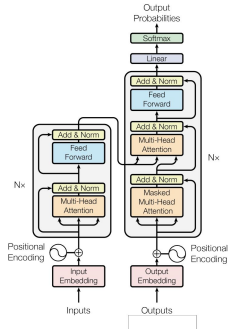
TensorFlow
User Group Lucknow

LUCHNOW AI

Build, Train & Deploy
A hands on workshop from Basic Neural Networks to Generative AI

BERT

Encoder



GPT

Decoder

Google Developer Groups

About Me

🤩

Cool Research | Trekking | Boxing | Skiing | Chess

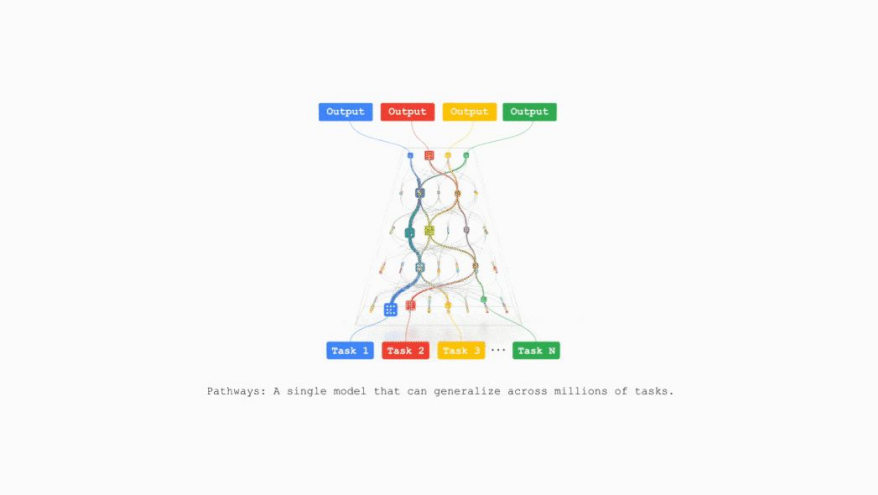
- Senior AI Research Engineer @ Saama AI Research Lab
- Research interests involve
  Representation Learning on Graphs and Manifolds
- Generative Modeling, MLOps, Signal Processing
  and their applications in Healthcare data

Setup Colab

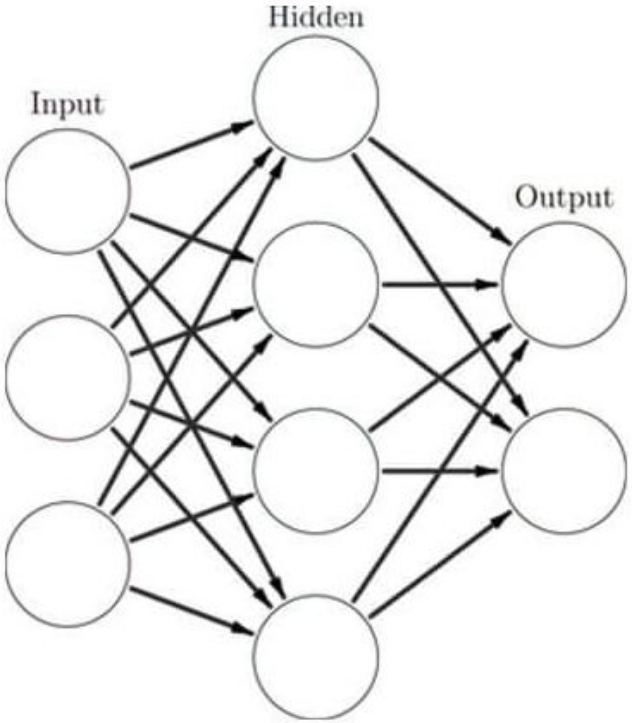
https://colab.research.google.com/

Topic for today

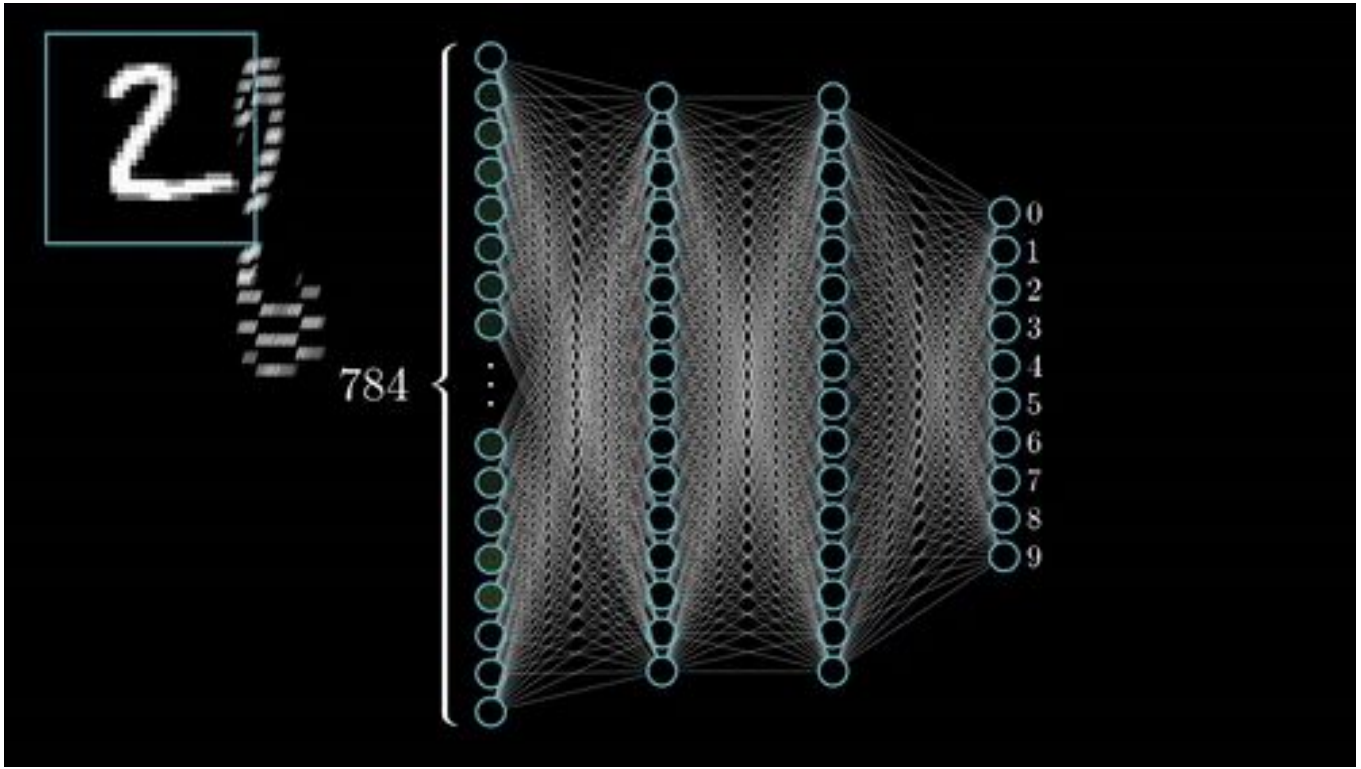
- Understanding Basics of Neural Networks & Practical code
- Embeddings & Practical code
- Attention Mechanism
- Transformers [encoders & decoders]
- Bert Model & Practical code
- GPT Model



What are Neural Networks?



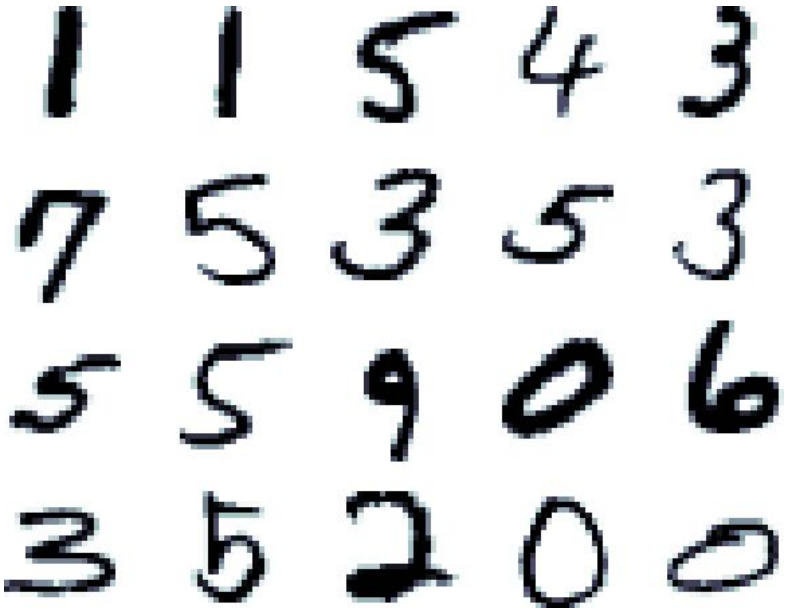
AI/ML



Source : http://tinyurl.com/ypa76y9m

Neural Networks

classify the each number's image from 0-9

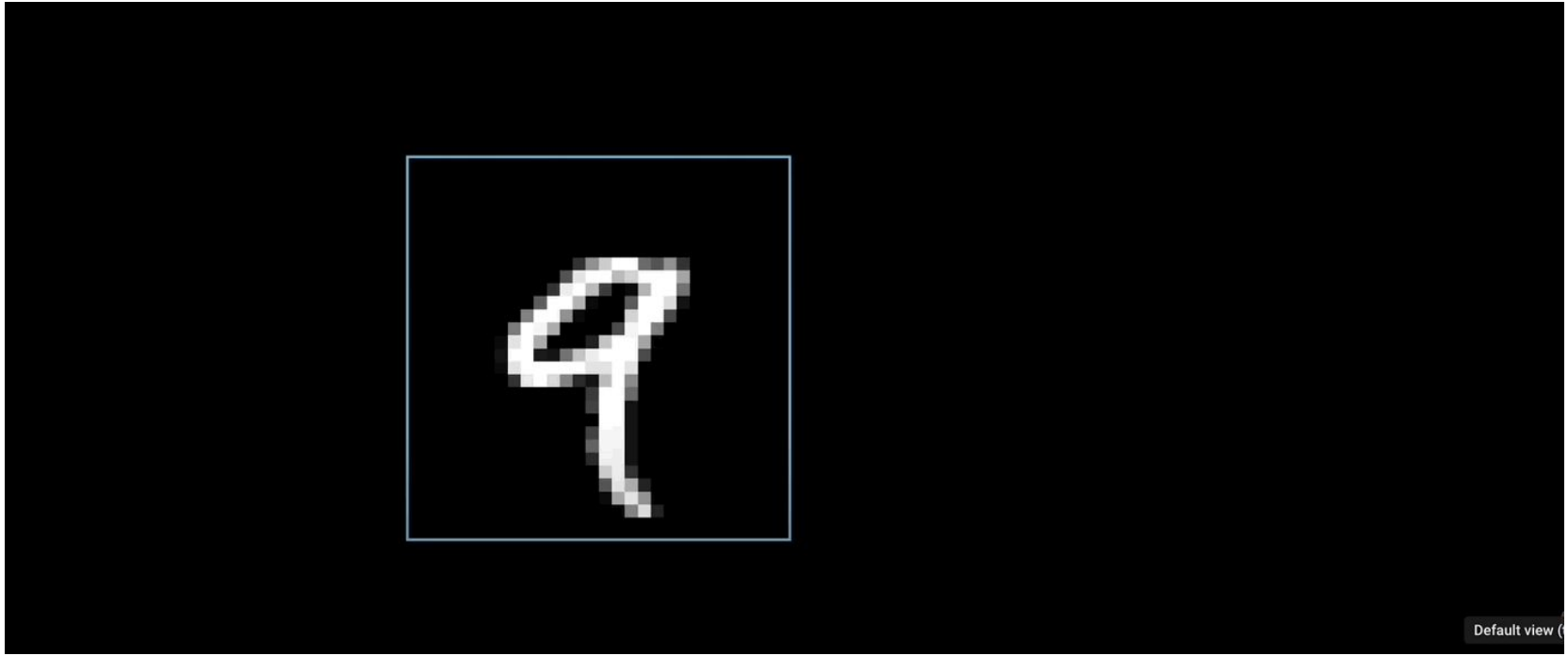


Neural Networks

?

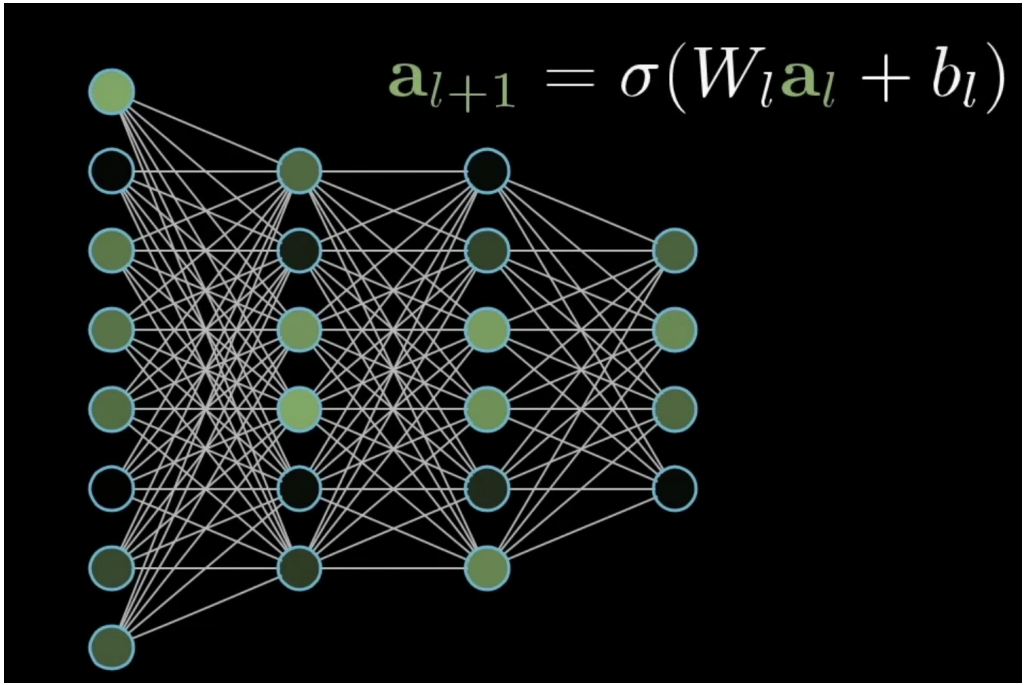
Which number is it?

Neural Networks

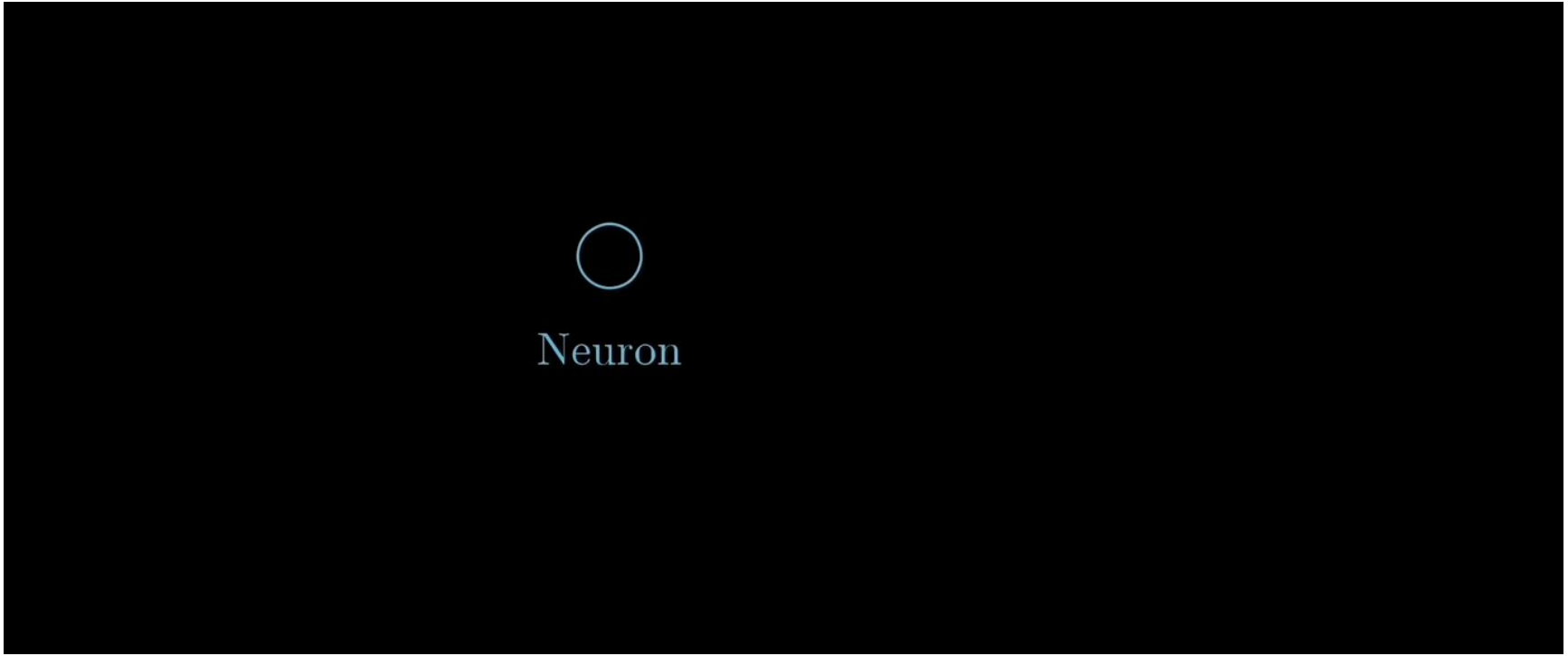


Source : http://tinyurl.com/ypa76y9m

Neural Networks

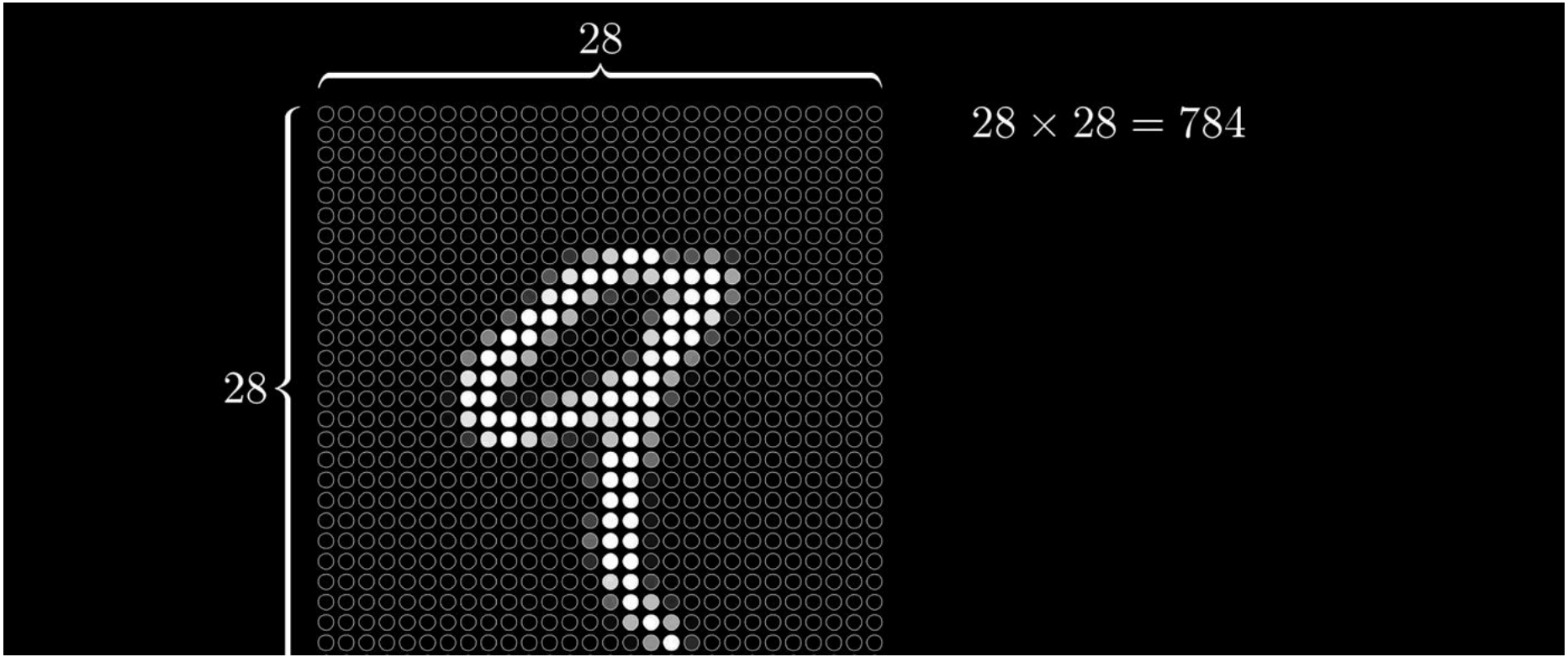


Neural Networks



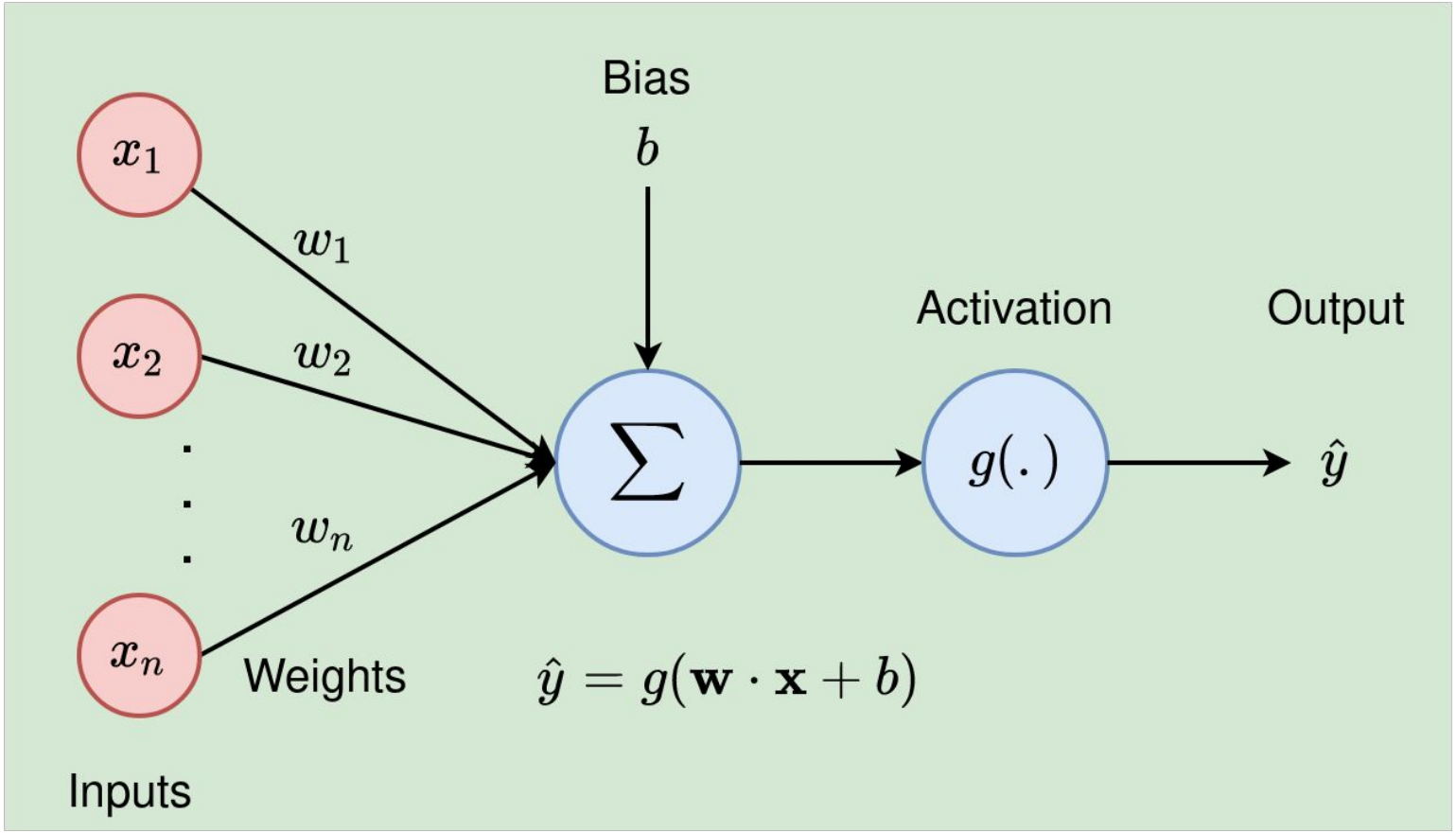
Source : http://tinyurl.com/ypa76y9m

Neural Networks



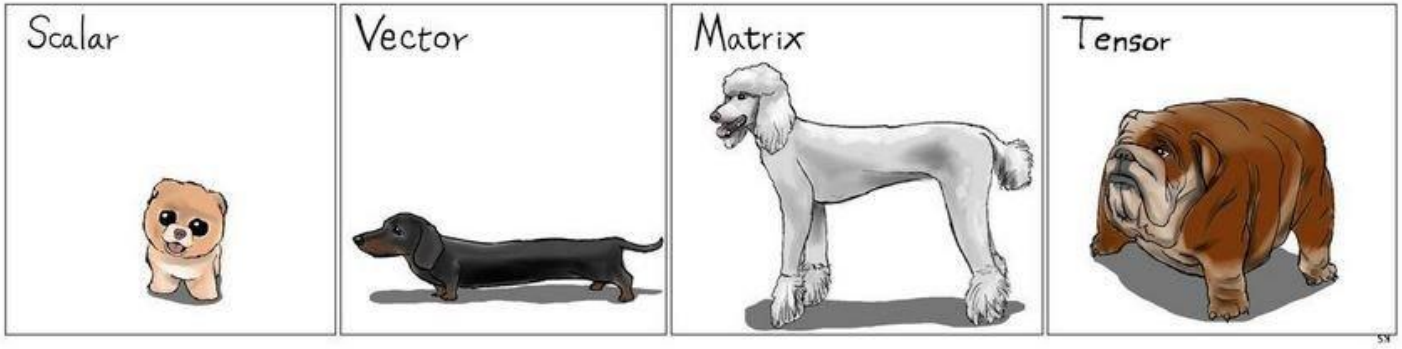
Source : http://tinyurl.com/ypa76y9m

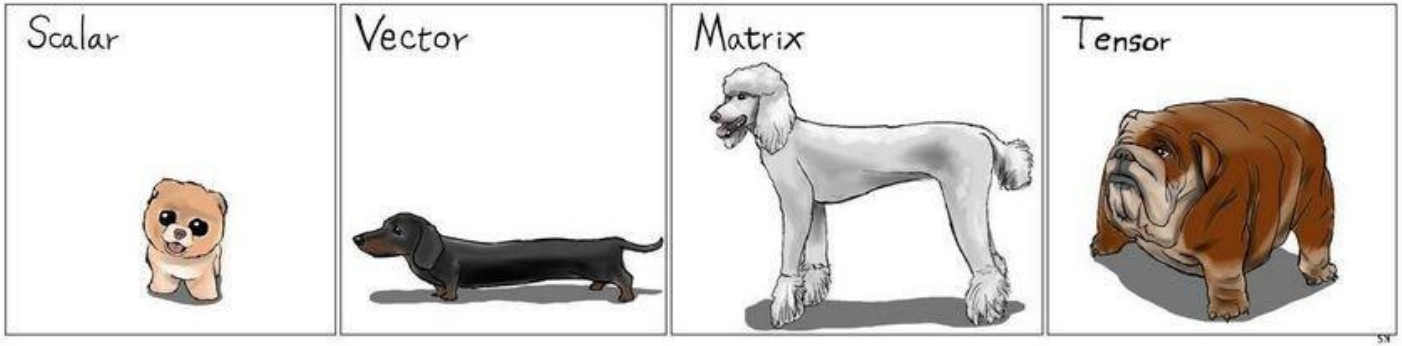
Google Developer Groups


Bias
b
x_1
x_2
x_n
w_1
w_2
w_n
$\sum$
Activation
$g(.)$
Output
$\hat{y}$
Weights
$\hat{y} = g(\mathbf{w} \cdot \mathbf{x} + b)$
Inputs


AAYEIN

What are matrices, scalars?


Scalar
Vector
Matrix
Tensor

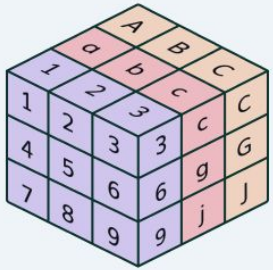


(11)

5	3	7

5
1.5
2

$$\begin{bmatrix} 4 & 19 & 8 \\ 16 & 3 & 5 \end{bmatrix}$$



Scalar

Row Vector
(shape 1×3)

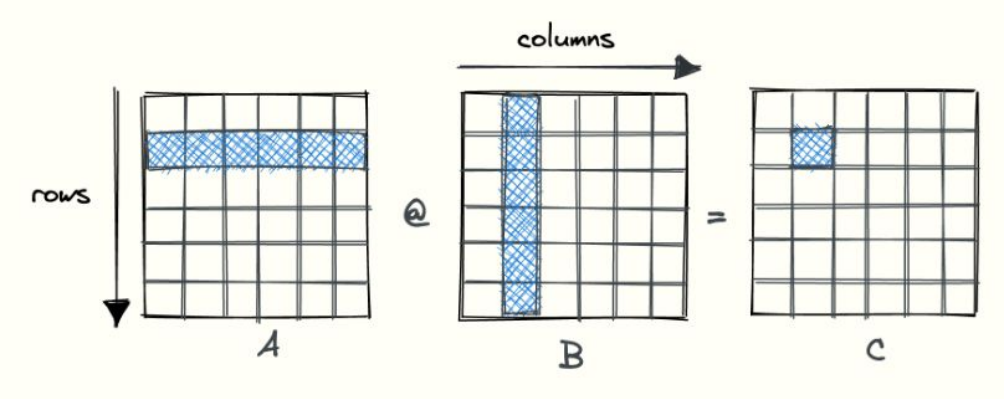
Column Vector
(shape 3×1)

Matrix

Tensor

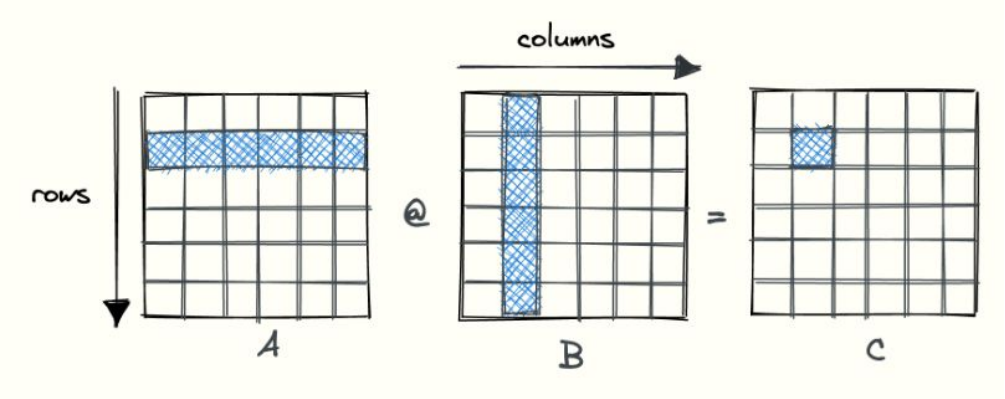

Scalar
Vector
Matrix
Tensor
1
1
2
1 2
3 4
1 2
1 7
3 2
5 4

Matrix Multiplication



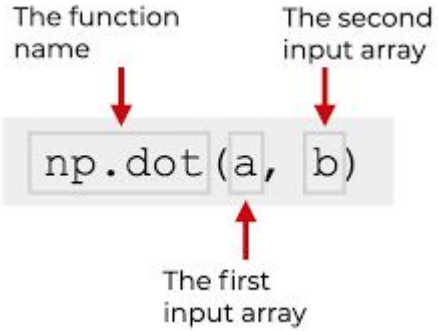
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ & \end{bmatrix}$$

$$(1, 2, 3) \bullet (8, 10, 12) = 1{\times}8 + 2{\times}10 + 3{\times}12 = 64$$

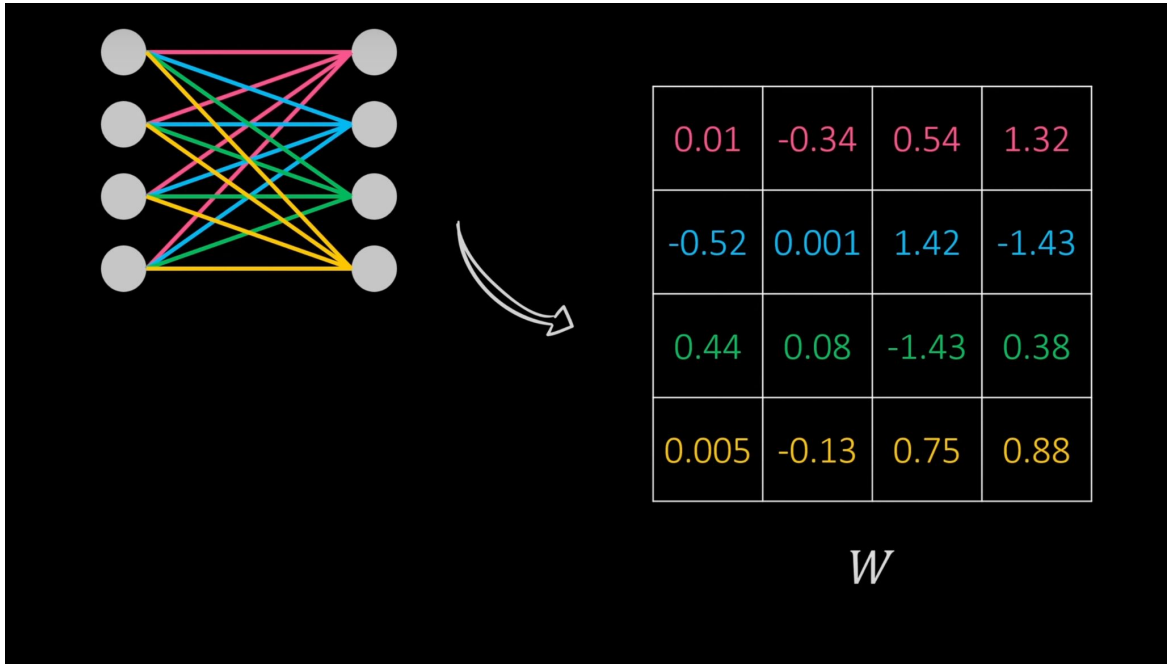


$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ \end{bmatrix}$$

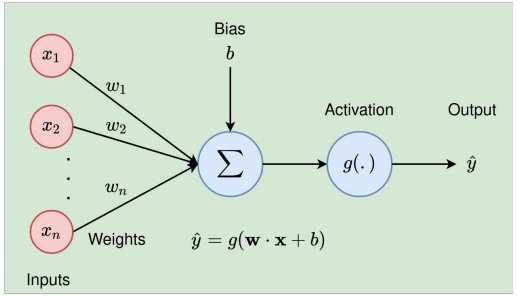
$$(1, 2, 3) \bullet (8, 10, 12) = 1 \times 8 + 2 \times 10 + 3 \times 12 = 64$$

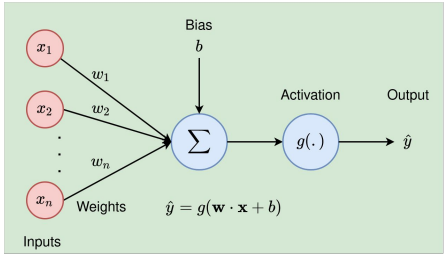


Parameters in Machine Learning Model



Let’s Implement our first Neural Network


Bias
b
x_1
w_1
x_2
w_2
w_n
x_n
$\sum$
Activation
$g(.)$
Output
$\hat{y}$
Weights
$\hat{y} = g(\mathbf{w} \cdot \mathbf{x} + b)$
Inputs



$$
\begin{aligned}
ouput &= np.dot(wieghts, input) + bias \\
&= np.dot([0.4,0.6,-0.7,1.1],[1.2,2.2,3.3,2.5]) \\
&= ((0.4 * 1.2) + (0.6 * 2.2) + (-0.7 * 3.3) + (1.1 * 2.5)) + 2 \\
&= (0.48 + 1.32 - 2.31 + 2.75) + 2 \\
&= 2.24 + 2 \\
&= 4.24
\end{aligned}
$$

Code for Simple Neural Networks v1 & v2

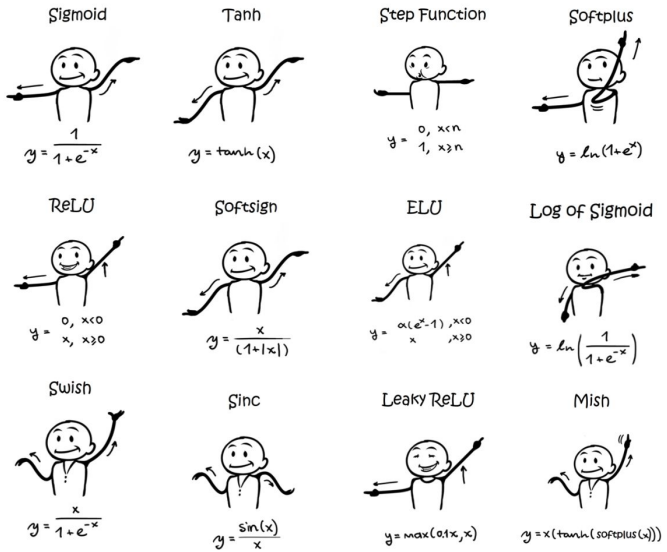
Code for Simple Neural Networks v1 & v2

```python
import math

def simple_neural_network(inputs, weights, bias, activation_function):
    """
    Represents a simple neural network with a single artificial neuron using pure
Python.

    Parameters:
    - inputs (list): Input values (features).
    - weights (list): Weights corresponding to the inputs.
    - bias (float): Bias term.
    - activation_function (callable): Activation function to apply to the weighted
sum.

    Returns:
    - Output of the neuron after applying the activation function.
    """

    # Calculate the weighted sum of inputs and bias using pure Python
    weighted_sum = sum(weight * input for weight, input in zip(weights, inputs)) +
bias

    # Apply the activation function
    output = activation_function(weighted_sum)

    return output
```

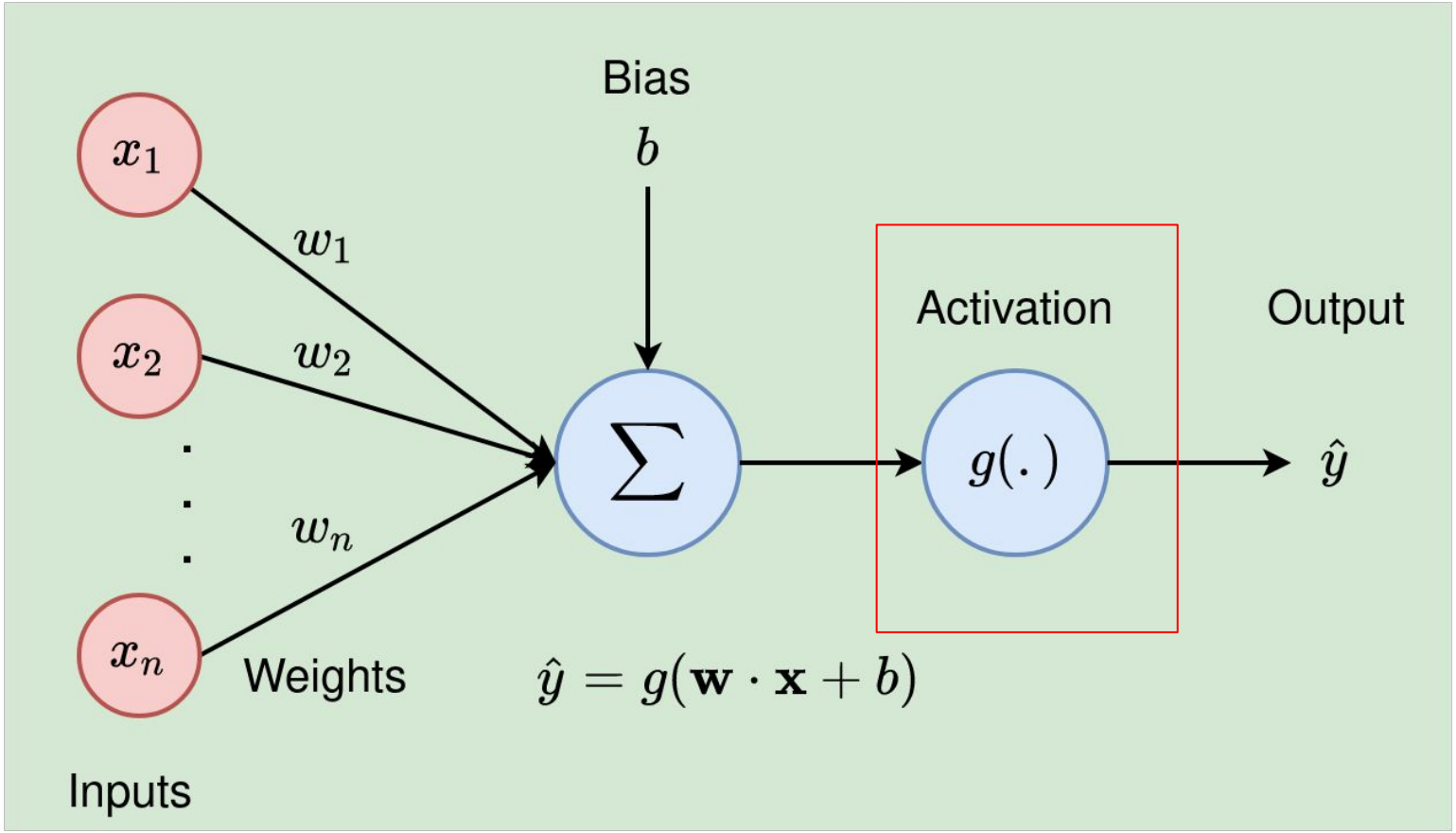
```python
# Example usage:
# Define input features
inputs = [1.0, 2.0, 3.0]  # Replace with actual input values

# Define weights
weights = [0.2, 0.8, -0.5]  # Replace with actual weights

# Define bias
bias = 2.0  # Replace with actual bias

# Define activation function, for example, sigmoid
def sigmoid(x):
    return 1 / (1 + math.exp(-x))

# Compute the output of the neuron
output = simple_neural_network(inputs, weights, bias, sigmoid)
output
```

Activation Function

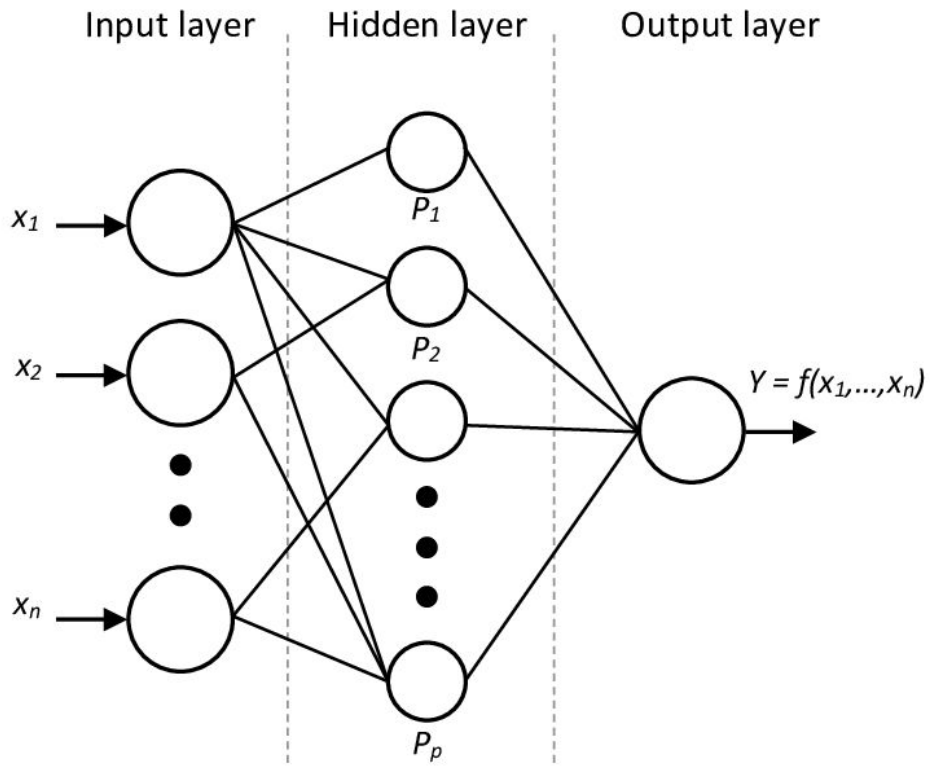
The purpose of the activation function is to introduce non-linearity into the output of a neuron.



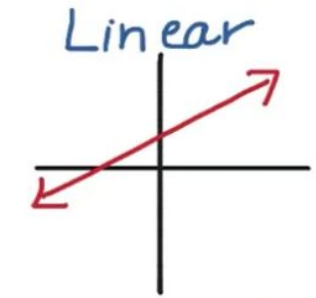
Google Developer Groups


Bias
b
x_1
x_2
x_n
w_1
w_2
w_n
$\sum$
Activation
$g(.)$
Output
$\hat{y}$
Weights
$\hat{y} = g(\mathbf{w} \cdot \mathbf{x} + b)$
Inputs

Activation Function



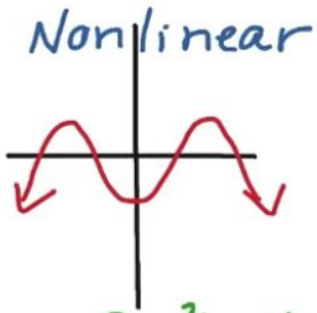
Linear data vs Non-Linear data



$y = 6x - 3$

x	y	
1	8	
2	16	+8
3	24	+8
		+1
		+1

slope (m) = 8/1 or 8



$y = 8x^2 + 11$

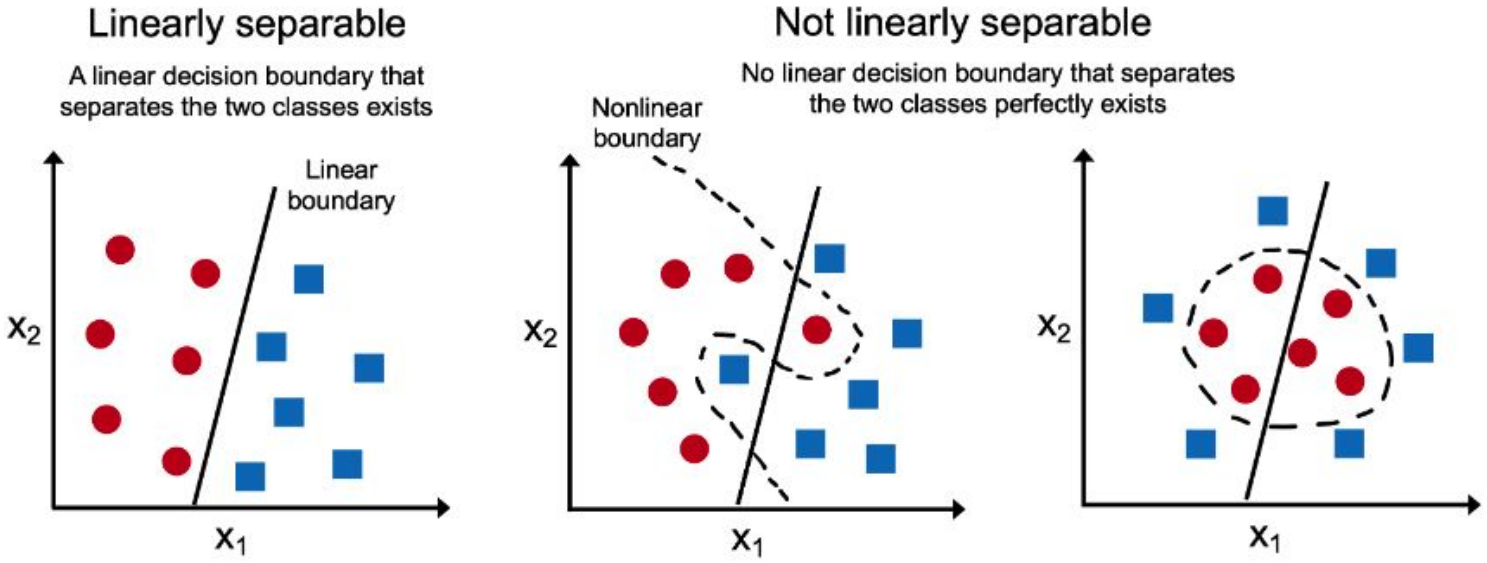
x	y	
1	4	
2	12	+8
3	15	+3
		+1
		+1

No constant rate of change

Linear data vs Non-Linear data

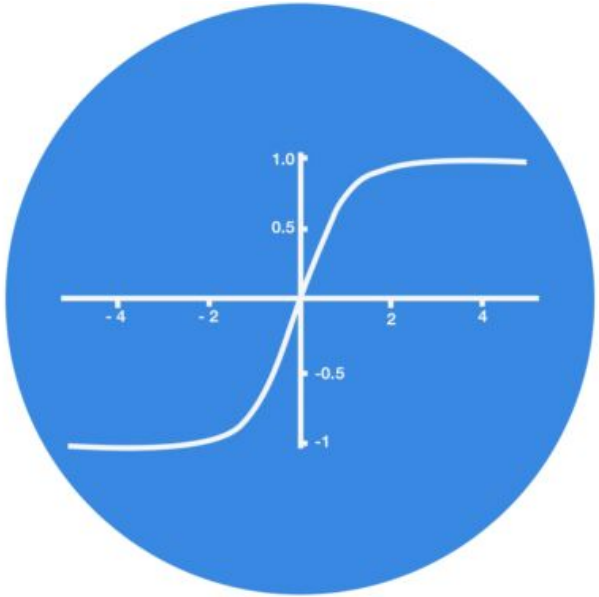
Weather Patterns, Population Growth of Species, Stock Market Prices all are non-linear

Linear data vs Non-Linear data



Activation Function

5
0.1
-0.5



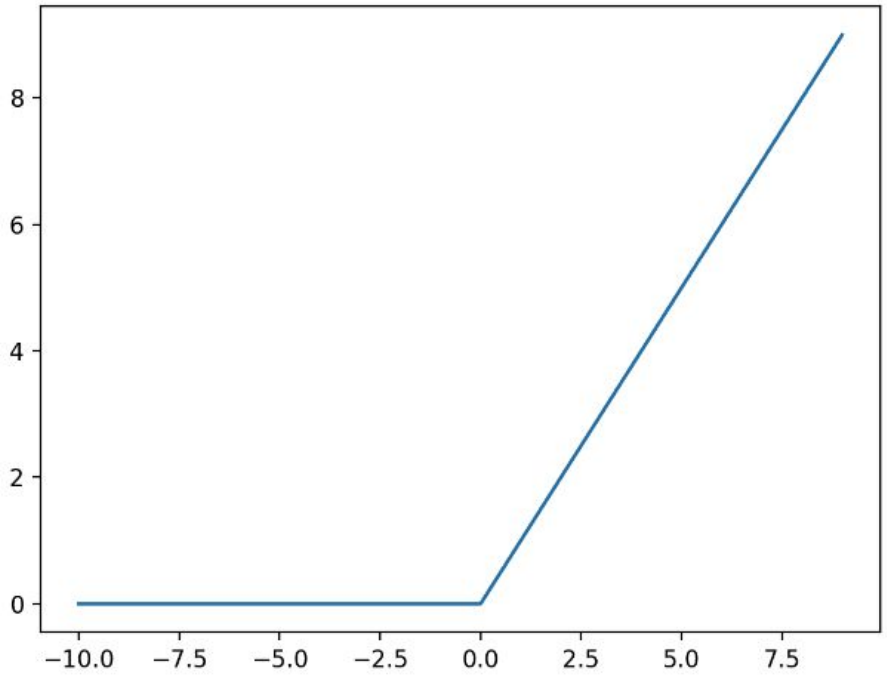
A threshold value determines whether a neuron should be activated or not activated

Activation Function

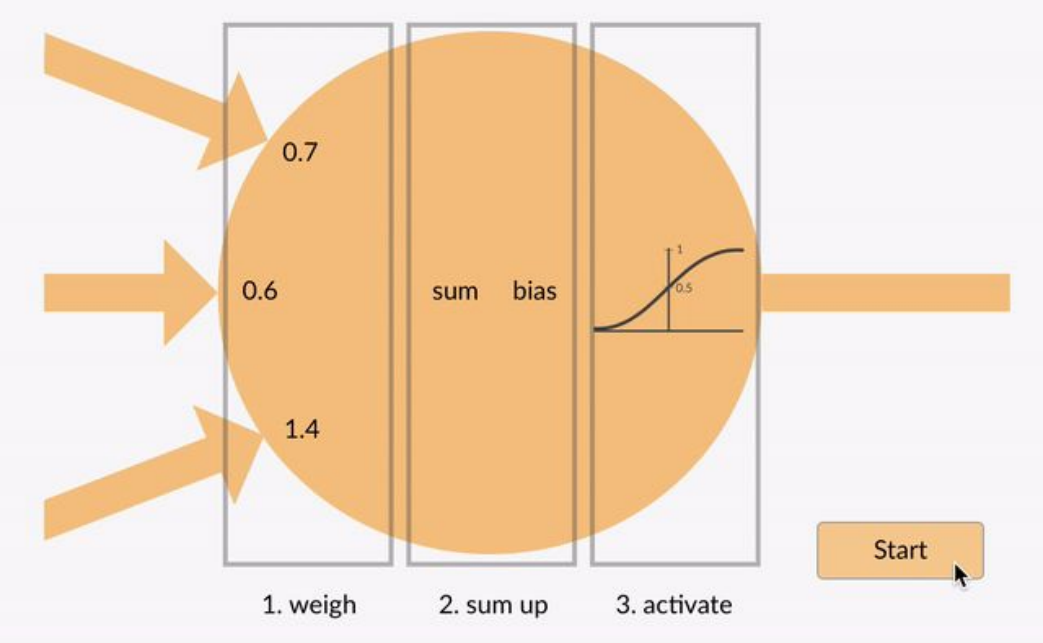
Rectified Linear Unit
Activation Function
(ReLU)

max(0,x)

Activation Function



Activation Function



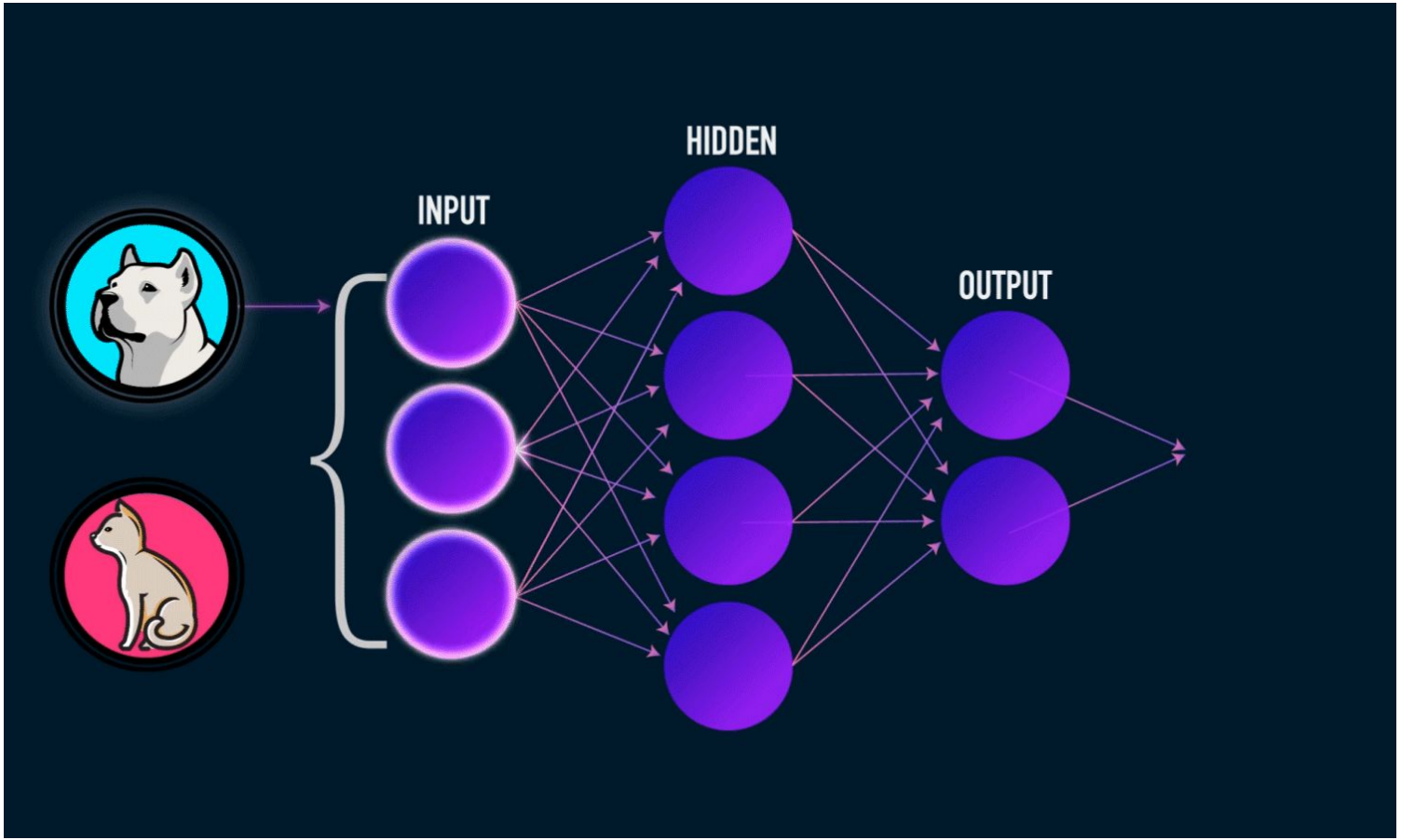
Three steps

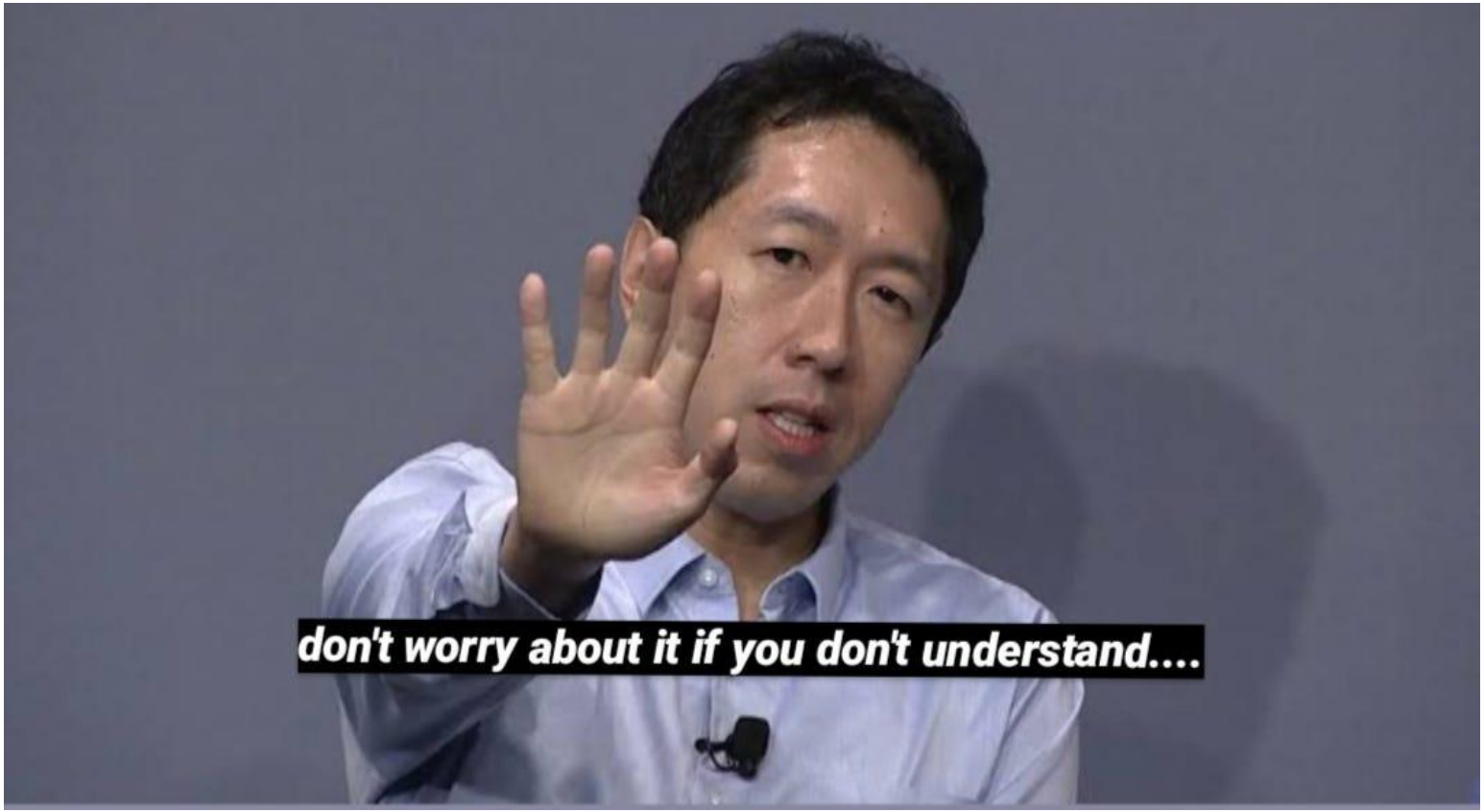
Code

`Simple Neural Network Notebook V3`

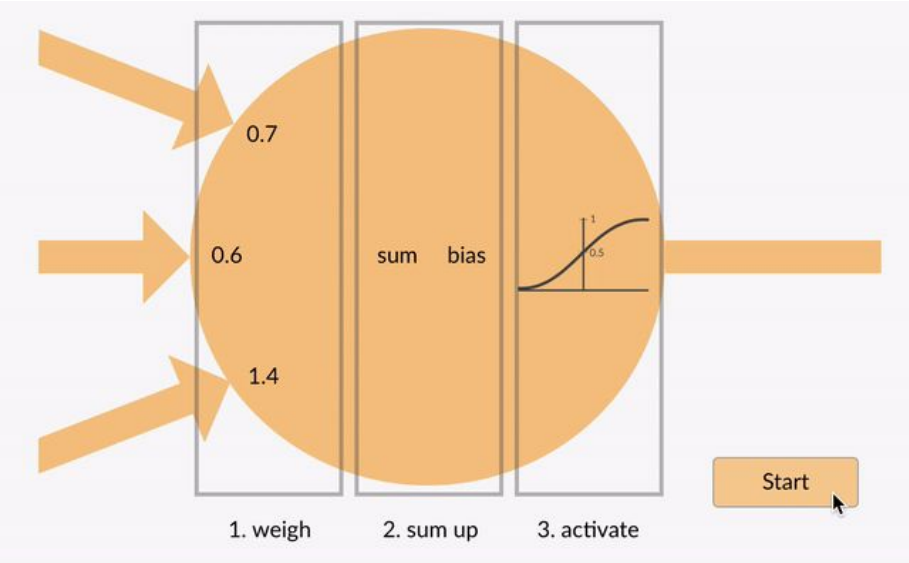
```python
import numpy as np

class DenseLayer:
    def __init__(self, input_size, output_size, activation_function=None):

        self.weights = np.random.randn(input_size, output_size) * 0.01
        self.biases  = np.zeros((1, output_size))
        self.activation_function = activation_function

    def forward(self, inputs):

        self.inputs = inputs
        self.output = np.dot(inputs, self.weights) + self.biases
        if self.activation_function:
            self.output = self.activation_function(self.output)
        return self.output


# Example activation function
def relu(x):
    return np.maximum(0, x)
```

Dense Layer is simple layer of neurons in which each neuron receives input from all the neurons of previous layer, thus called as dense


HIDDEN
INPUT
OUTPUT


don't worry about it if you don't understand....

How we give sentence or word inputs to Neural networks



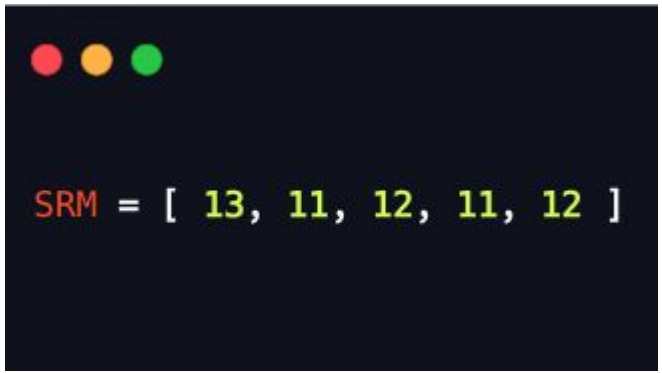
Embeddings

	AI	Cyber Security	Flutter	Javascript	Blockchain
SRM	13	11	12	11	12
BBD	13	4	31	13	22
Goel	11	4	31	13	31
Amity	31	23	33	33	42
Lucknow University	11	12	12	14	55

	AI	Cyber Security	Flutter	Javascript	Blockchain
SRM	13	11	12	11	12
BBD	13	4	31	13	22
Goel	11	4	31	13	31
Amity	31	23	33	33	42
Lucknow University	11	12	12	14	55

6 x 4 Matrix

	AI	Cyber Security	Flutter	Javascript	Blockchain
SRM	13	11	12	11	12
BBD	13	4	31	13	22
Goel	11	4	31	13	31
Amity	31	23	33	33	42
Lucknow University	11	12	12	14	55



	AI	Cyber Security	Flutter	Javascript	Blockchain
SRM	13	11	12	11	12
BBD	13	4	31	13	22
Goel	11	4	31	13	31
Amity	31	23	33	33	42
Lucknow University	11	12	12	14	55

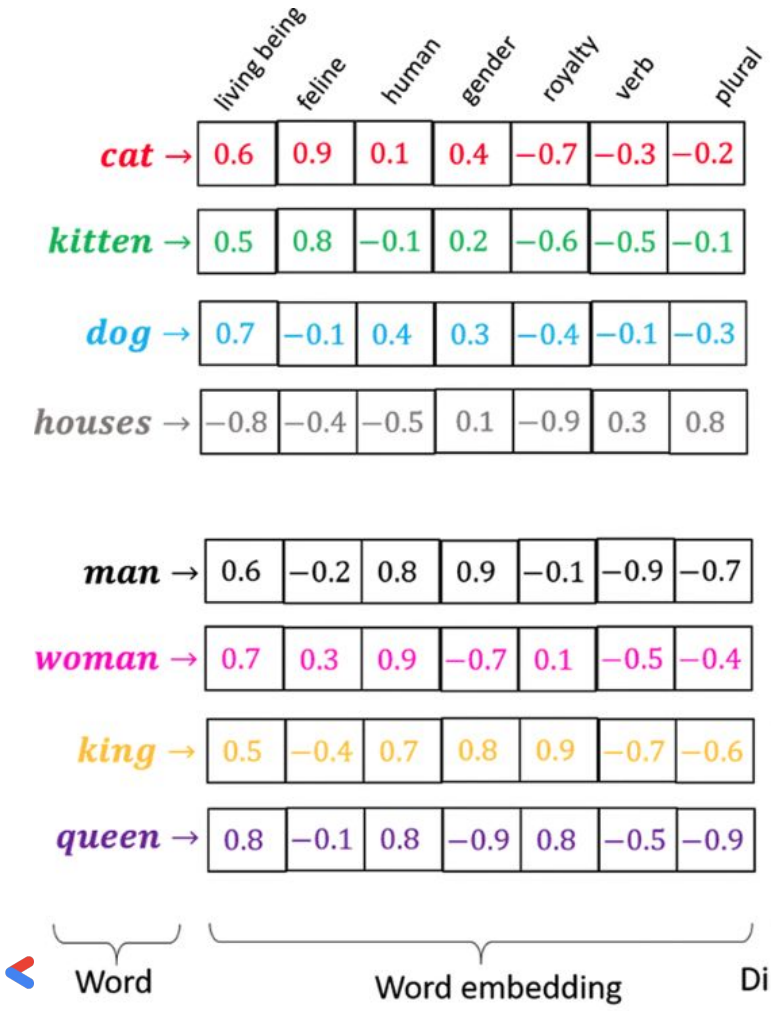
```
SRM = [ 13, 11, 12, 11, 12 ]
```

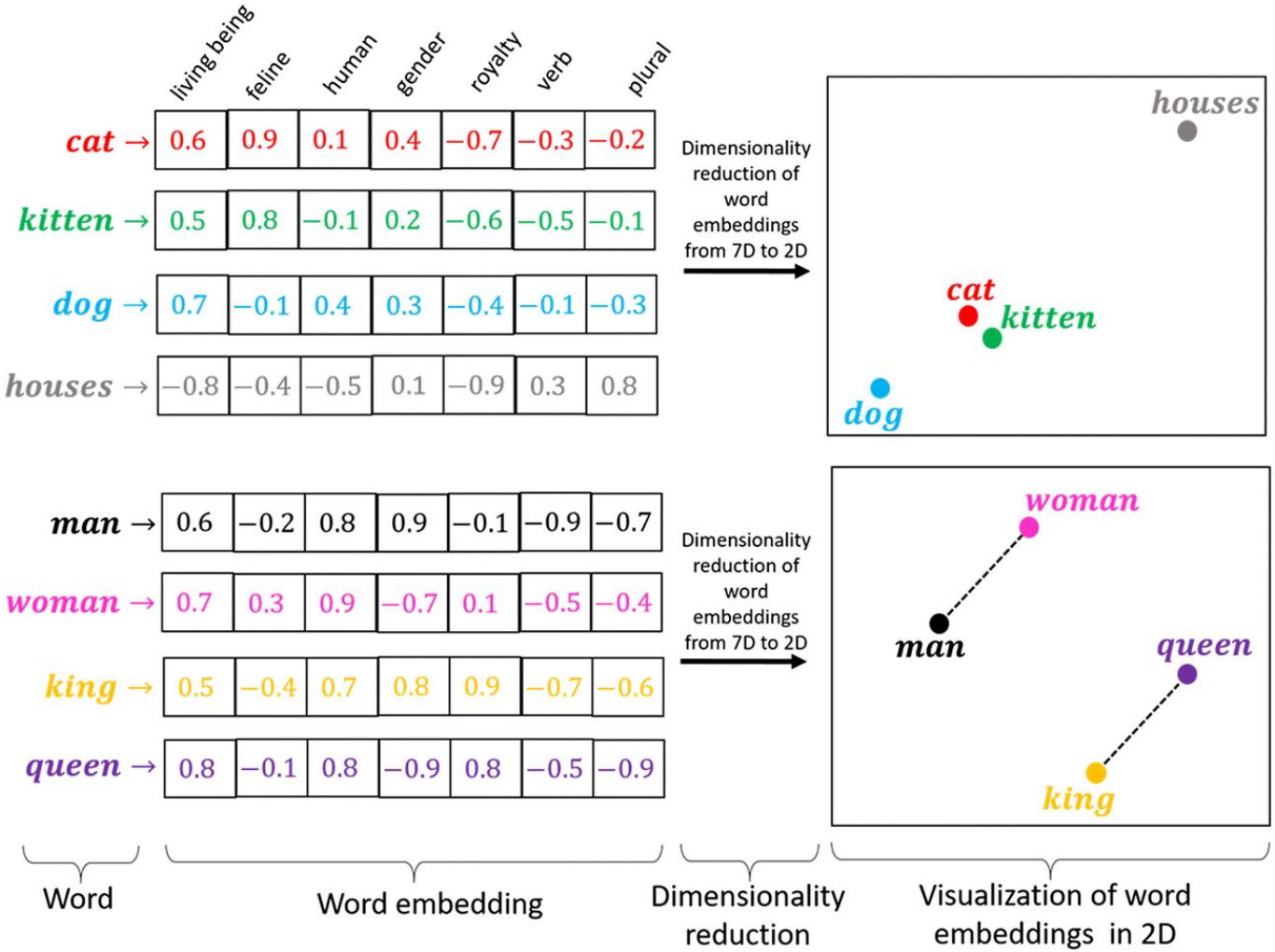
```
BBD = [ 13, 4, 31, 13, 22 ]
```

	AI	Cyber Security	Flutter	Javascript	Blockchain
SRM	13	11	12	11	12
BBD	13	4	31	13	22
Goel	11	4	31	13	31
Amity	31	23	33	33	42
Lucknow University	11	12	12	14	55

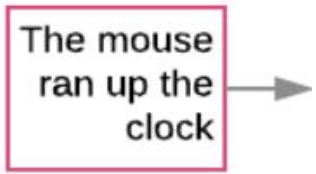
EMBEDDING MATRIX OF Colleges with 5 dimensions

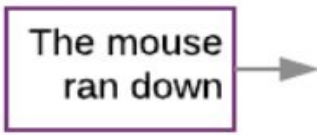
Columns are the Dim of matrix

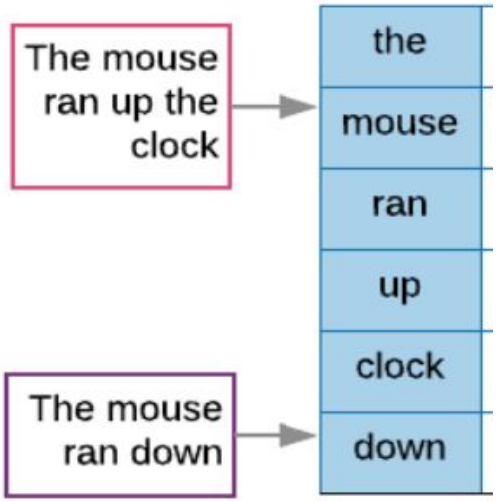

living being
feline
human
gender
royalty
verb
plural
cat → 0.6 0.9 0.1 0.4 −0.7 −0.3 −0.2
kitten → 0.5 0.8 −0.1 0.2 −0.6 −0.5 −0.1
dog → 0.7 −0.1 0.4 0.3 −0.4 −0.1 −0.3
houses → −0.8 −0.4 −0.5 0.1 −0.9 0.3 0.8
man → 0.6 −0.2 0.8 0.9 −0.1 −0.9 −0.7
woman → 0.7 0.3 0.9 −0.7 0.1 −0.5 −0.4
king → 0.5 −0.4 0.7 0.8 0.9 −0.7 −0.6
queen → 0.8 −0.1 0.8 −0.9 0.8 −0.5 −0.9
Word
Word embedding
Di


living being
feline
human
gender
royalty
verb
plural
cat → 0.6 0.9 0.1 0.4 −0.7 −0.3 −0.2
kitten → 0.5 0.8 −0.1 0.2 −0.6 −0.5 −0.1
dog → 0.7 −0.1 0.4 0.3 −0.4 −0.1 −0.3
houses → −0.8 −0.4 −0.5 0.1 −0.9 0.3 0.8
Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
man → 0.6 −0.2 0.8 0.9 −0.1 −0.9 −0.7
woman → 0.7 0.3 0.9 −0.7 0.1 −0.5 −0.4
king → 0.5 −0.4 0.7 0.8 0.9 −0.7 −0.6
queen → 0.8 −0.1 0.8 −0.9 0.8 −0.5 −0.9
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

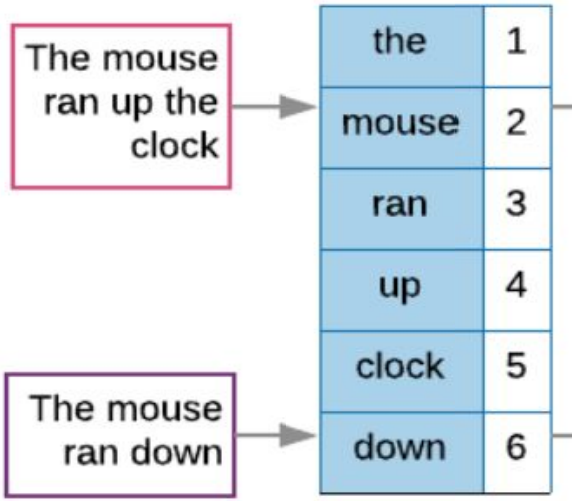
How to create the Embeddings?


The mouse ran up the clock

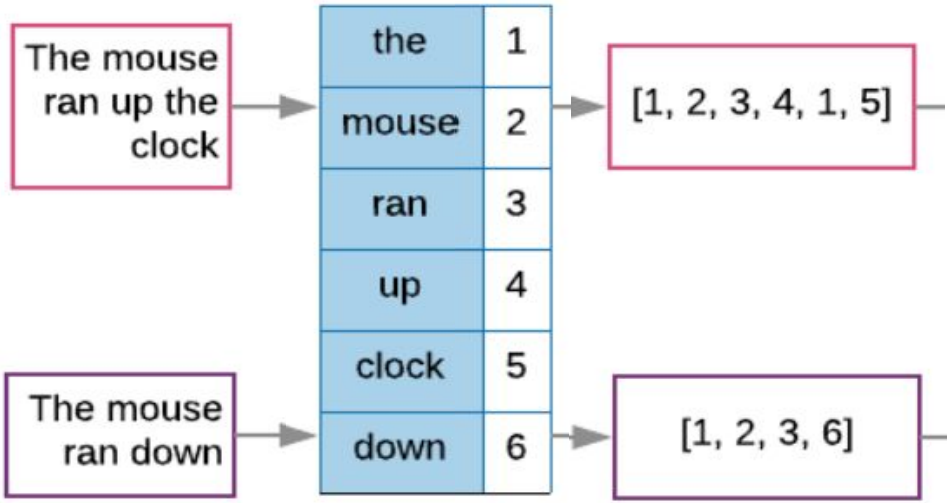

The mouse ran down



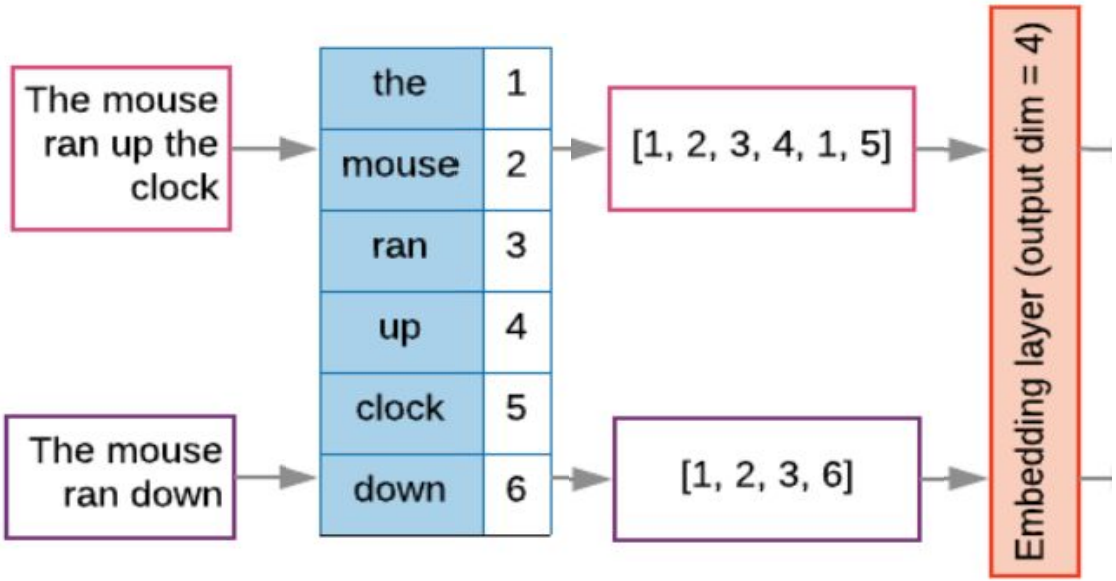
Step 1 : Creating Vocabulary of all the unique words



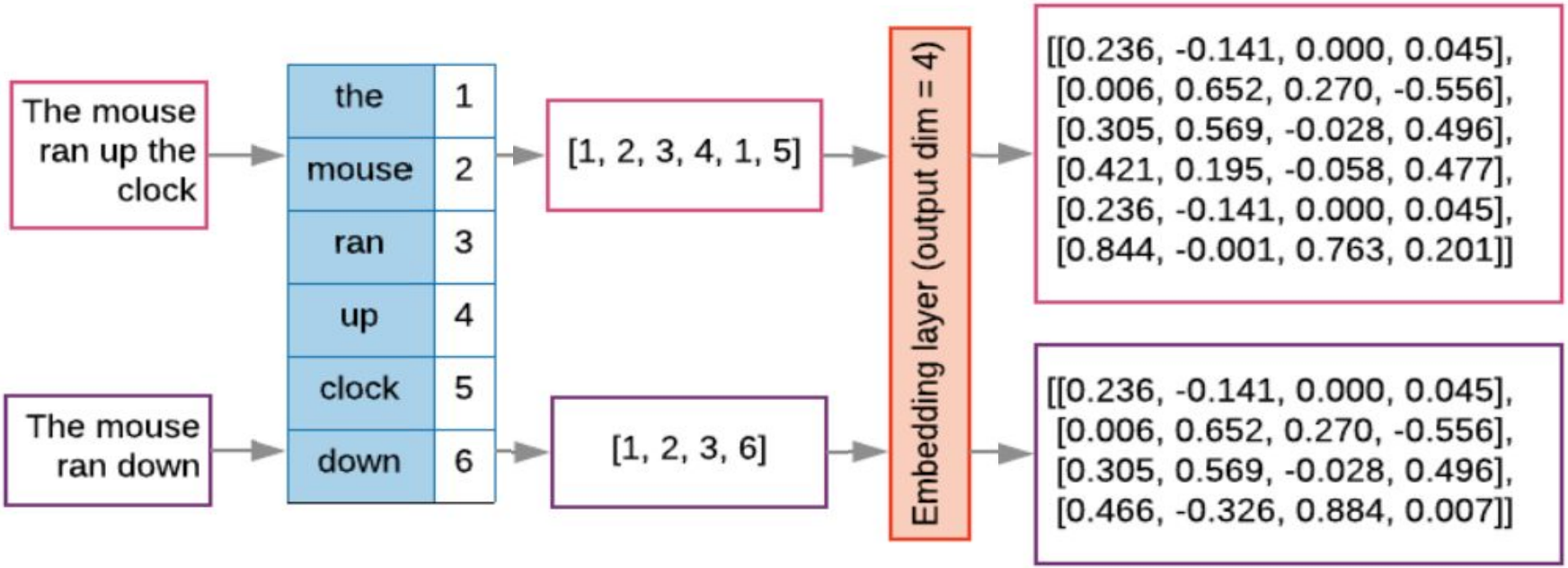
Step 2 : Assigning index numbers to each unique word



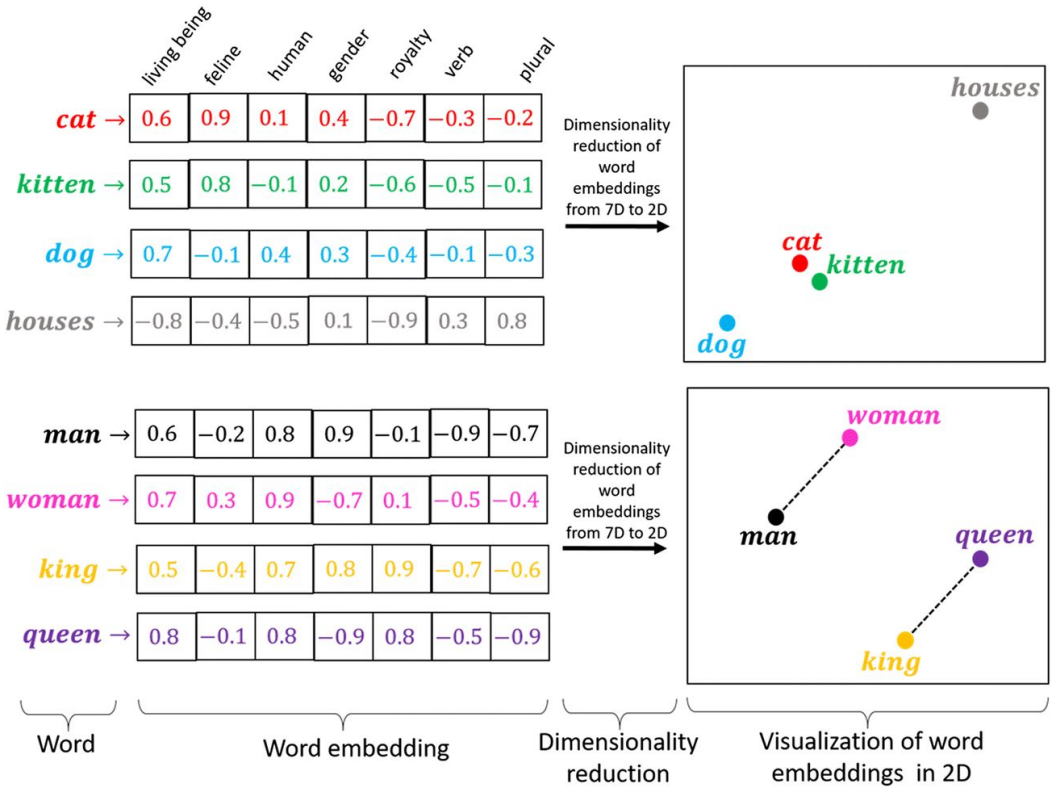
Step 3 : Mapping original words with index numbers

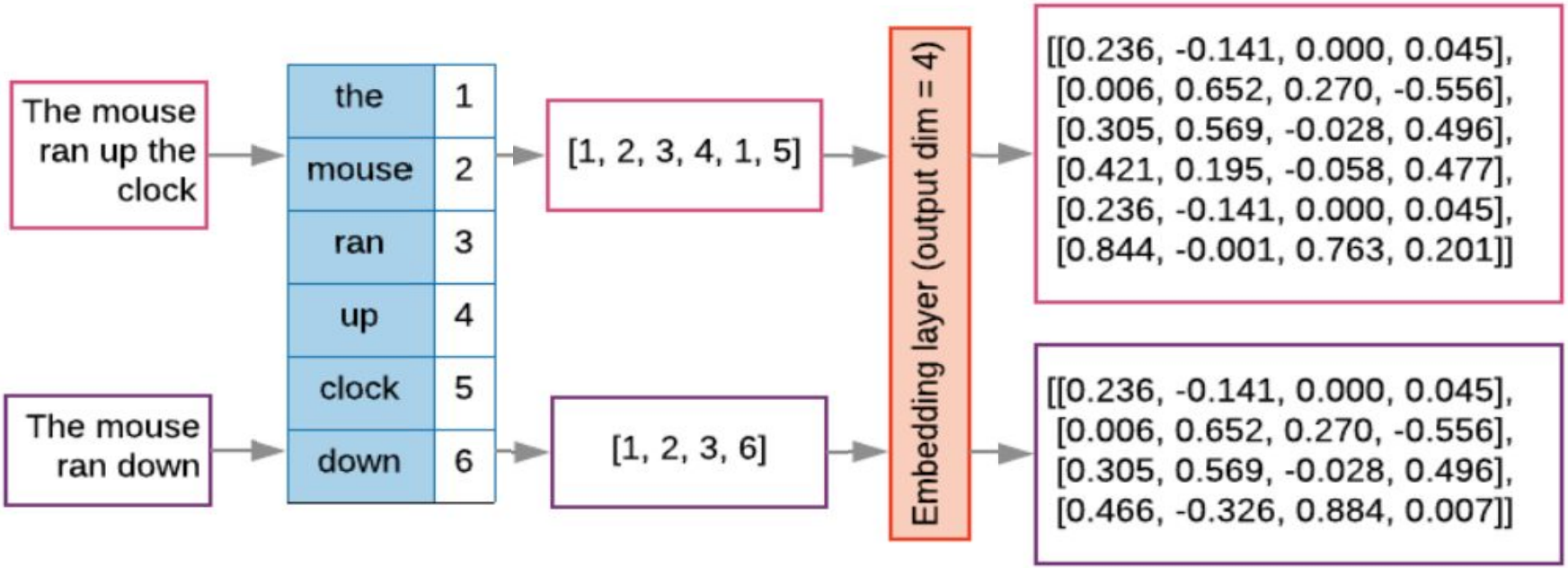


Step 4 : Sending mapped numbers Embedder for embeddings


The mouse ran up the clock
The mouse ran down
the 1
mouse 2
ran 3
up 4
clock 5
down 6
[1, 2, 3, 4, 1, 5]
[1, 2, 3, 6]
Embedding layer (output dim = 4)
[[0.236, -0.141, 0.000, 0.045],
[0.006, 0.652, 0.270, -0.556],
[0.305, 0.569, -0.028, 0.496],
[0.421, 0.195, -0.058, 0.477],
[0.236, -0.141, 0.000, 0.045],
[0.844, -0.001, 0.763, 0.201]]
[[0.236, -0.141, 0.000, 0.045],
[0.006, 0.652, 0.270, -0.556],
[0.305, 0.569, -0.028, 0.496],
[0.466, -0.326, 0.884, 0.007]]

Embedder


living being
feline
human
gender
royalty
verb
plural
cat → 0.6 0.9 0.1 0.4 −0.7 −0.3 −0.2
kitten → 0.5 0.8 −0.1 0.2 −0.6 −0.5 −0.1
dog → 0.7 −0.1 0.4 0.3 −0.4 −0.1 −0.3
houses → −0.8 −0.4 −0.5 0.1 −0.9 0.3 0.8
Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
man → 0.6 −0.2 0.8 0.9 −0.1 −0.9 −0.7
woman → 0.7 0.3 0.9 −0.7 0.1 −0.5 −0.4
king → 0.5 −0.4 0.7 0.8 0.9 −0.7 −0.6
queen → 0.8 −0.1 0.8 −0.9 0.8 −0.5 −0.9
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D


The mouse ran up the clock
The mouse ran down
the 1
mouse 2
ran 3
up 4
clock 5
down 6
[1, 2, 3, 4, 1, 5]
[1, 2, 3, 6]
Embedding layer (output dim = 4)
[[0.236, -0.141, 0.000, 0.045],
[0.006, 0.652, 0.270, -0.556],
[0.305, 0.569, -0.028, 0.496],
[0.421, 0.195, -0.058, 0.477],
[0.236, -0.141, 0.000, 0.045],
[0.844, -0.001, 0.763, 0.201]]
[[0.236, -0.141, 0.000, 0.045],
[0.006, 0.652, 0.270, -0.556],
[0.305, 0.569, -0.028, 0.496],
[0.466, -0.326, 0.884, 0.007]]

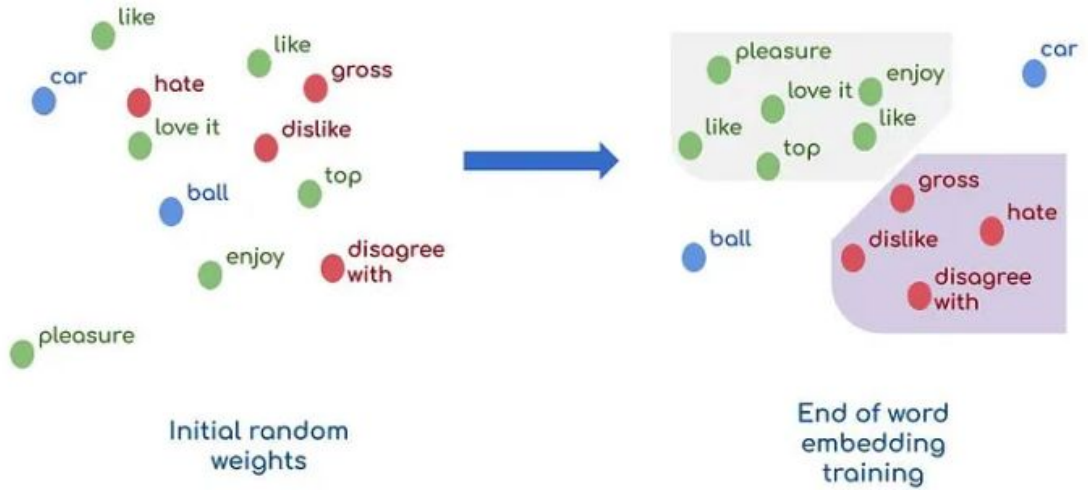
A 4-dimensional embedding

cat =>	1.2	-0.1	4.3	3.2
mat =>	0.4	2.5	-0.9	0.5
on =>	2.1	0.3	0.1	0.4

Columns are the Dim of matrix

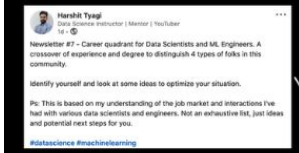
Let’s Implement Embeddings Code v1 & v2

Embeddings



by Zeineb Ghrib

ML models Leveraging Linear Algebra


Harshit Tyagi
Data Science Instructor | Mentor | YouTuber
1d •

Newsletter #7 – Career quadrant for Data Scientists and ML Engineers. A crossover of experience and degree to distinguish 4 types of folks in this community.

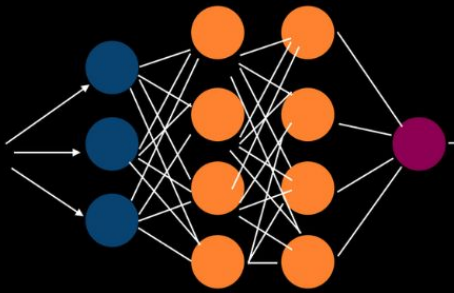
Identify yourself and look at some ideas to optimize your situation.

Ps: This is based on my understanding of the job market and interactions I've had with various data scientists and engineers. Not an exhaustive list, just ideas and potential next steps for you.

#datascience #machinelearning

[[12, 213, 44, 15],
[20, 113, 39, 10],
[13, 117, 45, 13],
[25, 120, 49, 14],
…,
…,
]

Tensor



[[0.98, 0.45, 0.44, 0.95],
[0.23, 0.13, 0.39, 0.89],
[0.13, 0.77, 0.55, 0.23],
[0.25, 0.20, 0.49, 0.14],
…,
…,
]

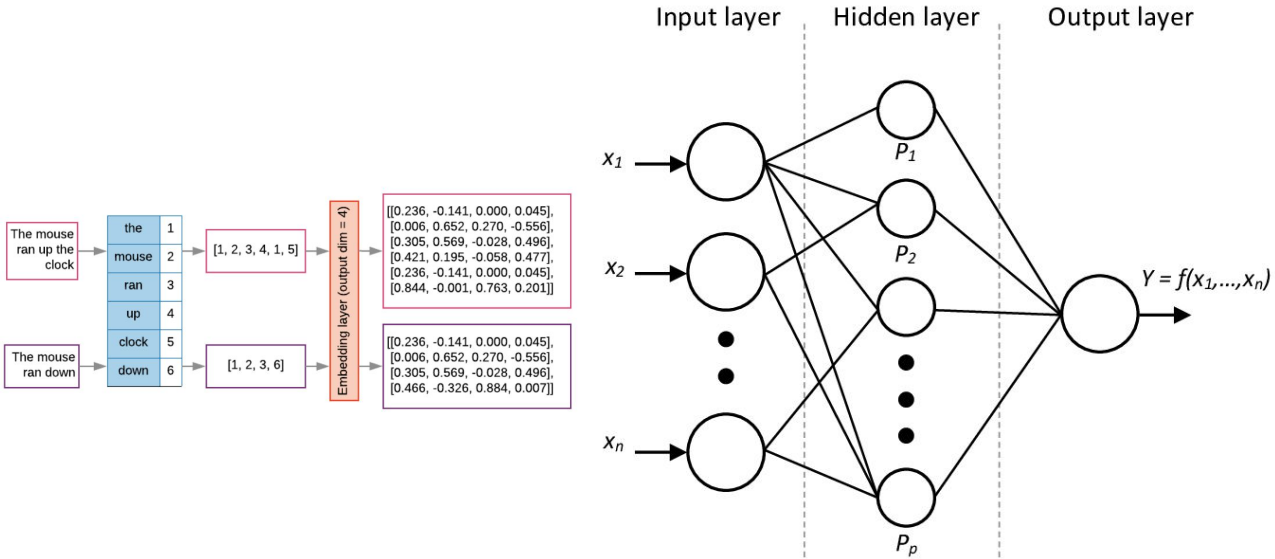
Tensor

Attend the session!

Atelactasis

Ok Google, set alarm for 5 AM.

Feeding Embeddings to Neural Networks

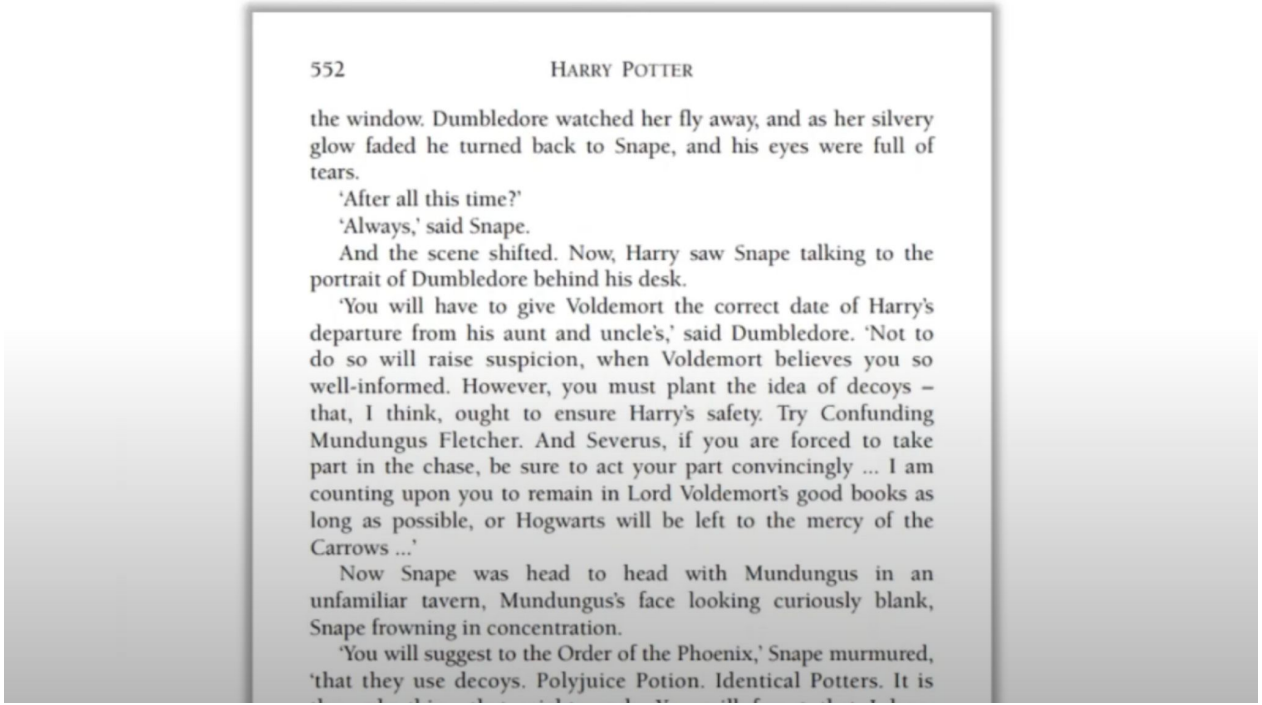


Feedforward neural network Code walk-through

Transformers

Attention Mechanism

Attention in Human Visual Processing System



552 HARRY POTTER

the window. Dumbledore watched her fly away, and as her silvery glow faded he turned back to Snape, and his eyes were full of tears.

'After all this time?'

'Always,' said Snape.

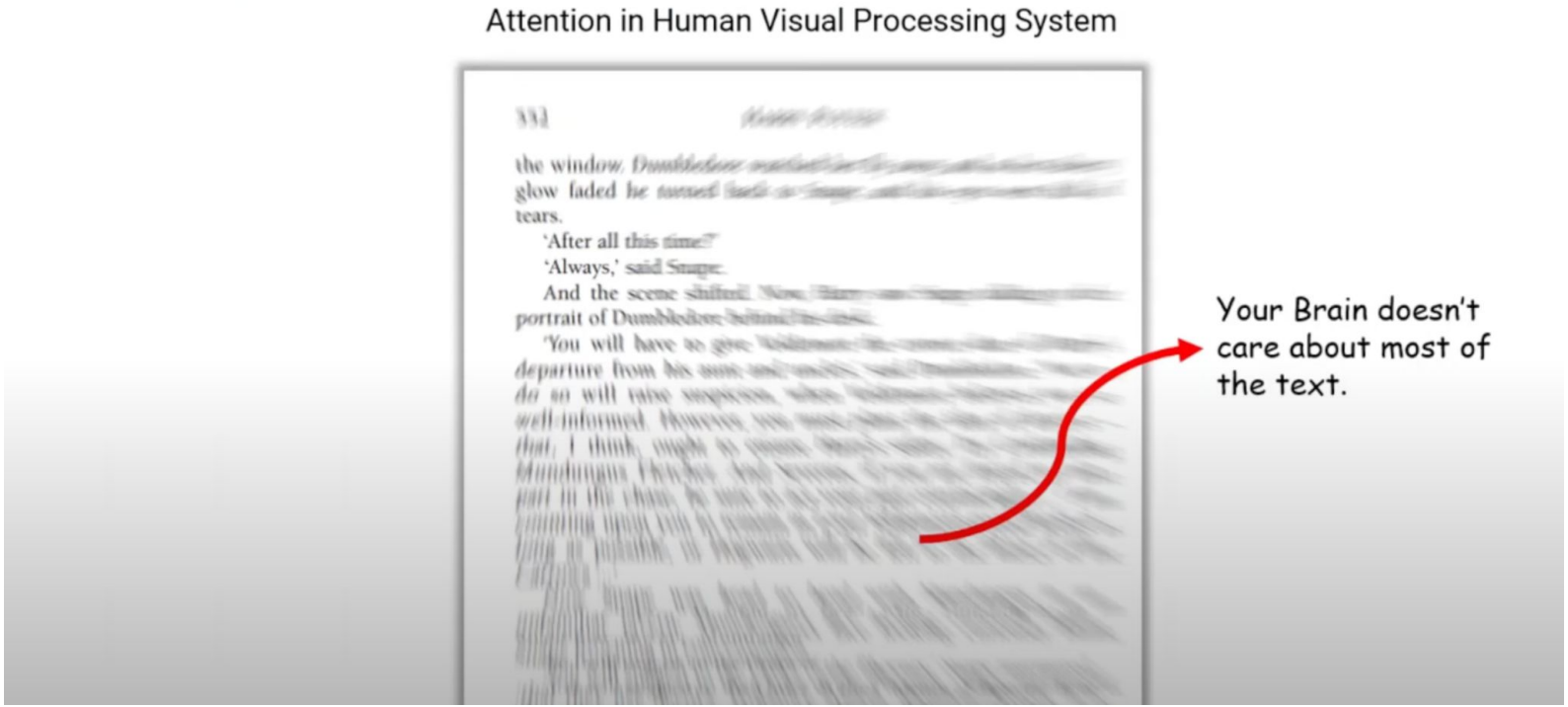
And the scene shifted. Now, Harry saw Snape talking to the portrait of Dumbledore behind his desk.

'You will have to give Voldemort the correct date of Harry's departure from his aunt and uncle's,' said Dumbledore. 'Not to do so will raise suspicion, when Voldemort believes you so well-informed. However, you must plant the idea of decoys – that, I think, ought to ensure Harry's safety. Try Confunding Mundungus Fletcher. And Severus, if you are forced to take part in the chase, be sure to act your part convincingly ... I am counting upon you to remain in Lord Voldemort's good books as long as possible, or Hogwarts will be left to the mercy of the Carrows ...'

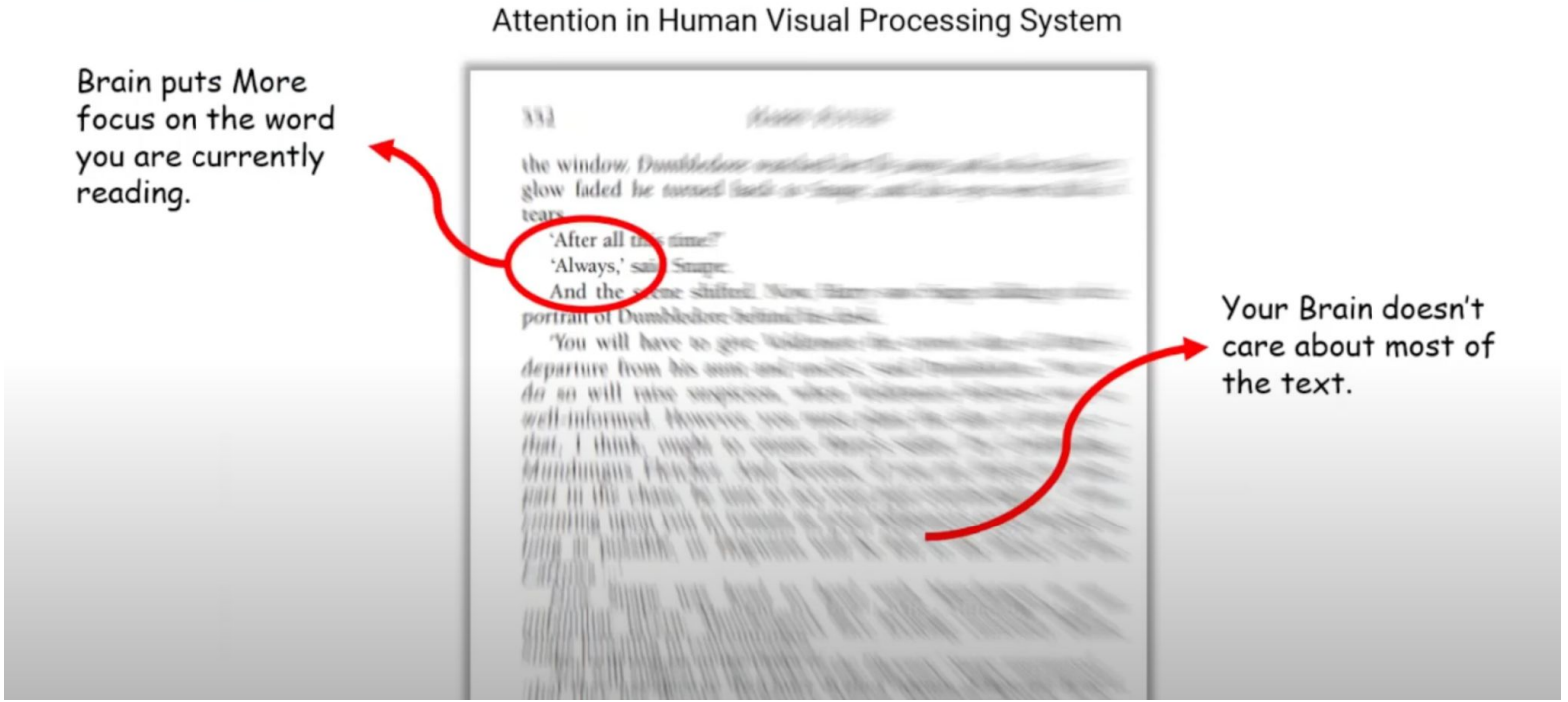
Now Snape was head to head with Mundungus in an unfamiliar tavern, Mundungus's face looking curiously blank, Snape frowning in concentration.

'You will suggest to the Order of the Phoenix,' Snape murmured, 'that they use decoys. Polyjuice Potion. Identical Potters. It is

Attention Mechanism

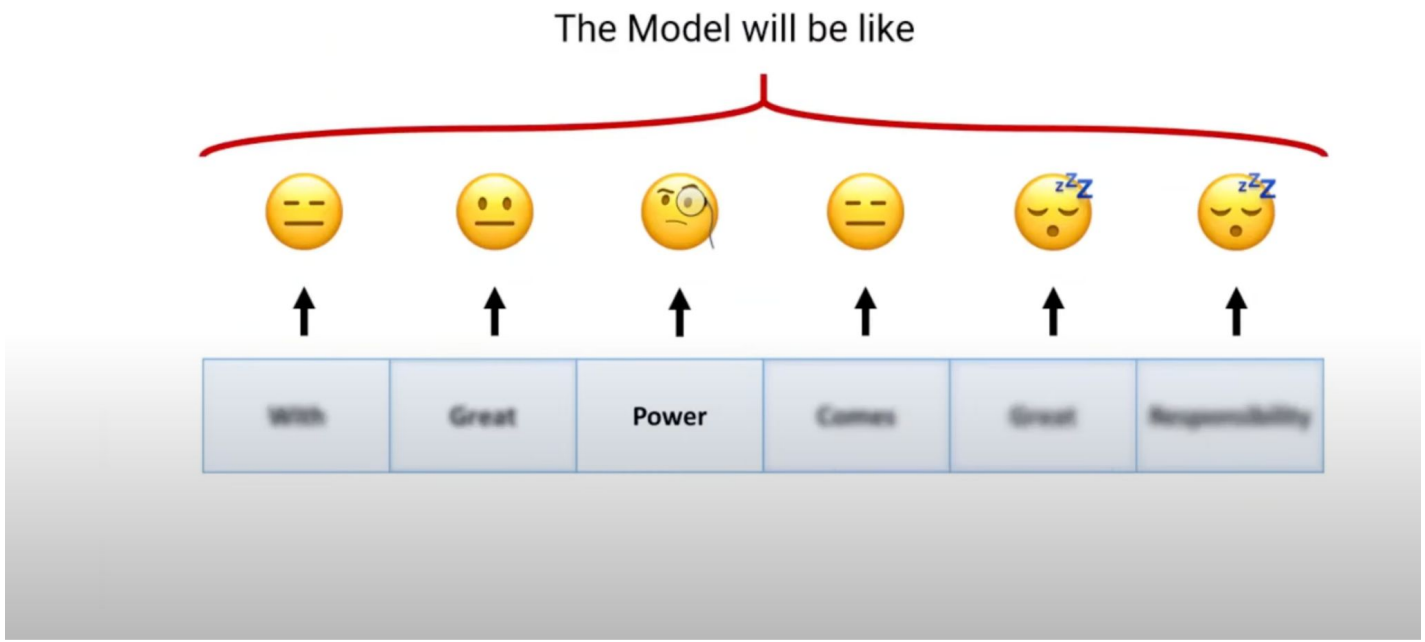


Attention Mechanism

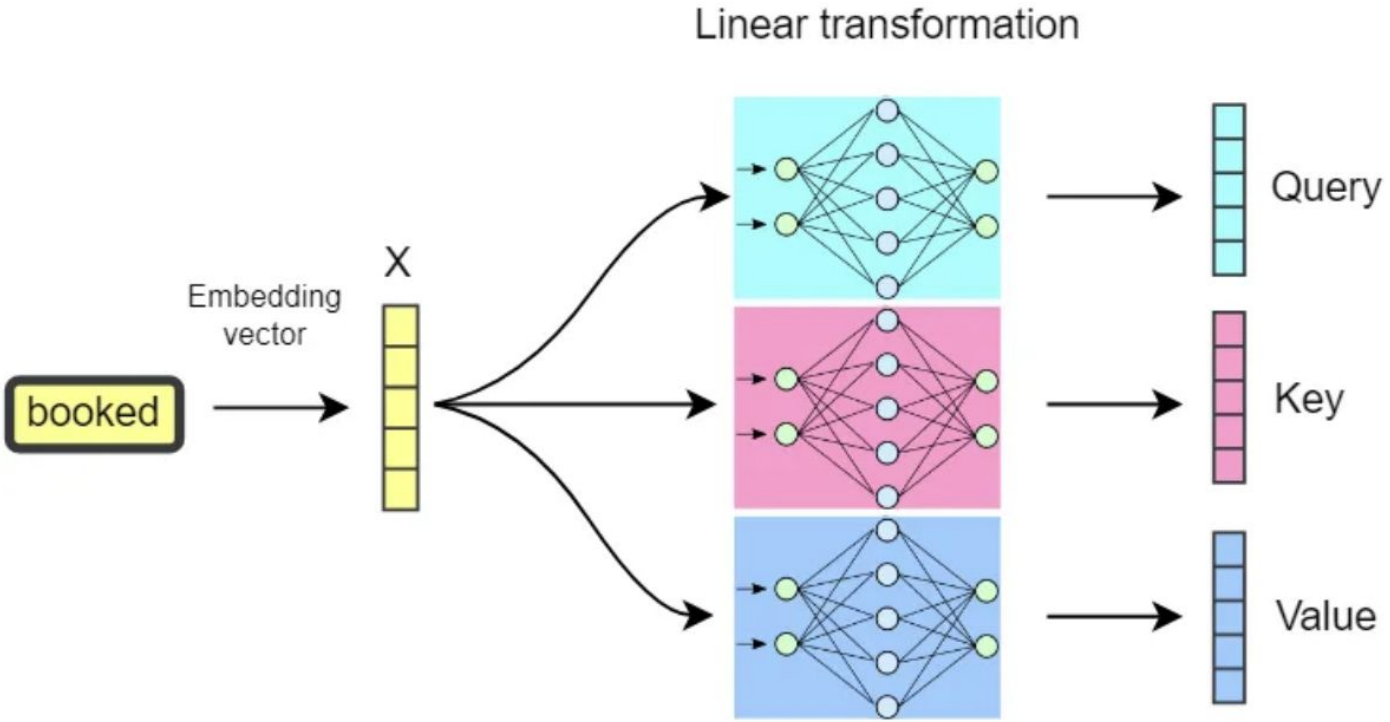


Attention Mechanism

Attention Mechanism



Attention Mechanism



Attention Mechanism

```
{("Zhang", "Aston"),
 ("Lipton", "Zachary"),
 ("Li", "Mu"),
 ("Smola", "Alex"),
 ("Hu", "Rachel"),
 ("Werness", "Brent")}
```

Last name being the key and the first name being the value.

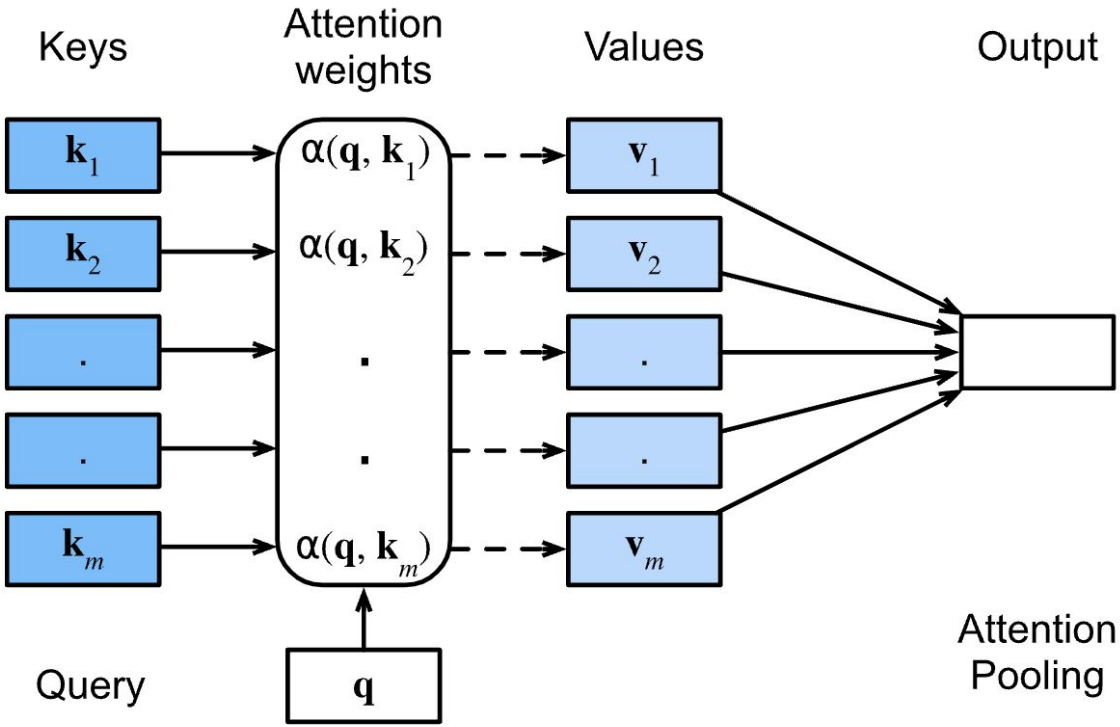
Attention Mechanism

Query (q) for “Li” which

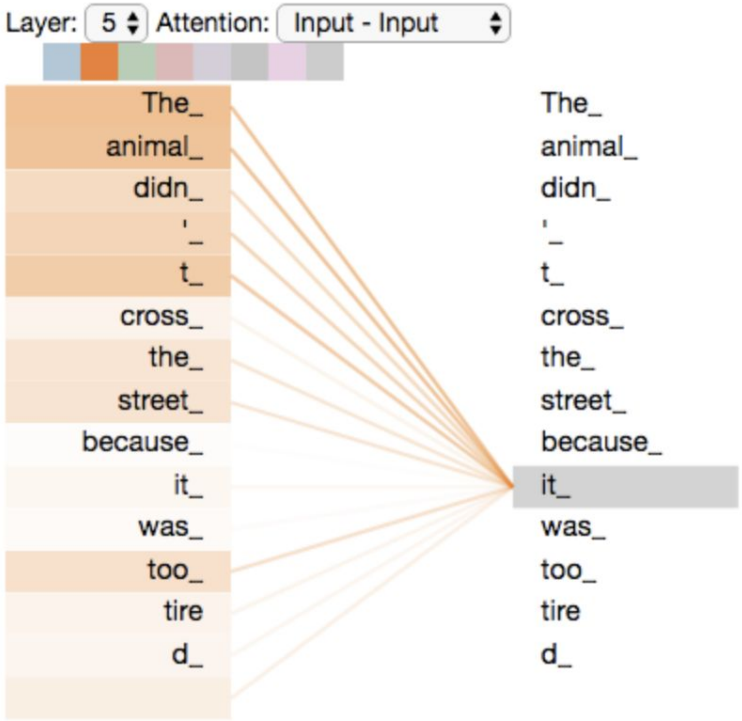
would return the value “Mu”.

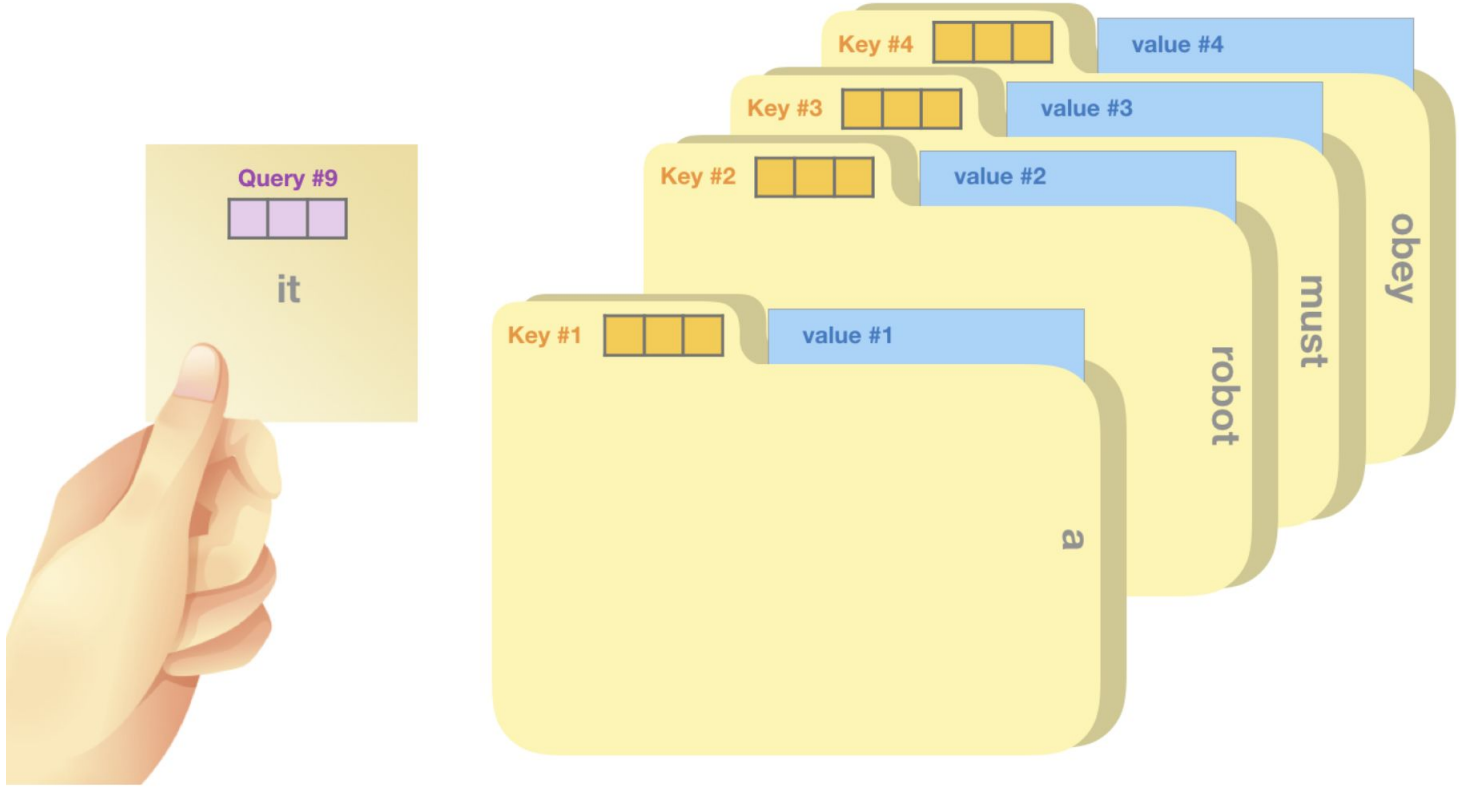
Last name being the key and the first name being the value.

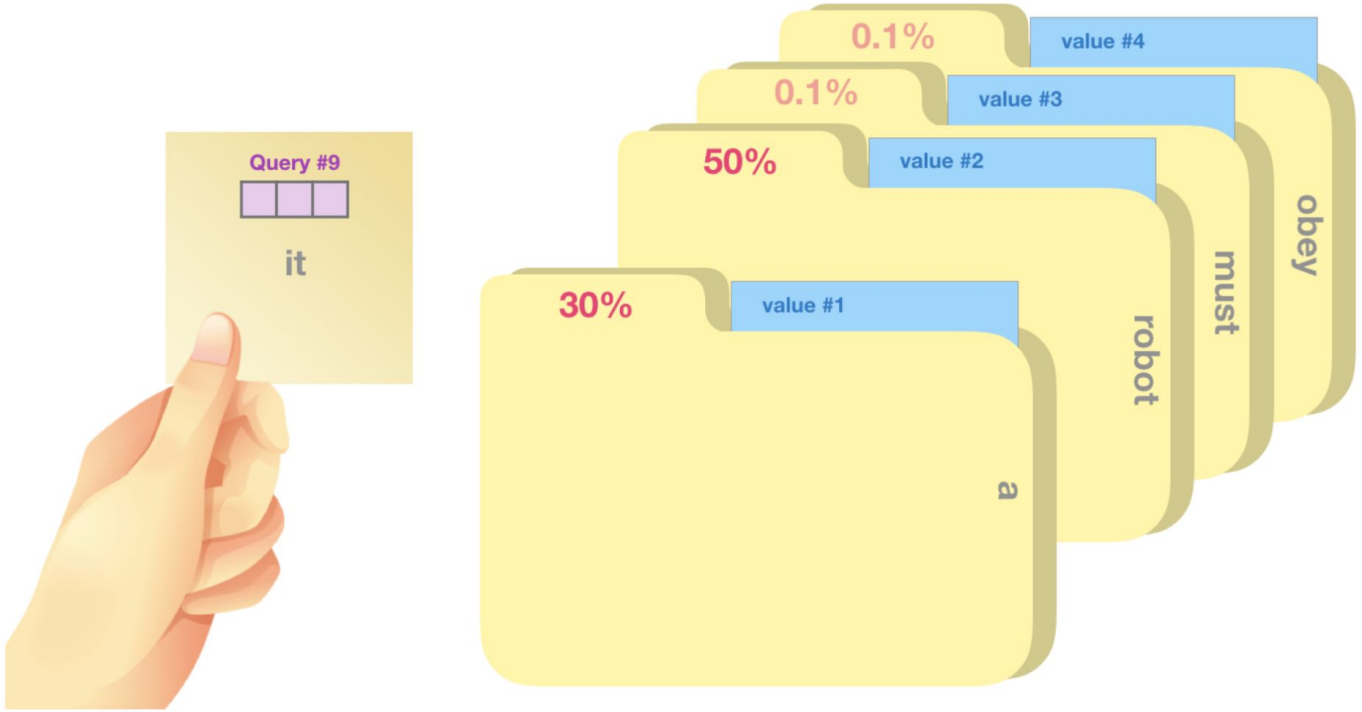
Attention Mechanism



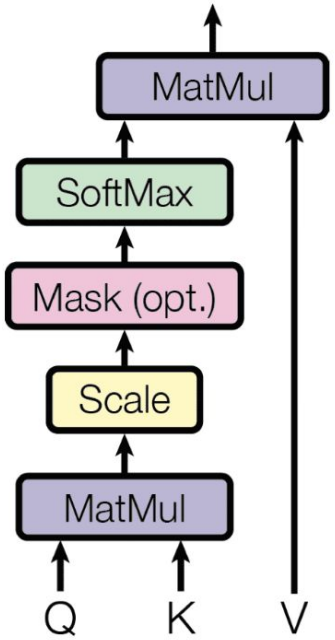
Attention!!


Layer: 5
Attention: Input - Input
The_
animal_
didn_
'_
t_
cross_
the_
street_
because_
it_
was_
too_
tire
d_
The_
animal_
didn_
'_
t_
cross_
the_
street_
because_
it_
was_
too_
tire
d_

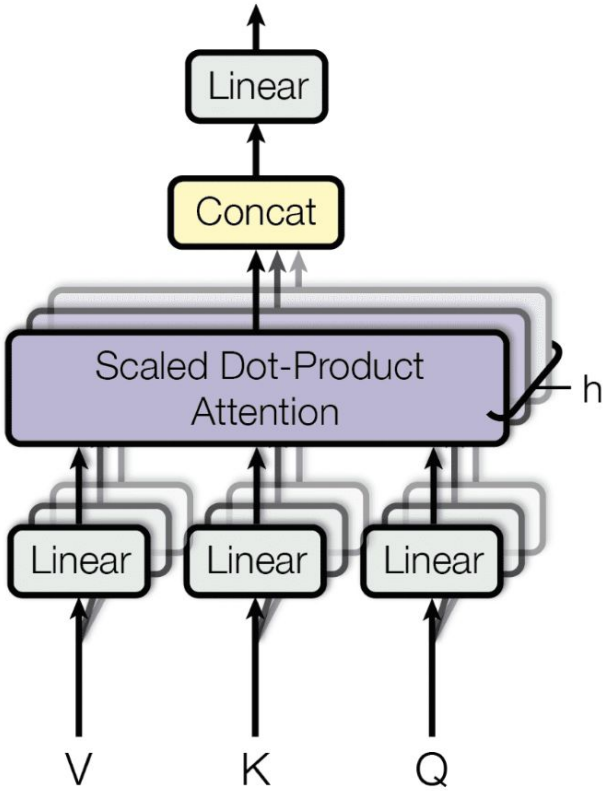

Query #9
it
Key #4
value #4
Key #3
value #3
Key #2
value #2
Key #1
value #1
obey
must
robot
a


Query #9
it
0.1%
value #4
obey
0.1%
value #3
must
50%
value #2
robot
30%
value #1
a

Scaled Dot-Product Attention



Multi-Head Attention

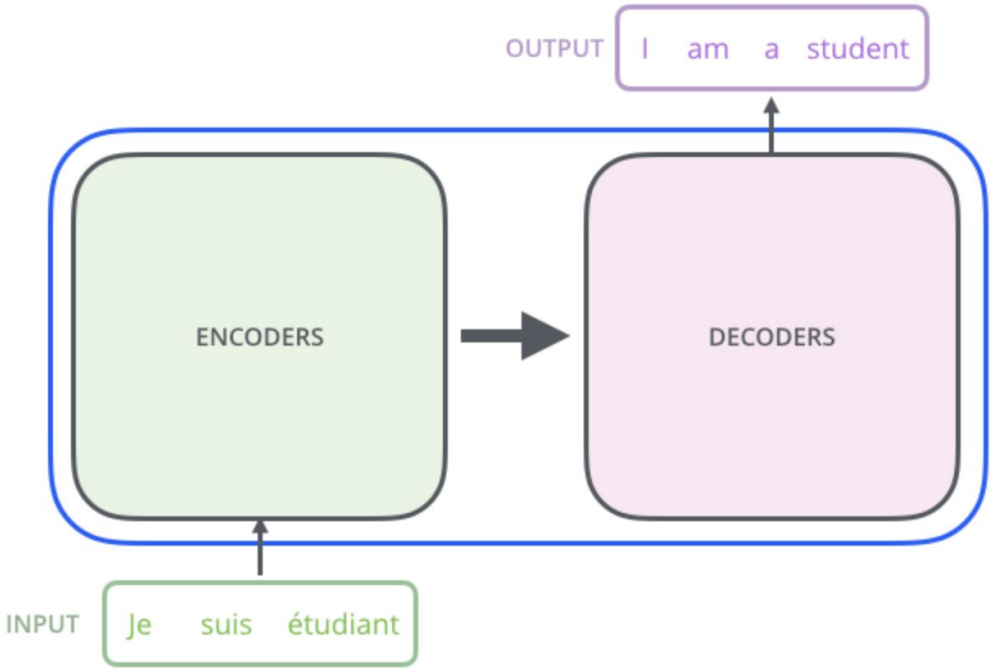


Second Law of Robotics
A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.

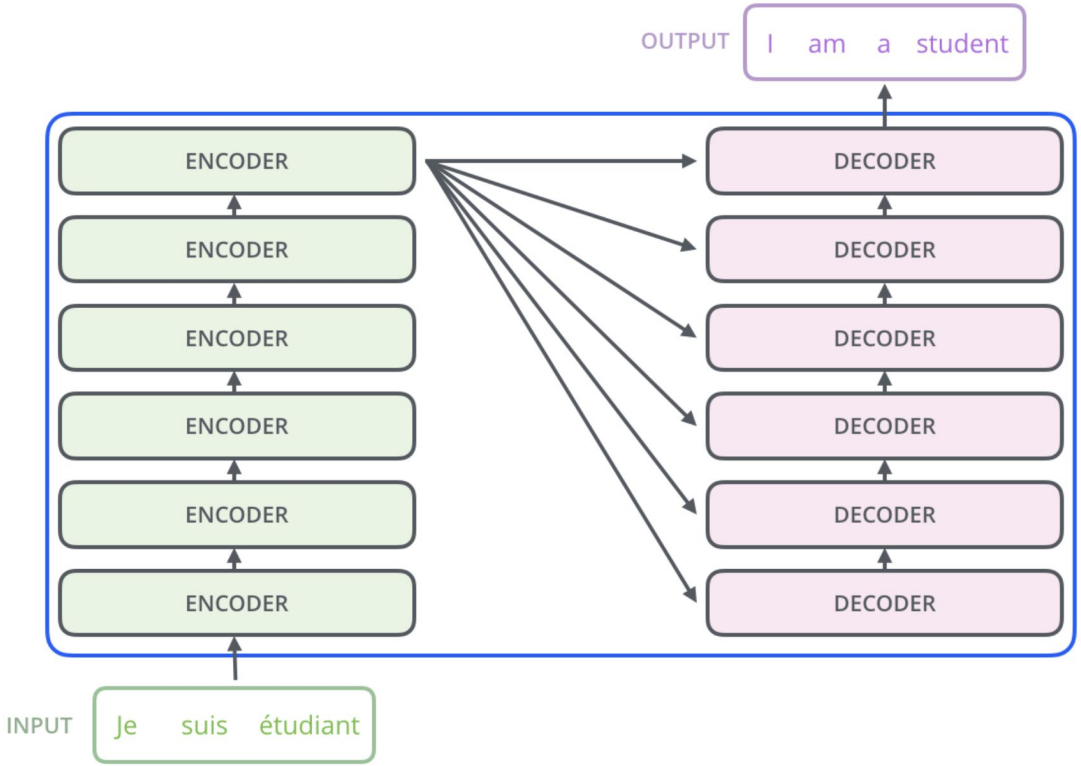
Word	Value vector	Score	Value X Score
<s>		0.001	
a		0.3	
robot		0.5	
must		0.002	
obey		0.001	
the		0.0003	
orders		0.005	
given		0.002	
it		0.19	
		Sum:	

Transformers

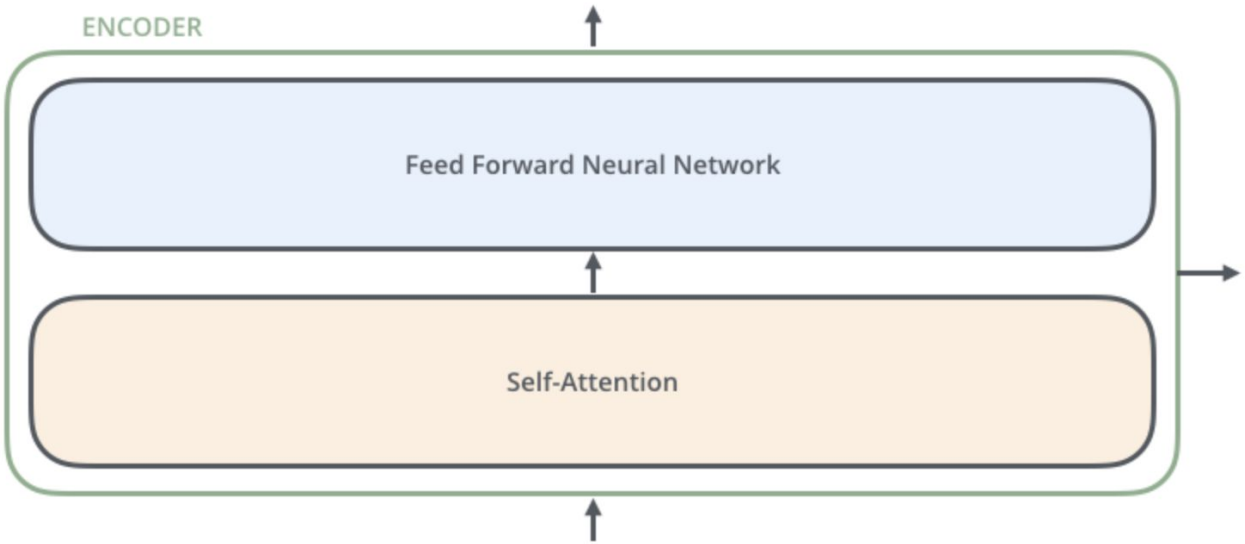
Transformers



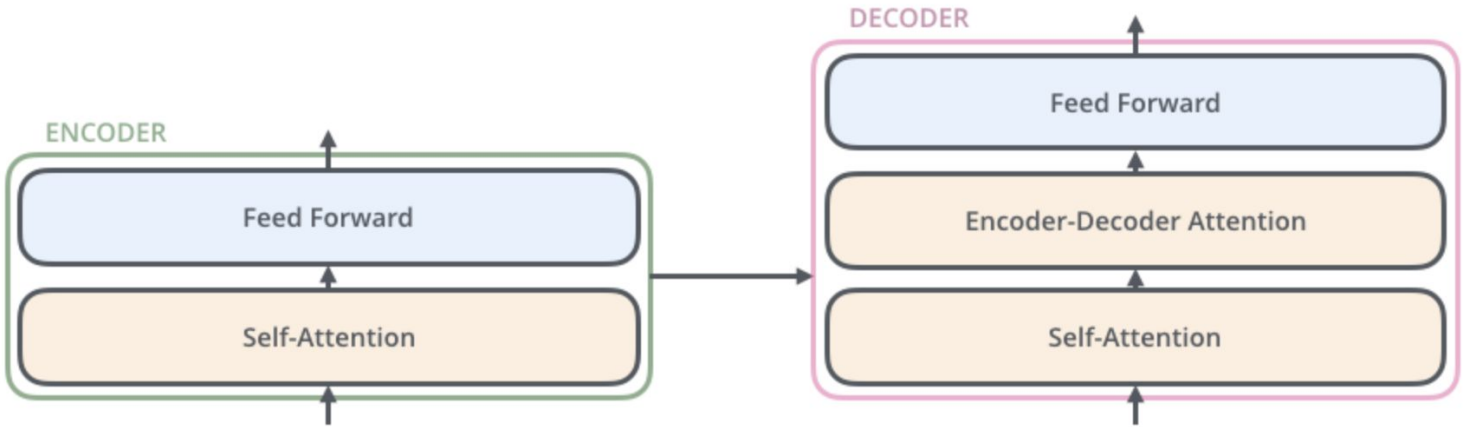
Transformers



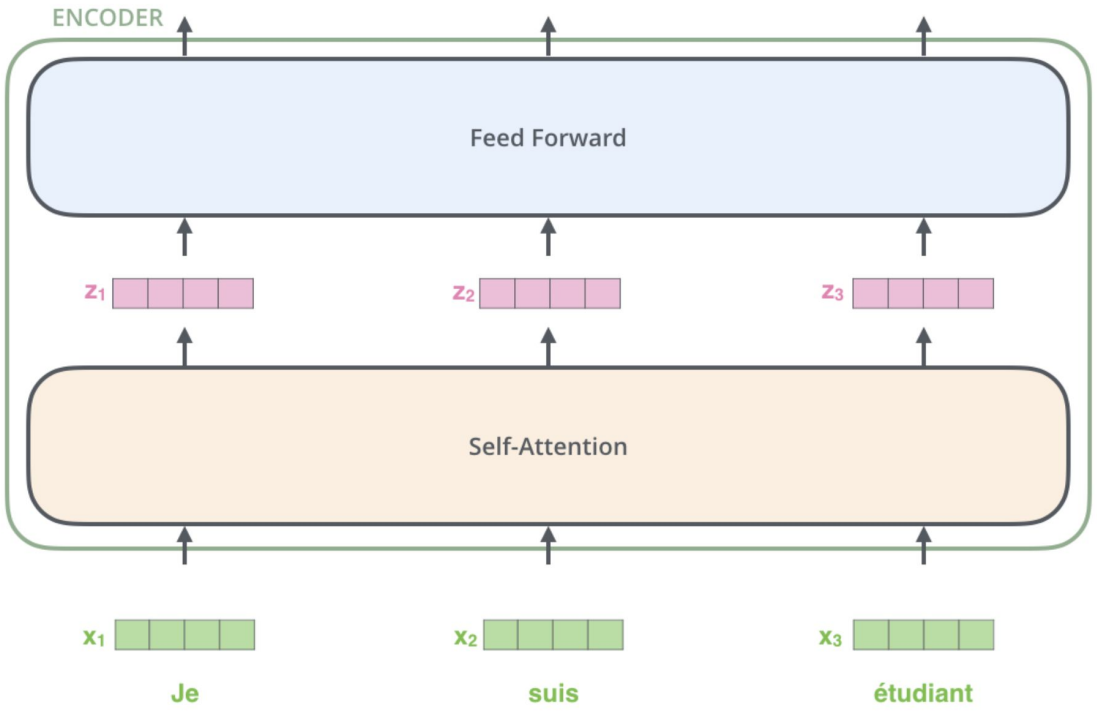
Transformers



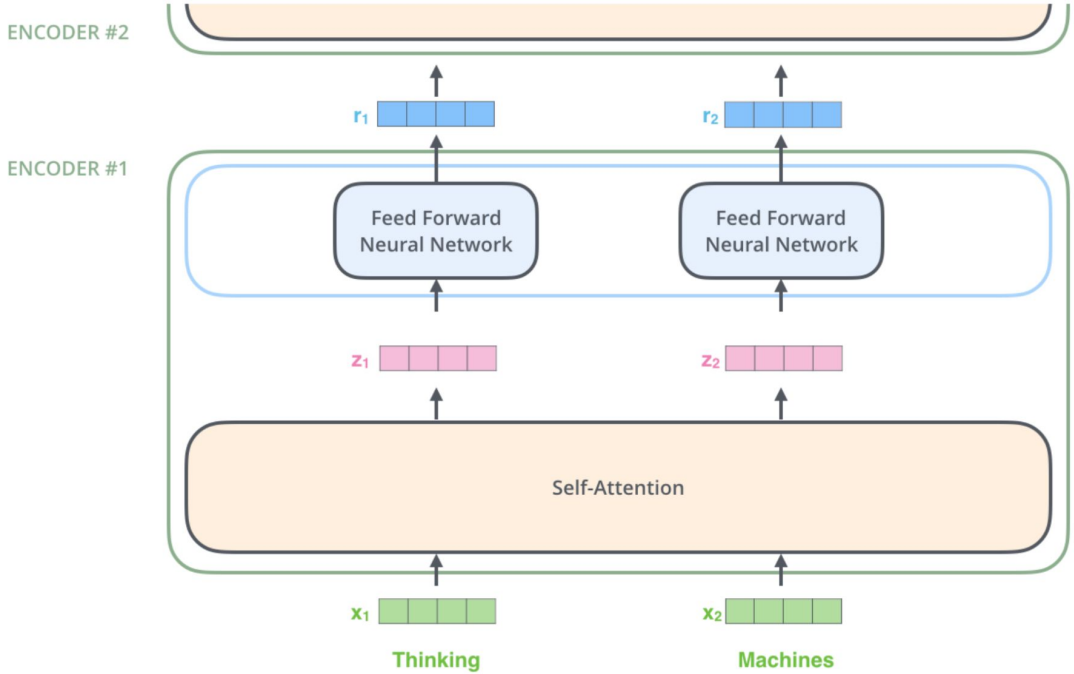
Transformers



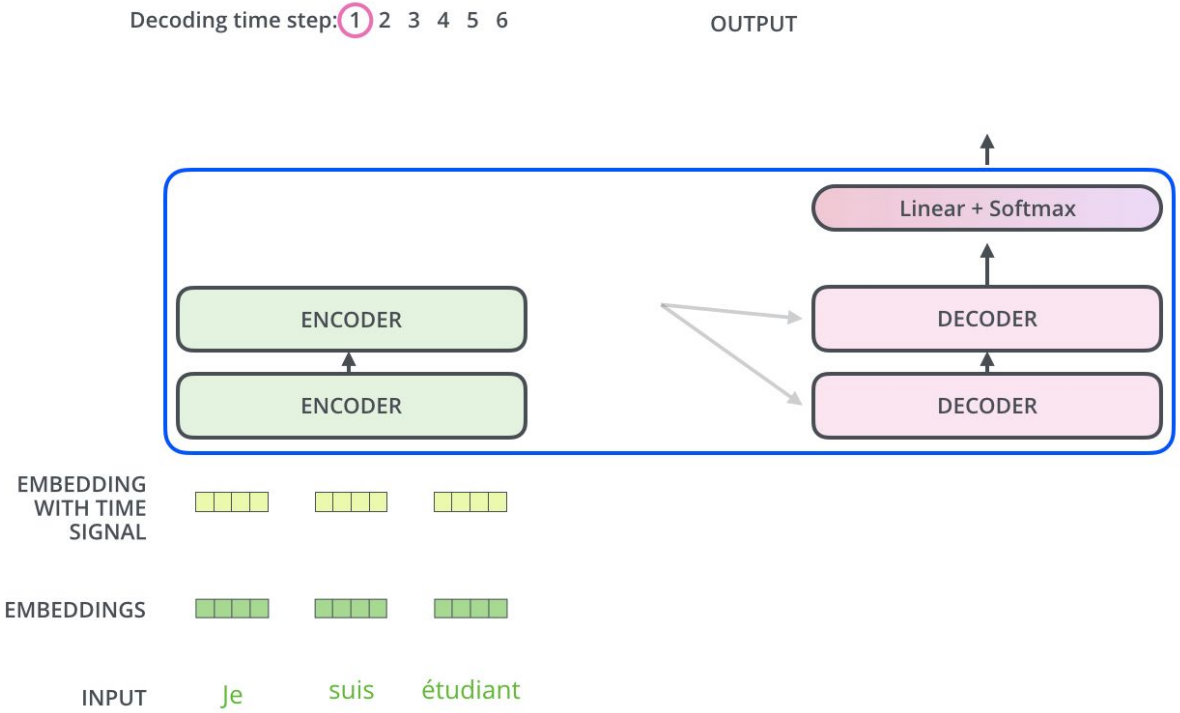
Transformers



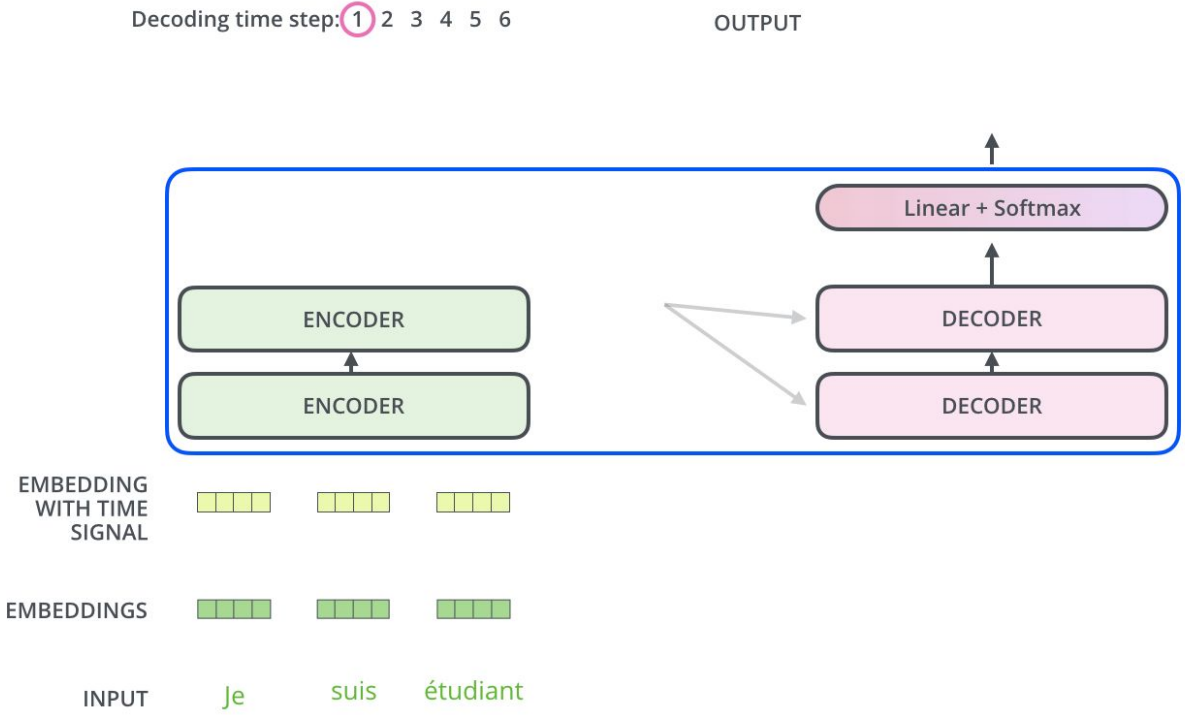
Transformers

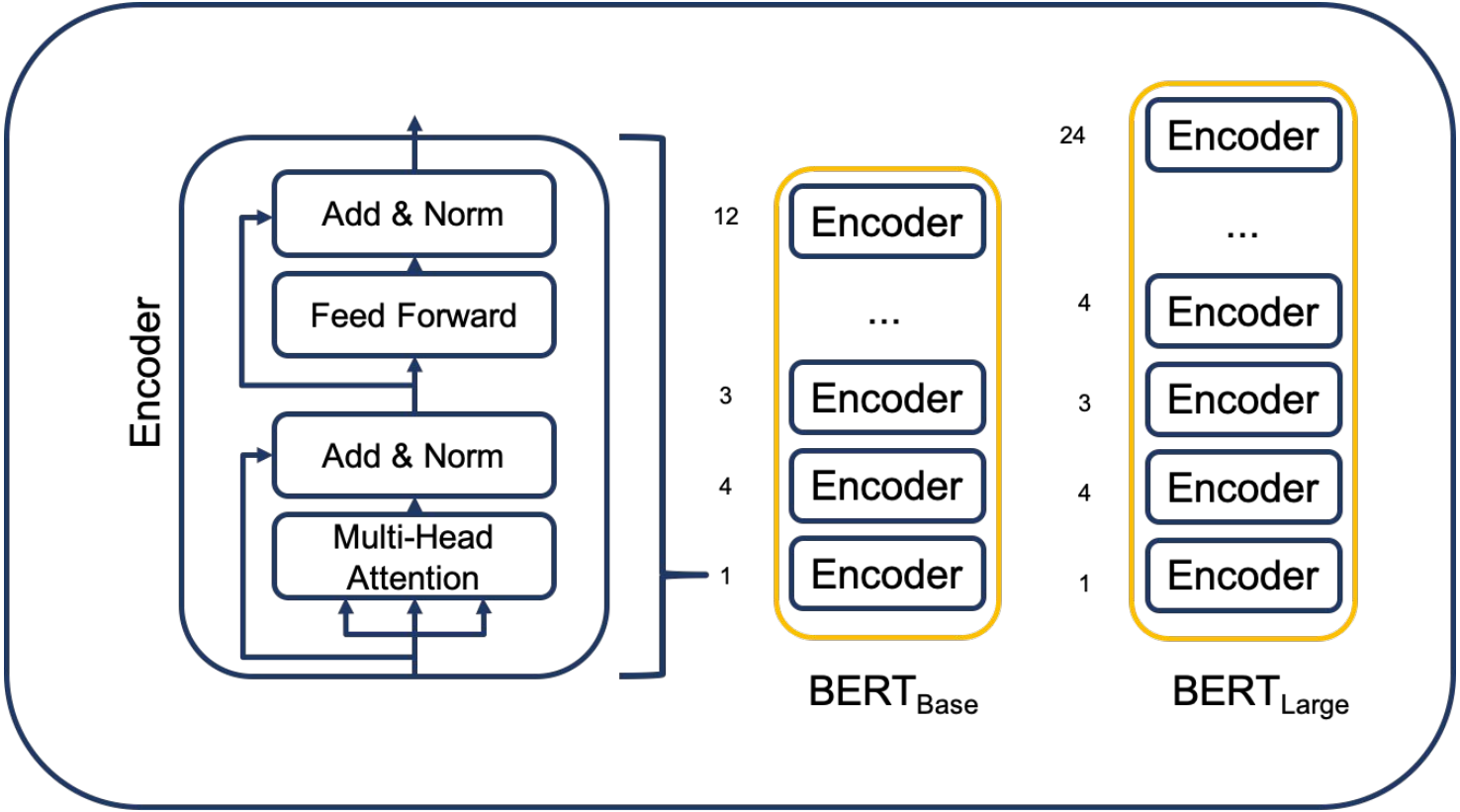


Transformers



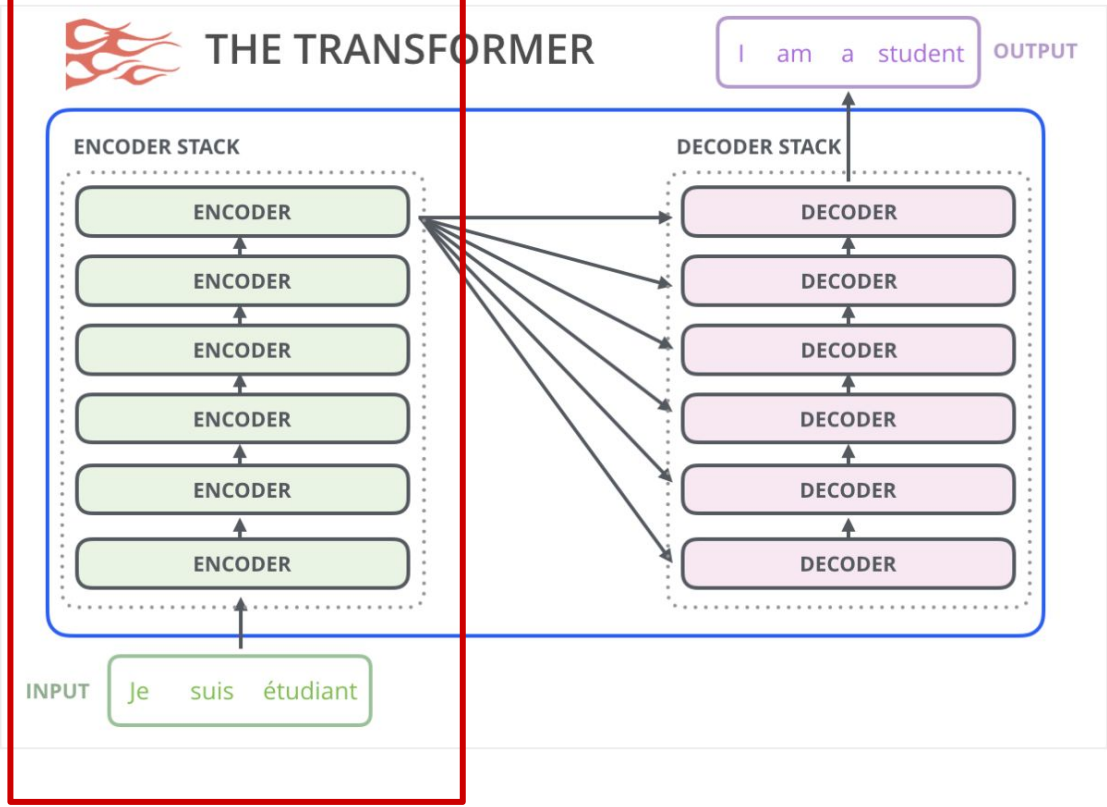
Transformers




Encoder
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
12
3
4
1
Encoder
...
Encoder
Encoder
Encoder
BERT Base
24
4
3
4
1
Encoder
...
Encoder
Encoder
Encoder
Encoder
BERT Large

Bert Model

The BERT model is trained on two tasks: Masked Language Modeling, where the objective is to predict the masked word based on the context


THE TRANSFORMER
I am a student
OUTPUT
ENCODER STACK
DECODER STACK
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
DECODER
DECODER
DECODER
DECODER
DECODER
DECODER
INPUT
Je suis étudiant

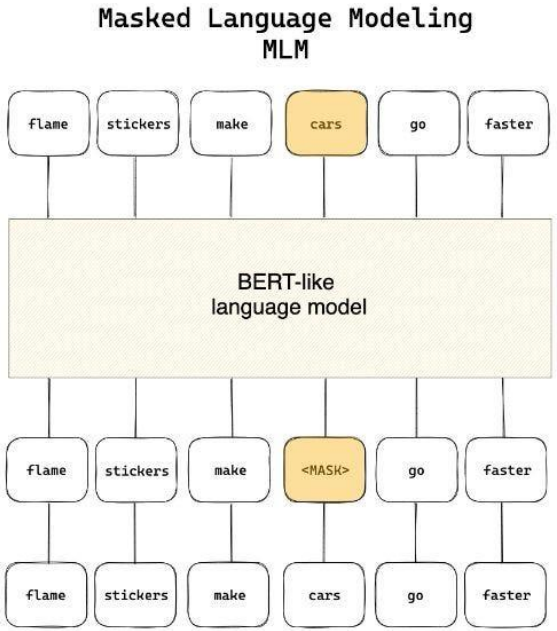
How does the BERT model learn?

Training

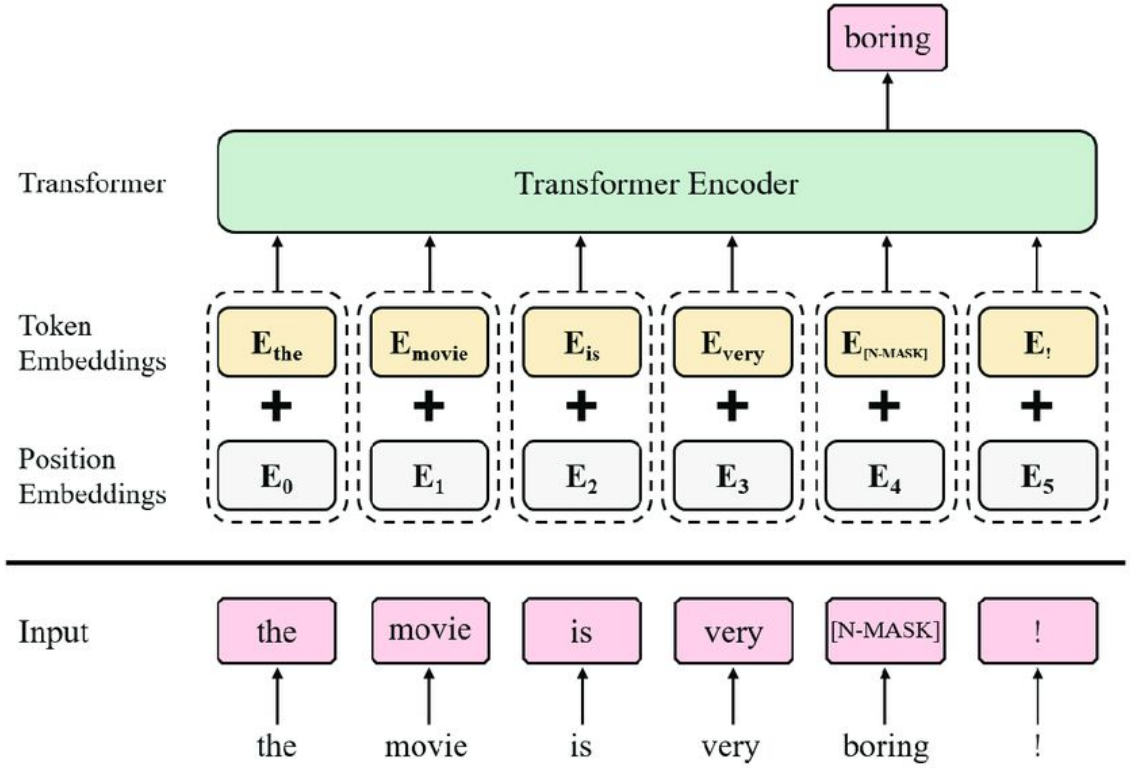
- Pre-training
- Fine-tuning

Pre-Training

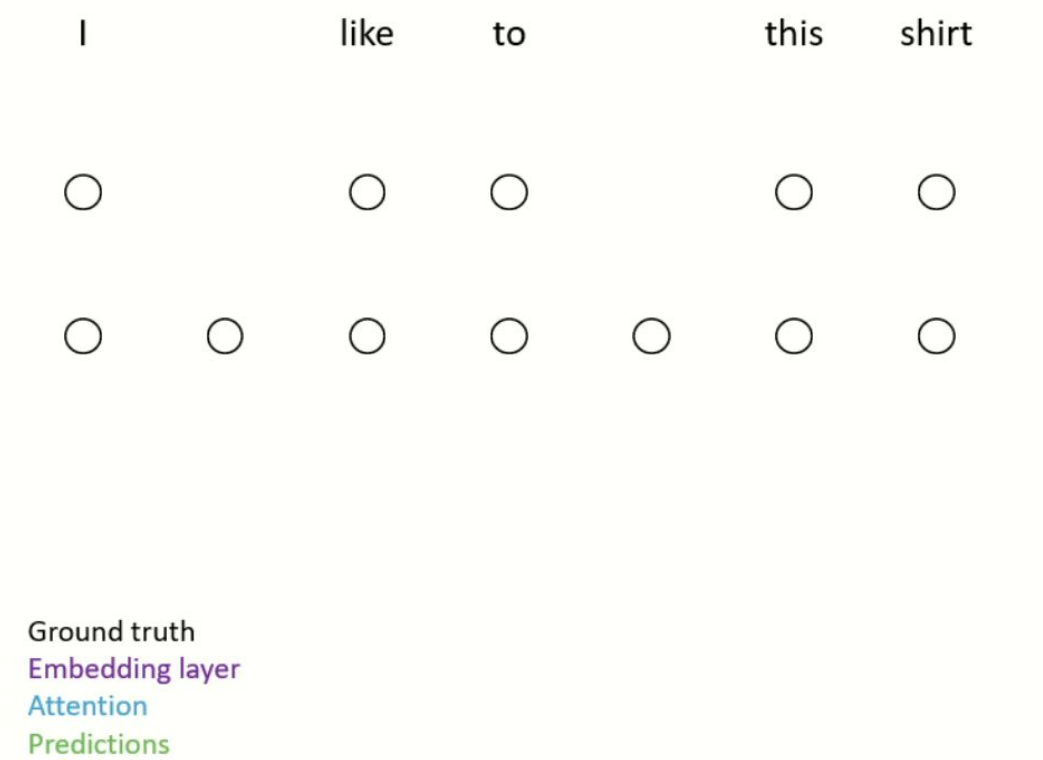
Understanding word in the context -> Understanding the context



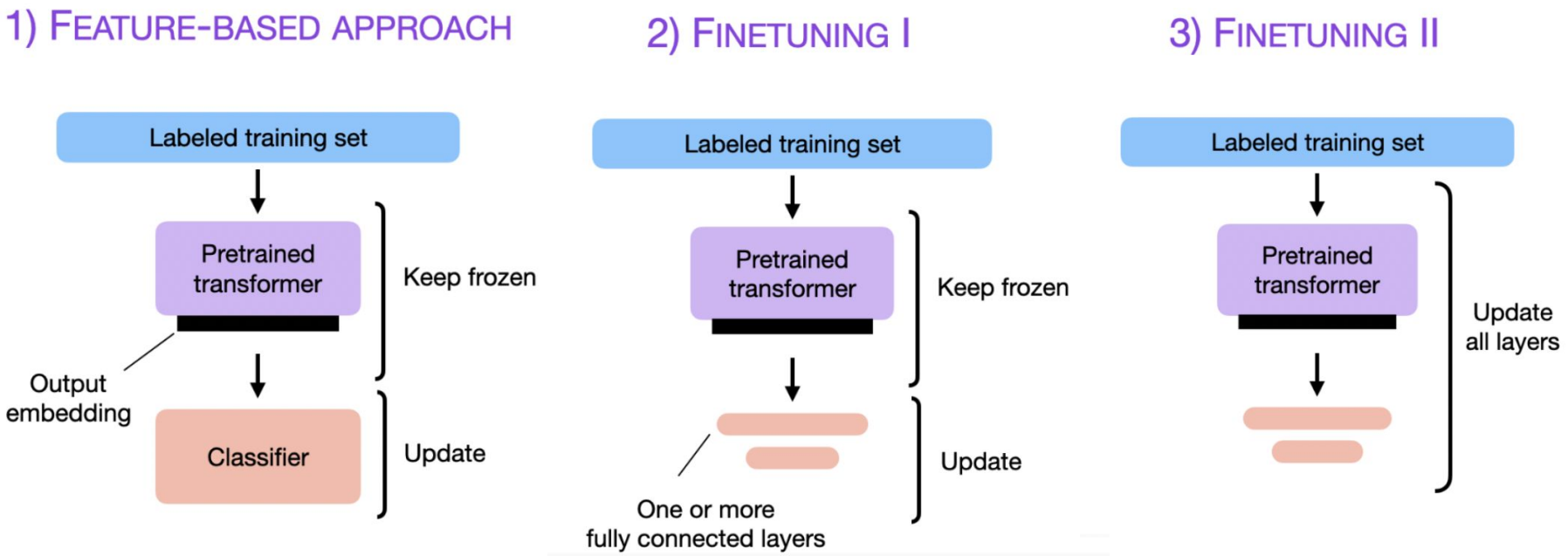
BERT was pre-trained with two unsupervised tasks, Masked LM and Next Sentence Prediction.



BERT's encoder-only architecture, context-dependent word embeddings

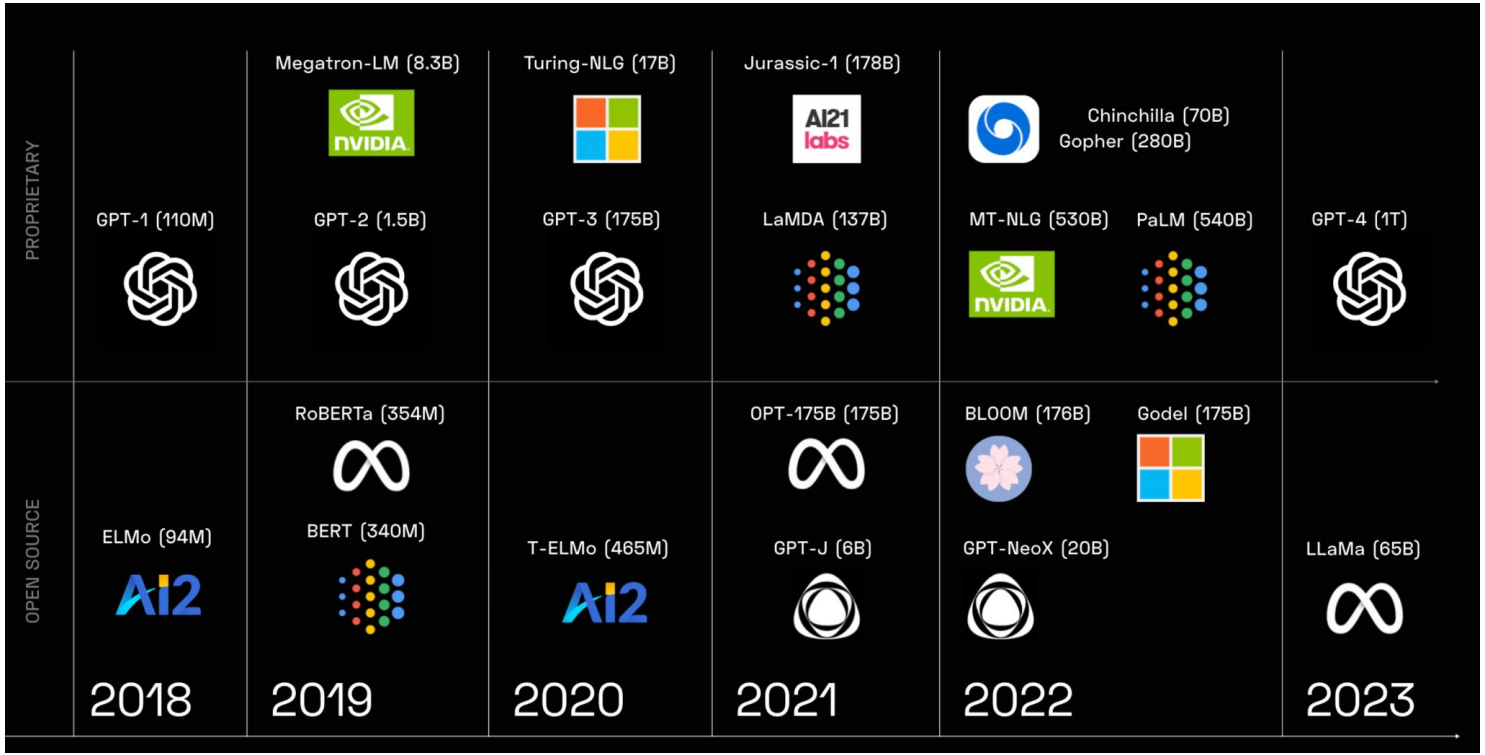

I
like
to
this
shirt
Ground truth
Embedding layer
Attention
Predictions

Fine-tune model via supervised learning



https://magazine.sebastianraschka.com/p/finetuning-large-language-models

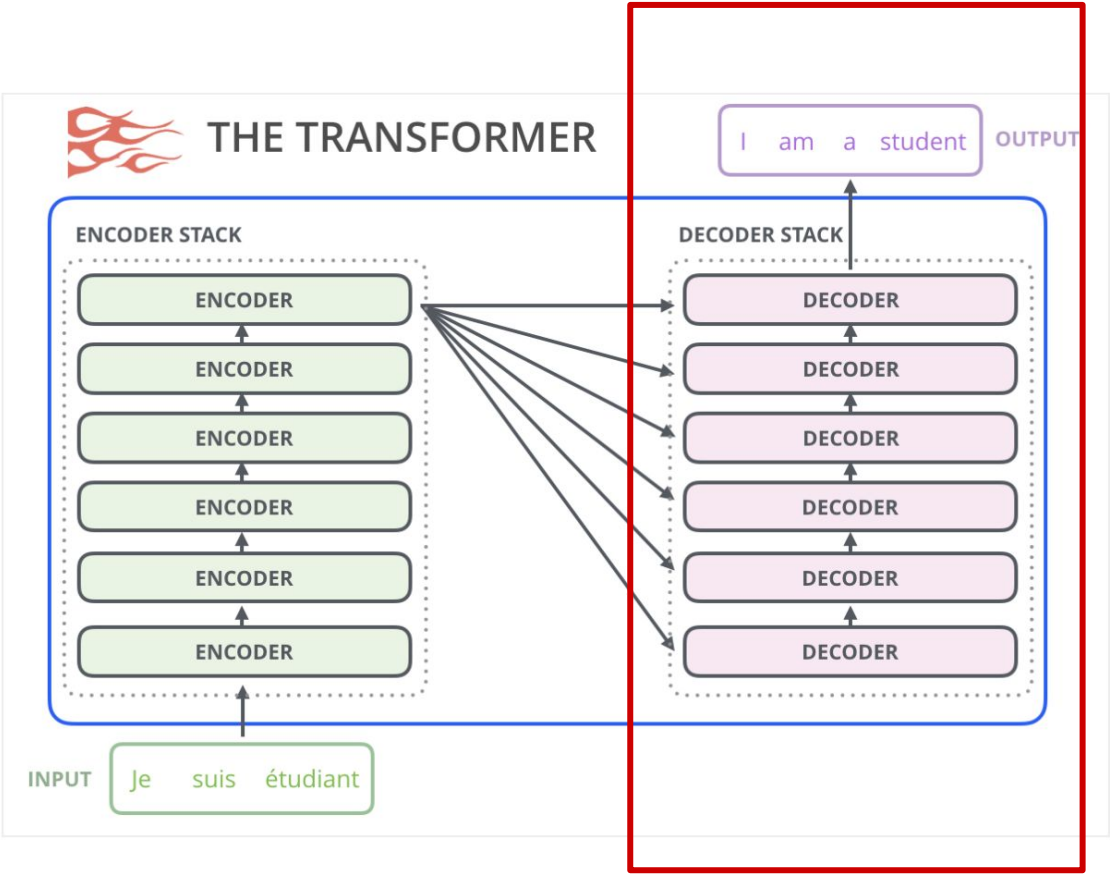
GPT Models

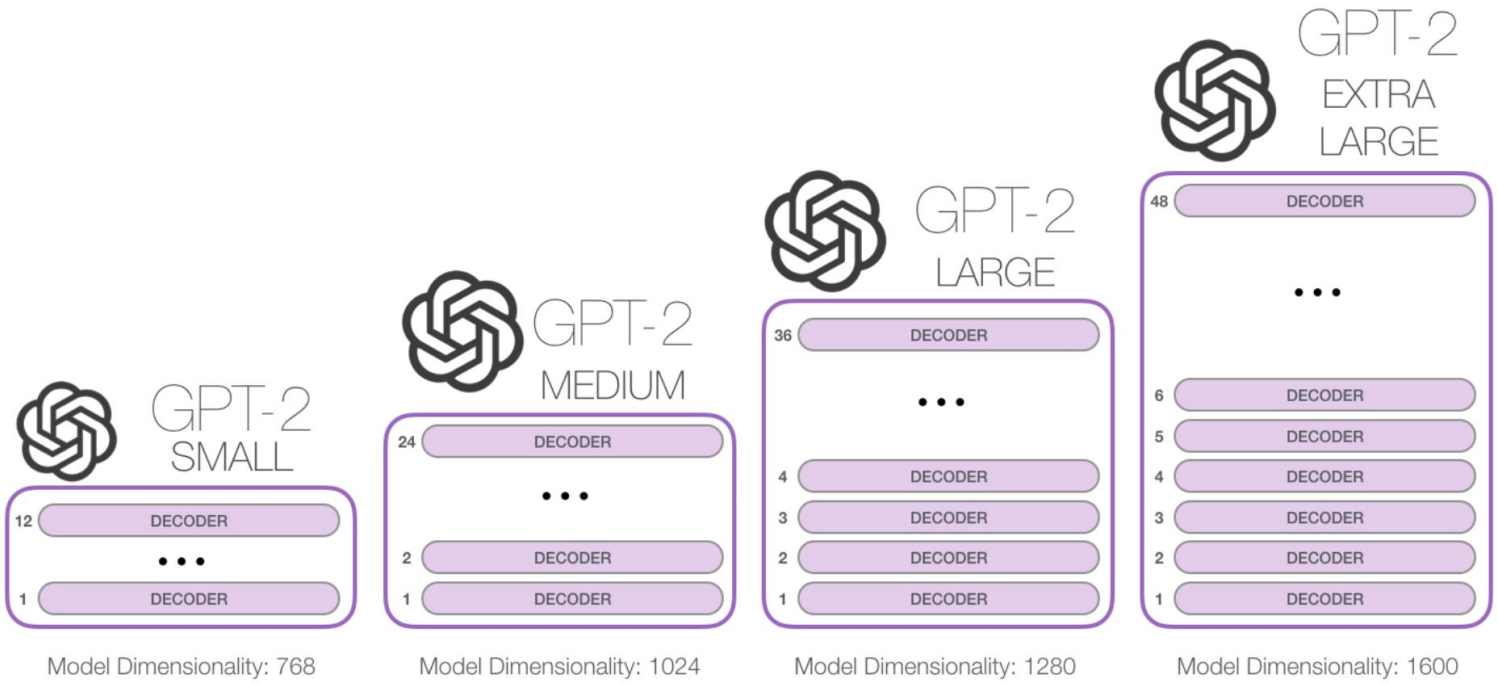

PROPRIETARY
OPEN SOURCE
GPT-1 (110M)
Megatron-LM (8.3B)
GPT-2 (1.5B)
Turing-NLG (17B)
GPT-3 (175B)
Jurassic-1 (178B)
LaMDA (137B)
Chinchilla (70B)
Gopher (280B)
MT-NLG (530B)
PaLM (540B)
GPT-4 (1T)
ELMo (94M)
RoBERTa (354M)
BERT (340M)
T-ELMo (465M)
OPT-175B (175B)
GPT-J (6B)
BLOOM (176B)
Godel (175B)
GPT-NeoX (20B)
LLaMa (65B)
2018
2019
2020
2021
2022
2023

How does the GPT model learn?

Training

- Pre-training
- Fine-tuning

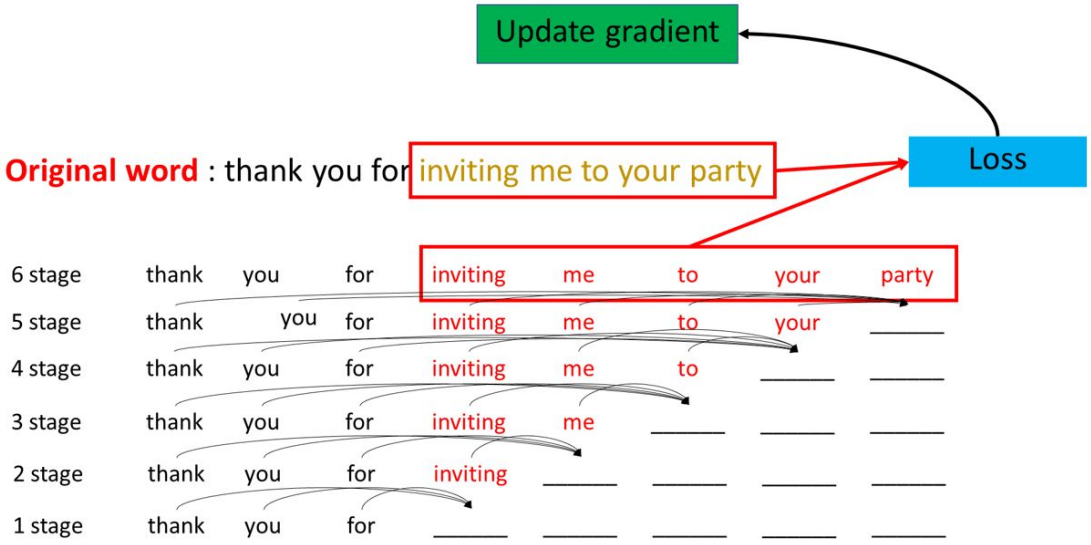

THE TRANSFORMER
I am a student
OUTPUT
ENCODER STACK
DECODER STACK
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
DECODER
DECODER
DECODER
DECODER
DECODER
DECODER
INPUT
Je suis étudiant



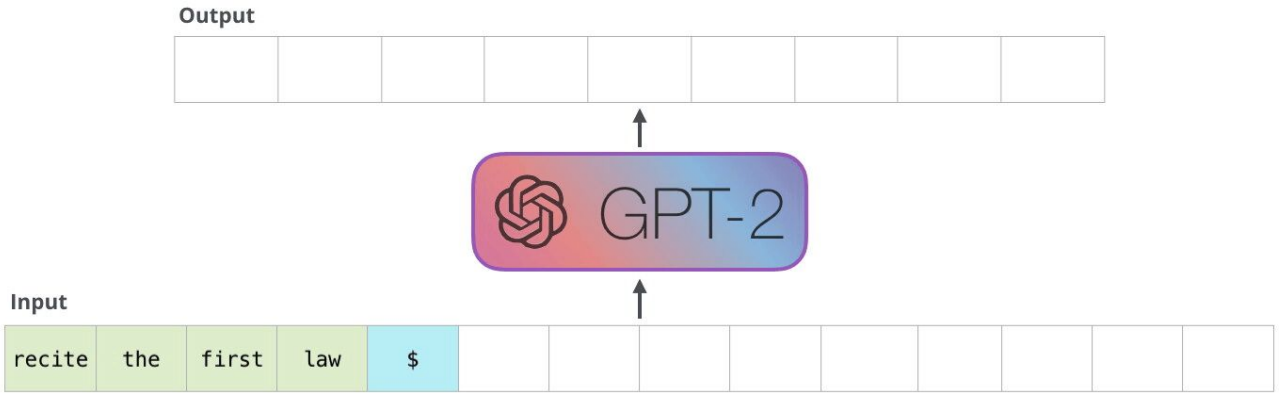
GPT is based on the decoder-only Transformer Architecture

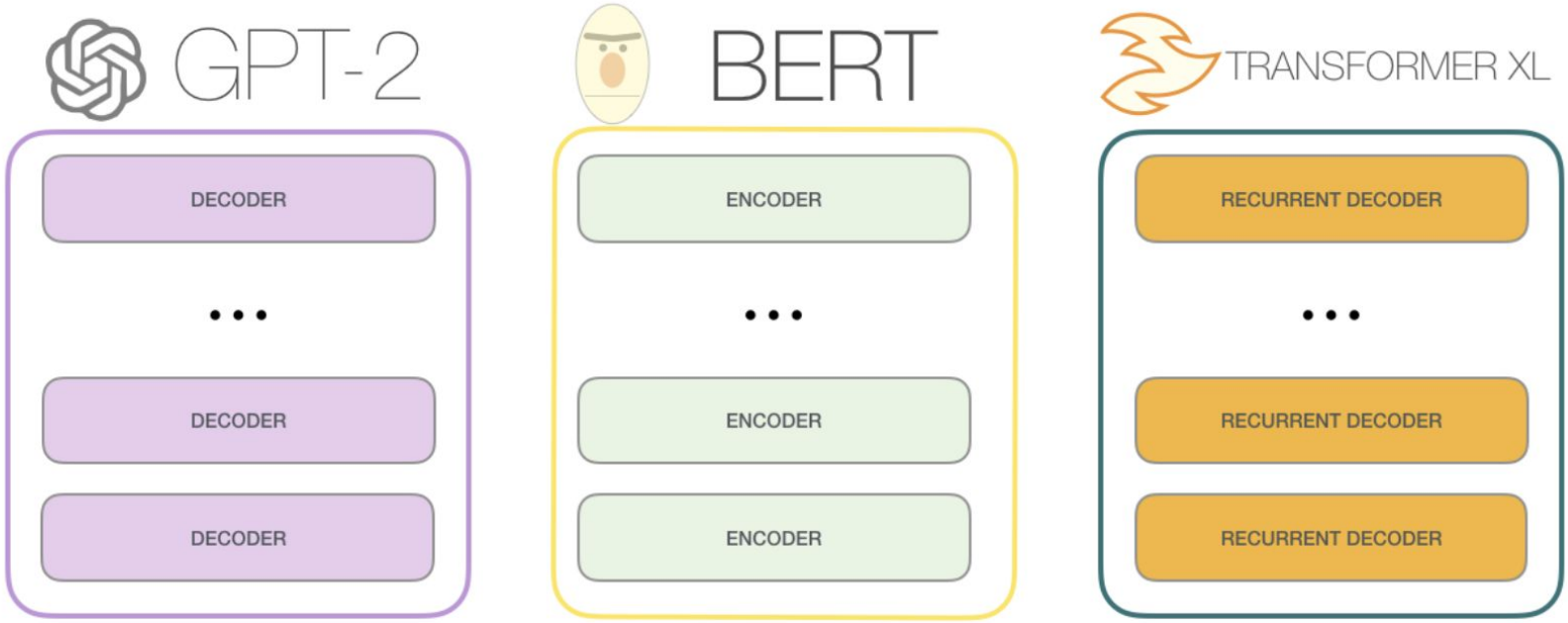
Pre-Training

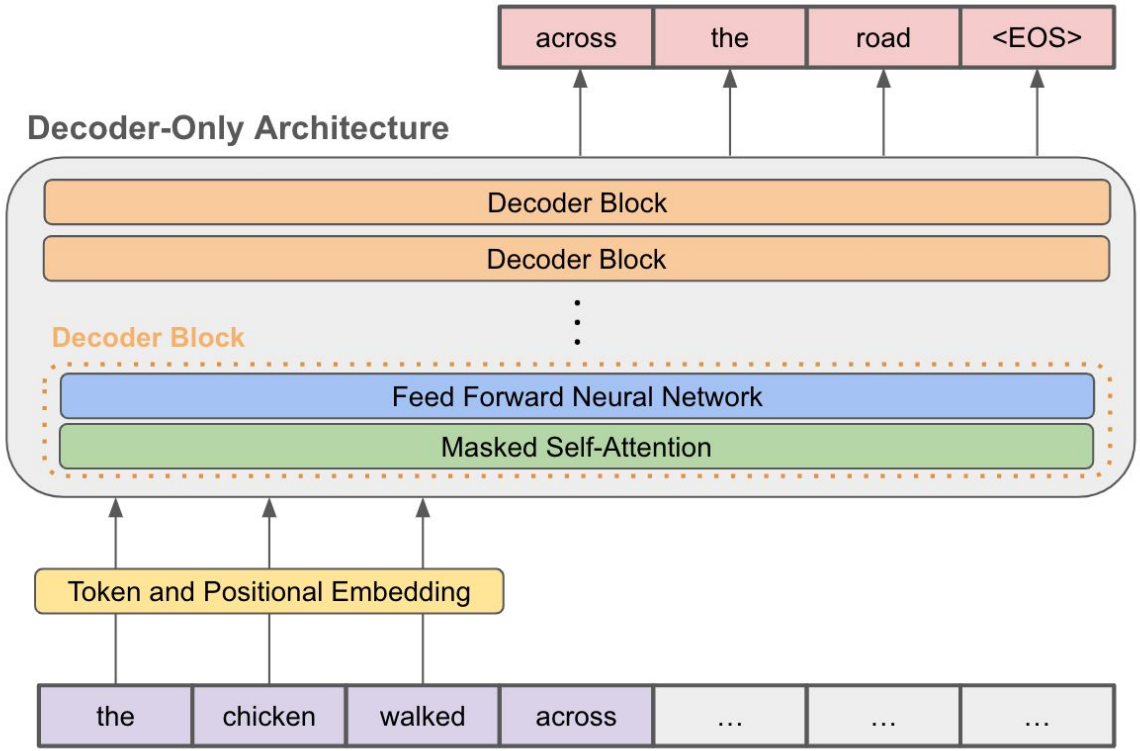
Predicting the next word -> Grasping the flow of dialogue



GPT was pre-trained using Autoregressive Language Modeling for contextual prediction


Output
GPT-2
Input
recite
the
first
law
$


GPT-2
DECODER
DECODER
DECODER
BERT
ENCODER
ENCODER
ENCODER
TRANSFORMER XL
RECURRENT DECODER
RECURRENT DECODER
RECURRENT DECODER


across
the
road
<EOS>
Decoder-Only Architecture
Decoder Block
Decoder Block
Decoder Block
Feed Forward Neural Network
Masked Self-Attention
Token and Positional Embedding
the
chicken
walked
across
…
…
…

Chat-GPT Based Models - Instruct Tuned

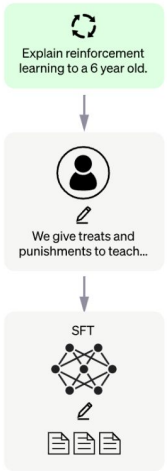
Step 1

Collect demonstration data and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3.5 with supervised learning.



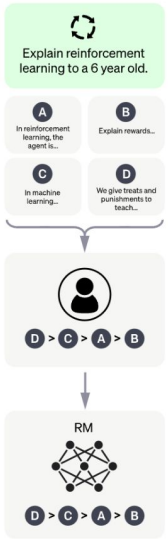
Step 2

Collect comparison data and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

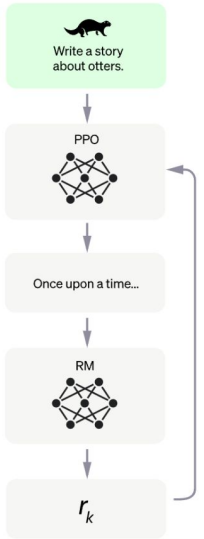
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Hands on Bert based Transformer model to detect if paragraph is human written or GPT-Generated

Google Developer Groups

Final Notes

Simple Neural Networks (MLP)

Activation Functions

Embeddings

Attention Mechanism

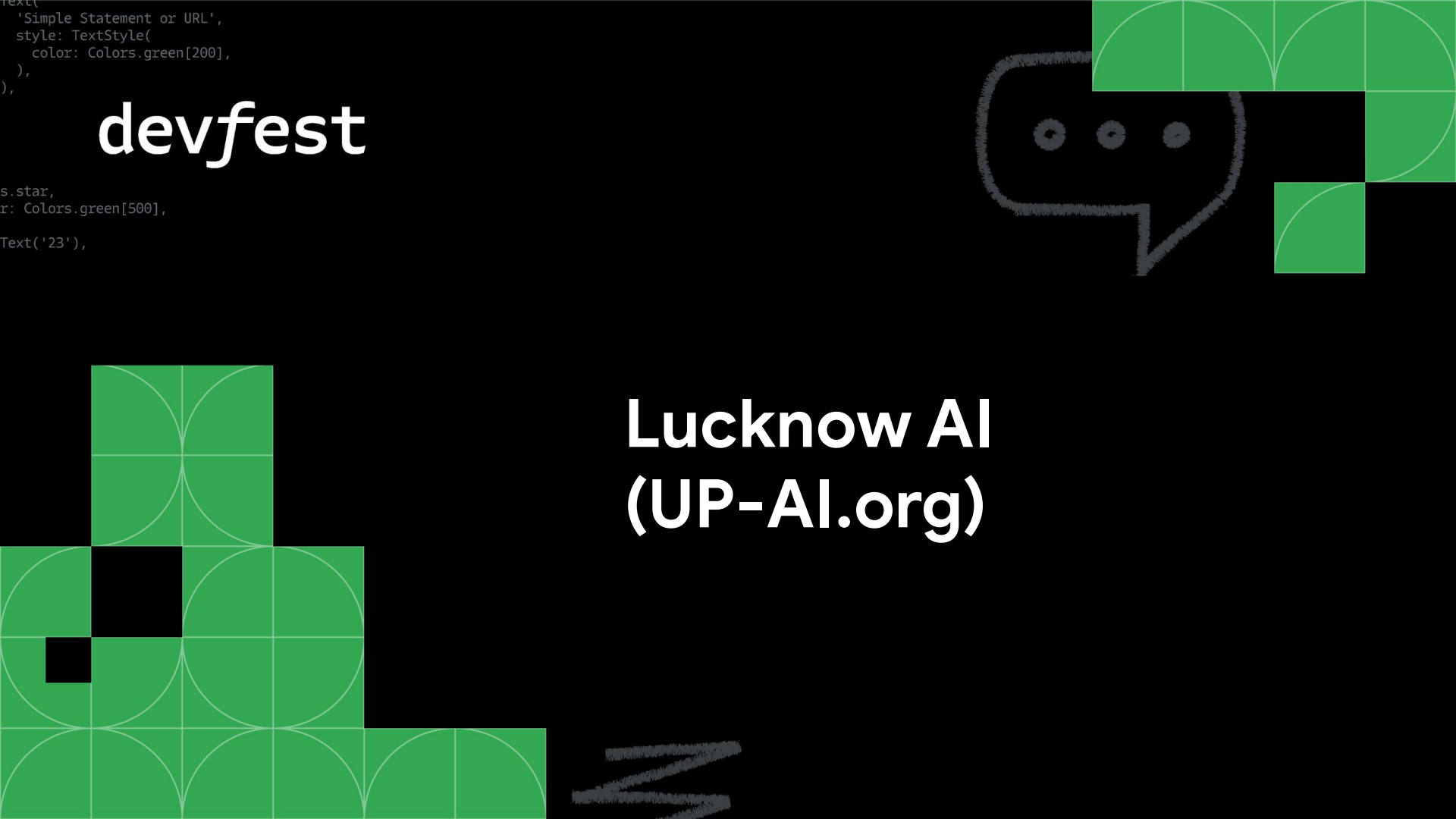
Transformers

Bert & GPT

GPT Detect Model

Thank you!

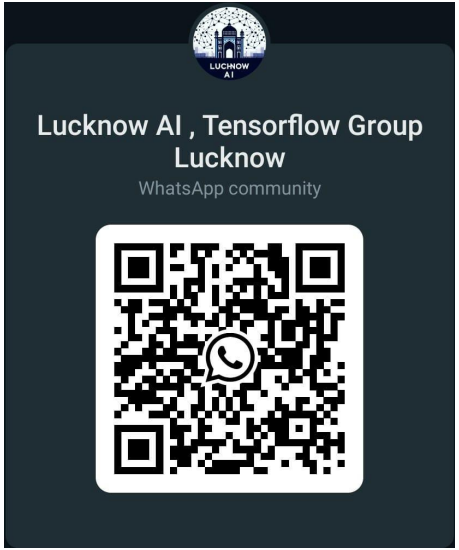
Let's get connected on Twitter! I am @adityaura

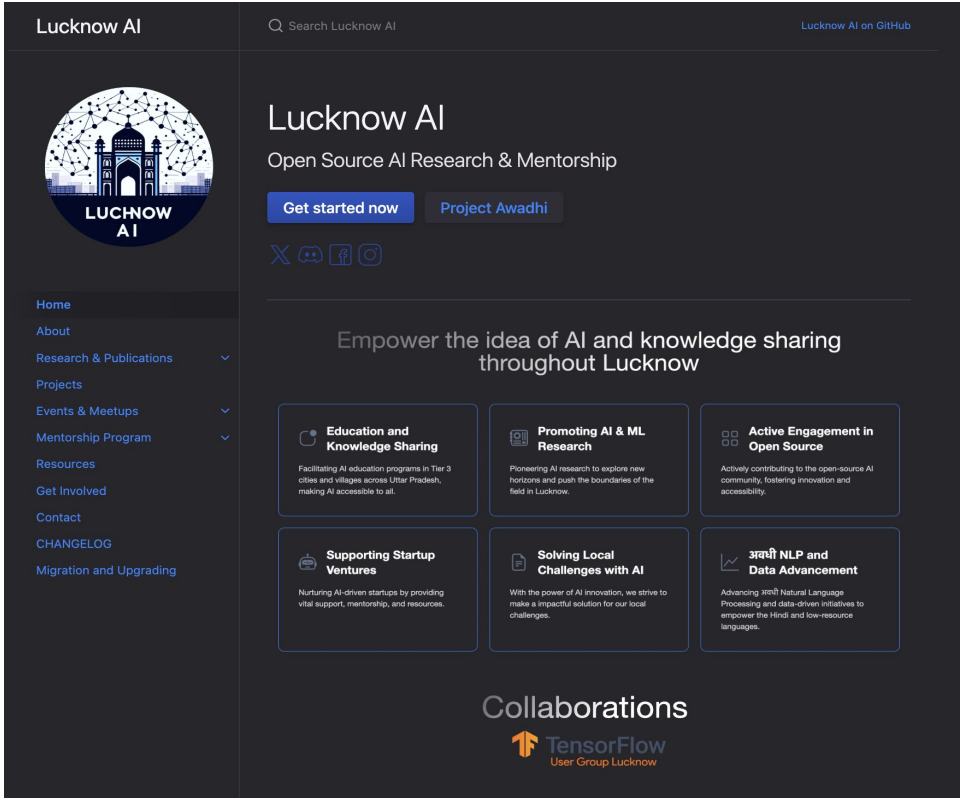

'Simple Statement or URL',
style: TextStyle(
color: Colors.green[200],
),
),
devfest
s.star,
r: Colors.green[500],
Text('23'),
Lucknow AI
(UP-AI.org)

Lucknow AI

@ailucknow

http://lucknowai.org





Google Developer Groups

TensorFlow
User Group Lucknow