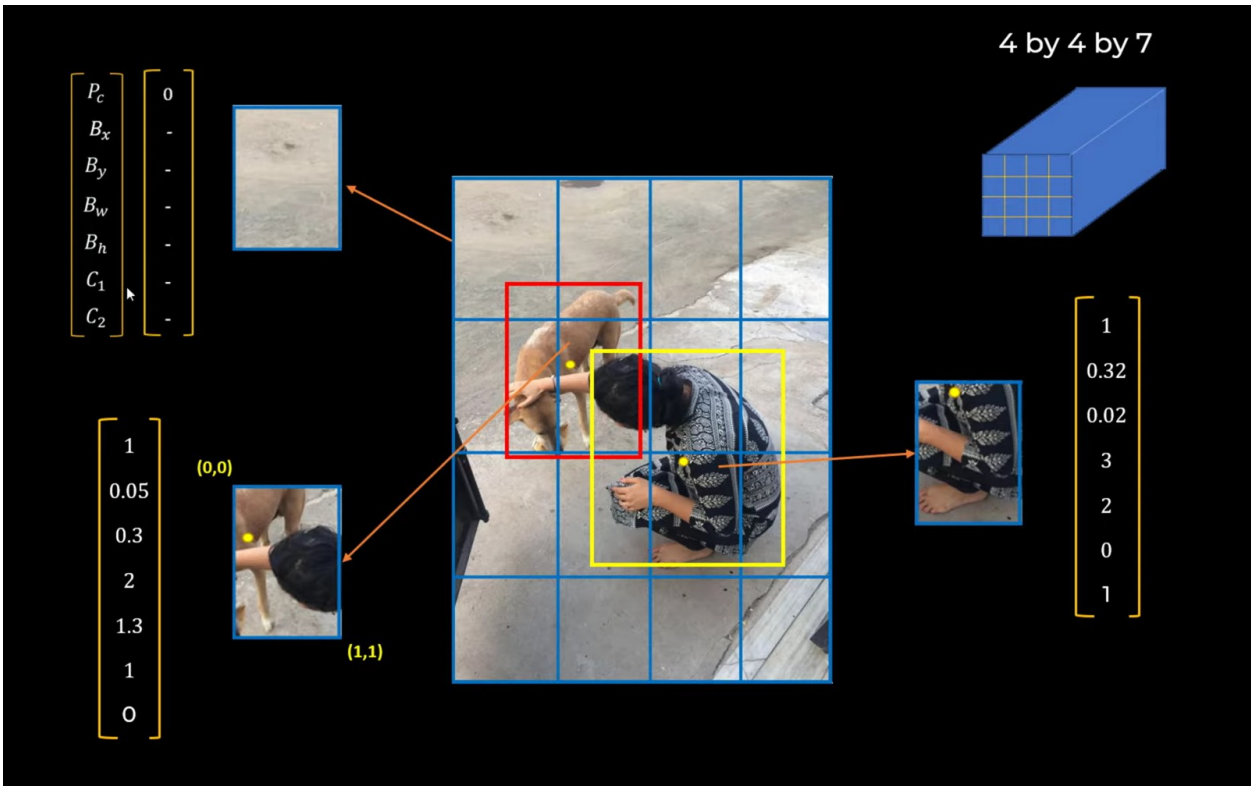
Dog = 1
Person = 0

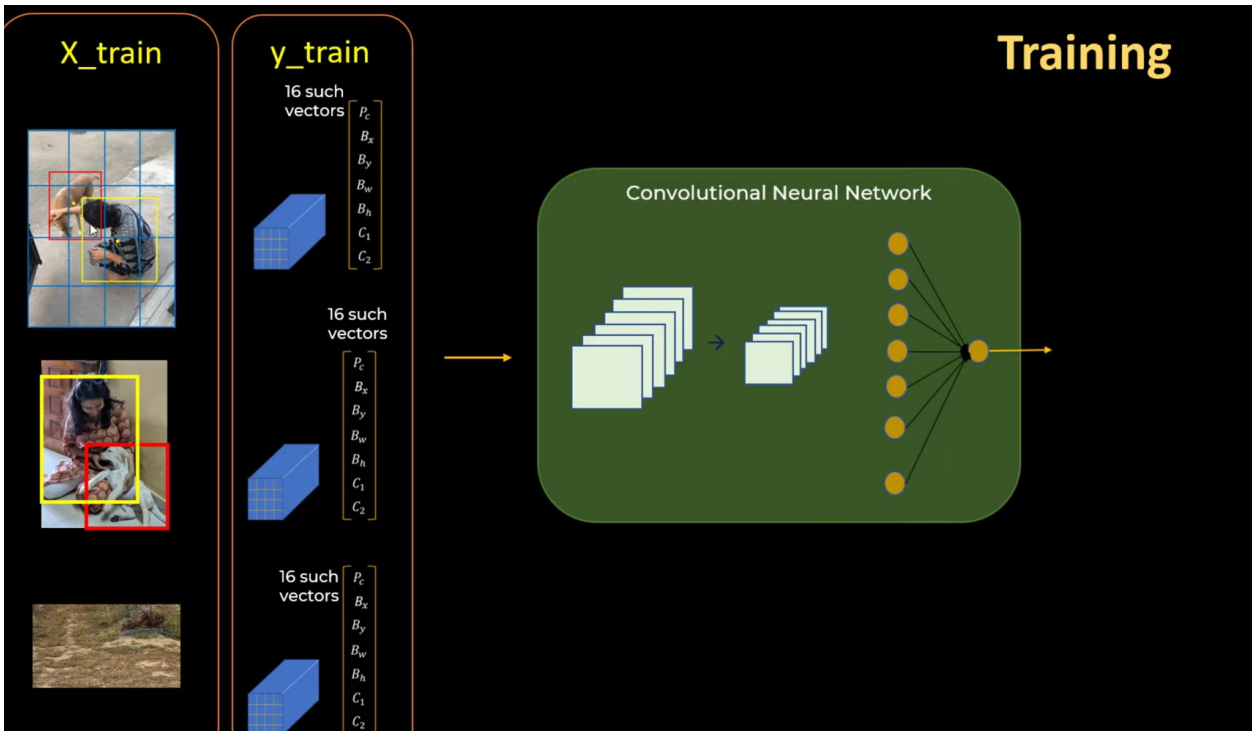
Object Localization

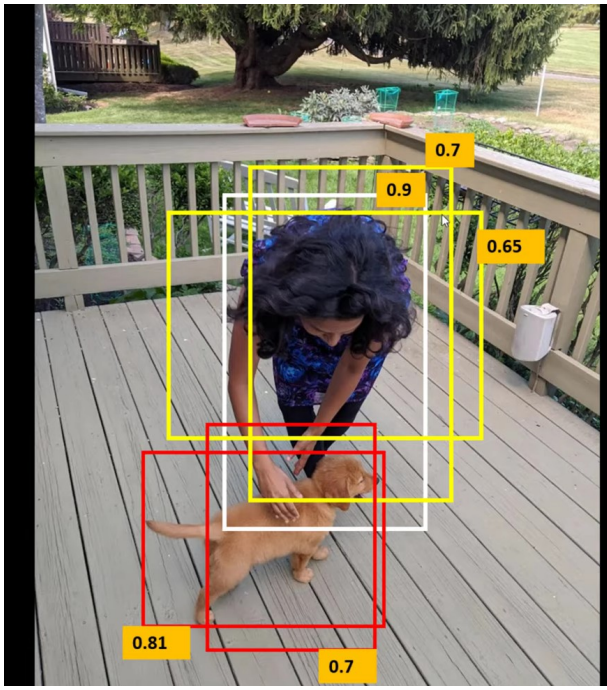
Where exactly is the dog in this image?

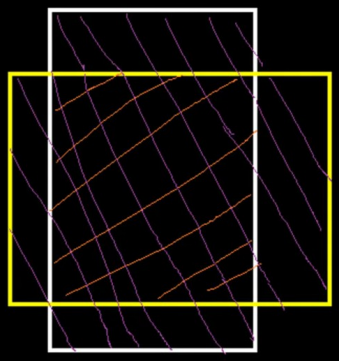
Neural Network Output

Dog = 1
Person = 0
+
Bounding Box



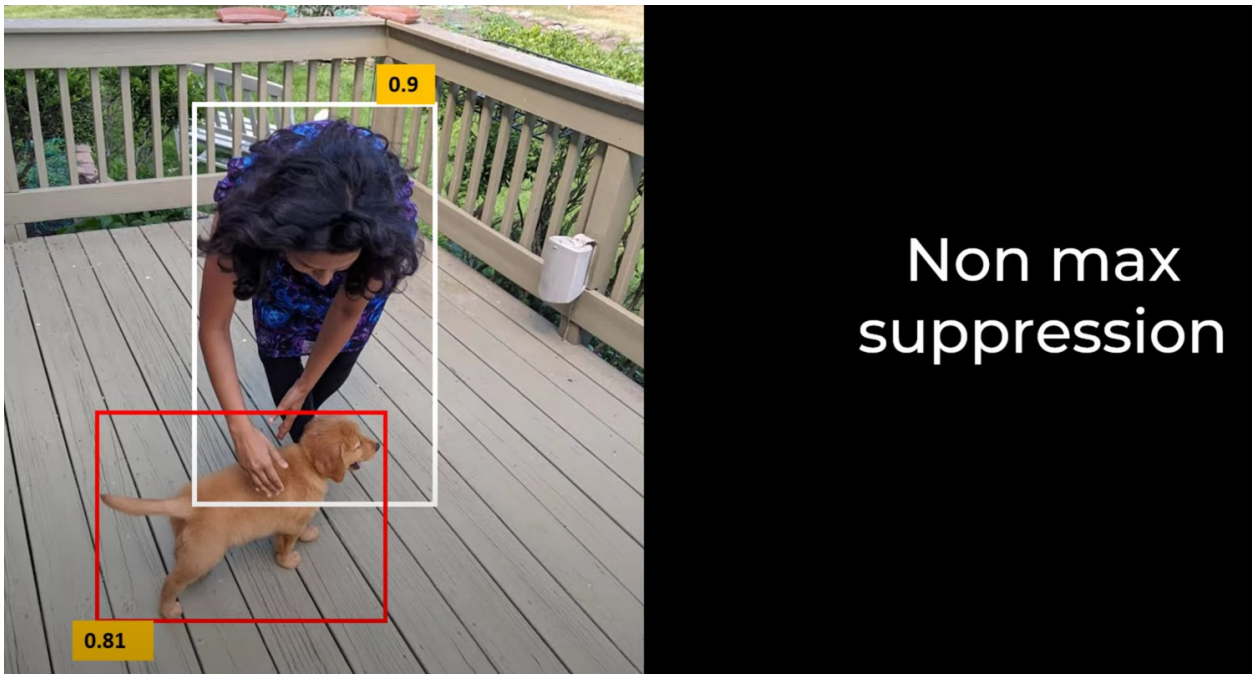


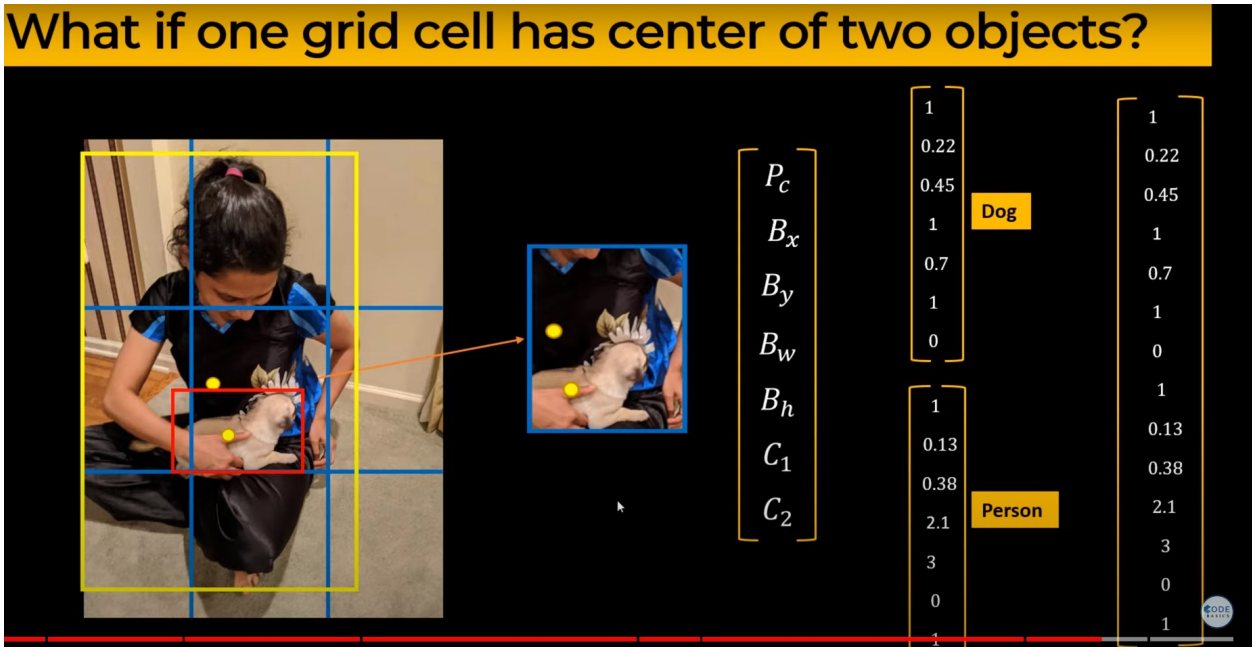


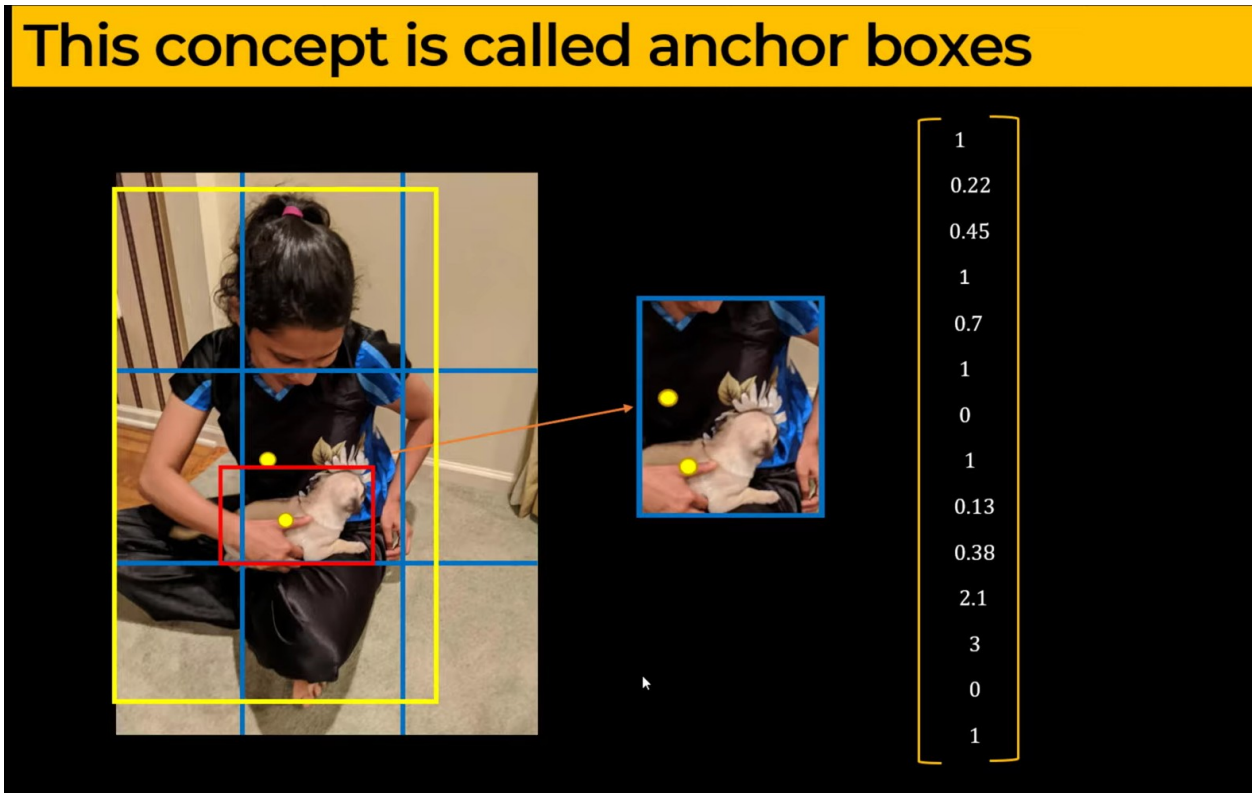


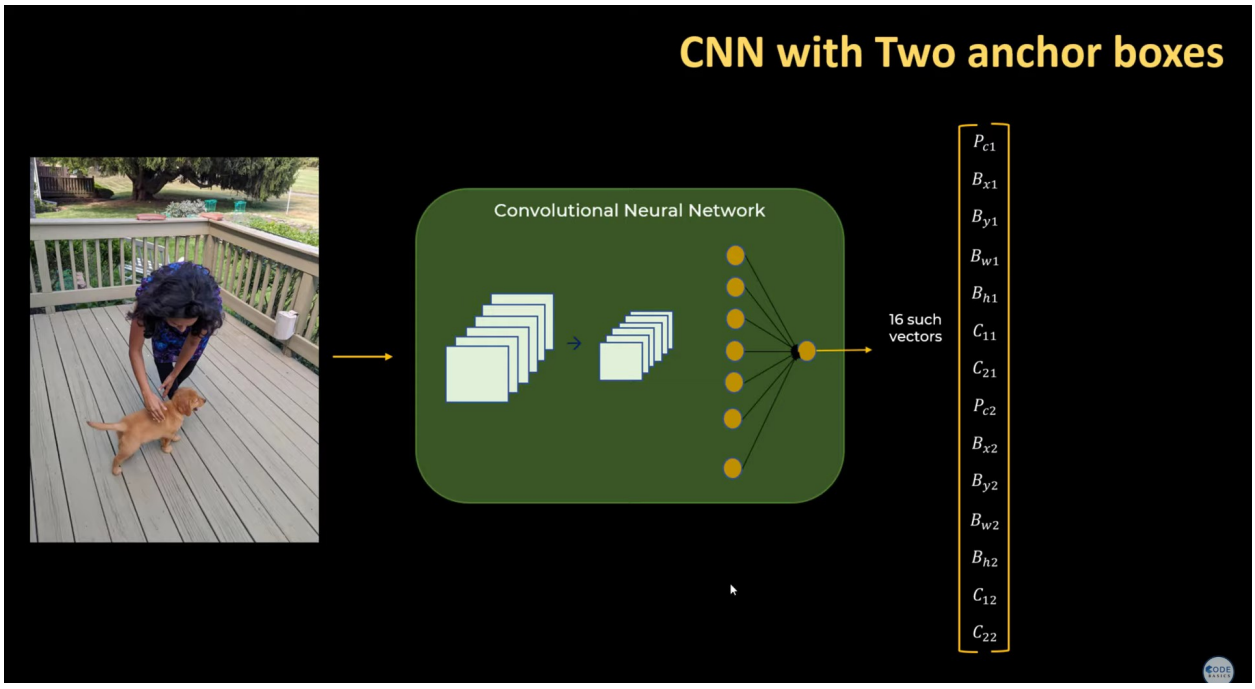
Intersection over union = intersect area / union area

Intersection over union : IOU


0.9
0.81
Non max
suppression


What if one grid cell has center of two objects?
P_c
B_x
B_y
B_w
B_h
C_1
C_2
1
0.22
0.45
1
0.7
1
0
Dog
1
0.13
0.38
2.1
3
0
Person
1
0.22
0.45
1
0.7
1
0
1
0.13
0.38
2.1
3
0
1

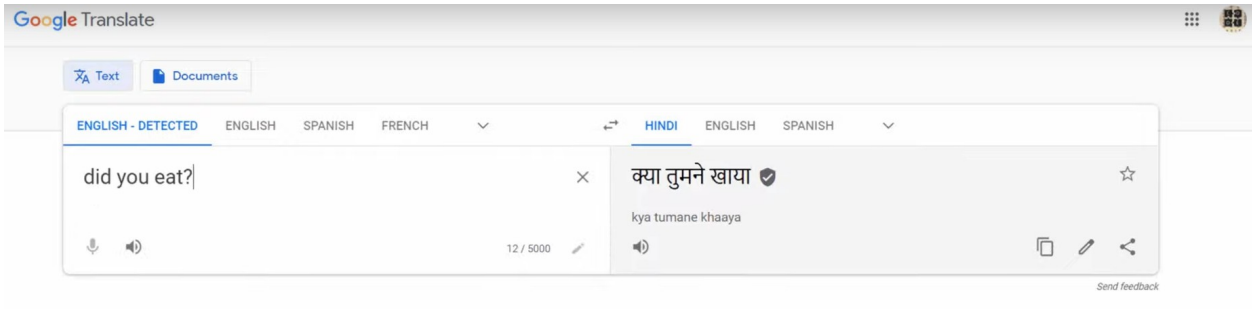


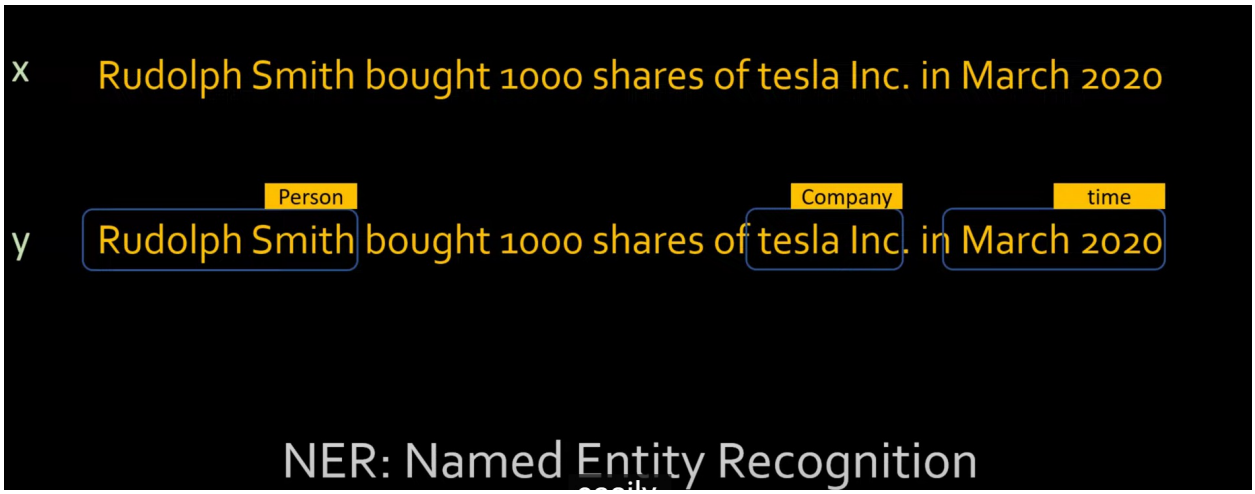


RNN

RNN or Recurrent Neural Network are also known as sequence models that are used mainly in the field of natural language processing as well as some other areas such as speech to text

translation, video activity monitoring, etc. In this video we will understand the intuition behind RNN and see how RNN's work.

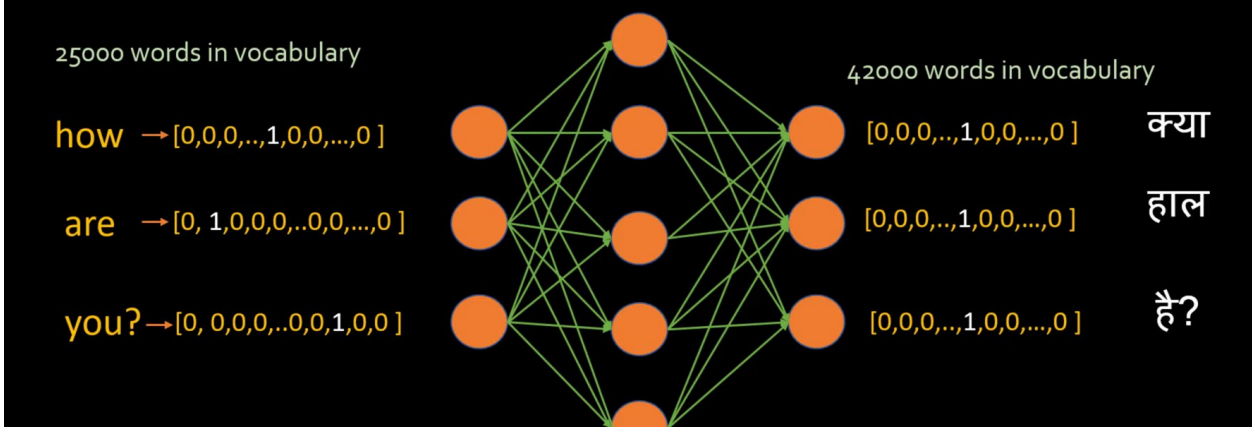




	x		y
auto complete	not interested at	→	this time
translation	how are you?	→	क्या हाल है?
NER	Rudolph Smith bought 1000 shares of tesla Inc. in March 2020	→	Person: Rudolph Smith bought 1000 shares of Company: tesla Inc. in time: March 2020
Sentiment Analysis	Not only the fan was expensive, but it was broken when it arrived.	→	★☆☆☆☆

Issue # 1: No fixed size of neurons in a layer

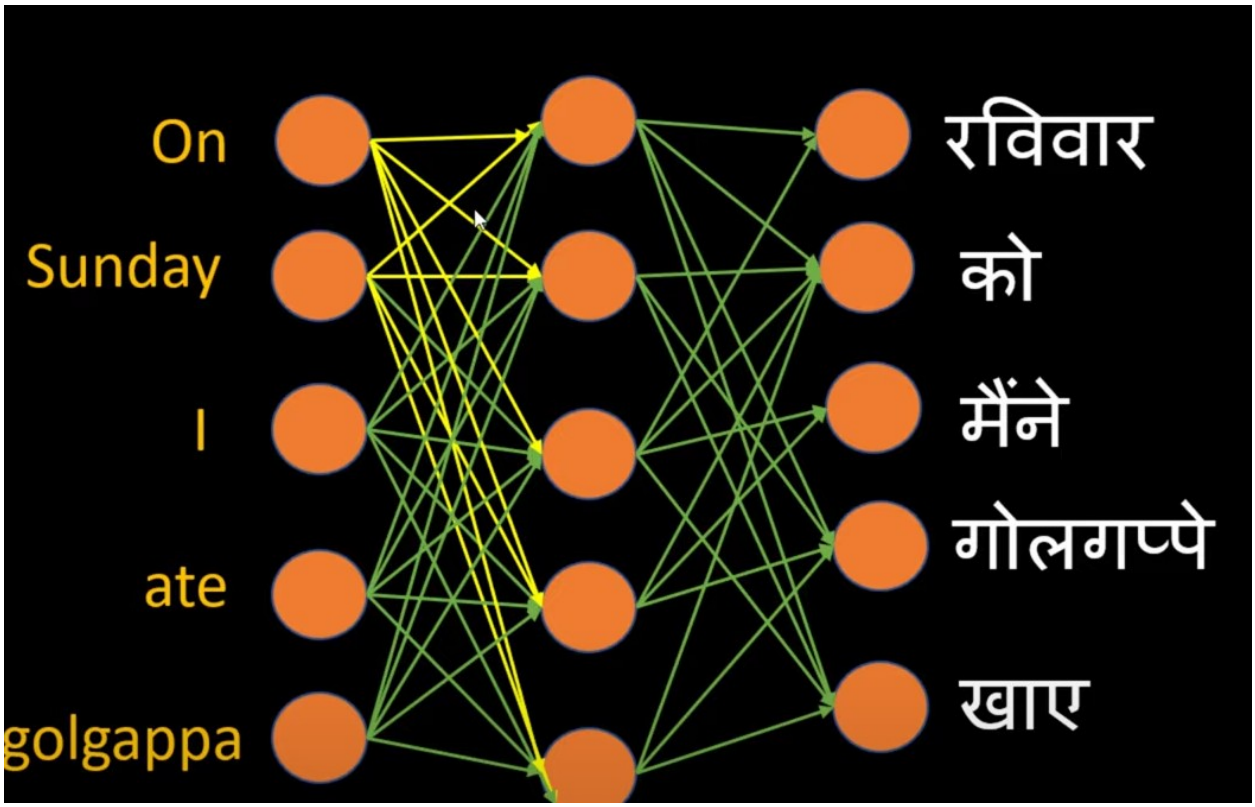
Issue # 2: Too much computation

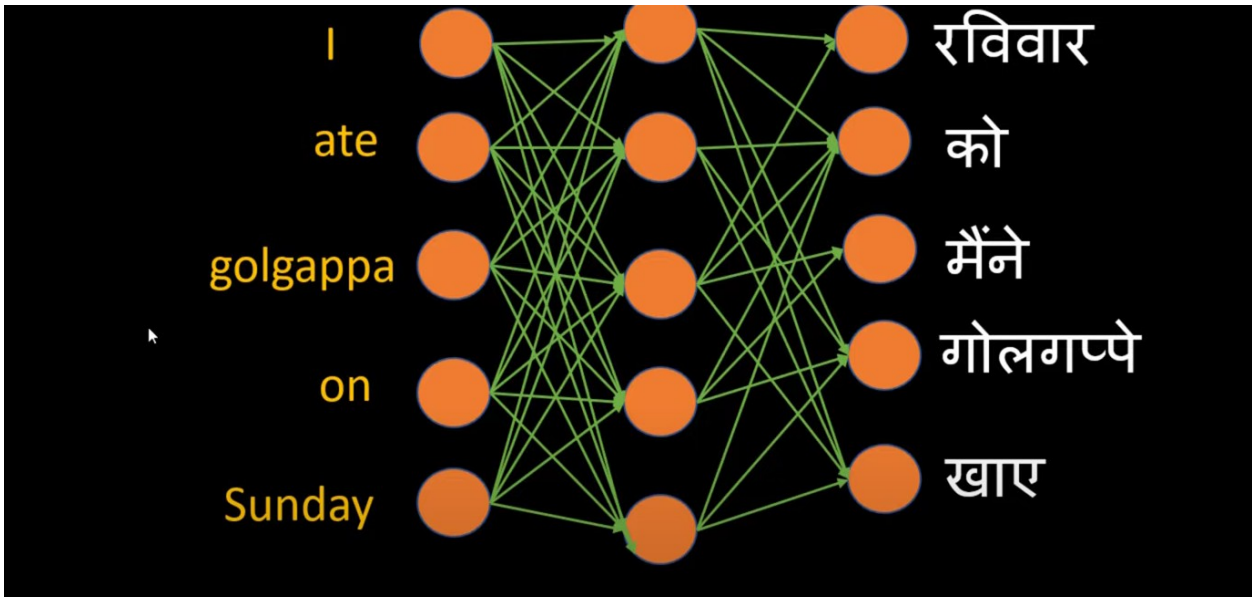


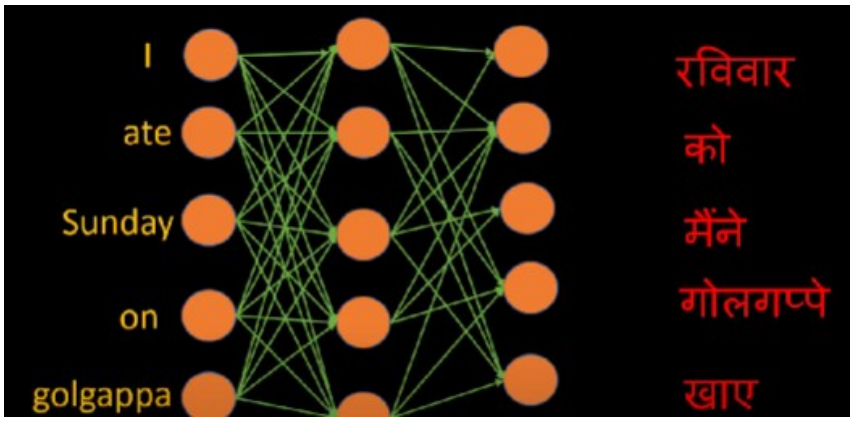
On sunday I ate golgappa

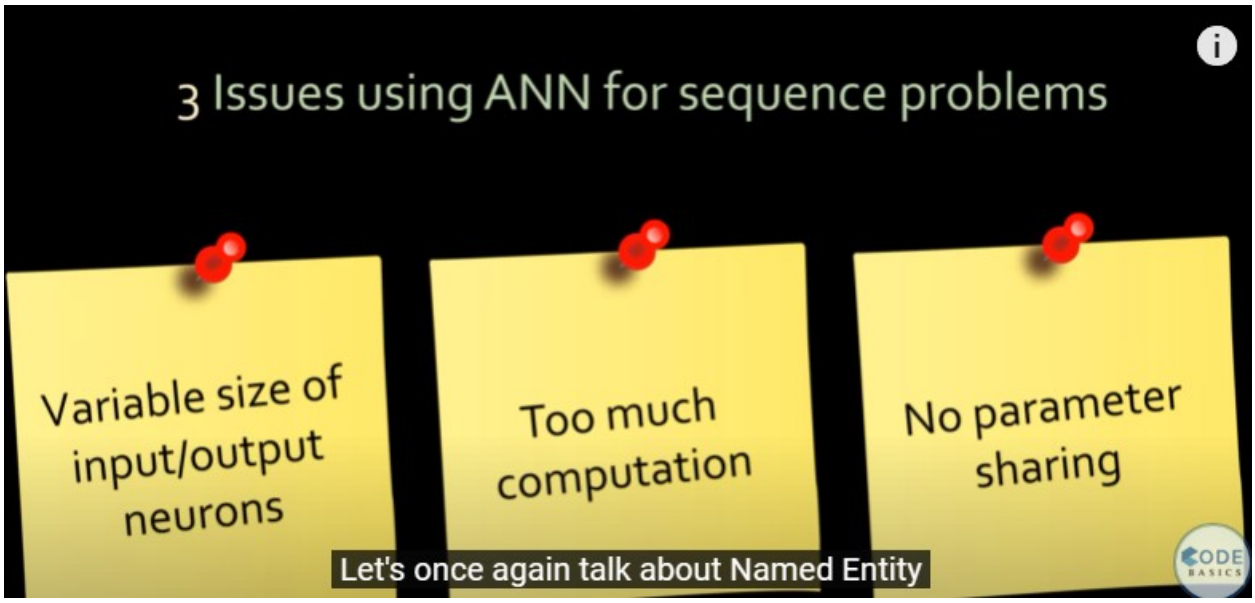
I ate golgappa on Sunday

रविवार को मैंने गोलगप्पे खाए




I
ate
golgappa
on
Sunday
रविवार
को
मैंने
गोलगप्पे
खाए


I
ate
Sunday
on
golgappa
रविवार
को
मैंने
गोलगप्पे
खाए


3 Issues using ANN for sequence problems
Variable size of input/output neurons
Too much computation
No parameter sharing
Let's once again talk about Named Entity

Named Entity Recognition

Dhaval loves baby yoda

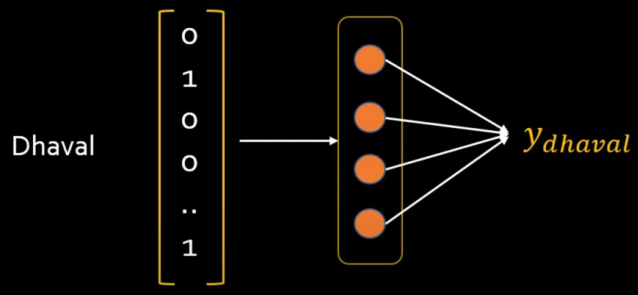
person person

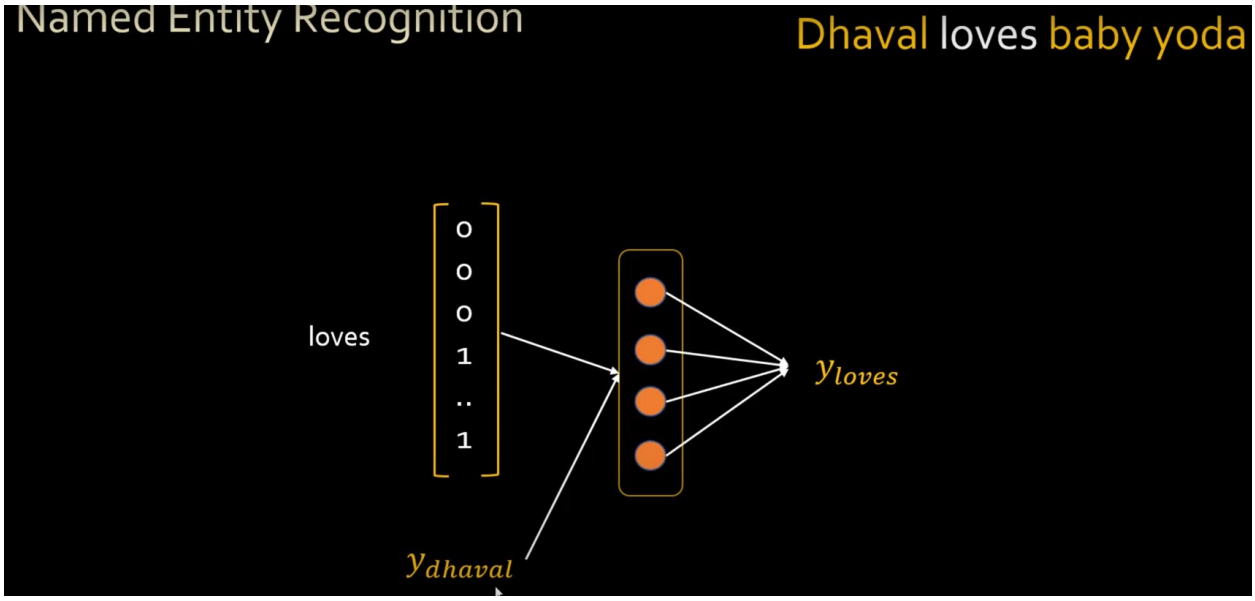
Dhaval loves baby yoda

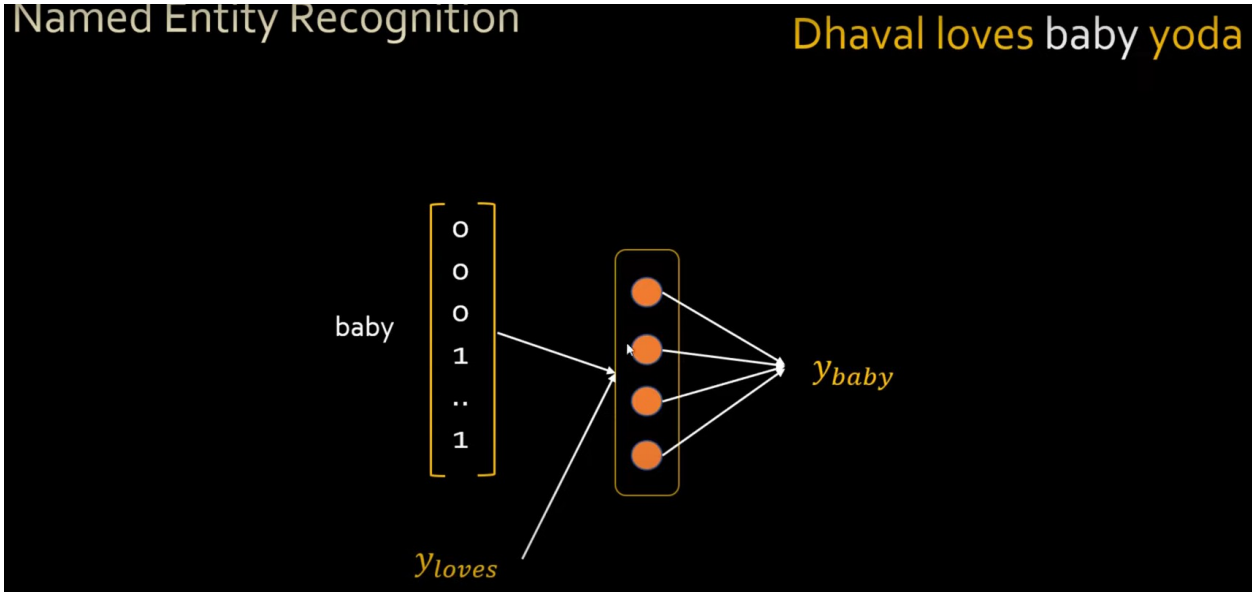
1 0 1 1

Named Entity Recognition

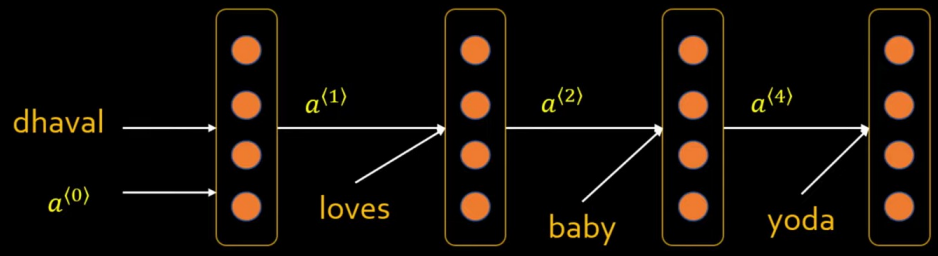
Dhaval loves baby yoda



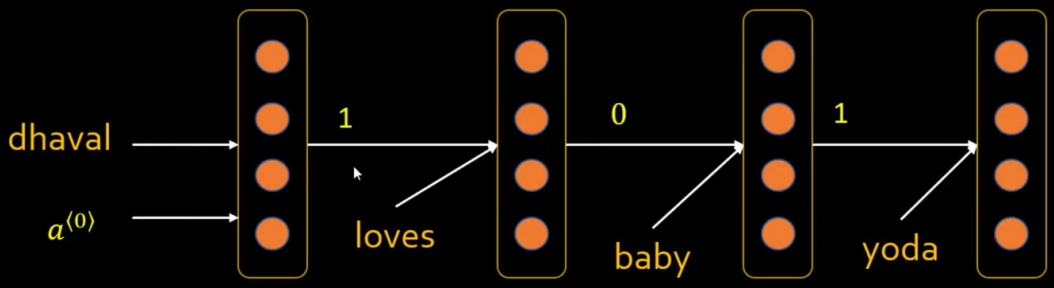

Named Entity Recognition
Dhaval loves baby yoda
loves
0
0
0
1
..
1
y_{loves}
y_{dhaval}


Named Entity Recognition
Dhaval loves baby yoda
baby
0
0
0
1
..
1
y_{baby}
y_{loves}

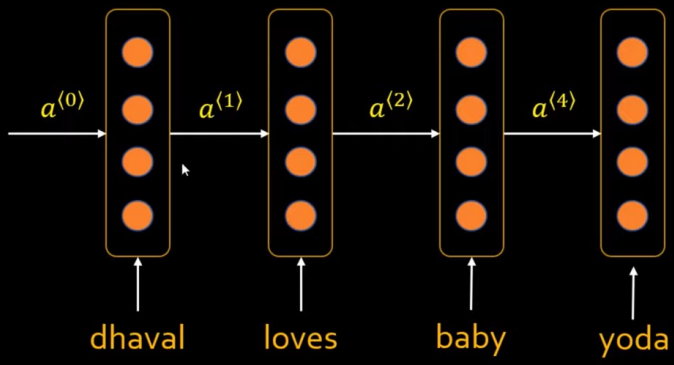
Named Entity Recognition



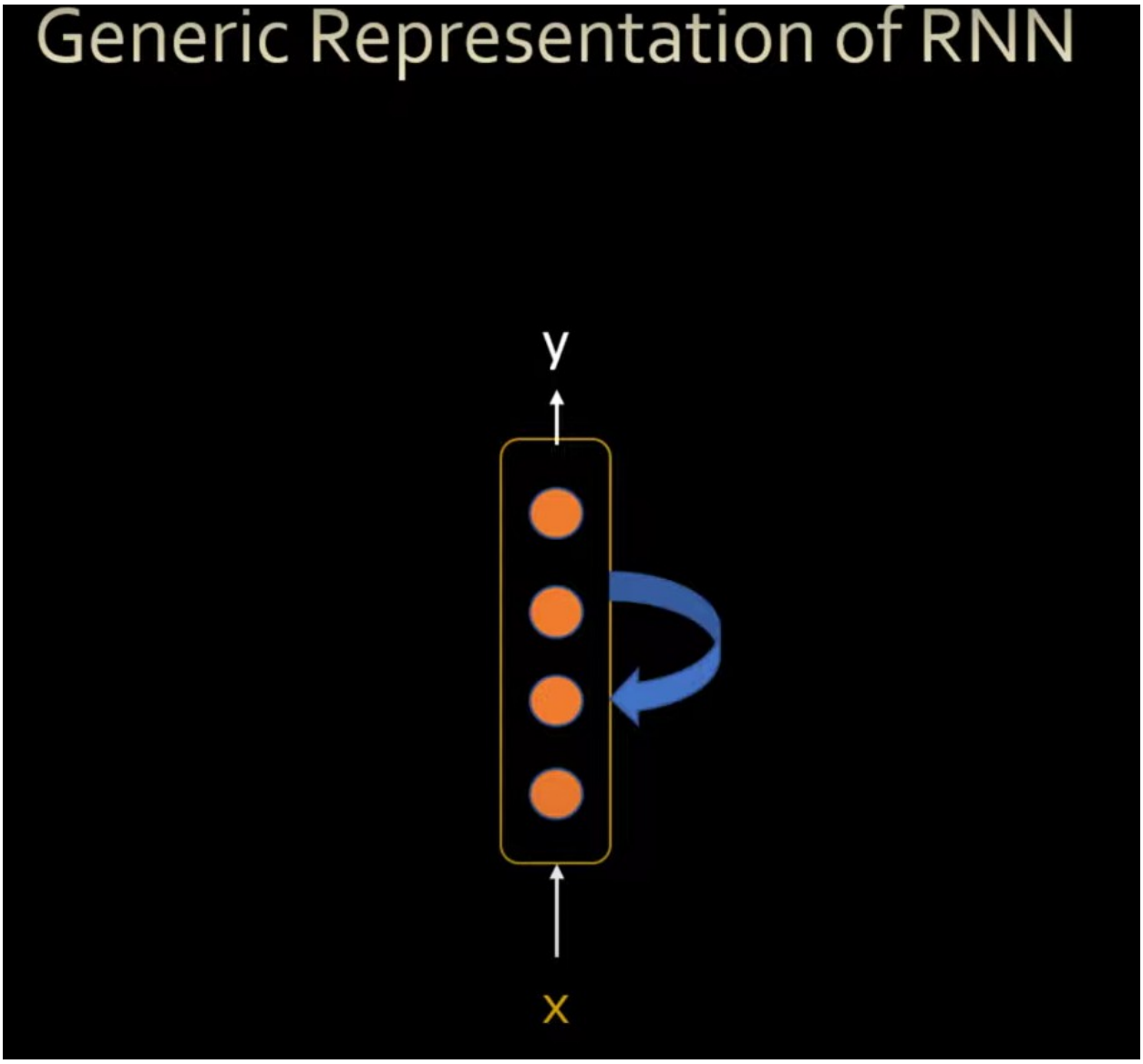
Named Entity Recognition: once network is trained



Named Entity Recognition



Generic Representation of RNN

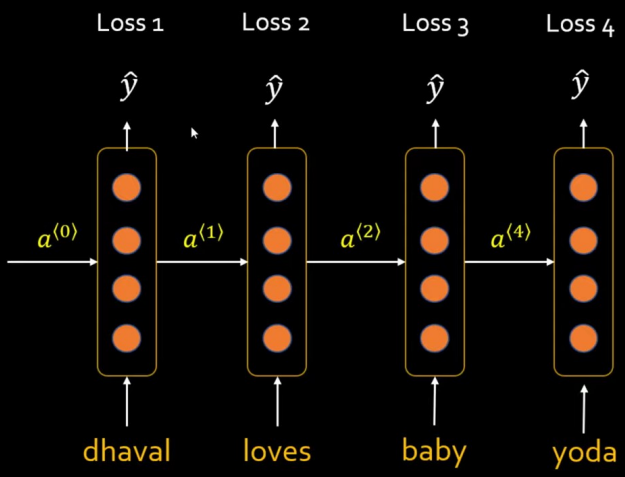


Training : Named Entity Recognition (NER)

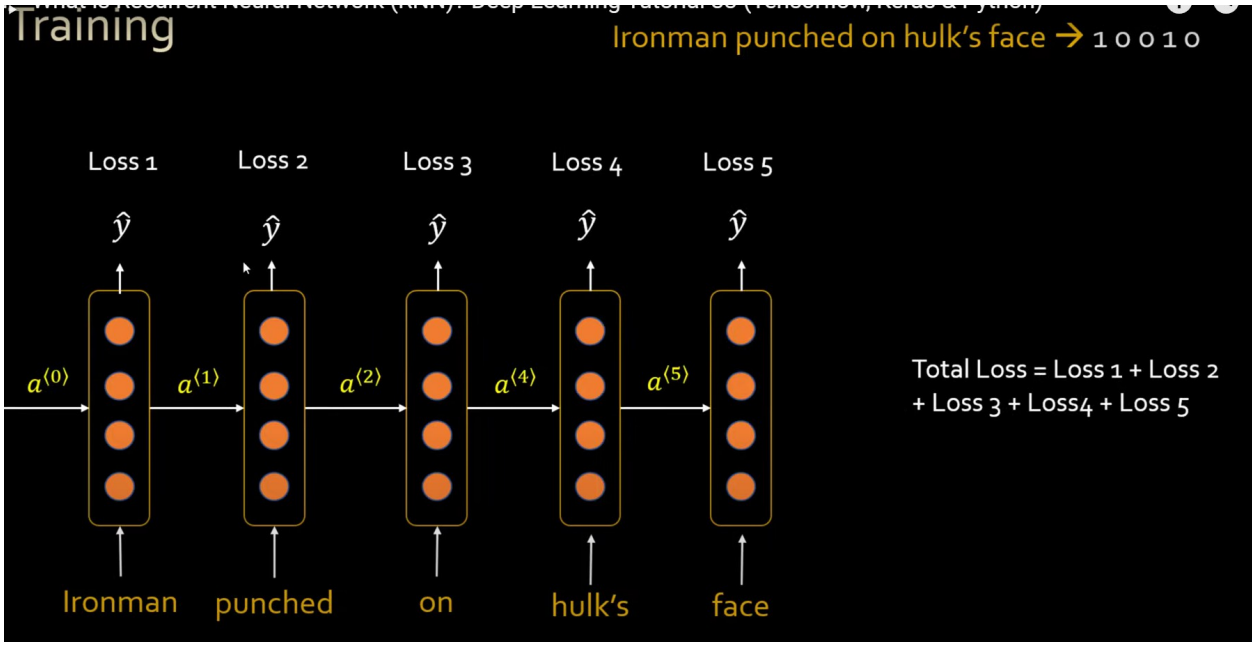
X	y
Dhaval loves baby yoda	1 0 1 1
Bob told Ahmed that pizza is delivered	1 0 1 0 0 0 0
Ironman punched on hulk's face	1 0 0 1 1

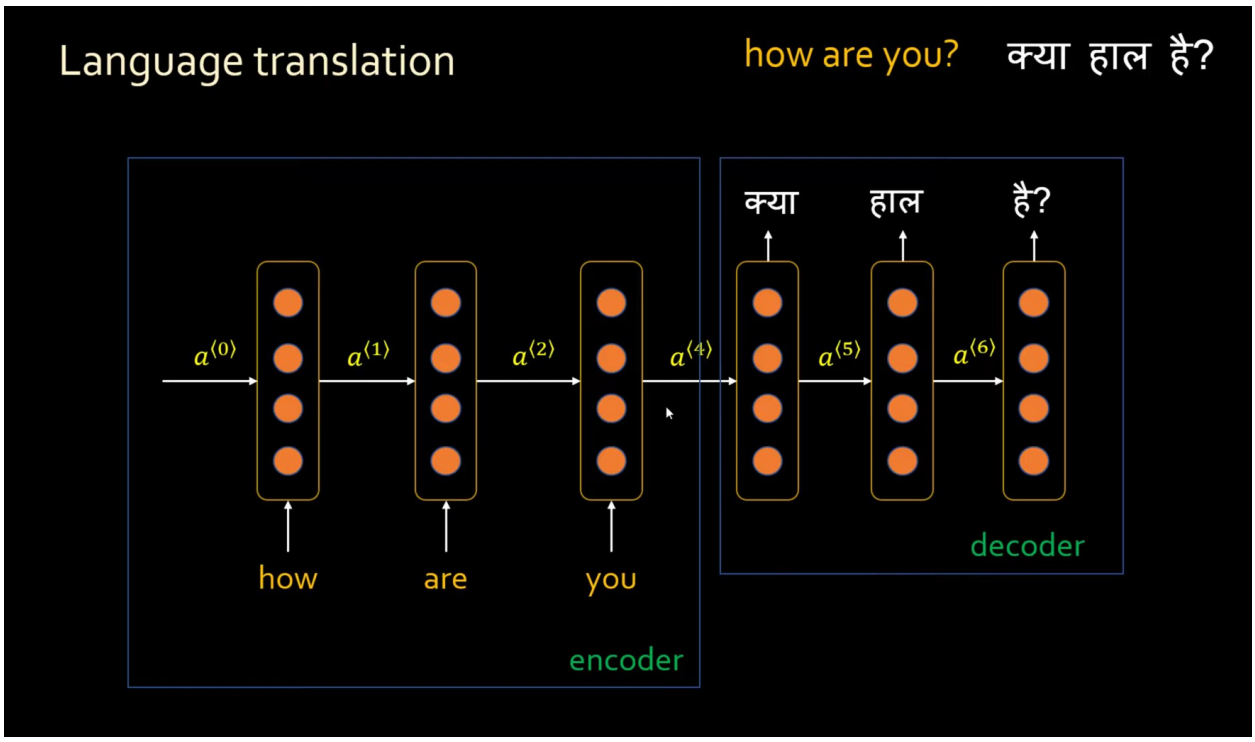
Training

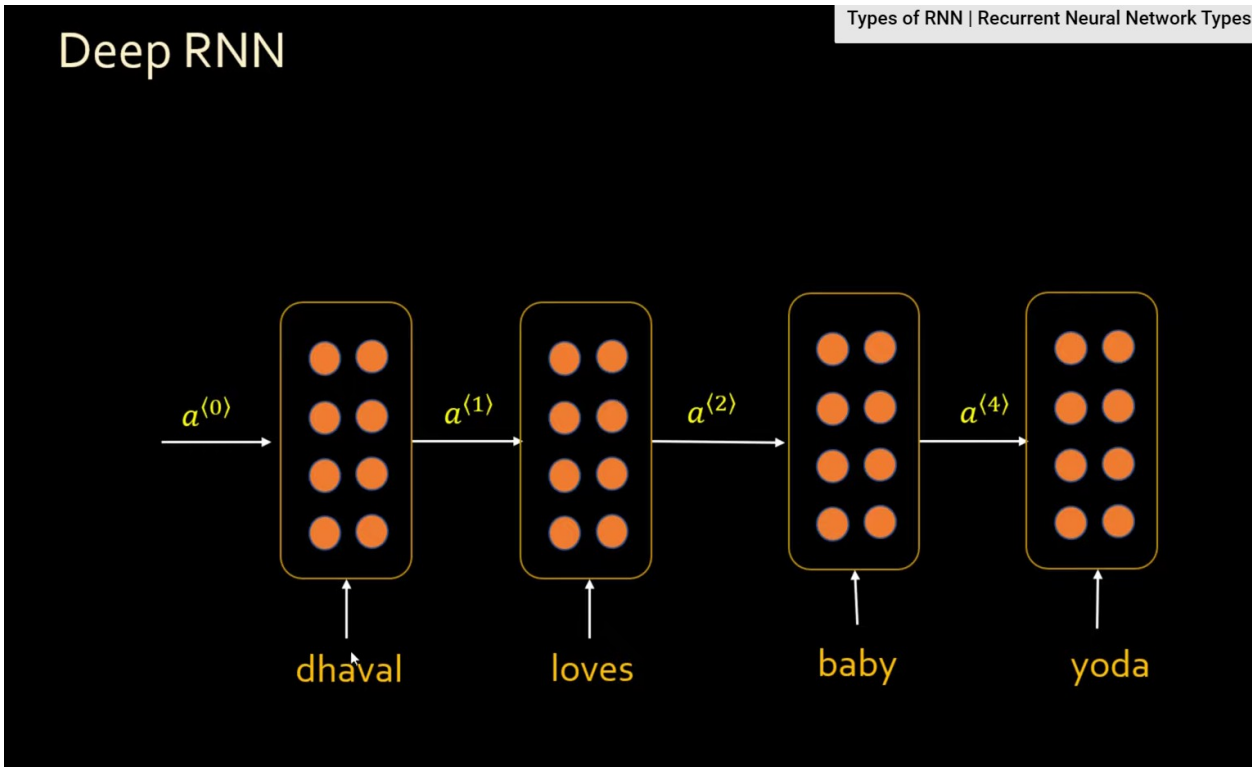
Dhaval loves baby yoda → 1 0 1 1



Total Loss = Loss 1 + Loss 2 + Loss 3 + Loss4

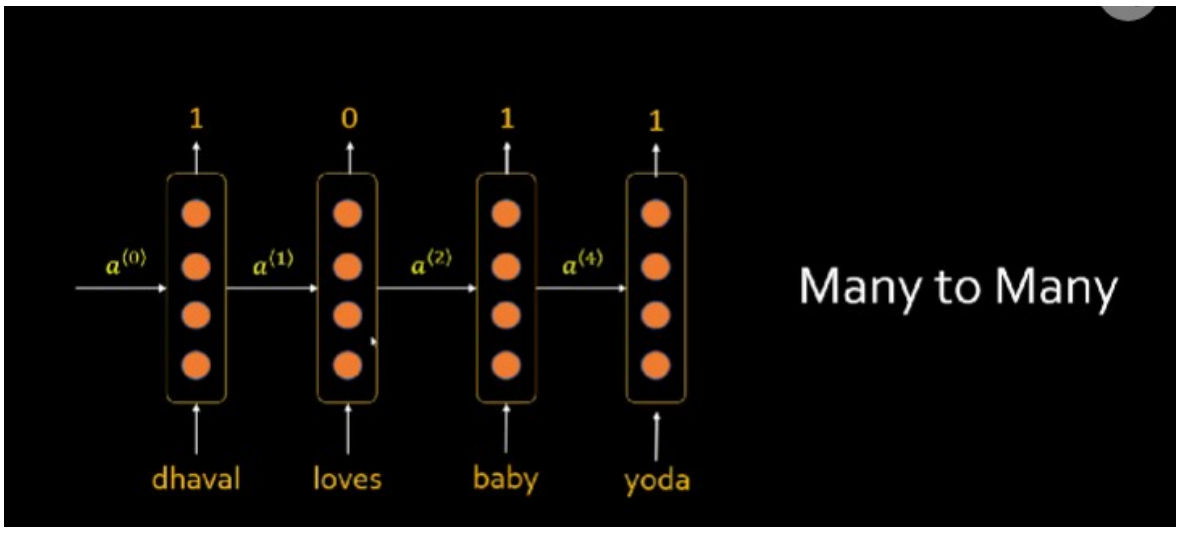

Training
Ironman punched on hulk's face → 1 0 0 1 0
Loss 1
Loss 2
Loss 3
Loss 4
Loss 5
$\hat{y}$
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 4 \rangle}$
$a^{\langle 5 \rangle}$
Ironman
punched
on
hulk's
face
Total Loss = Loss 1 + Loss 2 + Loss 3 + Loss4 + Loss 5

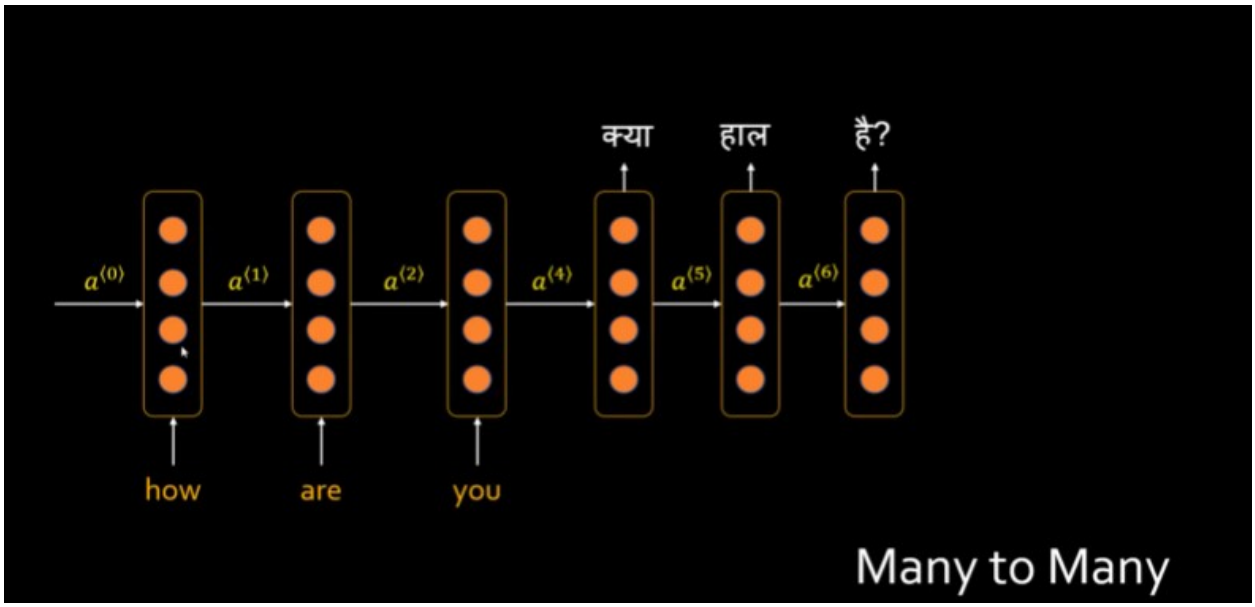

Language translation
how are you?
क्या हाल है?
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 4 \rangle}$
$a^{\langle 5 \rangle}$
$a^{\langle 6 \rangle}$
how
are
you
encoder
क्या
हाल
है?
decoder

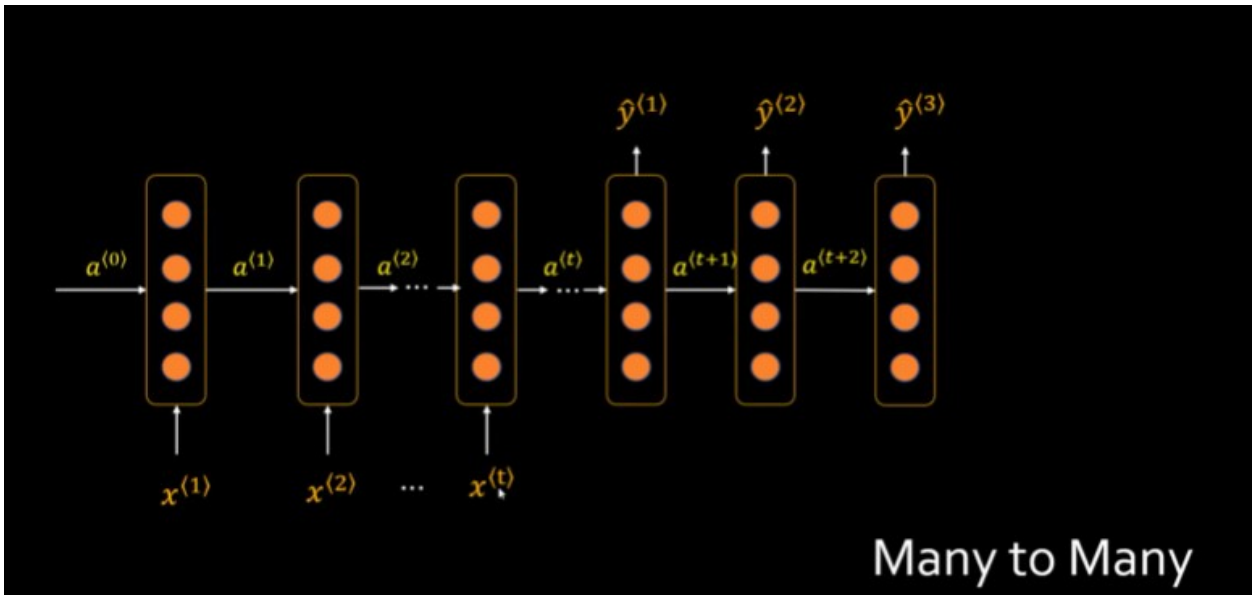


Deep RNN has multiple hidden layers.

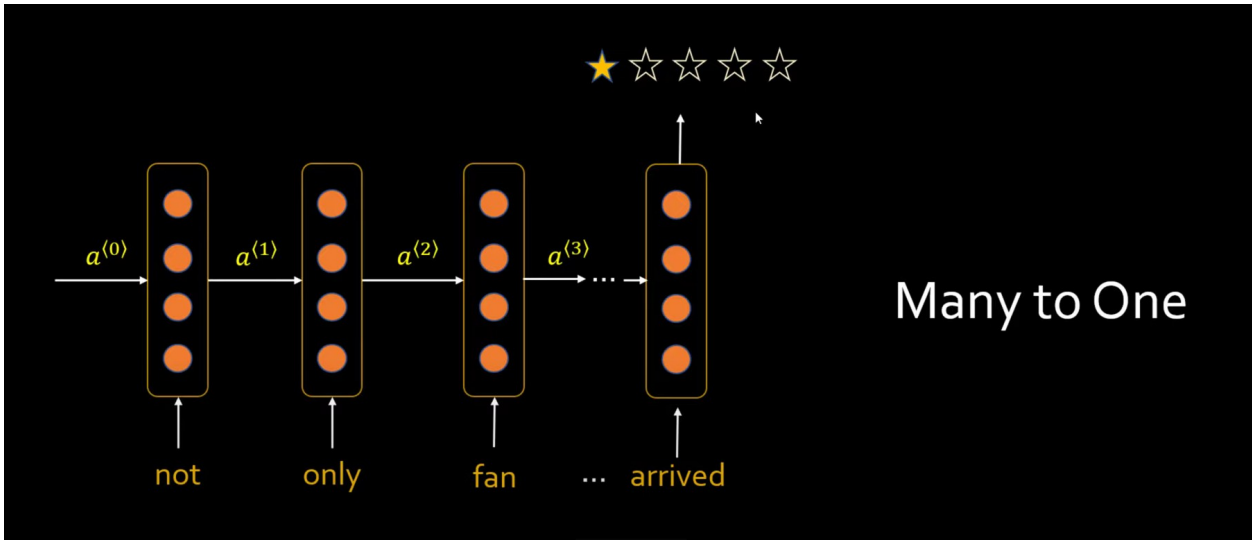
In this video we will discuss different types of RNN types such as, 1) One to many 2) Many to many 3) Many to one

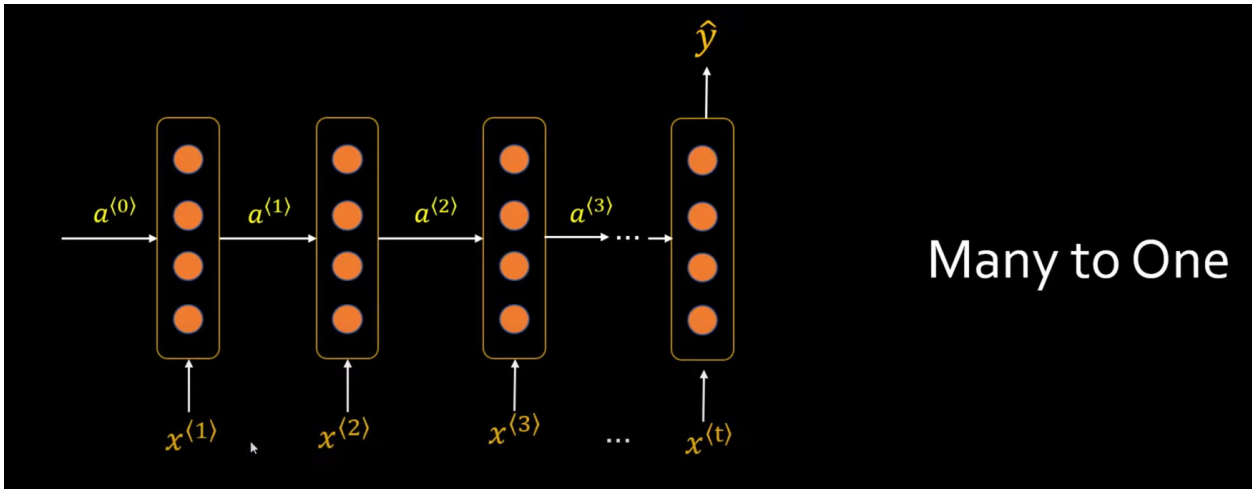


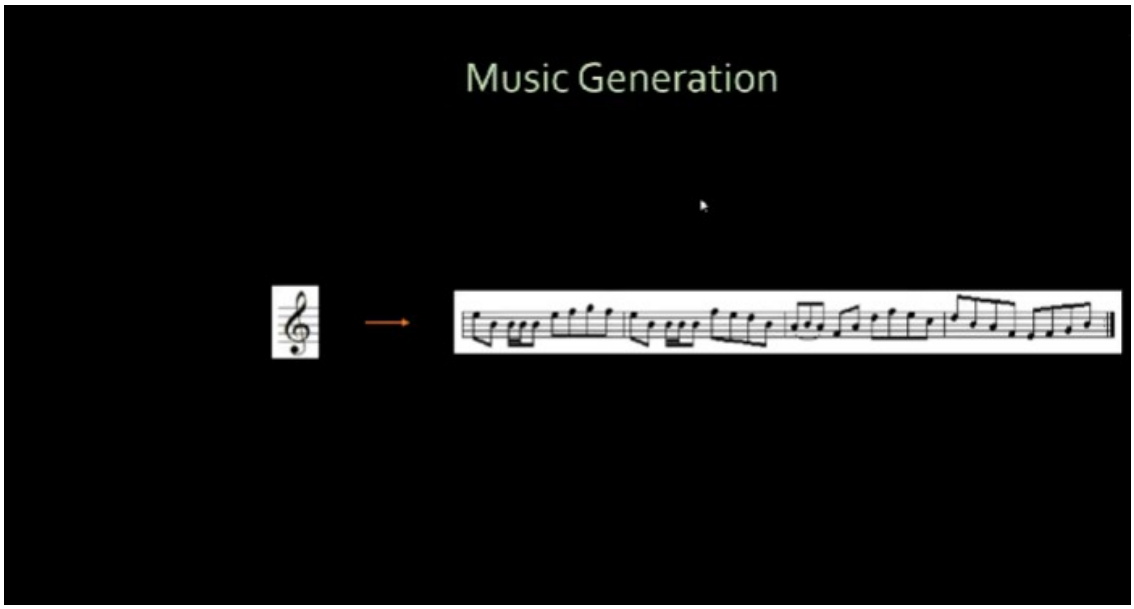

क्या
हाल
है?
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 4 \rangle}$
$a^{\langle 5 \rangle}$
$a^{\langle 6 \rangle}$
how
are
you
Many to Many

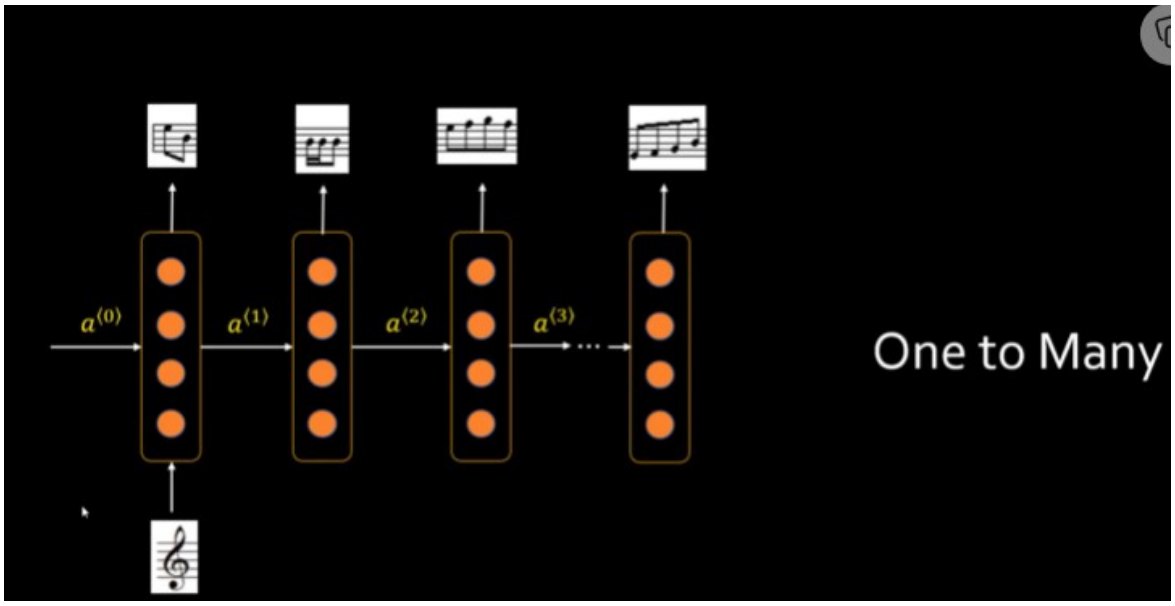

$\hat{y}^{\langle 1 \rangle}$
$\hat{y}^{\langle 2 \rangle}$
$\hat{y}^{\langle 3 \rangle}$
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle t \rangle}$
$a^{\langle t+1 \rangle}$
$a^{\langle t+2 \rangle}$
$x^{\langle 1 \rangle}$
$x^{\langle 2 \rangle}$
$x^{\langle t \rangle}$
Many to Many

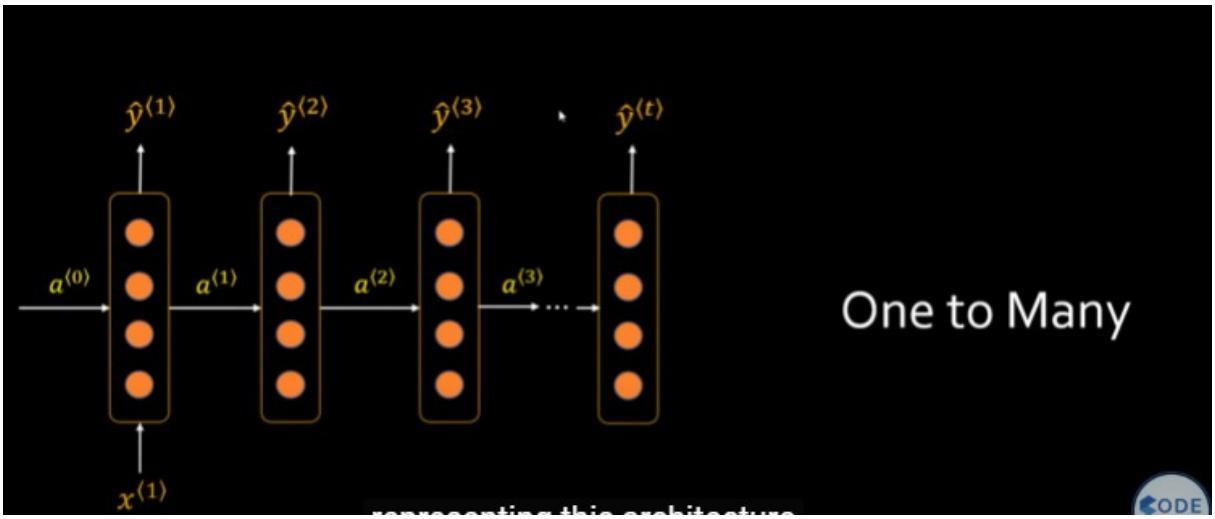

Sentiment Analysis
Not only the fan was expensive, but it was broken when it arrived.
The fan works like a charm, I wasn't expecting such a good quality at this cheap price


$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 3 \rangle}$
not
only
fan
... arrived
Many to One

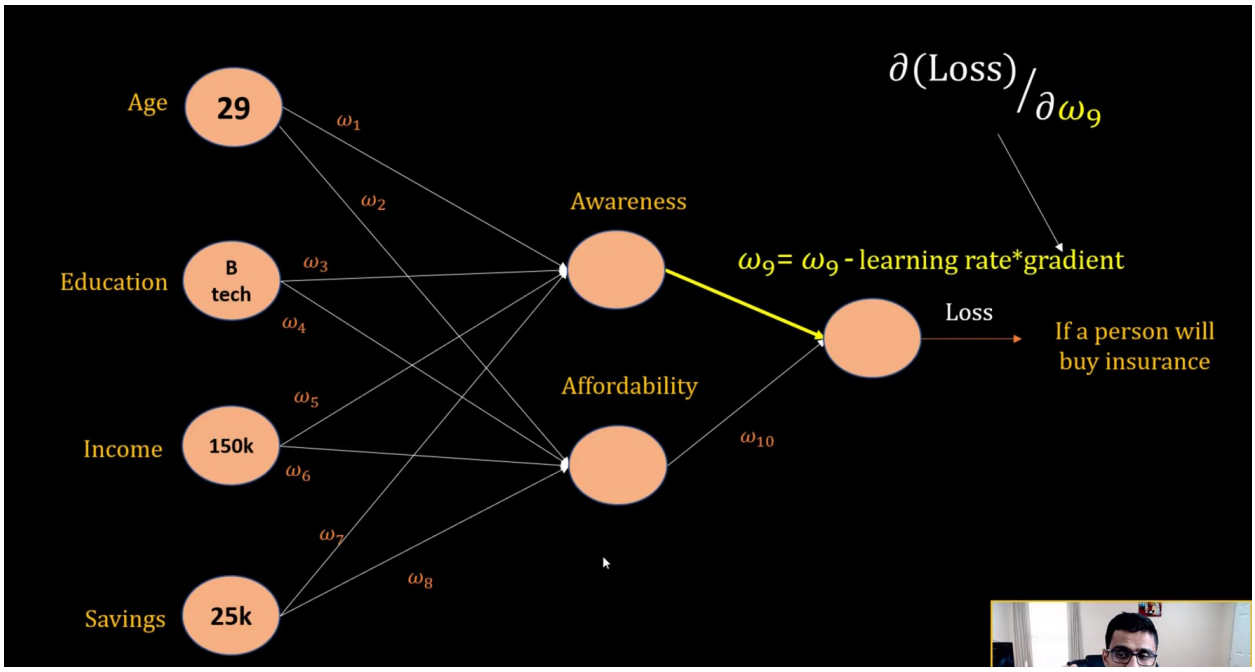

$\hat{y}$
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 3 \rangle}$
$x^{\langle 1 \rangle}$
$x^{\langle 2 \rangle}$
$x^{\langle 3 \rangle}$
$x^{\langle t \rangle}$
Many to One

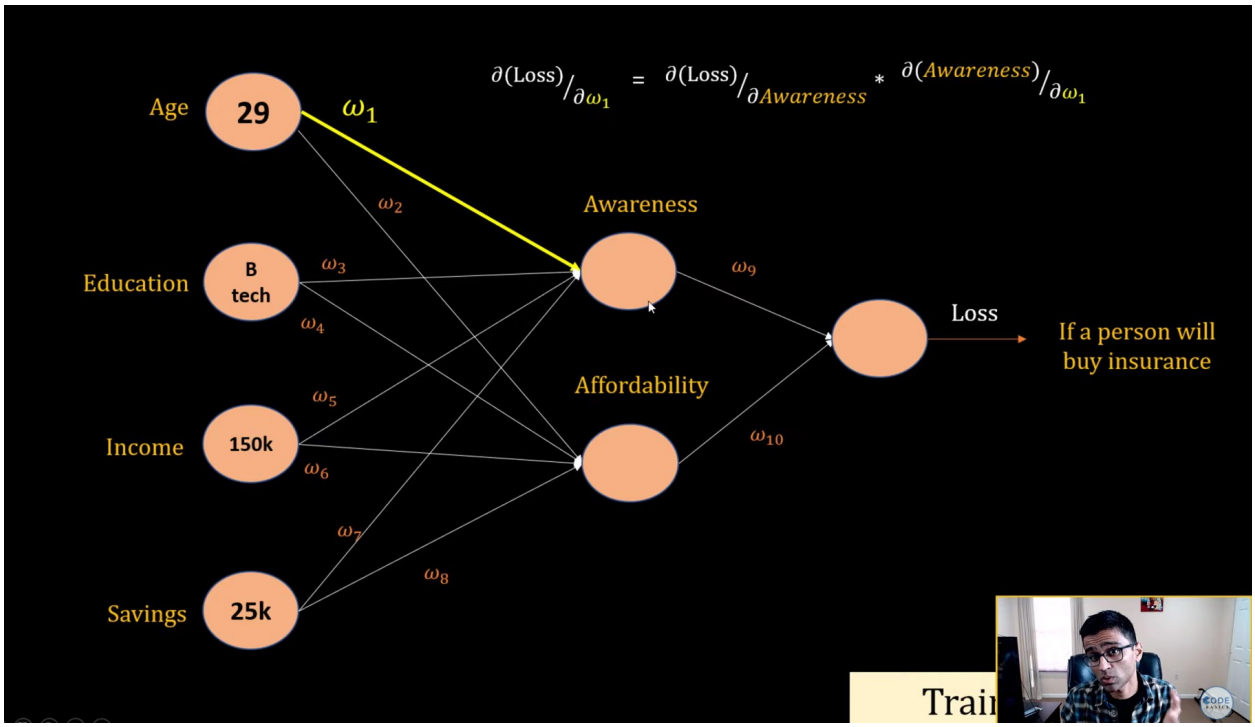

Music Generation


$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 3 \rangle}$
One to Many


$\hat{y}^{\langle 1 \rangle}$
$\hat{y}^{\langle 2 \rangle}$
$\hat{y}^{\langle 3 \rangle}$
$\hat{y}^{\langle t \rangle}$
$a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
$a^{\langle 3 \rangle}$
$x^{\langle 1 \rangle}$
One to Many

Vanishing gradient is a commong problem encountered while training a deep neural network with many layers. In case of RNN this problem is prominent as unrolling a network layer in time makes it more like a deep neural network with many layers. In this video we will discuss what vanishing and exploding gradients are in artificial neural network (ANN) and in recurrent neural network (RNN)





As number of hidden layers grow, gradient becomes very small and weights will hardly change . This will hamper the learning process.

Vanishing Gradients

When individual derivatives are large, the final derivate will also become huge and weights would change drastically.

Exploding Gradients

$$gradient = d1 * d2 * d3 * d4 * ... * dn$$

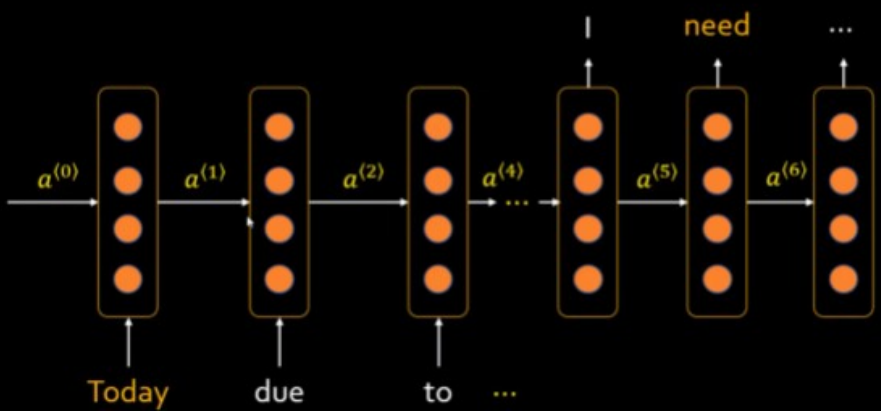
Vanishing gradient problem is more prominent in very deep neural networks.

Vanishing gradient problem in **RNN**

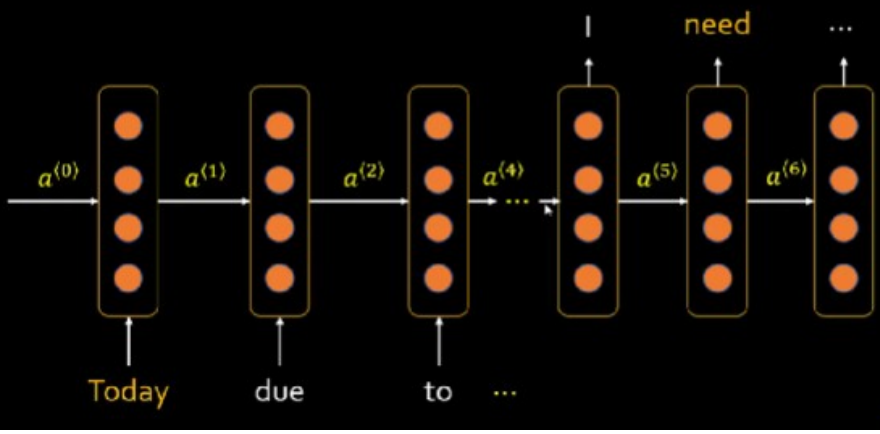
Today, due to my current job situation and family conditions, I need to take a loan.

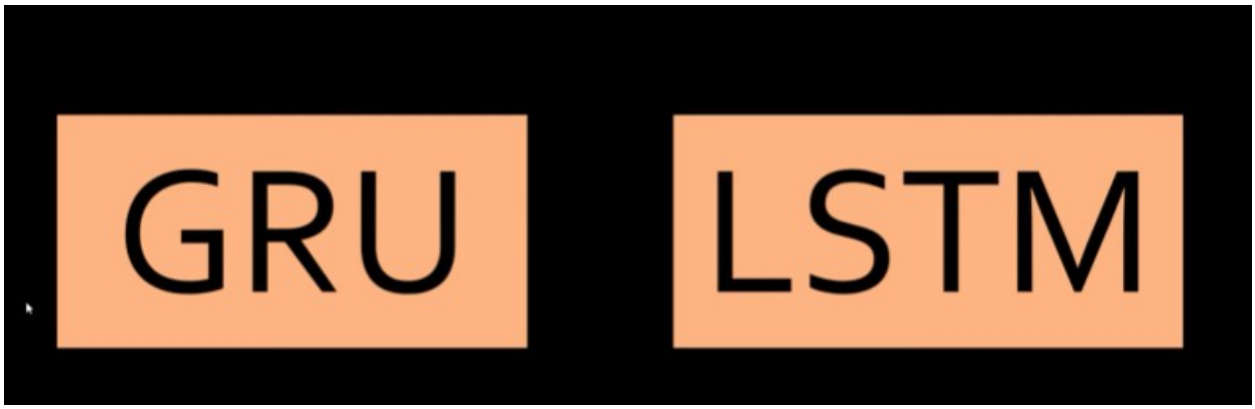
Last year, due to my current job situation and family conditions, I had to take a loan.

Today, due to my current job situation and family conditions, I need to take a loan.

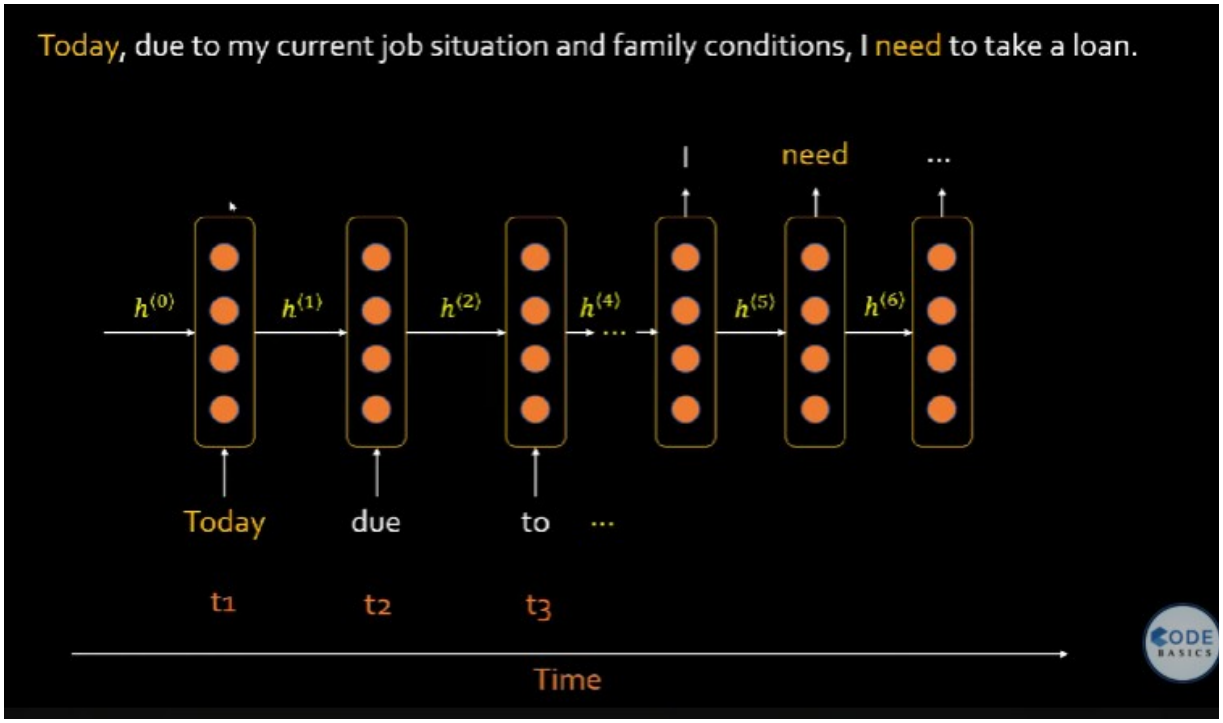


Today, due to my current job situation and family conditions, I need to take a loan.





To adress the problem of vanishing effect ,we use GRU and LSTM



Due to short length memory of RNN,using word "Today/Last Year" at beginning would not be able to predict "need/had" .

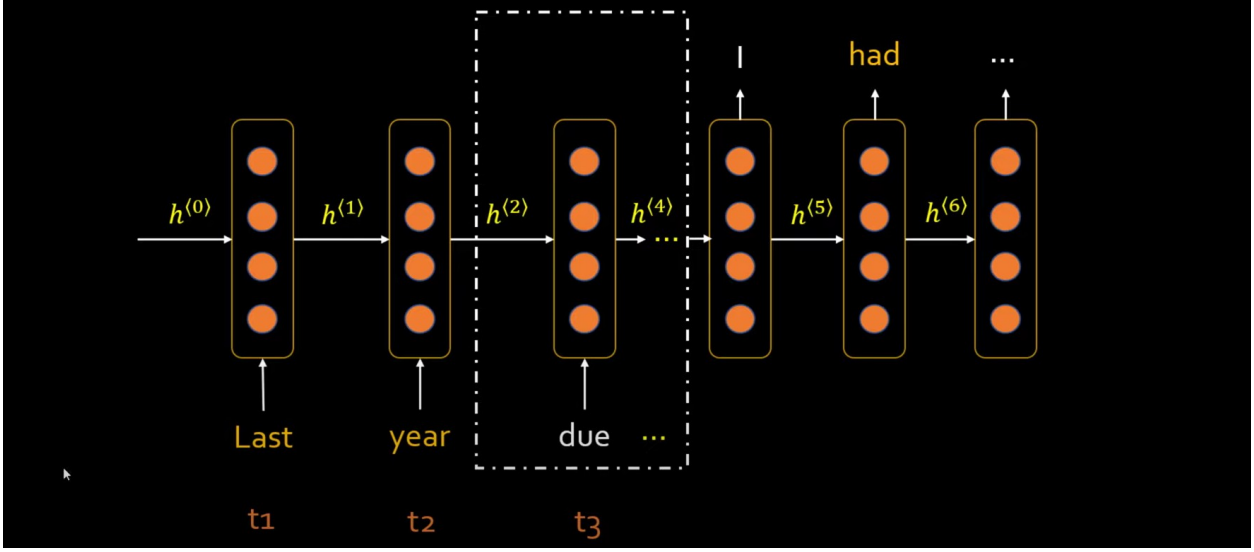
LSTM

LSTM or long short term memory is a special type of RNN that solves traditional RNN's short term memory problem. In this video I will give a very simple explanation of LSTM using some real life examples so that you can understand this difficult topic easily. Also refer to following blogs to explore math and understand few more details.

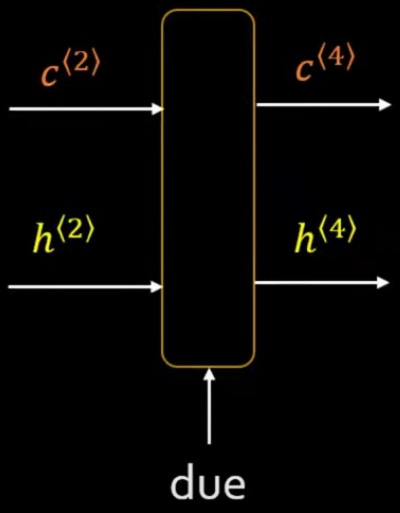
http://colah.github.io/posts/2015-08-...

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Last year, due to my current job situation and family conditions, I had to take a loan.

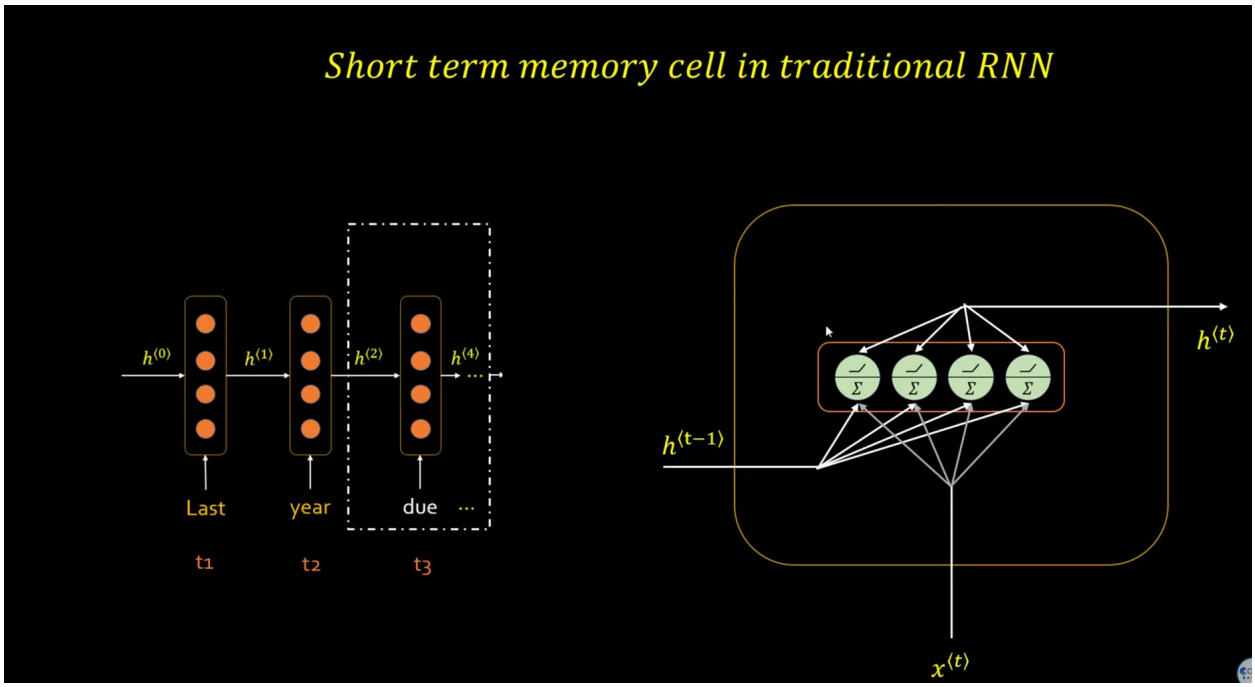


Memory cell

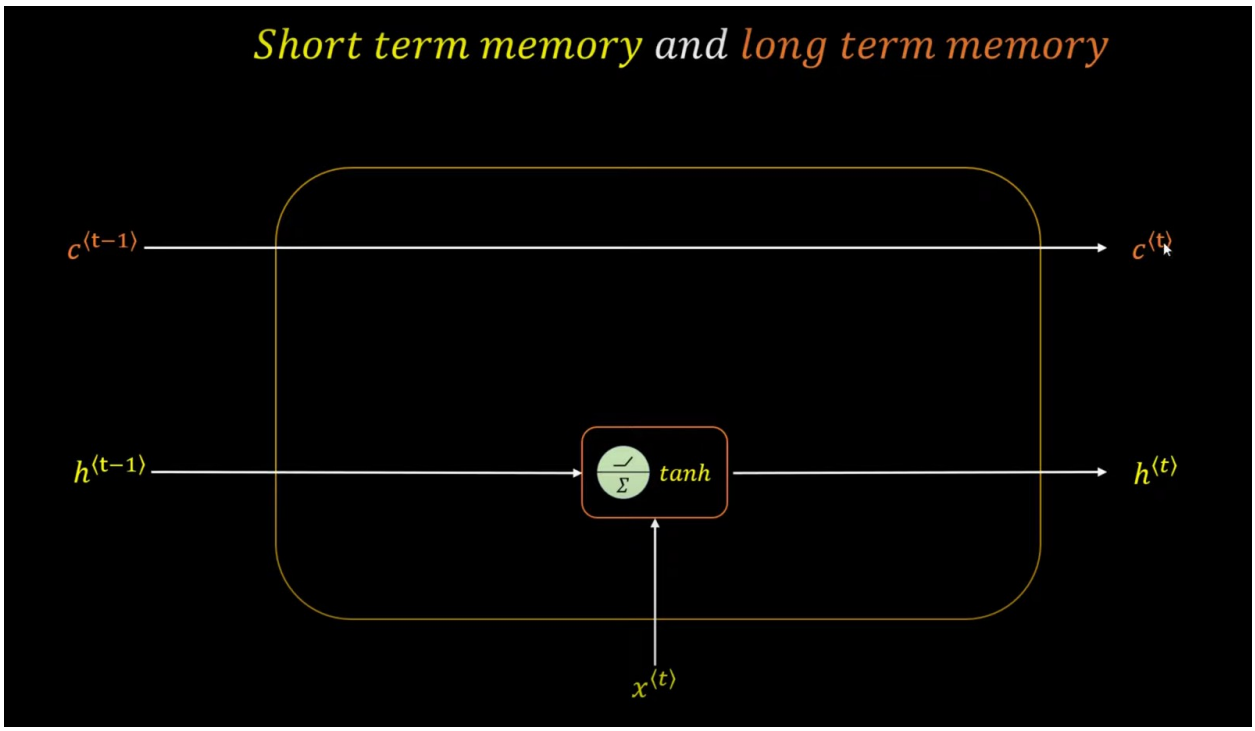


Long term memory

Short term memory

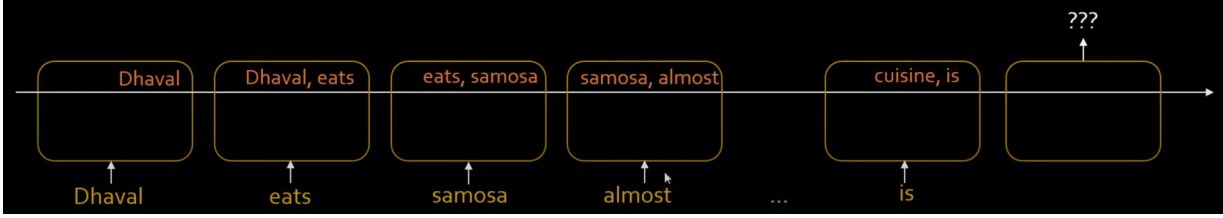

Short term memory cell in traditional RNN
$h^{\langle 0 \rangle}$
$h^{\langle 1 \rangle}$
$h^{\langle 2 \rangle}$
$h^{\langle 4 \rangle}$
Last
year
due
t1
t2
t3
$h^{\langle t-1 \rangle}$
$h^{\langle t \rangle}$
$x^{\langle t \rangle}$

#

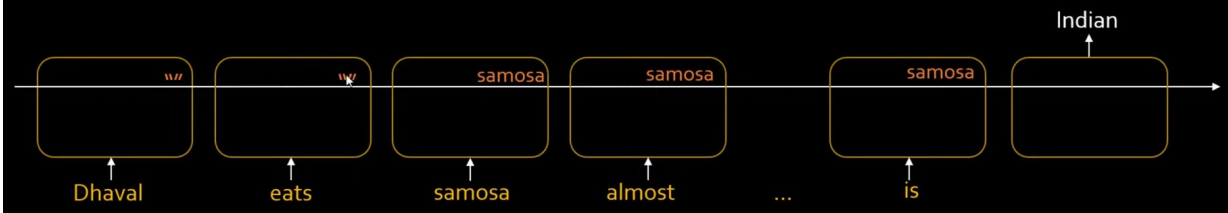

Short term memory and long term memory
$c^{\langle t-1 \rangle}$
$c^{\langle t \rangle}$
$h^{\langle t-1 \rangle}$
tanh
$h^{\langle t \rangle}$
$x^{\langle t \rangle}$

Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian

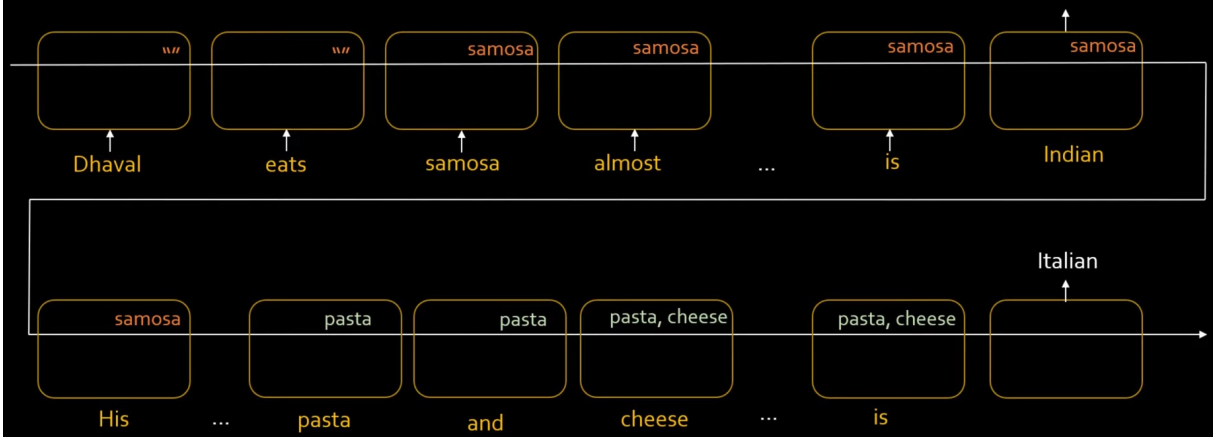
Traditional RNN

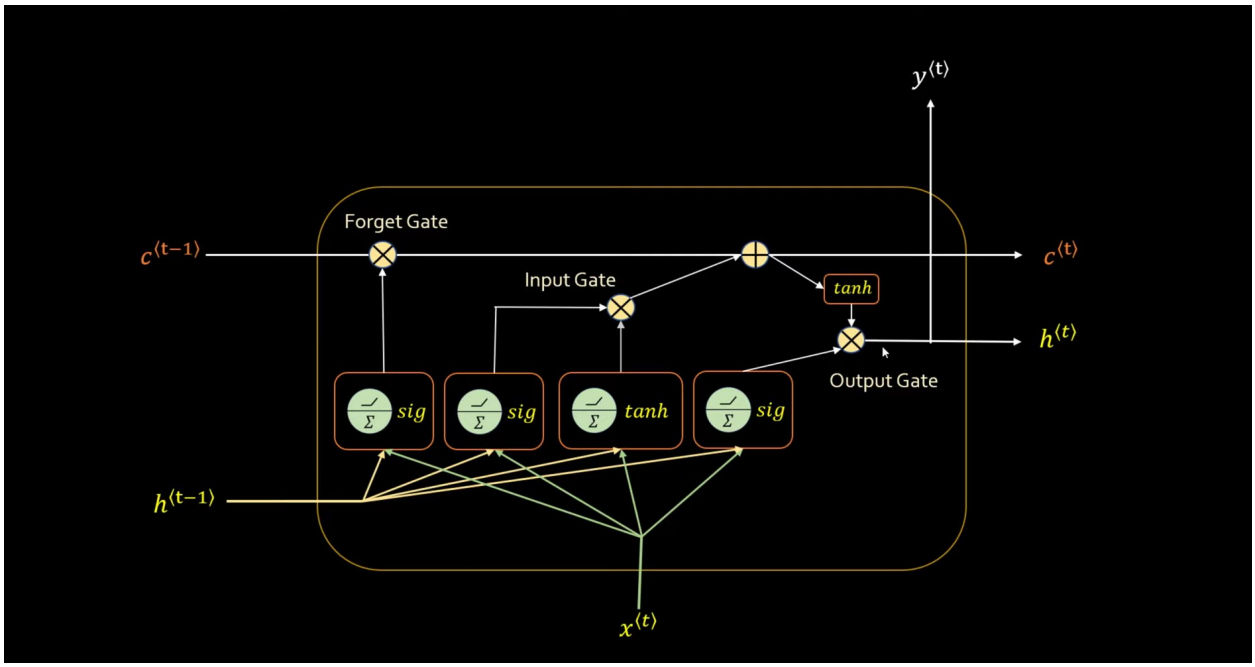


Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian



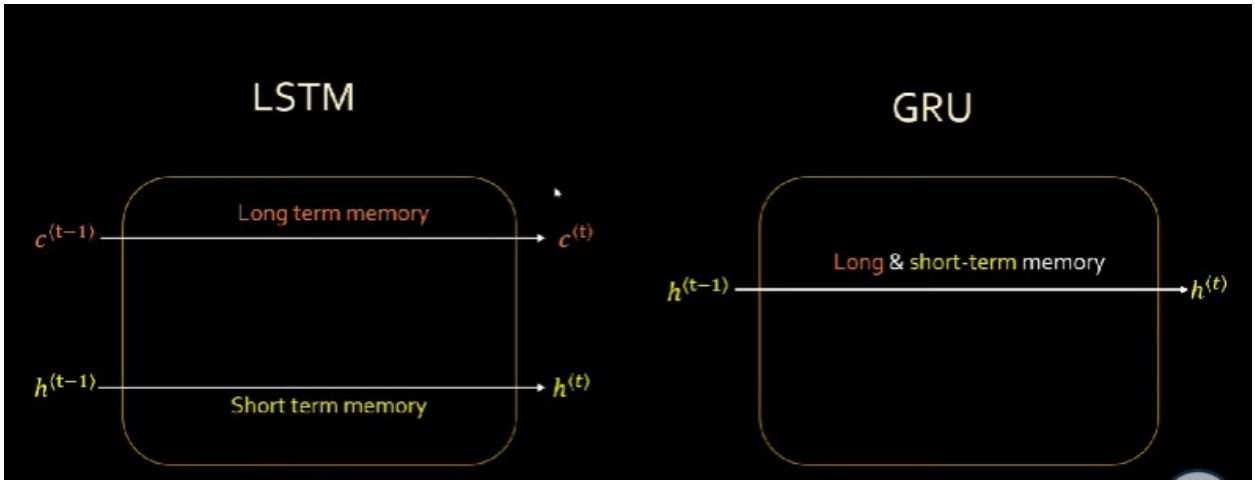
Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His brother Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian



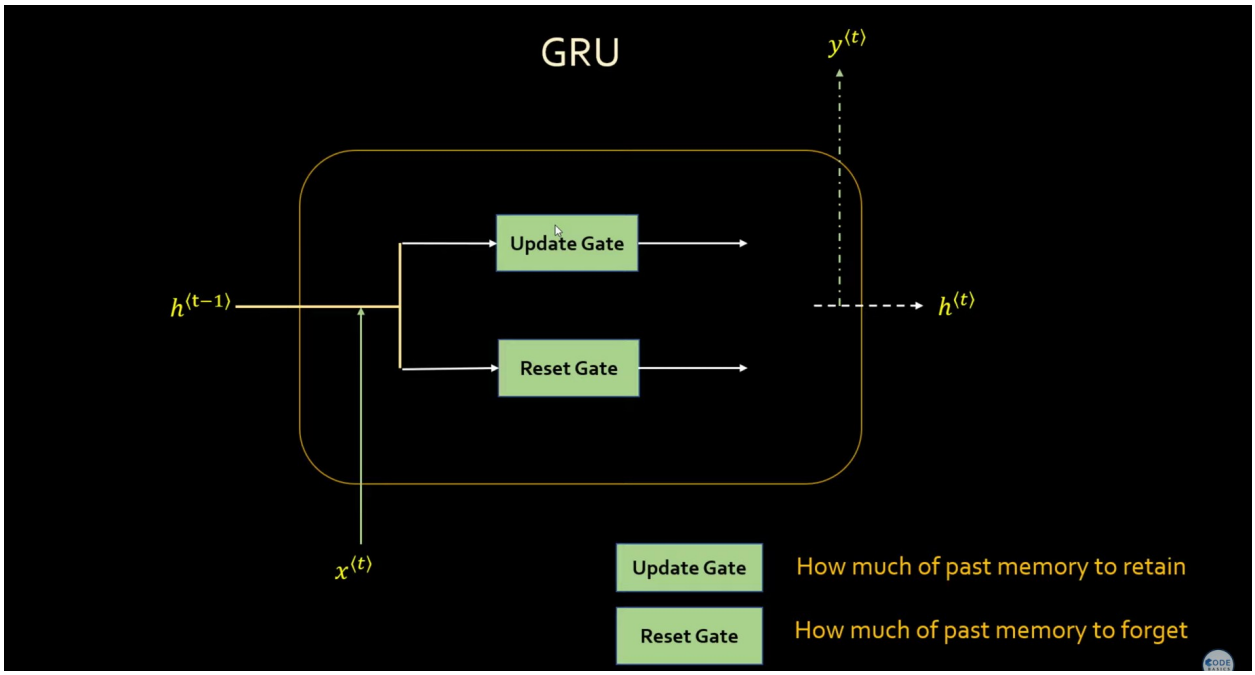


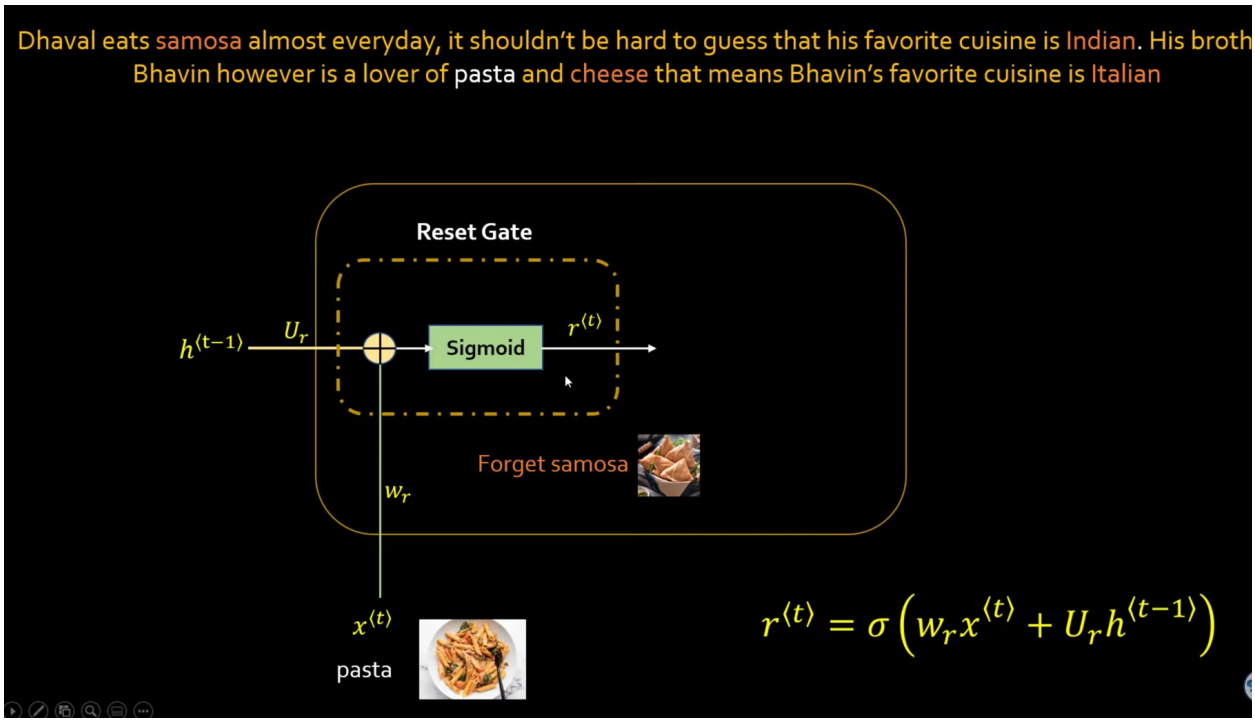
GRU

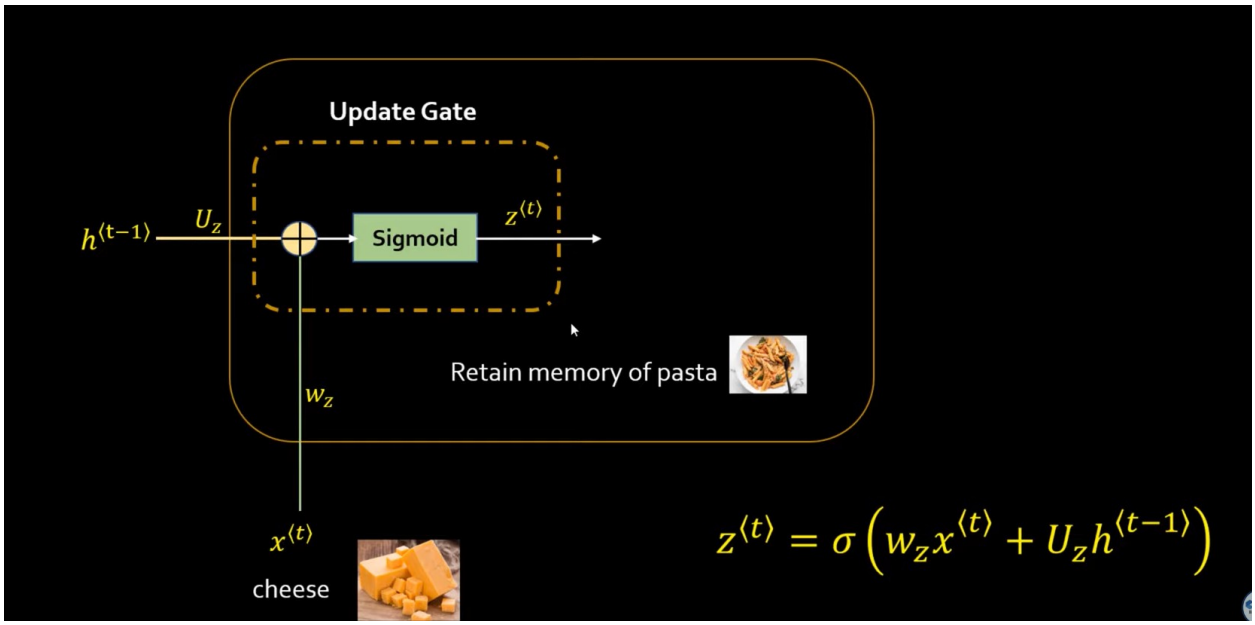
Simple Explanation of GRU (Gated Recurrent Units): Similar to LSTM, Gated recurrent unit addresses short term memory problem of traditional RNN. It was invented in 2014 and getting more popular compared to LSTM.

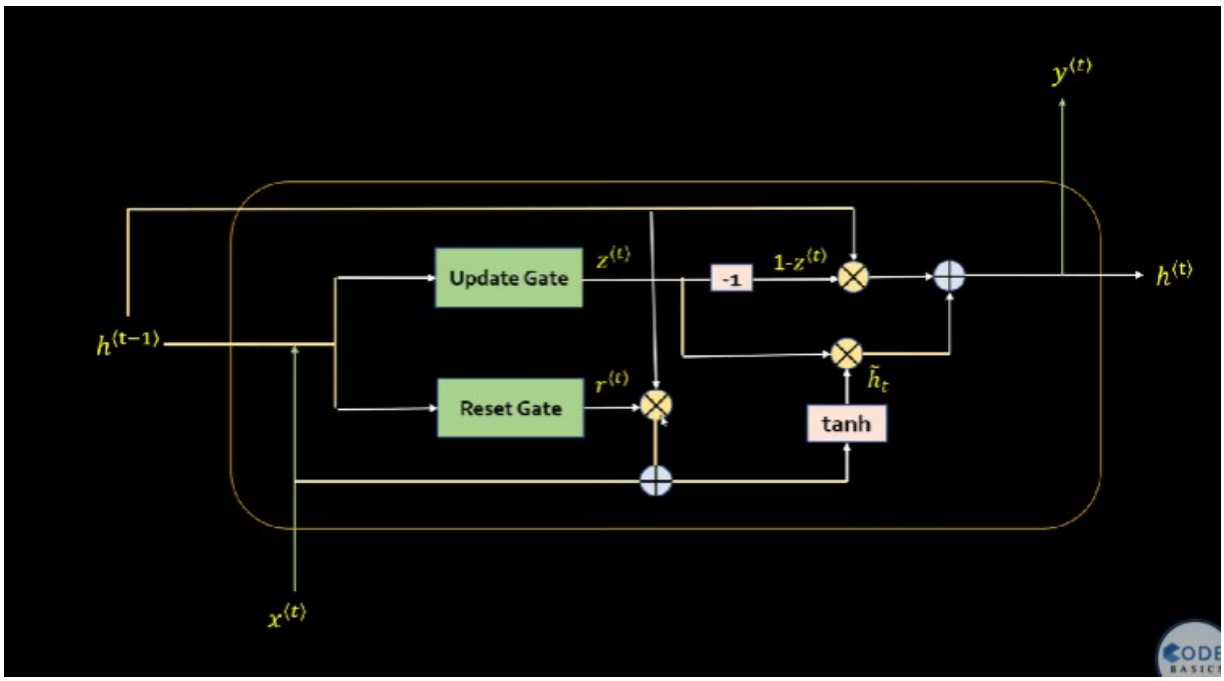


LSTM has two cell state (one is hidden state) whereas GRU has only hidden state as cell state which can combine both long and short term memory.LSTM has three gates:input,output and forget .GRU has two gates:Update gate and reset gate.


GRU
$y^{\langle t \rangle}$
Update Gate
Reset Gate
$h^{\langle t-1 \rangle}$
$h^{\langle t \rangle}$
$x^{\langle t \rangle}$
Update Gate
How much of past memory to retain
Reset Gate
How much of past memory to forget


Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His broth
Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian
Reset Gate
$h^{\langle t-1 \rangle}$
U_r
Sigmoid
$r^{\langle t \rangle}$
Forget samosa
W_r
$x^{\langle t \rangle}$
pasta
$r^{\langle t \rangle} = \sigma\left(W_r x^{\langle t \rangle} + U_r h^{\langle t-1 \rangle}\right)$


Update Gate
$h^{\langle t-1 \rangle}$
U_z
Sigmoid
$z^{\langle t \rangle}$
Retain memory of pasta
w_z
$x^{\langle t \rangle}$
cheese
$z^{\langle t \rangle} = \sigma\left(w_z x^{\langle t \rangle} + U_z h^{\langle t-1 \rangle}\right)$


$y^{\langle t \rangle}$
Update Gate
$z^{\langle t \rangle}$
-1
$1\text{-}z^{\langle t \rangle}$
$h^{\langle t \rangle}$
$h^{\langle t-1 \rangle}$
Reset Gate
$r^{\langle t \rangle}$
$\tilde{h}_t$
tanh
$x^{\langle t \rangle}$

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$
$$\hat{h}_t = \phi_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t$$

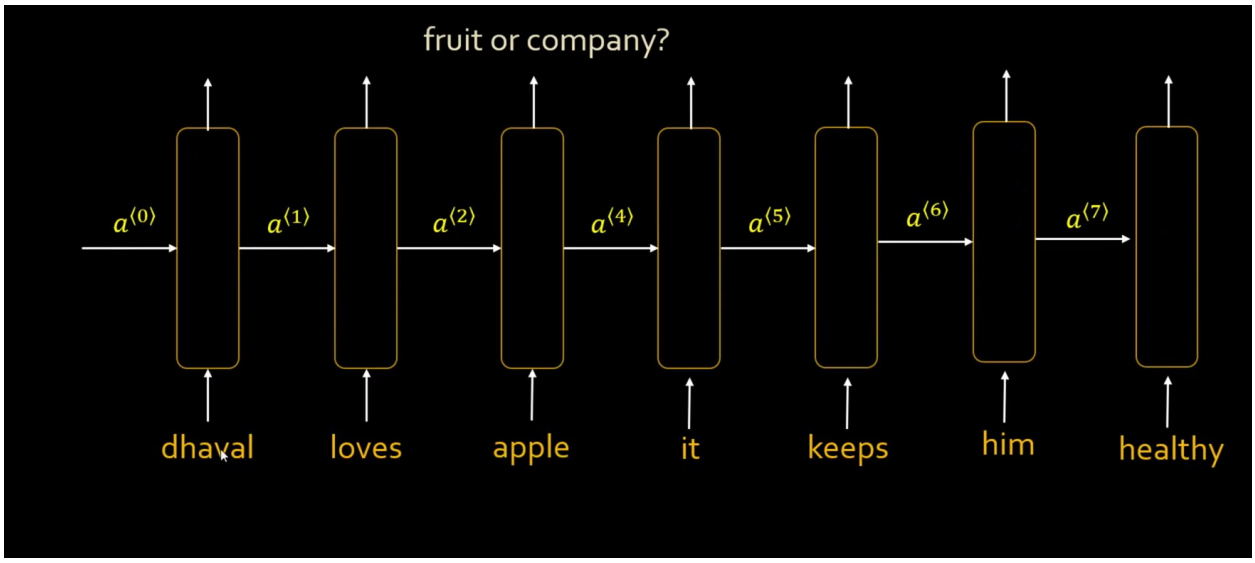
Variables

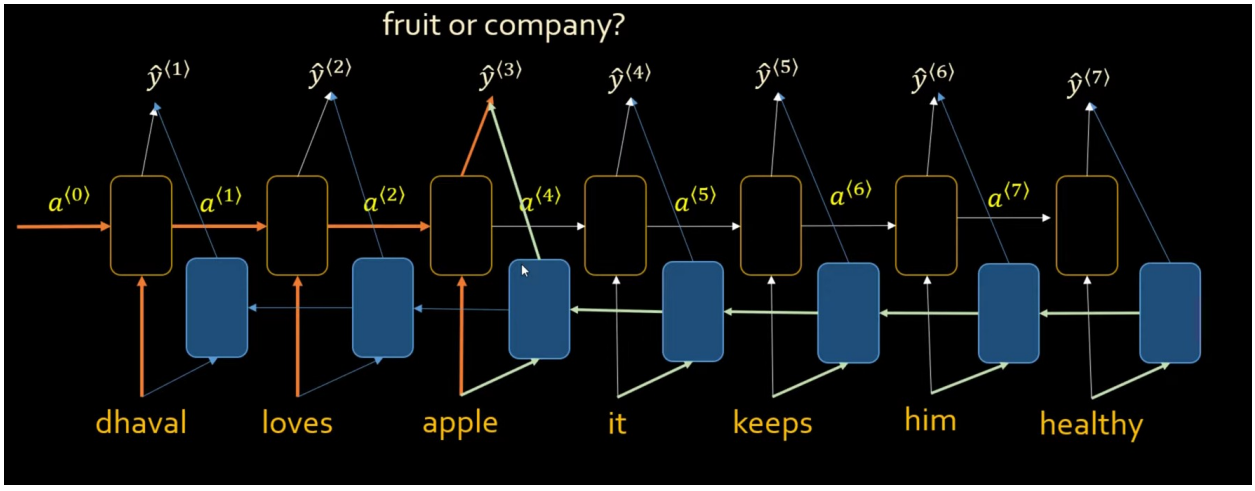
- x_t: input vector
- h_t: output vector
- $\hat{h}_t$: candidate activation vector
- z_t: update gate vector
- r_t: reset gate vector
- W, U and b: parameter matrices and vector

LSTM	GRU
3 Gates: Input, output, forget	2 Gates: reset, update
More accurate on longer sequence, less efficient	More efficient computation wise. Getting more popular
Invented: 1995 - 1997	Invented: 2014

Bi directional RNNs

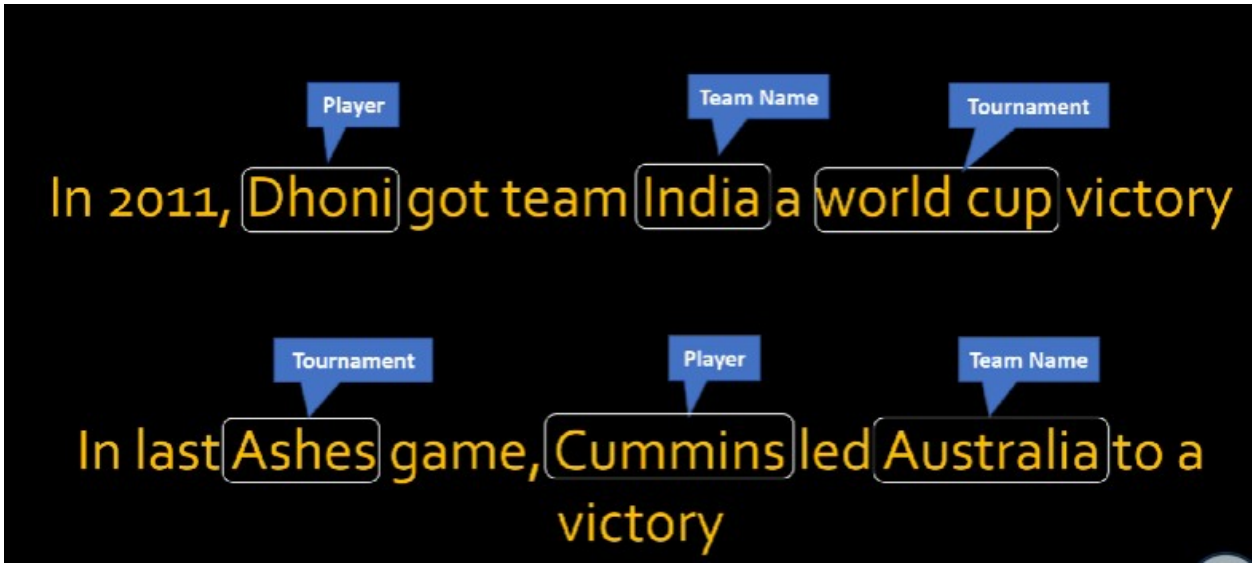
Bi directional RNNs are used in NLP problems where looking at what comes in the sentence after a given word influences final outcome. In this short video we will look at bi directional RNN architecture using a very simple example of named entity recognition.

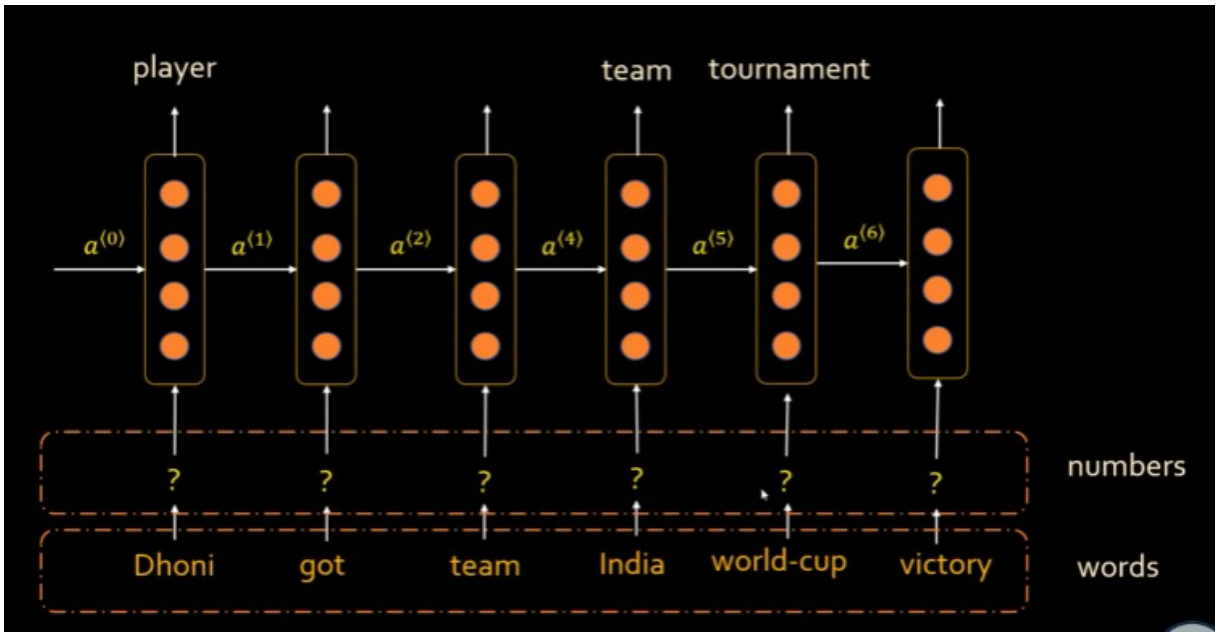


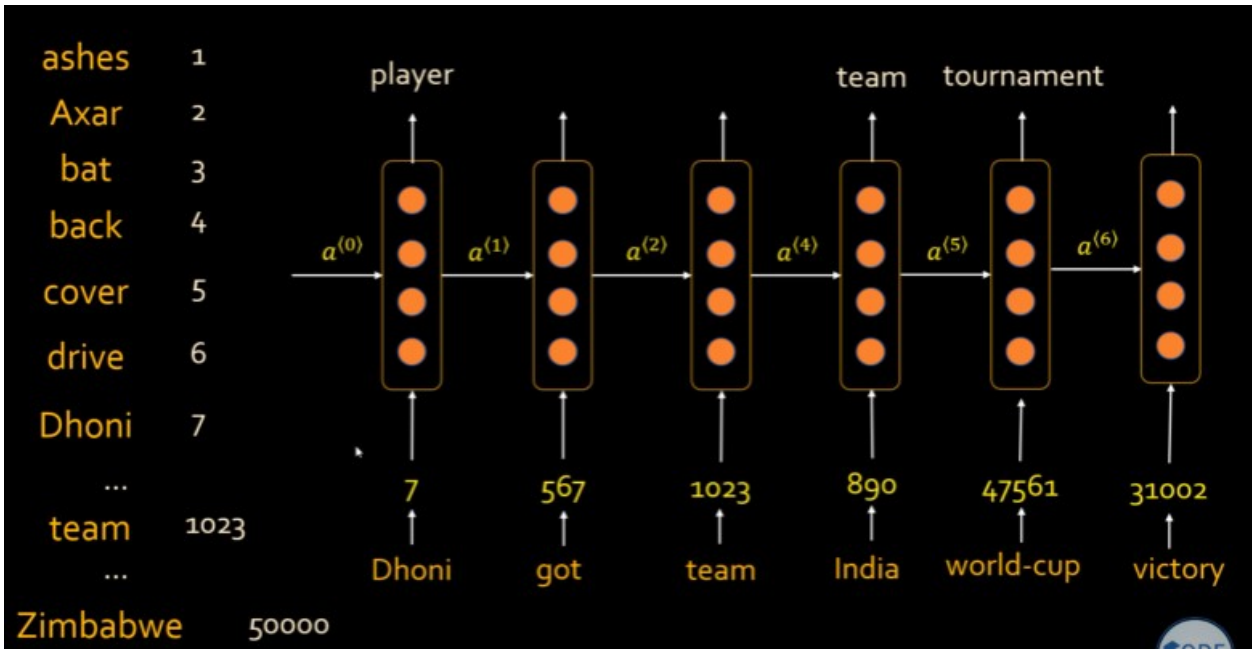


Machine learning models don't understand words. They should be converted to numbers before they are fed to RNN or any other machine learning model. In this tutorial, we will look into various techniques for converting words to numbers. These techniques are, 1) Using unique numbers 2) One hot encoding 3) Word embeddings

Word embeddings







Issue with option 1: unique numbers

Numbers are random. They don't capture relationship between words

Issue with option 2: one hot encoding

1. Doesn't capture relationship between words
2. Computationally in-efficient

	ashes	Axar	Bat	back	cover	drive	dhoni	..	Zimbabve
ashes	1	0	0	0	0	0	0	...	0
Axar	0	1	0	0	0	0	0	...	0
Bat	0	0	1	0	0	0	0	...	0
Back	0	0	0	1	0	0	0	...	0
...									
Zimbabwe	0	0	0	0	0	0	0	...	1

How can we capture similarities between two words?

Dhoni

Person: 1
Healthy/fit: 0.9
Location: 0
Has two eyes: 1
Has government: 0

Cummins

Person: 1
Healthy/fit: 0.87
Location: 0
Has two eyes: 1
Has government: 0

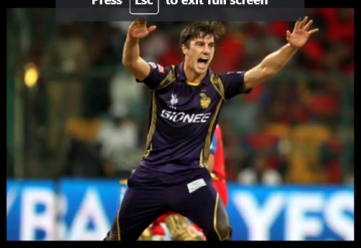
Australia

Person: 0
Healthy/fit: 0.7
Location: 1
Has two eyes: 0
Has government: 1

Press Esc to exit full screen

Dhoni

$$\begin{bmatrix} 1 \\ 0.9 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$



Cummins

$$\begin{bmatrix} 1 \\ 0.87 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Australia

$$\begin{bmatrix} 0 \\ 0.7 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

	ashes	Australia	Bat	Cummins	cover	Dhoni	World cup	..	Zimbabwe
Person	0	0.02	0.1	**0.95**	0.03	**0.96**	0.67	...	0.04
Country	0	**0.97**	0	0	0	0	0	...	**1**
Healthy & Fit	0	0	0.3	**0.87**	0	**0.9**	0	...	0
Event	**1**	0.1	0	0	0.4	0	**1**	...	0
gear	0	0	**1**	0	0	0	0	...	0

	ashes	Australia	Bat	Cummins	cover	Dhoni	World cup	..	Zimbabwe
Person	0	0.02	0.1	**0.95**	0.03	**0.96**	0.67	...	0.04
Country	0	**0.97**	0	0	0	0	0	...	**1**
Healthy & Fit	0	0	0.3	**0.87**	0	**0.9**	0	...	0
Event	**1**	0.1	0	0	0.4	0	**1**	...	0
gear	0	0	**1**	0	0	0	0	...	0

Cummins	Dhoni	Australia	Zimbabwe
0.95	0.96	0.02	0.04
0	0	0.97	1
1	1	0	0
0	0	0.1	0
0	0	0	0

Converting words to numbers

1. Unique numbers
2. One hot encoding
3. Word embeddings
   1. TF-IDF
   2. Word2Vec

Embeddings are not hand crafted. Instead, they are learnt during neural network training

Techniques to compute word embeddings

1. Using **supervised** learning
2. Using **self-supervised** learning
    1. Word2vec
    2. Glove

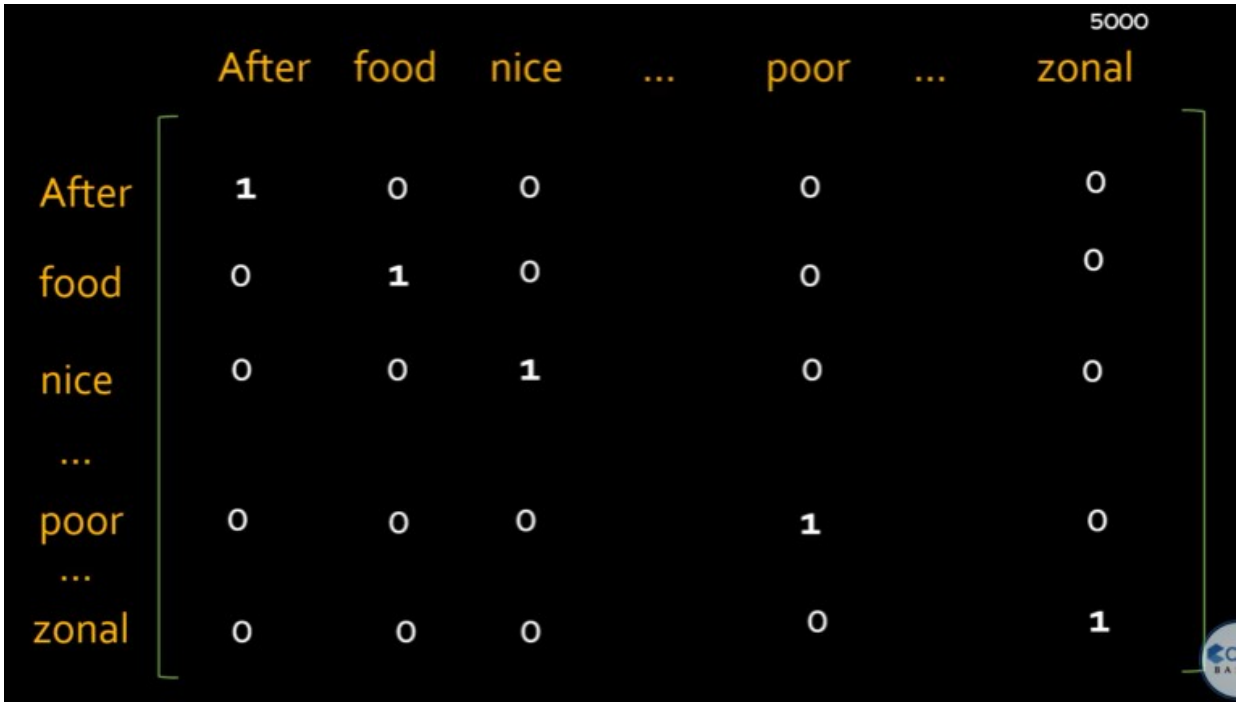
1. Using **supervised** learning

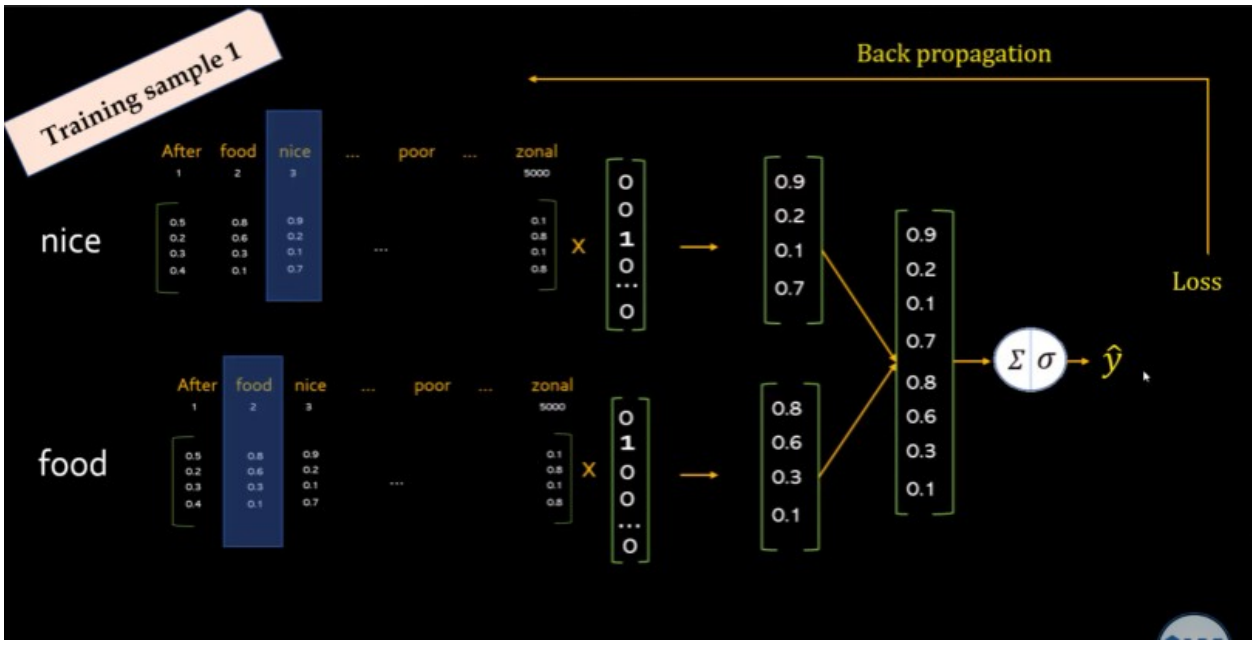
Take an NLP problem and try to solve it. In that pursuit as a side effect, you get word embeddings

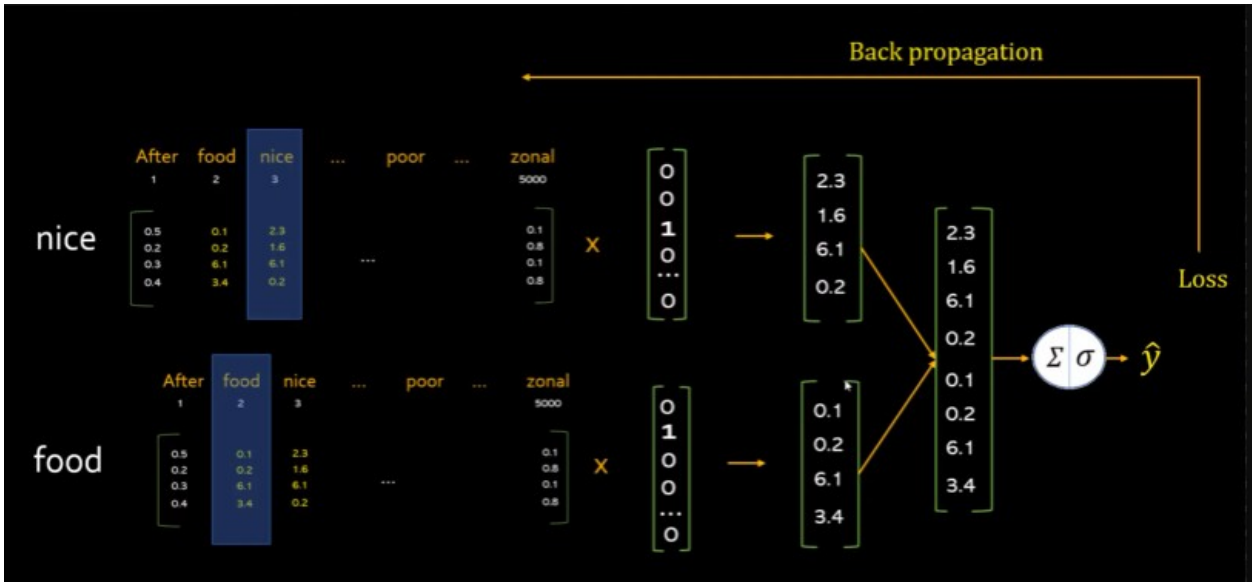

nice food. The pasta dish was too good!
poor quality food. I would never go there again!

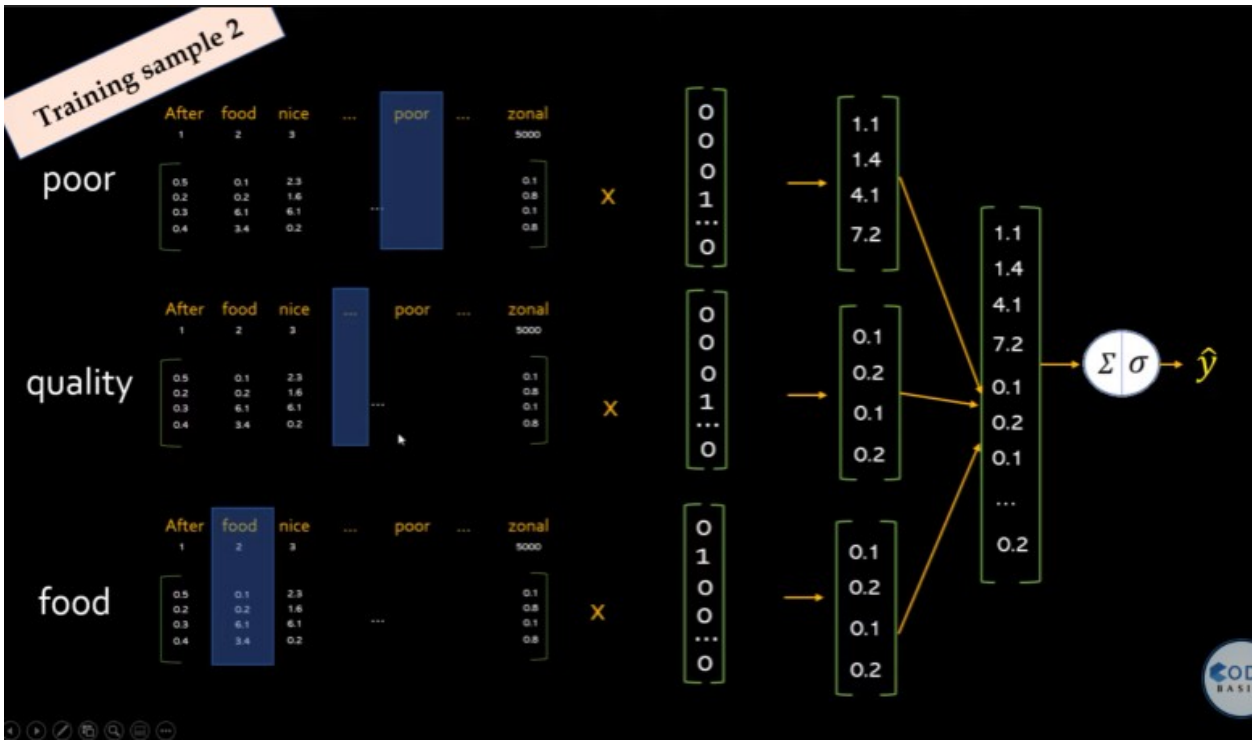

After
food
nice
...
poor
...
zonal
1
2
3
5000
0.5
1.2
0.7
3.1
4.3
0.1
0.9
5.5
0.4
8.1
8.8
4.2
9.8
0.6
2.2
1.3
4-dimensional word embedding vector

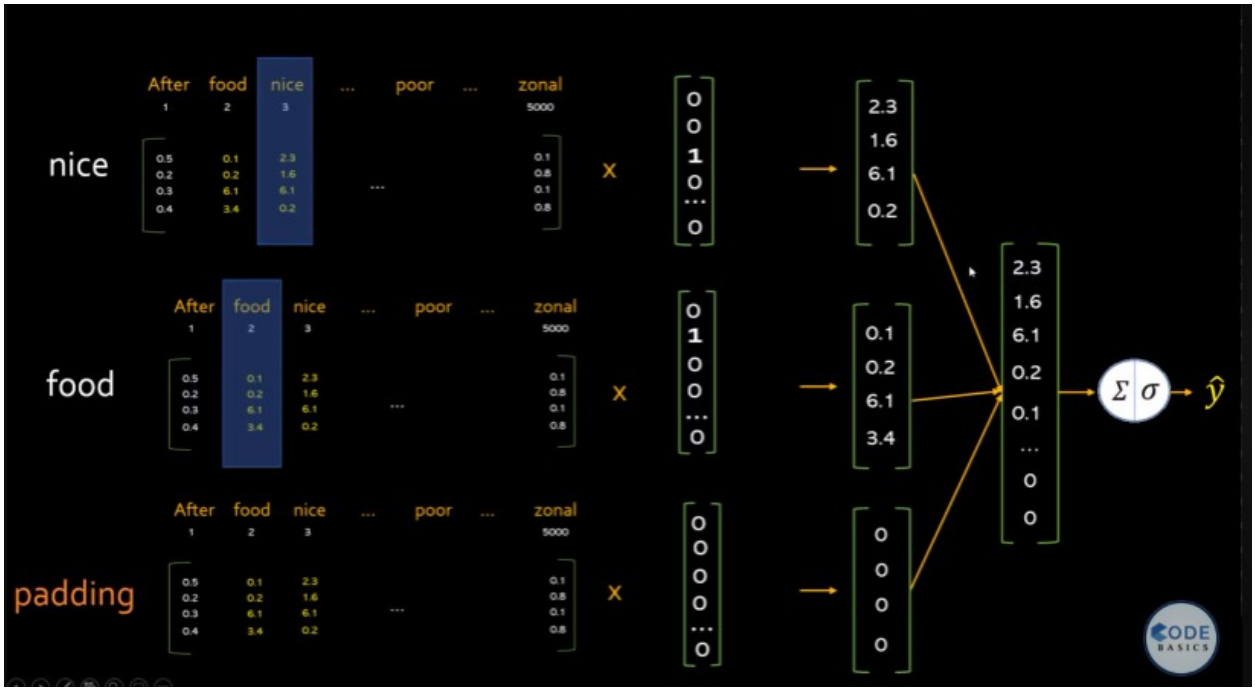

After food nice ... poor ... zonal
1 2 3 5000
0.5 4.3 0.4 9.8
1.2 0.1 8.1 0.6
0.7 0.9 8.8 ... 2.2
3.1 5.5 4.2 1.3
4 by 5000 Embedding Matrix: E


5000
After food nice ... poor ... zonal
After 1 0 0 0 0
food 0 1 0 0 0
nice 0 0 1 0 0
...
poor 0 0 0 1 0
...
zonal 0 0 0 0 1


Training sample 1
Back propagation
After food nice ... poor ... zonal
1 2 3 5000
nice
food
Loss
Σ σ
$\hat{y}$


Back propagation
After food nice ... poor ... zonal
1 2 3 5000
nice
food
Loss
Σ σ
$\hat{y}$


Training sample 2
poor
quality
food
After food nice ... poor ... zonal
1 2 3 5000
x
1.1
1.4
4.1
7.2
0.1
0.2
0.1
0.2
Σ σ
ŷ


nice
food
padding
After food nice ... poor ... zonal
1 2 3 5000
x
2.3
1.6
6.1
0.2
0.1
0.2
6.1
3.4
0
0
Σ σ
ŷ
CODE BASICS

Similar words have almost similar vectors.