



# transformers and beyond

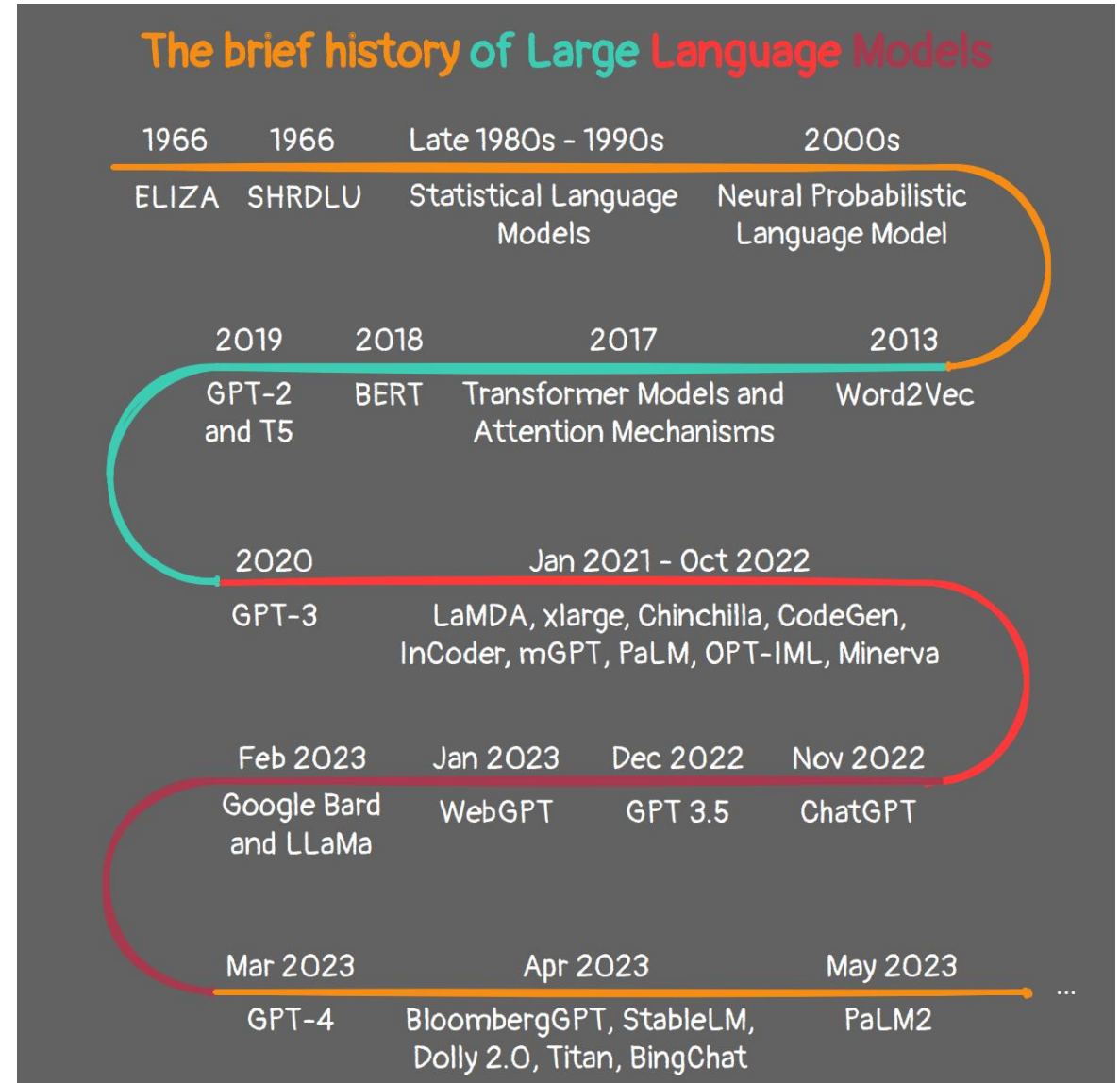
GESIS Fall Seminar 2023

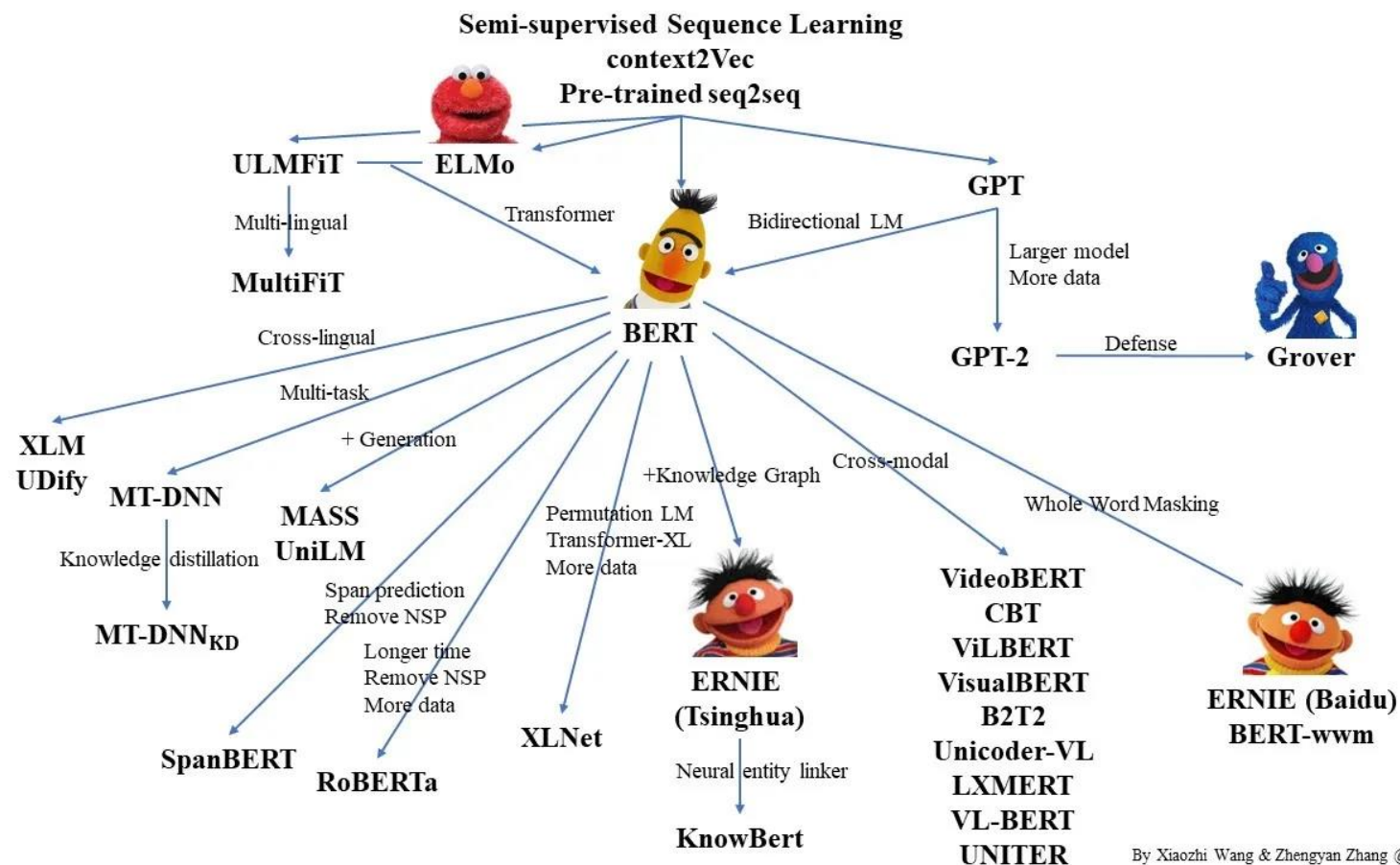
“From Embeddings to Transformers: Advanced  
Text Analysis in Python”

*[day 3, afternoon: attention]*

# some history

[[The brief history of Large Language Models: A Journey from ELIZA to GPT-4 and Google Bard](#) | by Armin Norouzi, Ph.D | [Level Up Coding](#) ([gitconnected.com](#))]





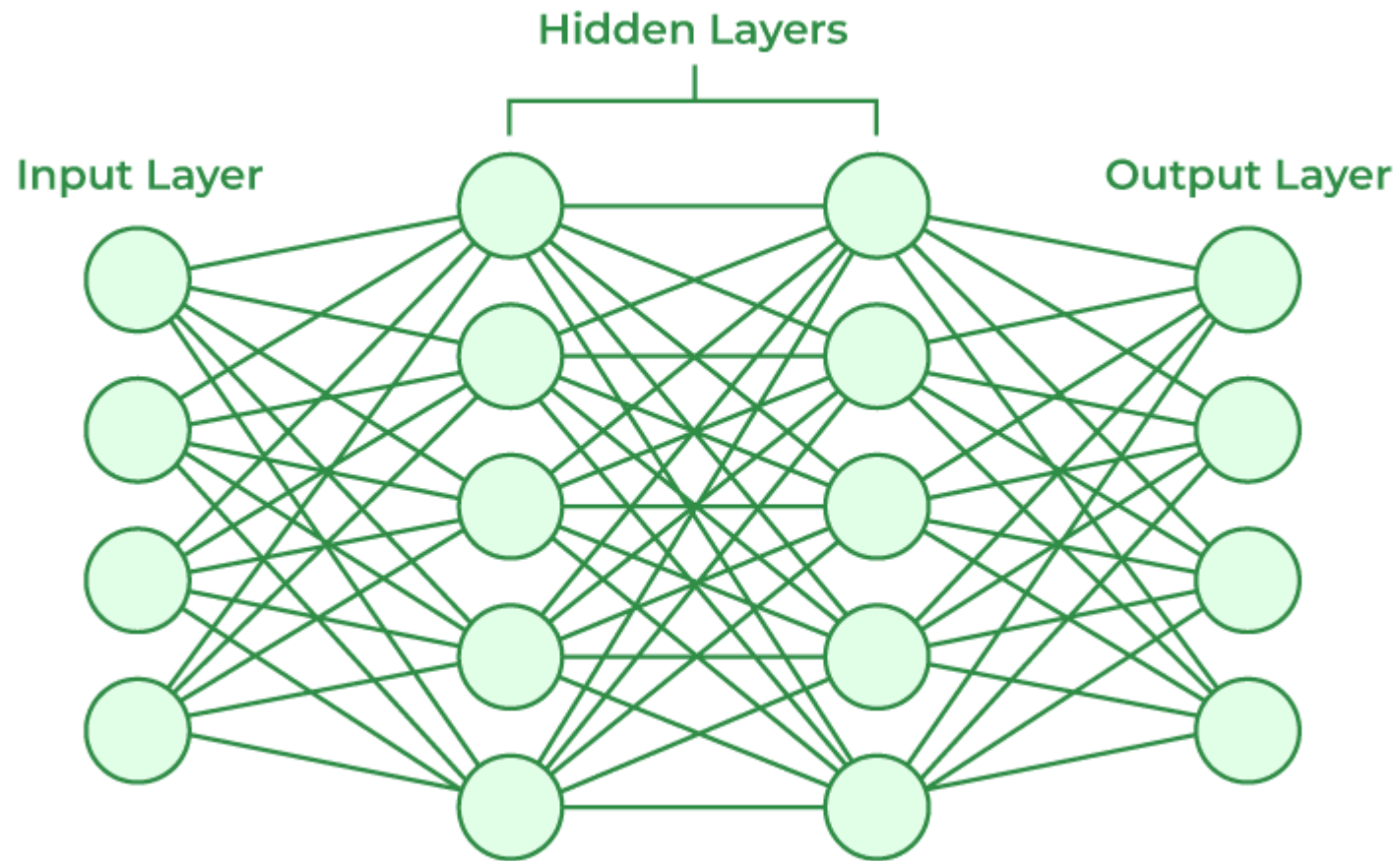
By Xiaozhi Wang & Zhengyan Zhang @THUNLP

# ELMo, BERT, & crew aka GPT

[[10 Things You Need to Know About BERT and the Transformer Architecture That Are Reshaping the AI Landscape \(neptune.ai\)](#)]

# neural networks

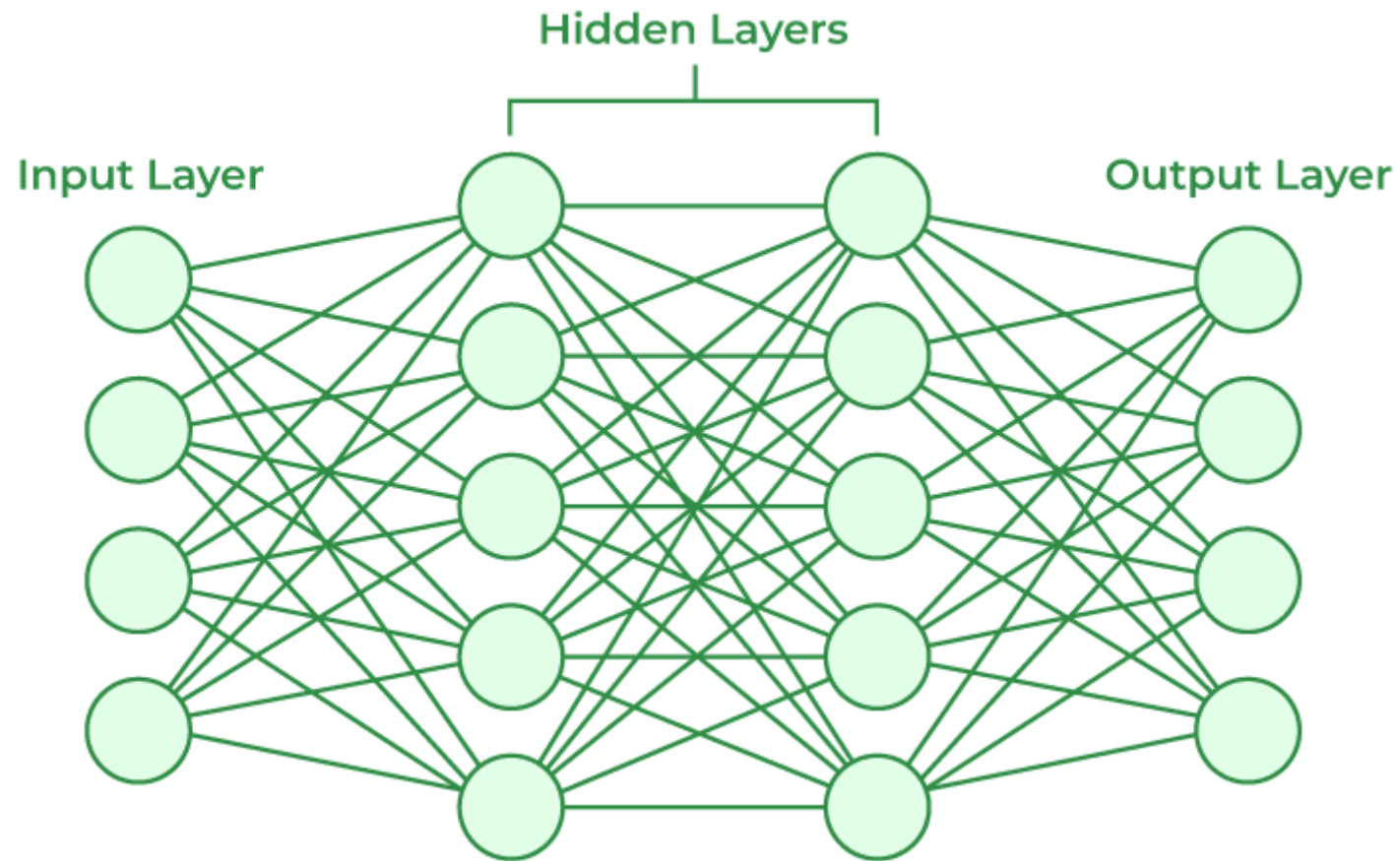
Neural Network In 5 Minutes | What Is A Neural  
Network? | How Neural Networks Work |  
Simplilearn - YouTube



# neural networks

[\[Artificial Neural Networks and its Applications - GeeksforGeeks\]](#)





does this  
look  
familiar?



don't think  
too far!

# regression?

[[Lesson 3: Neural Network is nothing but a Linear Regression | by Md. Asifur Rahman | Medium](#)]



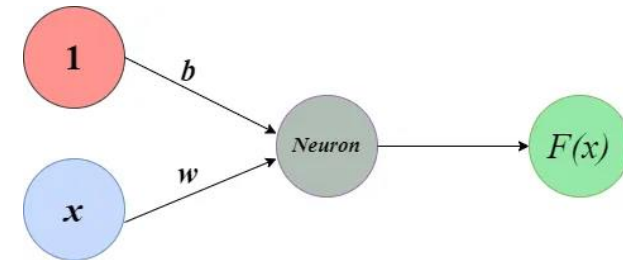
# linear regression!

# single neural network path!

$$y = \beta_0 + \beta_1 X + \varepsilon$$

$$y = mx + c$$

$y=mx+c$ ;  $x$  is feature vector,  $c$  is bias,  $y$  is output or dependent variable,  $m$  is weight vector.



$$F(x) = w^T x + b$$

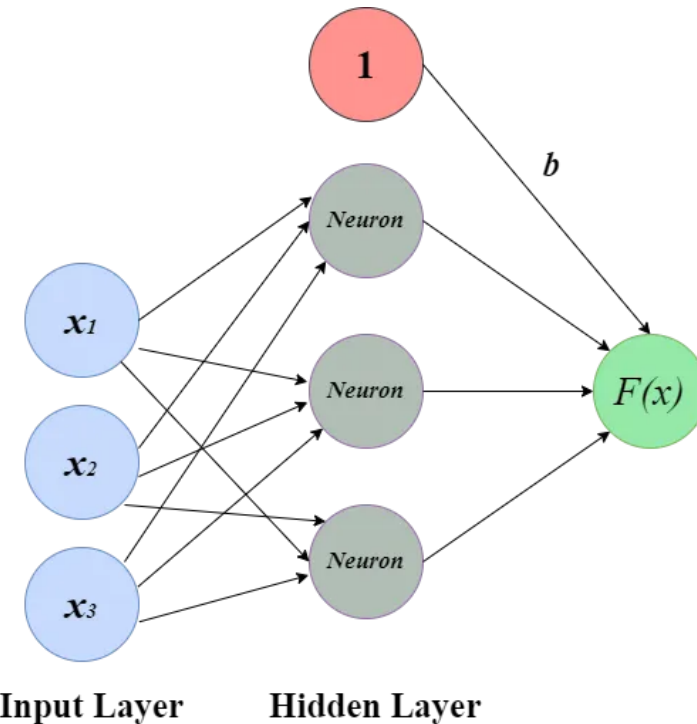
[[Lesson 3: Neural Network is nothing but a Linear Regression | by Md. Asifur Rahman | Medium](#)]

# multiple linear regression!

# simple perceptron!

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_i X_i$$

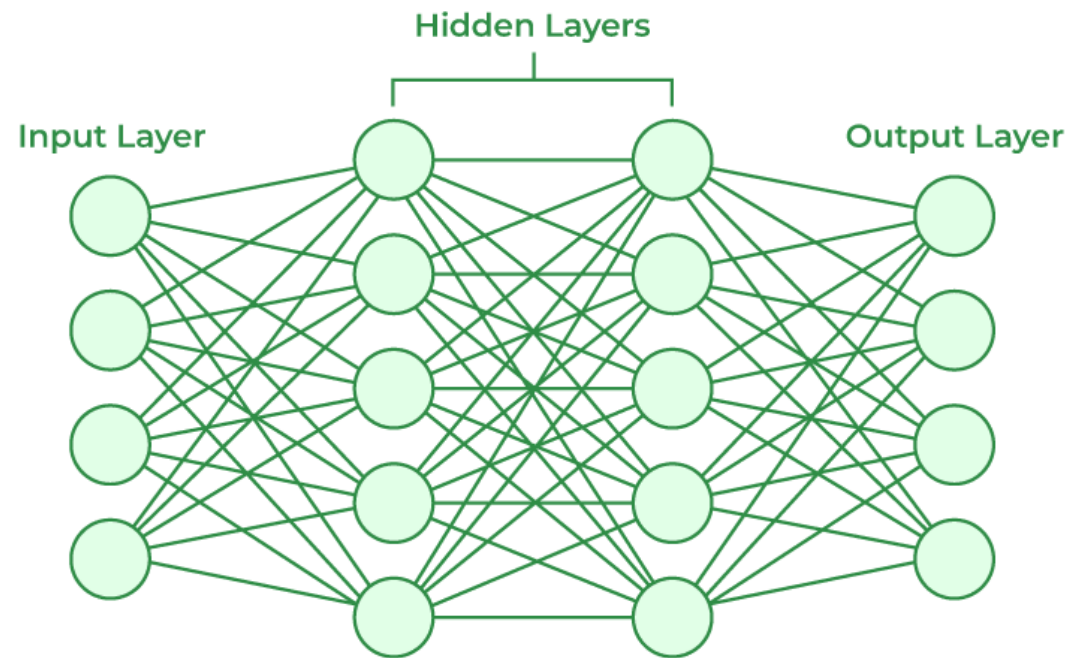
$Y$  : Dependent variable  
 $\beta_0$  : Intercept  
 $\beta_i$  : Slope for  $X_i$   
 $X$  = Independent variable



[[Lesson 3: Neural Network is nothing but a Linear Regression | by Md. Asifur Rahman | Medium](#)]

adding additional layers ➡ neural network!

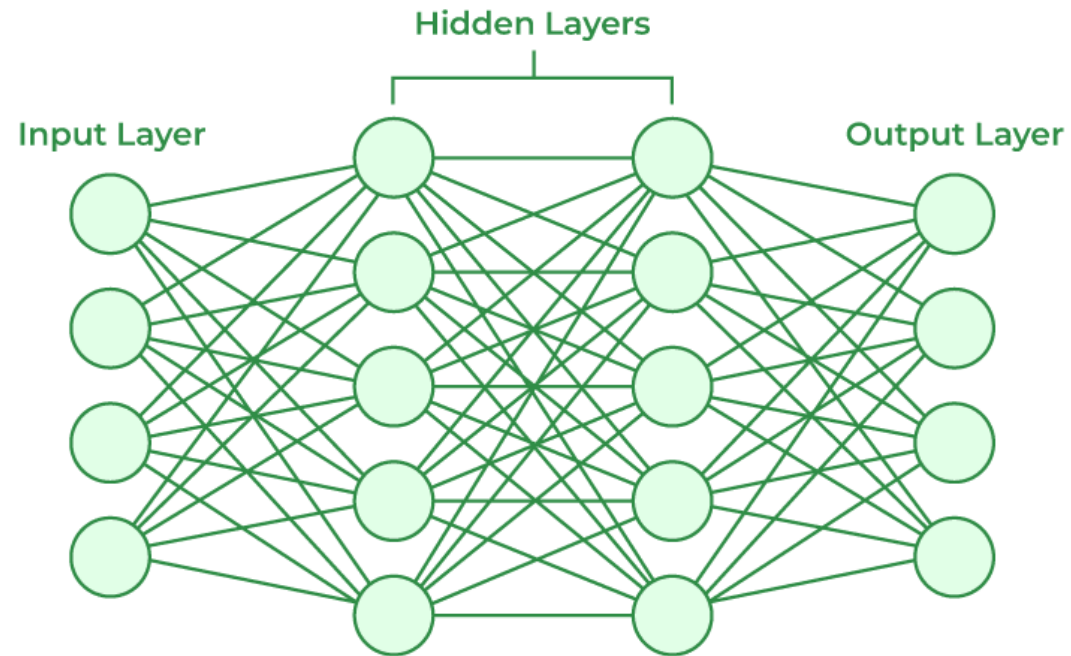
---



# but what about linearity and non-linearity?

linear regression = linear.

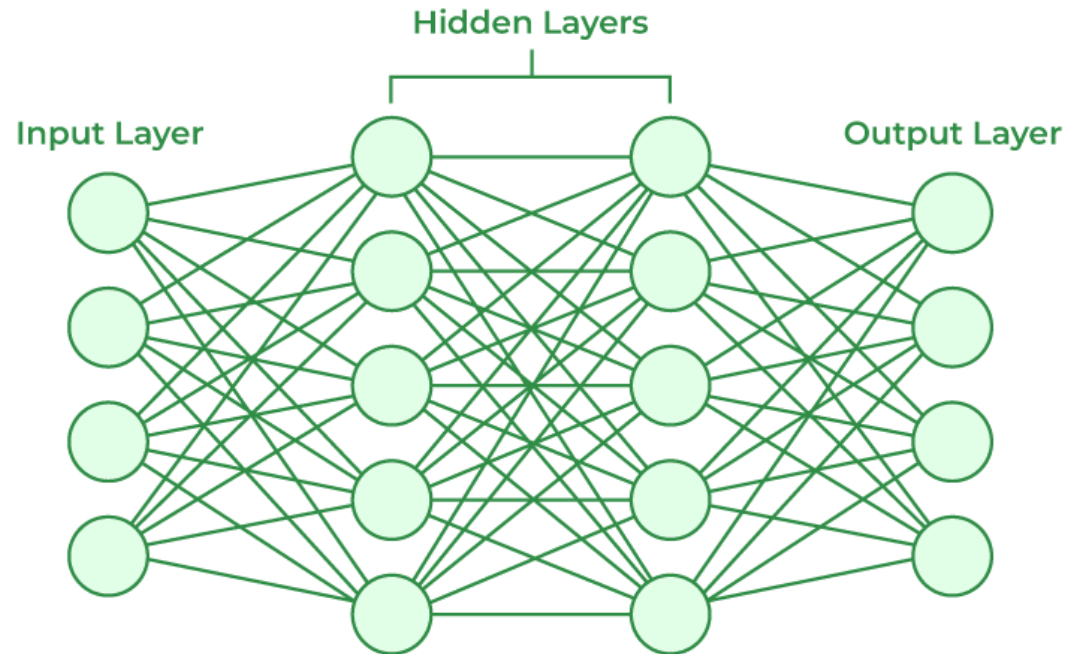
neural networks = solve complex non-linear problems.

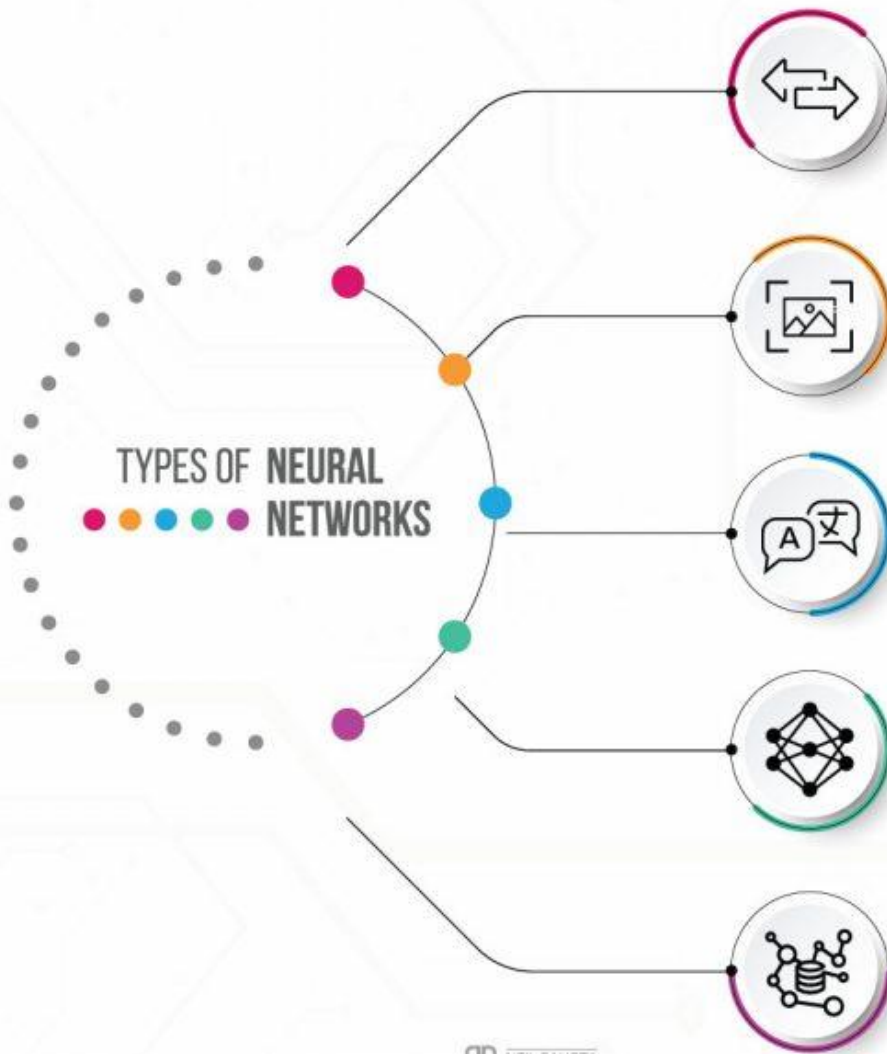


# remember the activation function

---

If we exchange a non-linear activation function  
with a linear activation function,  
the neural network in its function becomes a linear regression.





## FEEDFORWARD NEURAL NETWORKS

Feedforward neural networks are good at solving problems with a clear relationship between the input and the output, but may not be as effective at figuring out more complex relationships.

## CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks are used for tasks that involve data with a grid-like structure, such as image recognition, but may require a large amount of data and be slow.

## RECURRENT NEURAL NETWORKS

Recurrent neural networks are used for tasks involving data in a sequence, such as a language translation and speech recognition, but they may need help learning long-term relationships, which can be challenging to train.

## GENERATIVE ADVERSARIAL NETWORKS

Generative adversarial networks are composed of two neural networks that work together to generate synthetic data that appears real but may be challenging to train and require a large amount of data to perform well. They have been used for tasks such as creating realistic images and

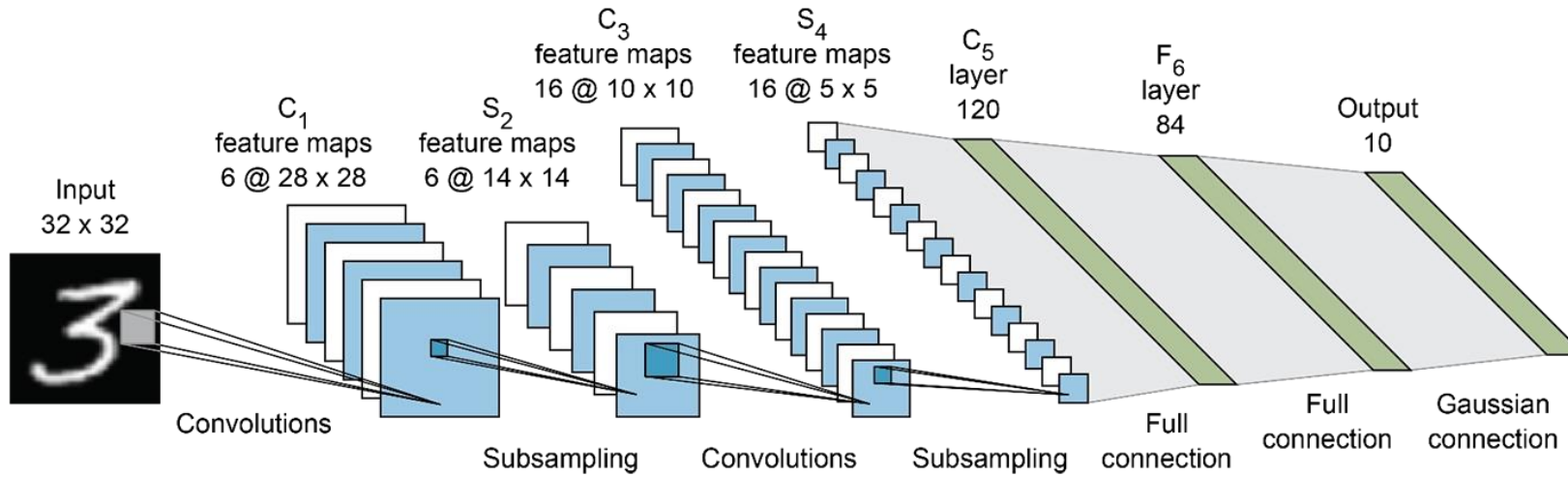
## AUTOENCODER NEURAL NETWORKS

Autoencoders are used to reduce the complexity of data and learn important features, but they may be sensitive to the settings used and may not always learn meaningful patterns in the data. They have been applied in tasks such as image and speech recognition.

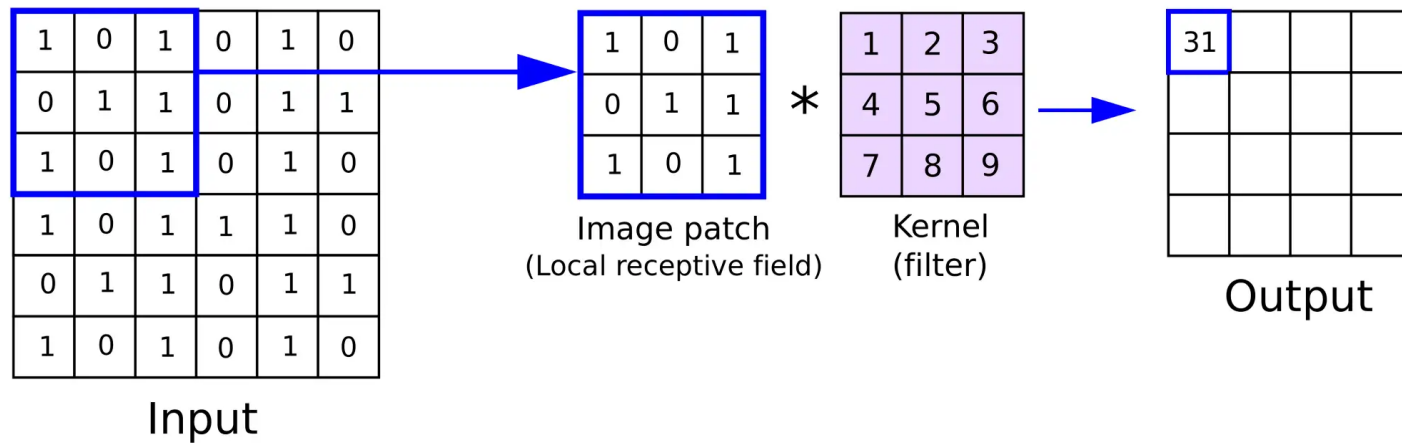
task-specific  
types of  
neural  
networks

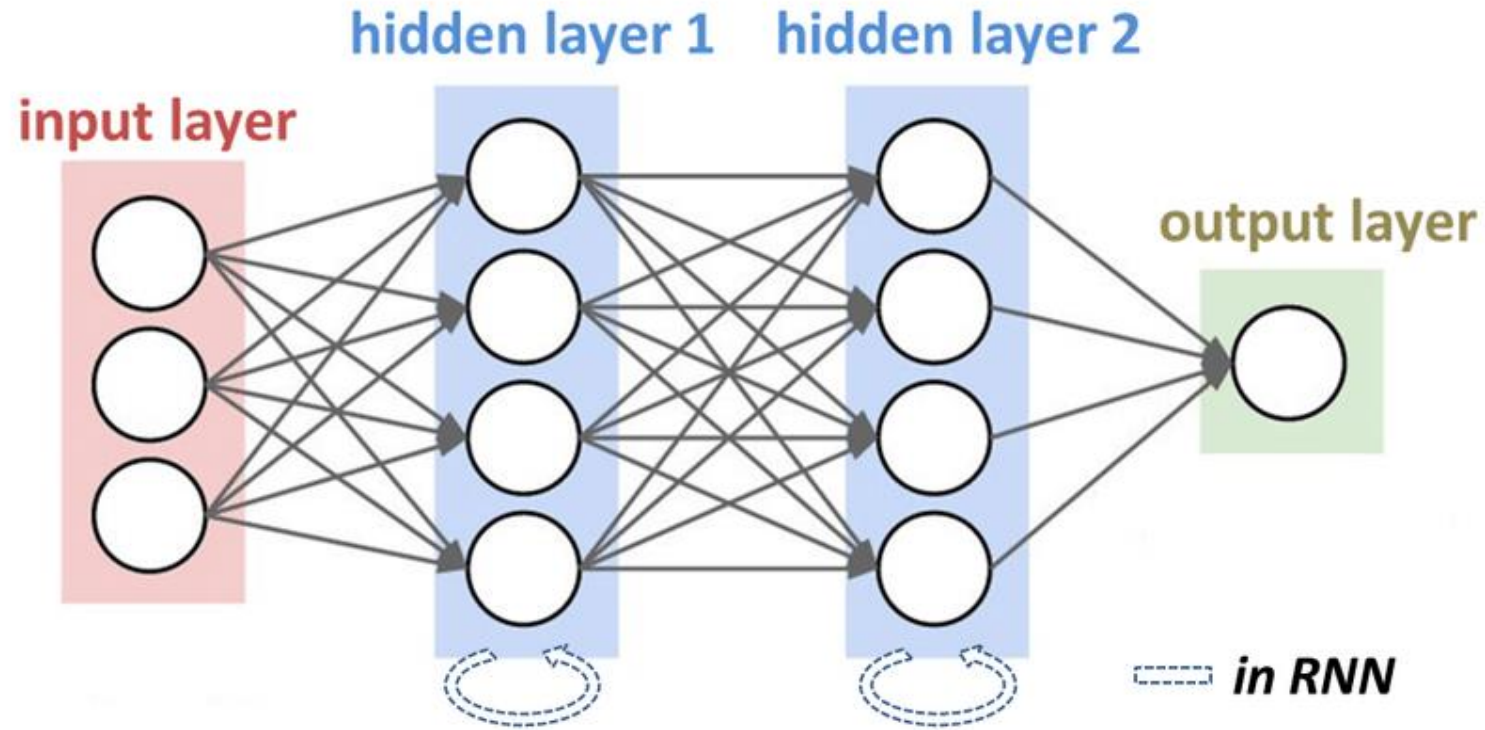
[[Neural Networks: Solving Complex Science Problems \(neilsahota.com\)](https://neilsahota.com)]





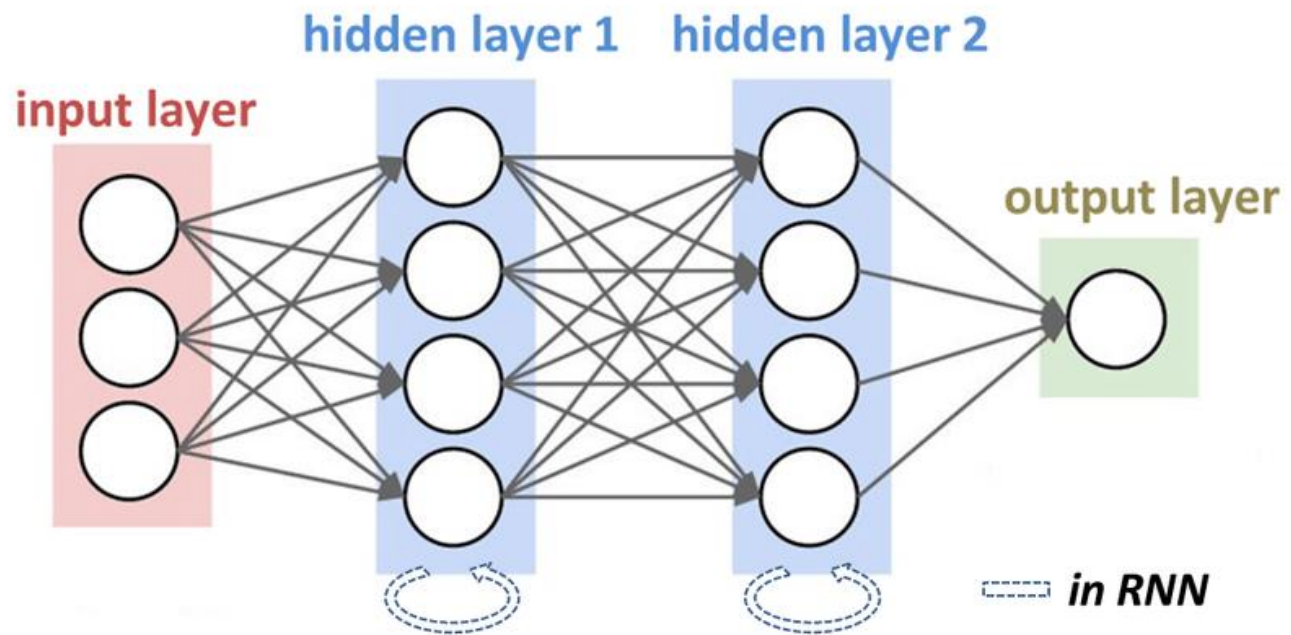
convolutional  
neural  
network





text =  
sequential  
processing

RNNs =  
recurrent  
neural  
networks

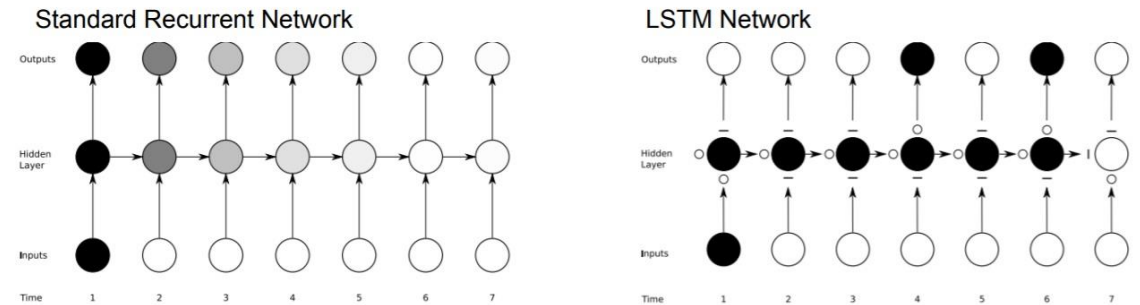


**problem: vanishing gradient**

# RNNs

# long short-term memory LSTMs

LSTMs reduce vanishing gradient problem



Graves et al 2013

- The darker the shade, the greater the sensitivity
- The sensitivity decays exponentially over time as new inputs overwrite the activation of hidden unit and the network 'forgets' the first input



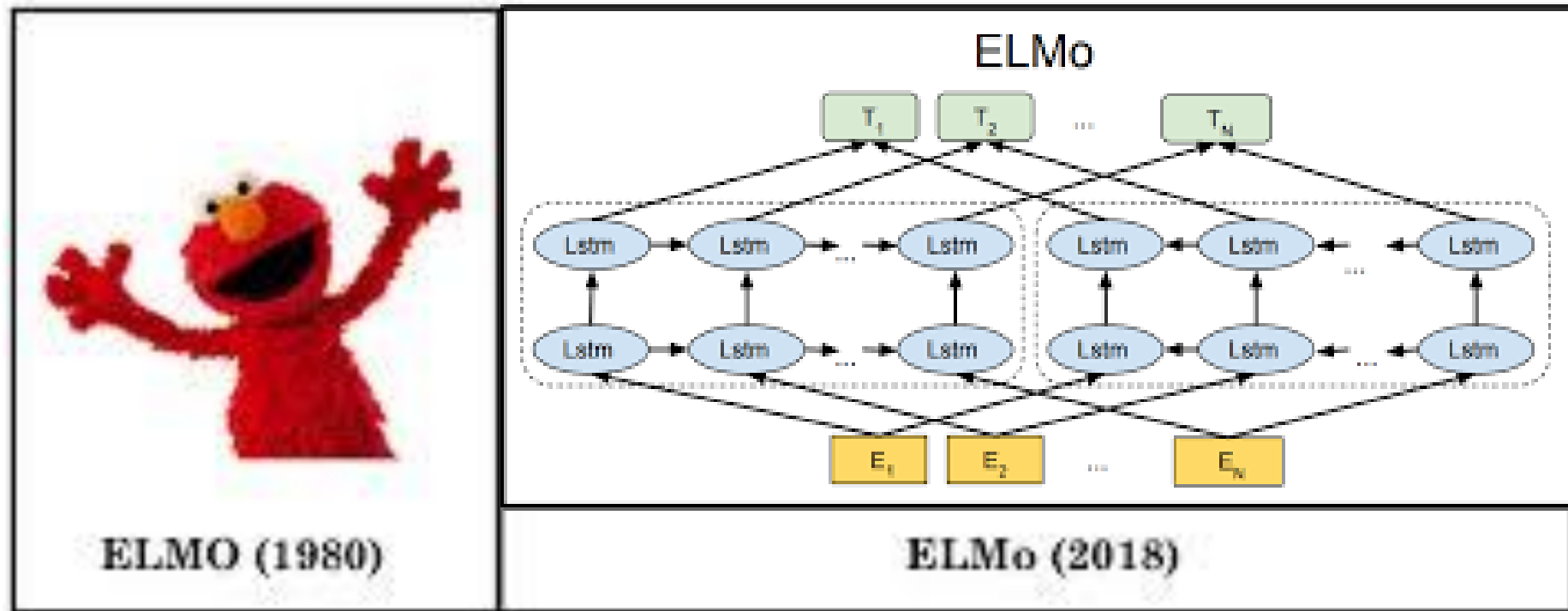


why  
transformers  
?!

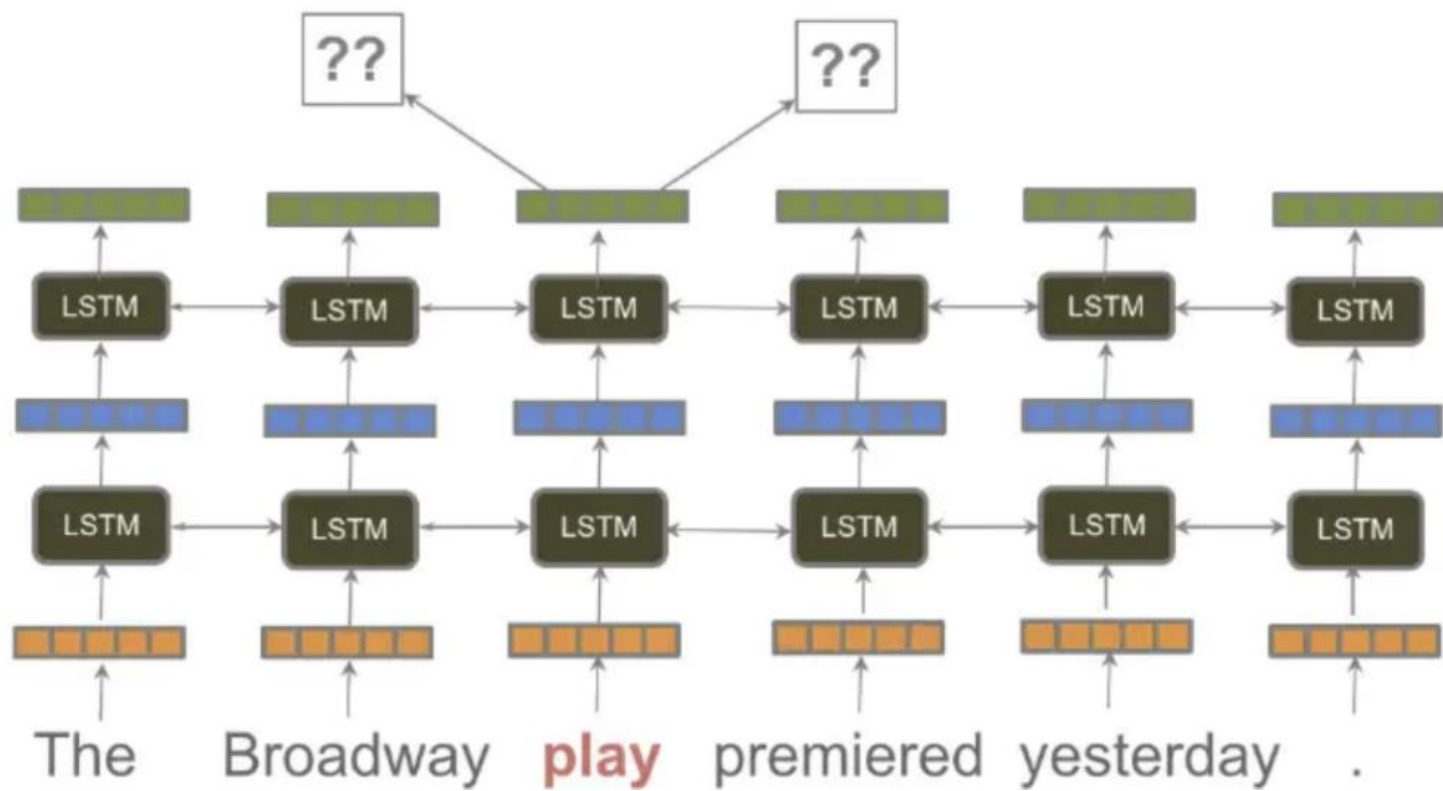
welcome to  
the world of  
transfer  
learning

# ELMo

## Embeddings from Language Models







ELMo

bidirectional  
stacked  
LSTM



# ELMo shortcomings

- **lack of ability to capture long range dependencies and global context (dynamic context window)**
  - **training complexity**
    - **computationally expensive and time-consuming**
  - **fixed architecture**
    - **not adaptable to different tasks and data sizes**
  - **lack of masked language modeling**
-



transformers

# transformer architecture

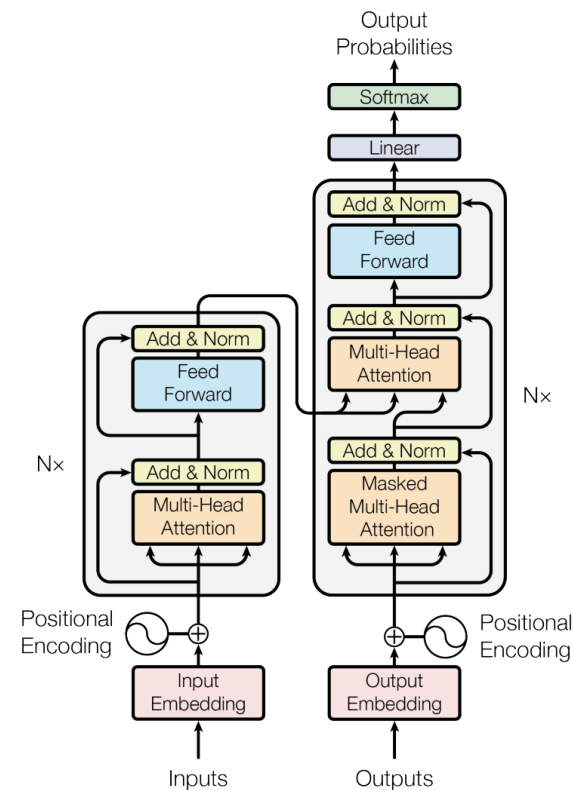
- Parallel processing: Increases performance and scalability
- Bidirectionality: Allows understanding of ambiguous words and coreferences

BERT

Encoder

GPT

Decoder



# transformer architecture

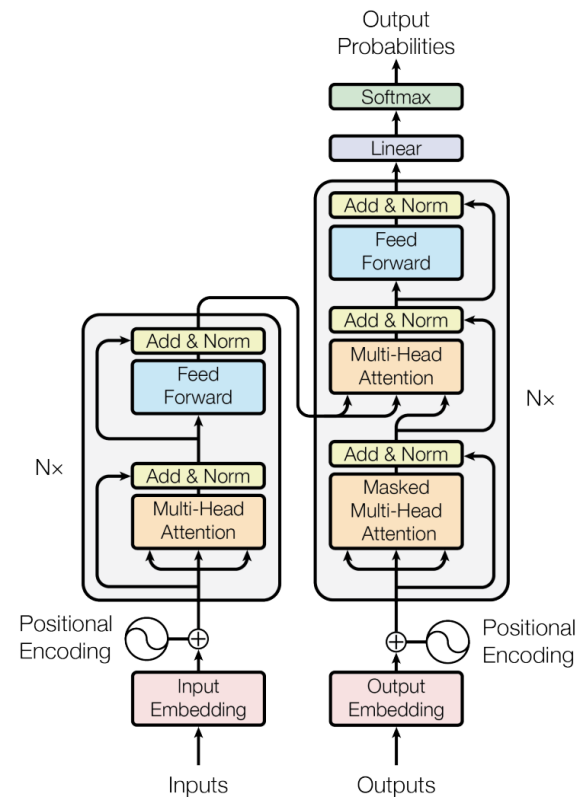
- BERT (encoder): classification (e.g., sentiment), questions and answers, summarization, named entity recognition
  - output: embeddings representing words with attention information in a certain context
- GPT (decoder): translation, generation (e.g., stories)
  - output: next words with probabilities

BERT

Encoder

GPT

Decoder



A mechanism that computes a weighted sum of values (elements in a sequence) based on their similarity to a given query.

It allows the model to dynamically emphasize or de-emphasize different elements in the sequence, enabling the capture of contextual information and relationships between elements.

[\[1706.03762\] Attention Is All You Need \(arxiv.org\)](#)


# attention




# a little help from ChatGPT

## Attention Mechanism in Transformers

(in Simple Terms):



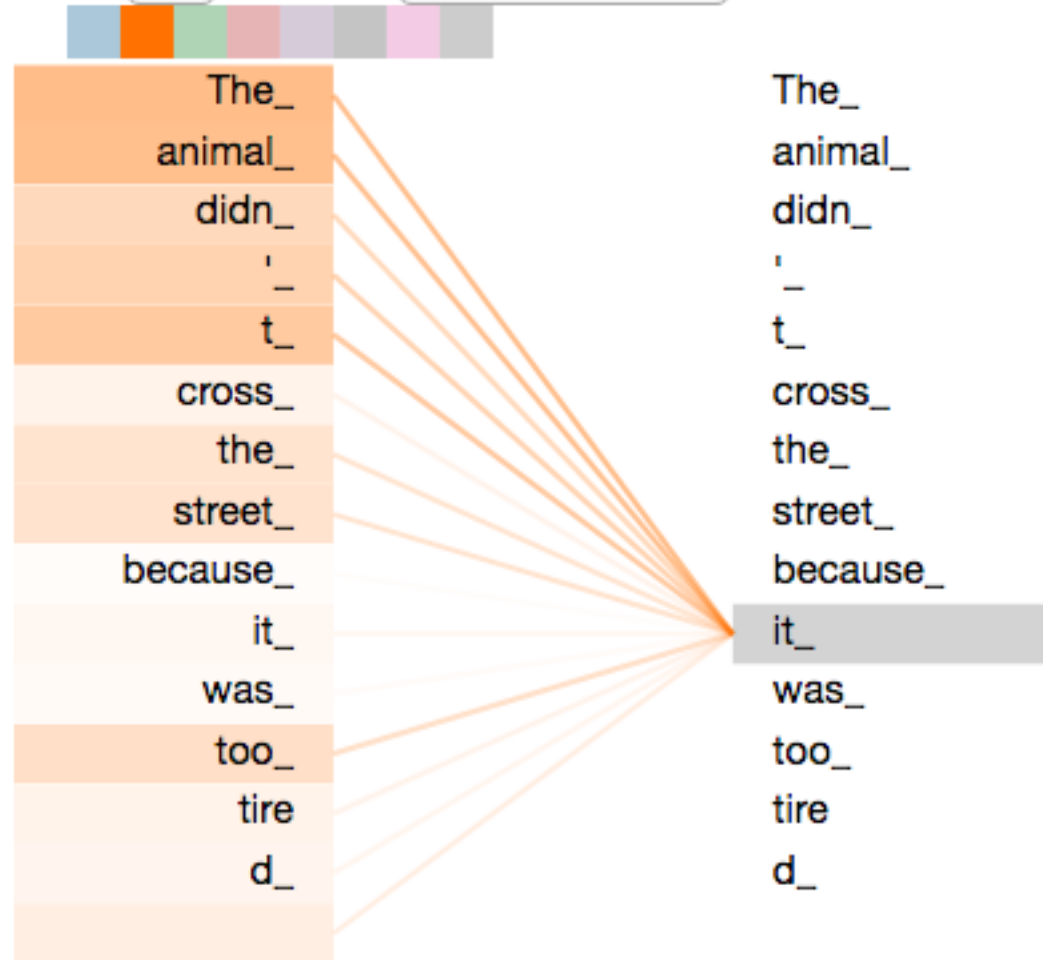
Imagine you're reading a sentence, and you want to understand the meaning of a specific word. Instead of just looking at that one word, you glance at all the words in the sentence. However, you pay more attention to the words that are more relevant to the word you're trying to understand. This way, you get a better grasp of the context and meaning.



The attention mechanism in Transformers works similarly. It helps the model focus on different parts of a sentence when trying to understand a specific word. It's like giving more importance to the words that matter most for understanding that word's context. This allows the model to create richer and more accurate representations of words and their relationships.

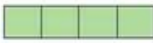
[bertviz tutorial.ipynb - Colaboratory \(google.com\)](#)

Layer: 5 Attention: Input - Input



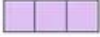
attention score  
needs query and key

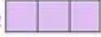
Embedding

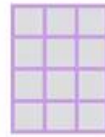
$x_1$  

$x_2$  

Queries


$q_1$  

$q_2$  



$W^Q$

Keys

$k_1$  

$k_2$  

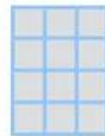


$W^K$

Values

$v_1$  

$v_2$  



$W^V$

attention score  
needs query  
and key



# why attention

- capturing context
- handling long-range dependencies
- parallel processing
- interpretable representations

Attention for Neural  
Networks, Clearly  
Explained!!! -  
YouTube

