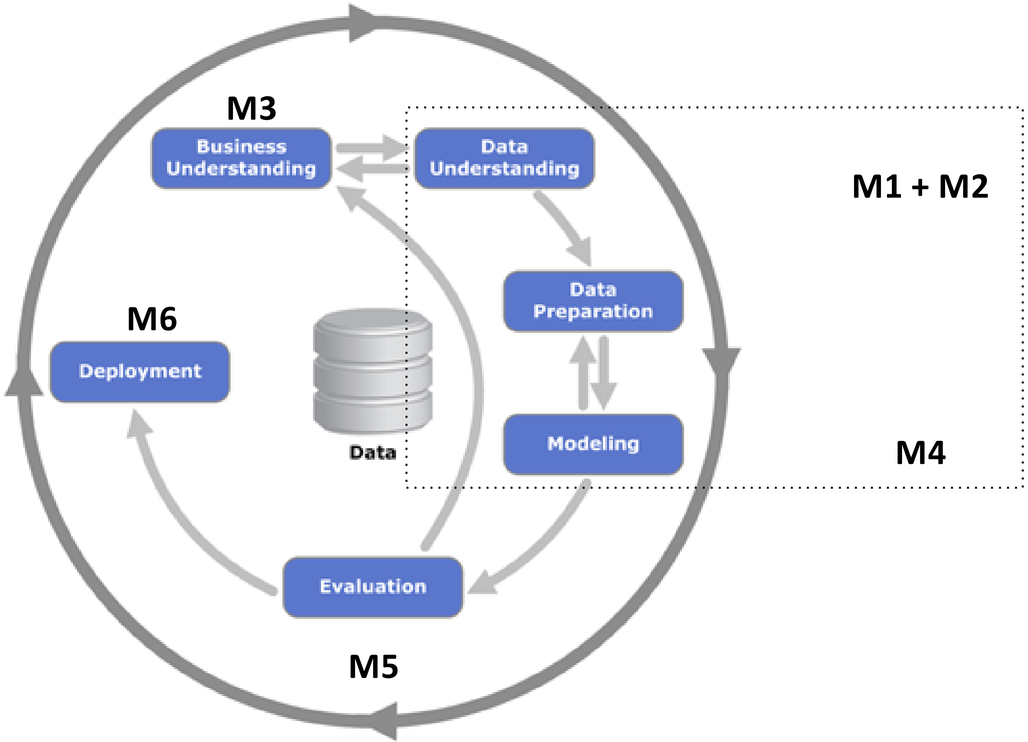
From Gradient to Attention

Hamid Bekamiri
Aalborg University Business School
Innovation, Knowledge, and Economic Development (IKE)
AI Denmark (AIDK)
hamidb@business.aau.dk
Feb 02, 2026

CRISP - DM



M3
Business Understanding
Data Understanding
M1 + M2
Data Preparation
M6
Deployment
Data
Modeling
M4
Evaluation
M5

M4: Applied DL and AI

Course Overview

This course explores advanced Deep Learning for NLP, progressing from foundational gradients to state-of-the-art Large Language Models. Students will gain hands-on experience with BERT, data pipelines, light agentic systems, and Graph-based Machine Learning.

Instructors: Hamid, Milad, Richard

Date	Time	Activity
Mon 02-02	12:30 - 16:15	Lecture **Session 1:** From Gradients to Attention
Fri 06-02	08:15 - 12:00	Workshop **Workshop 1:** Life2Vec to Market2Vec
Mon 09-02	08:15 - 12:00	Lecture **Session 2:** From Finetuning to Applications – BERT & SBERT (⚠ Deadline Assign. 1)
Mon 16-02	08:15 - 12:00	Lecture **Session 3:** Data Pipelines to Custom LLMs (⚠ Deadline Assign. 2)
Fri 20-02	08:15 - 12:00	Workshop **Workshop 2:** Fine-tuning
Mon 23-02	08:15 - 12:00	Lecture **Session 4:** Graph-based Machine Learning (⚠ Deadline Assign. 3)
Fri 27-02	12:30 - 14:15	Workshop **Workshop 3:** Workshop Session 3
Thu 05-03	-	**Final Assignment Due**
12 & 13-03	-	**Exam (Week 11)**

Assignments and Exam

Delivery

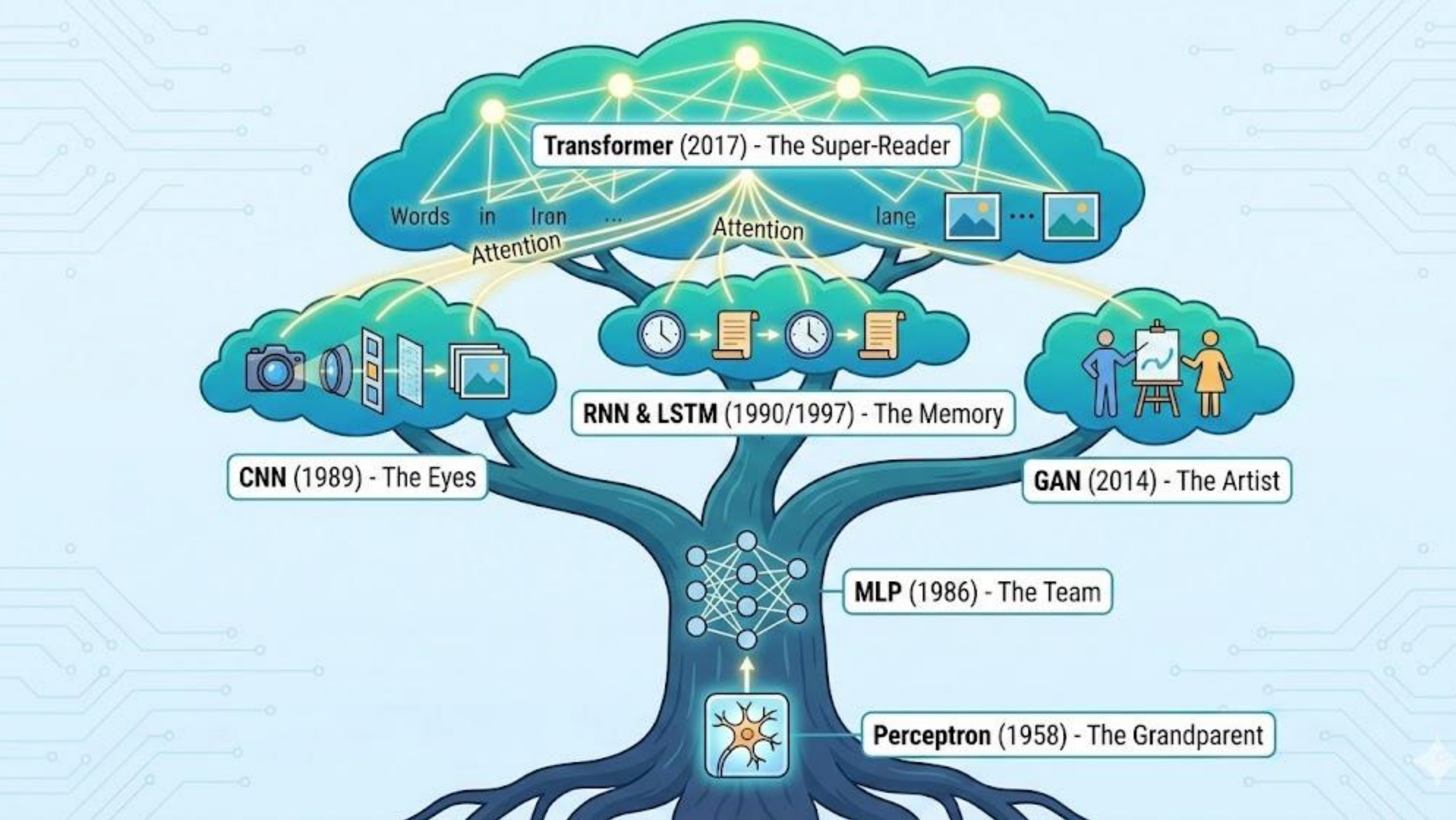
1. **GitHub Repository**
   - Create a repository containing your work.
   - Include a **README.md** with a brief description of your assignment and how to run the code/notebook.
2. **Colab Notebook**
   - Save your notebook in the repository.
3. **Group Work**
   - You may work in groups of up to 3 members.
   - Each group member's contribution should be briefly outlined in the README or the notebook.
4. **Technical Explainer Video**
   - Record a short (~5 minutes) **technical explainer video** presenting your main ideas, methodology, and results.
   - You may use Panopto, OBS Studio, Loom, or any other screen-recording tool.
   - Include the video link in your submission.
5. **Submission**
   - Send an email to Hamid (hamidb@business.aau.dk) with the link to your GitHub repository (and video) by the deadline.

Exam info

A prerequisite for participating in the exam is that the student has handed in written material.

The exam in Applied Deep Learning and Artificial Intelligence is an oral individual examination. based on a submitted assignment portfolio.
It is an internal examination with a 7-point grading scale and takes place **on March 12 and 13th , 2026.**

The submission date for the complete portfolio is March 5, at 10:00 AM, 2026.

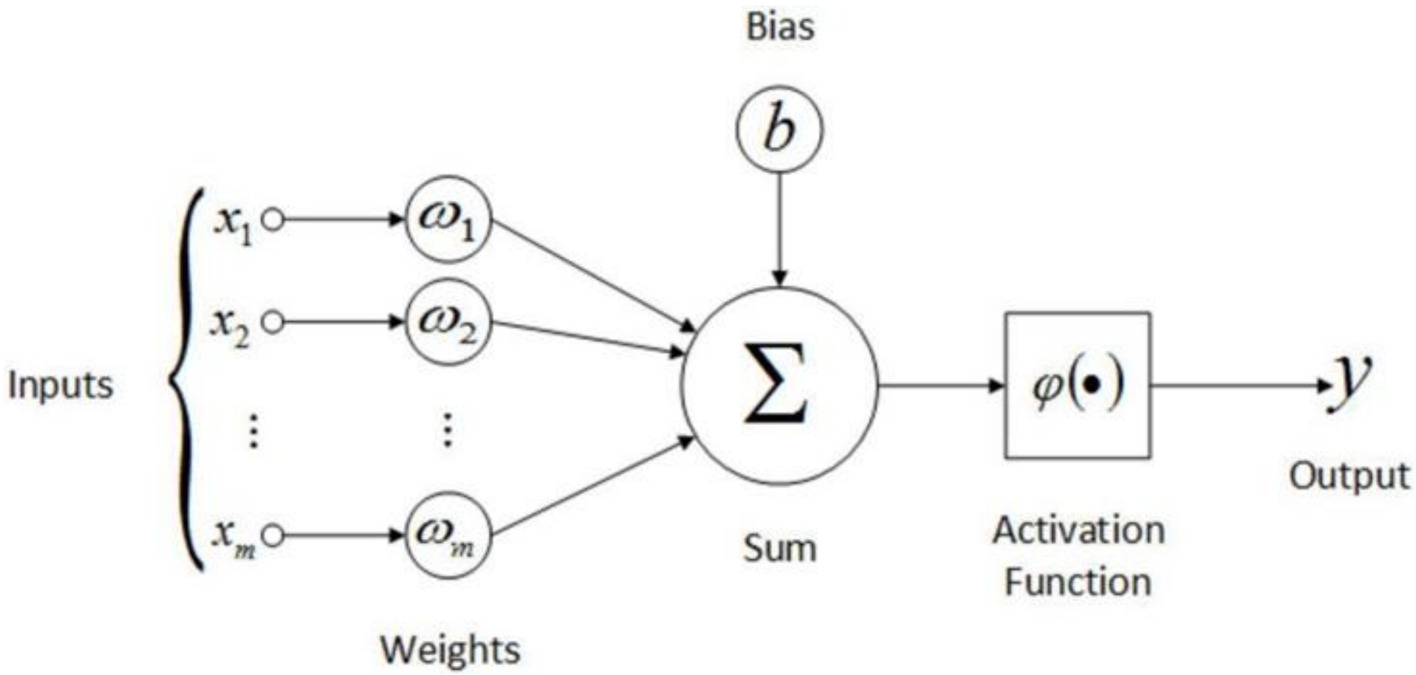


Transformer (2017) - The Super-Reader
Words
in
Iran
...
lang
Attention
Attention
RNN & LSTM (1990/1997) - The Memory
CNN (1989) - The Eyes
GAN (2014) - The Artist
MLP (1986) - The Team
Perceptron (1958) - The Grandparent

Deep Learning

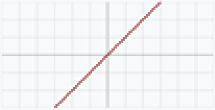
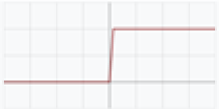
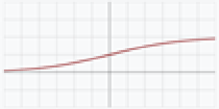
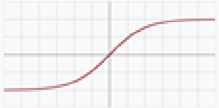
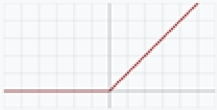
1. **What is the structure of an Artificial Neural Network?**
2. **How to train an Artificial Neural Network?**
   - 2.1 How can we assess the performance of our model?
   - 2.2 What methods can we use to determine the optimal values for parameters like weights and biases?
   - 2.3 How feasible is it to find the best parameter values when dealing with a massive number of parameters, such as 10 million?
   - 2.4 Can you highlight the differences between Batch Gradient Descent and Stochastic Gradient Descent in the context of Machine Learning?
3. **Why do we need Deep Learning frameworks like TesnsorFlow?**
   - 3.1 What is the procedure for constructing a Neural Network that encompasses various layers, including input, hidden, and output layers?
   - 3.2 What strategies can be employed to mitigate the issue of overfitting in a complex neural network?
   - 3.3 How to save and load a trained model using Torch?

1. What is the structure of an Artificial Neural Network?



Activation Function: A "gatekeeper" decides whether or not to pass a signal, and how strongly.

Activation Function: A Gatekeeper

Name	Plot	Function, $f(x)$	Derivative of f, $f'(x)$	Range
Identity		x	1	$(-\infty, \infty)$
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$\begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1+e^{-x}}$ [1]	$f(x)(1 - f(x))$	$(0, 1)$
tanh		$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - f(x)^2$	$(-1, 1)$
Rectified linear unit (ReLU)[11]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$	$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$

Activation Function: A "gatekeeper" decides whether or not to pass a signal, and how strongly.

2. How to train an Artificial Neural Network?

Build a NN	1. Creating a Feedforward Neural Network - 1.1. Structure (Architecture) of NN - 1.2. Loss Function - 1.3. Optimization Approach
Training Loop (Steps 2 through 5)	2. Forward Pass 3. FeedForward Evaluation 4. Backward Pass / Gradient Calculation 5. Back Propagation / Update Weights

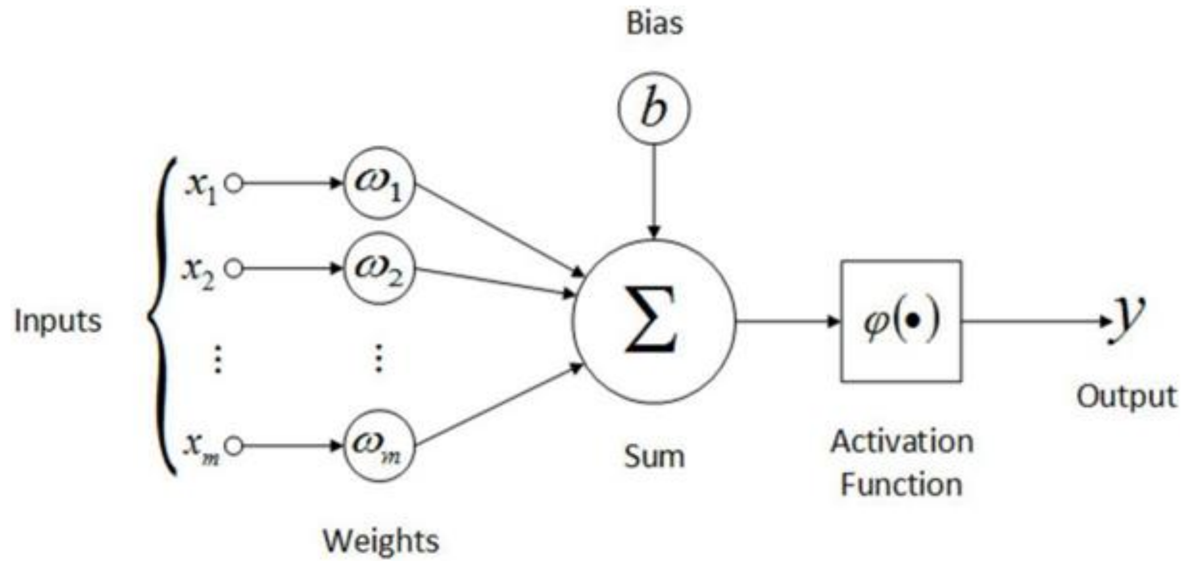
1.1. Structure (Architecture) of NN

Activation function Bias

$$\hat{y} = \sigma(w_0 + \textstyle\sum_{i=1}^{n} w_i x_i)$$

$$\hat{y} = \sigma(w_0 + W^T X)$$

$$\hat{y} = \sigma(Z)$$



1.2. Loss Function

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}|$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2$$

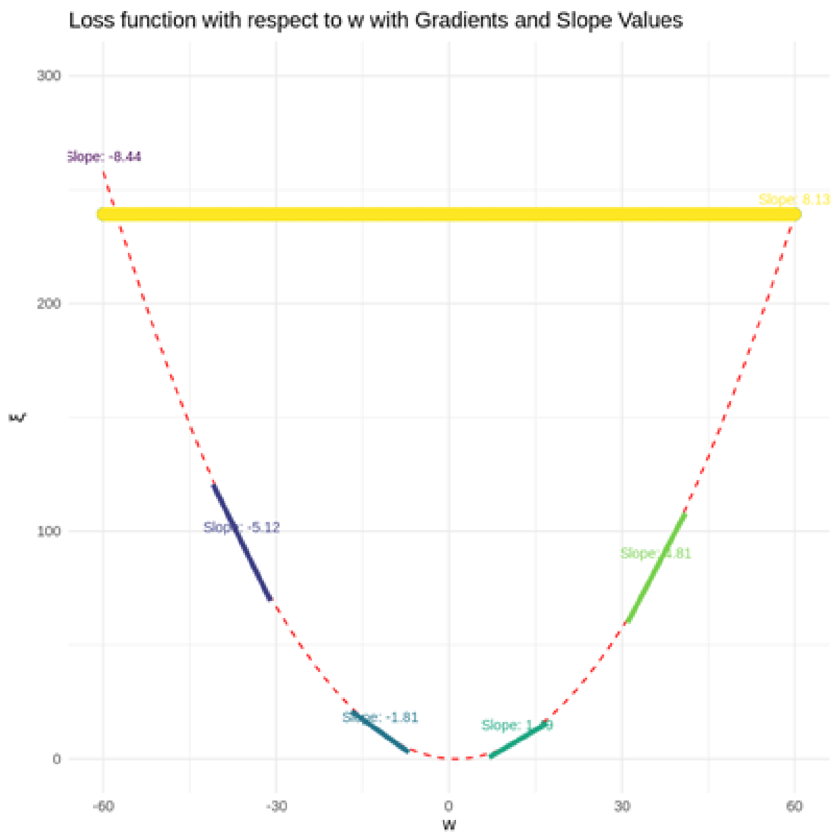
$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2}$$

$$R^2 = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2}$$

Where,

$\hat{y}$ − *predicted value of y*

$\bar{y}$ − *mean value of y*



1.3. Optimization Approach: From Slope to Direction

In mathematics, a derivative measures the rate at which a function changes at a specific point.The process of finding a derivative is called **differentiation**.

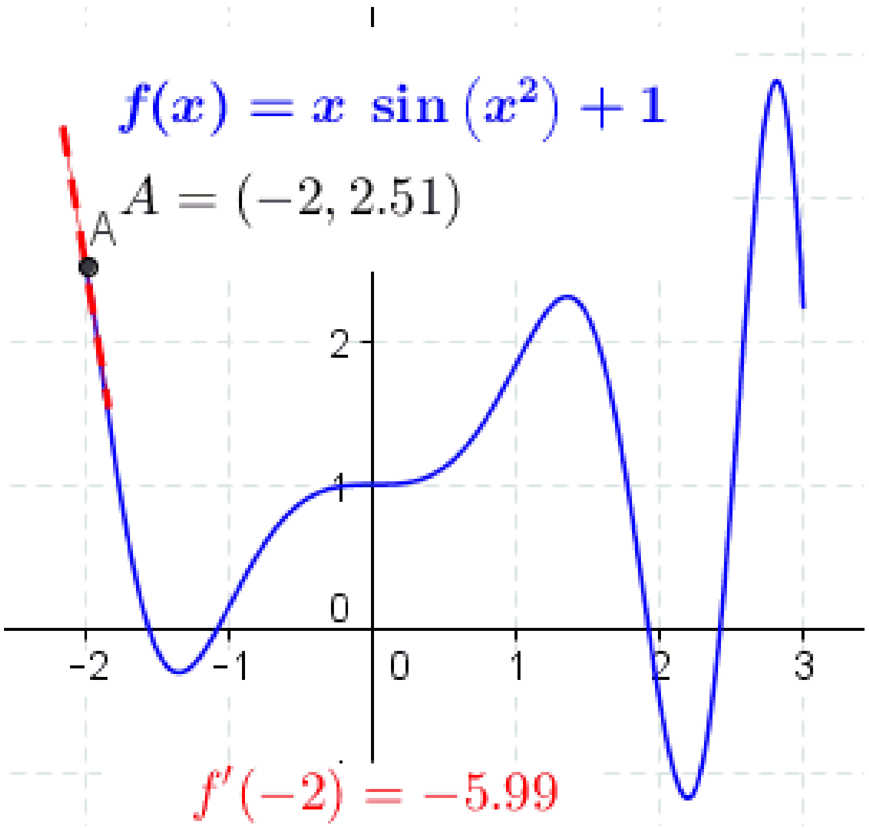
$$\frac{dy}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Applications of derivatives

- **Rate of change:** Used in physics to calculate velocity (the first derivative of position) and acceleration (the second derivative of position).

- **Optimization:** Finding the maximum and minimum values of a function to solve problems in business, engineering, and science.

Differencing is the **discrete** analog of **differentiation**.



Join at
slido.com
#2706 791

If the rate of change is positive, what is happening to the slope?

A) It is going uphill (rising) 📈
0%

B) It is going downhill (falling) 📉
0%

C) It is flat (no change) ↔️
0%

If the rate of change is negative, what does this tell us?

A) We are climbing higher

0%

B) We are going downhill (falling) 📉

0%

C) We are at the highest possible point

0%

Join at

slido.com

#2706 791

Join at

slido.com

#2706 791

If the rate of change is zero, what is the state of the slope?

A) It is at its steepest point

0%

B) it is dropping quickly

0%

C) It is perfectly flat (no movement) ↔️

0%

Forward Pass - Prediction

The forward pass is the process of predicting the output value given the input.

Backward Pass - Gradient Calculation

Backward Pass / Gradient Calculation

For a linear regression model with ($y_i = w_i x_i$), the gradient of the MSE with respect to the weight (w) is:

$$\frac{\partial L}{\partial w} = \frac{2}{N} \sum_{i=1}^{N} x_i(y_i - t_i)$$

Explanation:

1. The derivative of $(y_i - t_i)^2$ with respect to (y_i) is $2(y_i - t_i)$.
2. For ($y_i = w_i x_i$), the derivative of (y_i) with respect to (w_i) is (x_i).
3. Using the chain rule, the derivative of the loss with respect to (w_i) is ($2(y_i - t_i) \times x_i$).
4. Averaging over all data points gives the gradient: $\frac{\partial L}{\partial w} = \frac{2}{N} \sum_{i=1}^{N} x_i(y_i - t_i)$

This gradient provides direction and magnitude to adjust (w) to minimize the loss during gradient descent.

Chain Rule

Back Propagation - Update Weights

Back Propagation / Update Weights

The gradient descent formula is used to update the parameters of a model in order to minimize a cost or loss function. It's an iterative process that adjusts the parameters in the direction that reduces the cost function. The formula is as follows:

$$w_{\text{new}} = w_{\text{old}} - \alpha \cdot \nabla J(w_{\text{old}})$$

Where:

- (w_{new}) is the updated parameter vector.
- (w_{old}) is the current parameter vector.
- (α) is the learning rate, determining the step size in each iteration.
- $\nabla J(w_{\text{old}})$ is the gradient of the cost or loss function (L) with respect to the parameters w at the current values w_{old}.

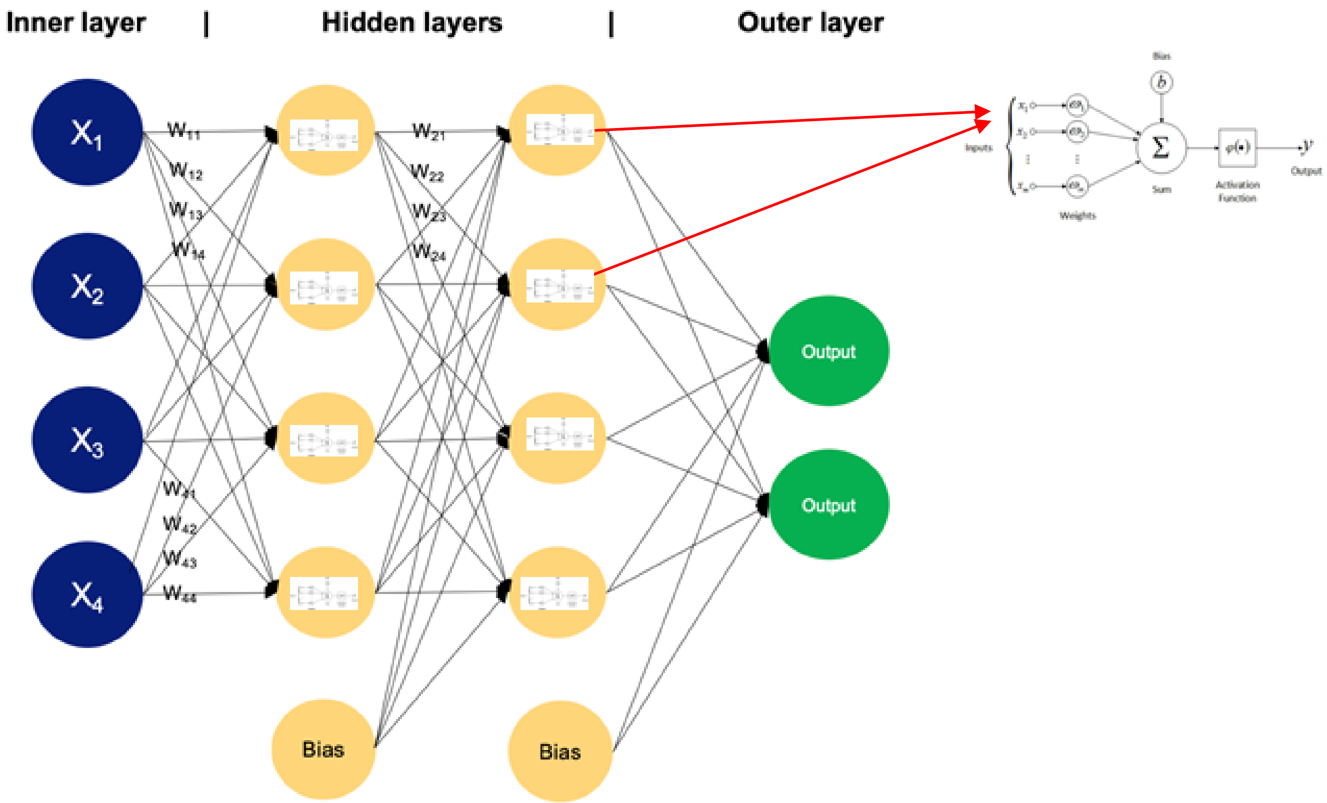
This formula is used iteratively until convergence to find the parameter values that minimize the cost function.

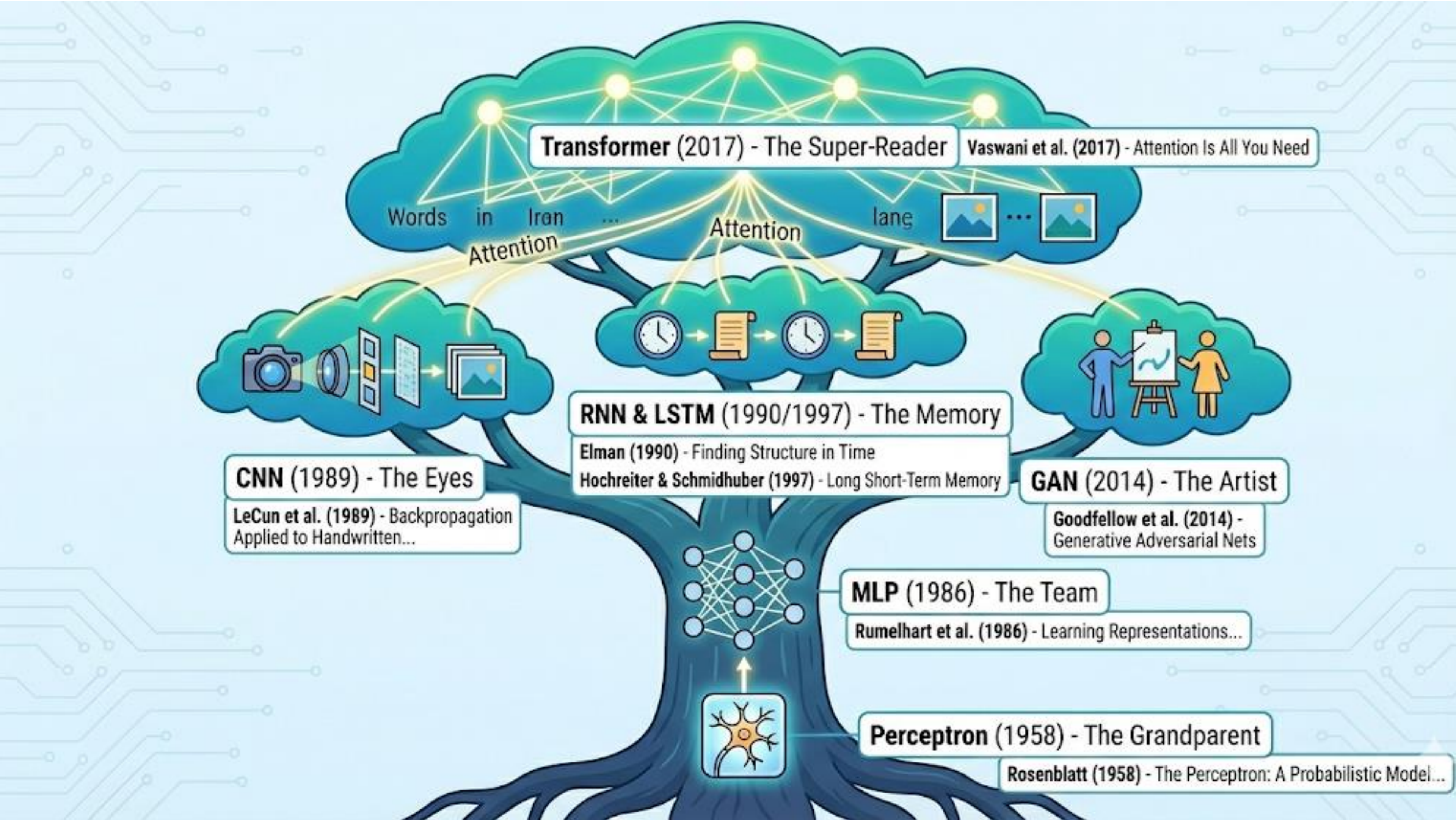
Different Type of Training Loop

There are three main types of gradient descent:

- **Batch Gradient Descent:** Batch Gradient Descent computes the gradient of the cost function with respect to the parameters for the entire training dataset. Computationally efficient when the dataset fits in memory because it can benefit from vectorized operations. Can be very slow for large datasets
- **Stochastic Gradient Descent:** Stochastic Gradient Descent (SGD) computes the gradient and updates the parameters for each training example one at a time. Can handle large datasets since it only requires one training example in memory at a time. Less accurate convergence. The path to the minimum is noisy compared to Batch Gradient Descent.
- **Mini-Batch Gradient Descent:** Mini-Batch Gradient Descent computes the gradient of the cost function and updates the parameters using a subset of the training data, rather than the entire dataset or a single training example. Faster computation than Batch Gradient Descent, as it doesn't need to process the entire dataset before making updates. The mini-batch size is an additional hyperparameter to tune, and finding the optimal size can be challenging.

MultiLayer Perceptron (MLP)



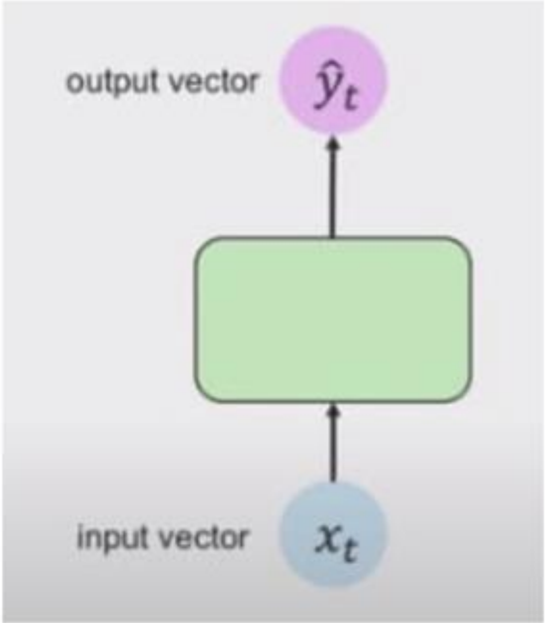


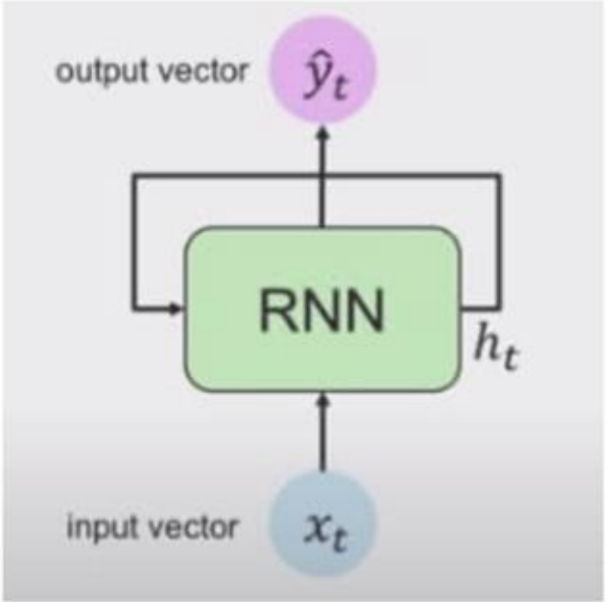
Transformer (2017) - The Super-Reader
Vaswani et al. (2017) - Attention Is All You Need
Words in Iron ...
lang
Attention
Attention
RNN & LSTM (1990/1997) - The Memory
Elman (1990) - Finding Structure in Time
Hochreiter & Schmidhuber (1997) - Long Short-Term Memory
CNN (1989) - The Eyes
LeCun et al. (1989) - Backpropagation Applied to Handwritten...
GAN (2014) - The Artist
Goodfellow et al. (2014) - Generative Adversarial Nets
MLP (1986) - The Team
Rumelhart et al. (1986) - Learning Representations...
Perceptron (1958) - The Grandparent
Rosenblatt (1958) - The Perceptron: A Probabilistic Model...

RNN & LSTM

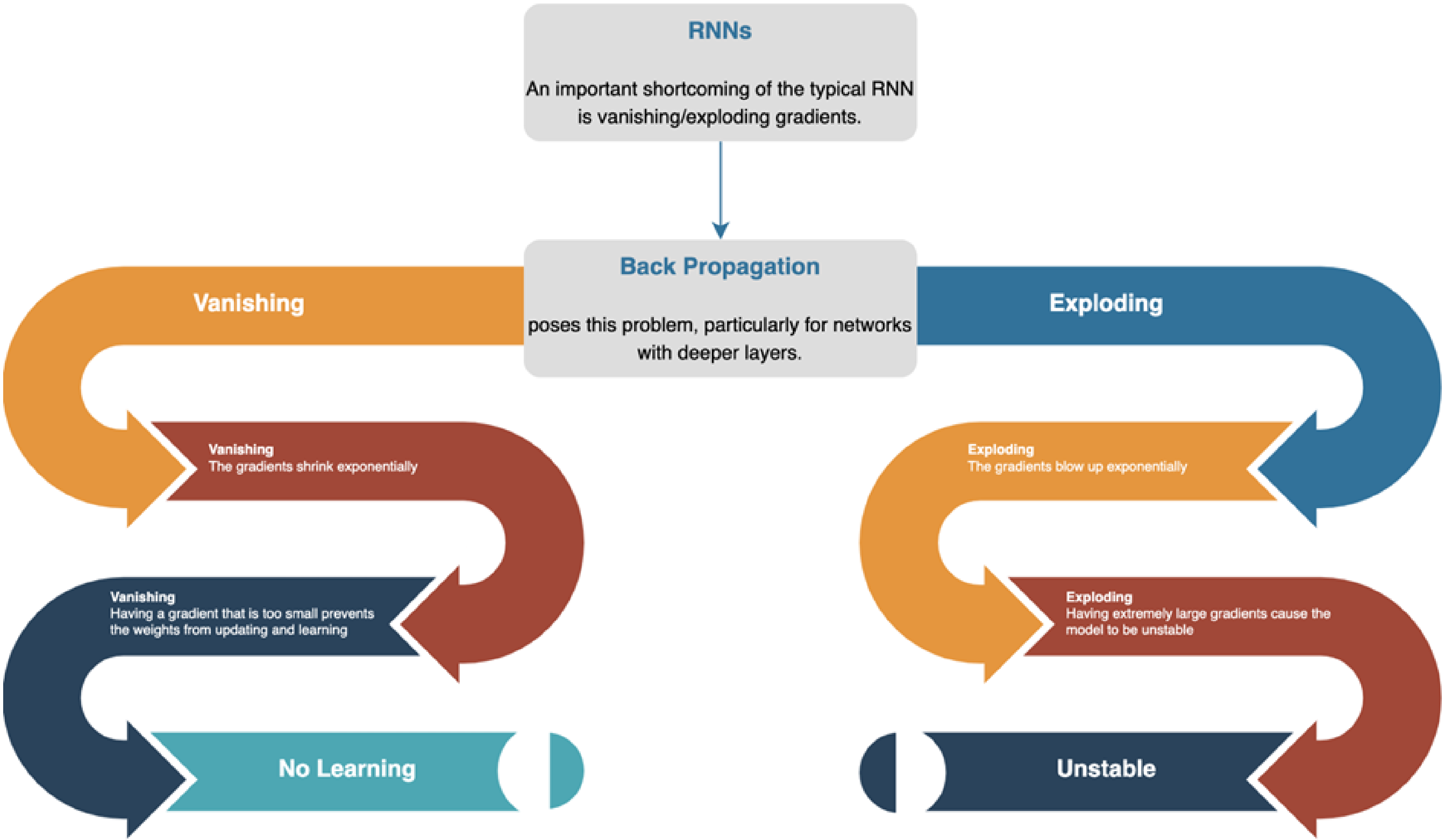
In an MLP, each word would be treated as a separate input and would be processed through separate hidden layers. There is no way for the network to share information across words in the sequence, such as information about the relationship between words or about common features that occur across different parts of the sequence.

Recurrent Neural Network

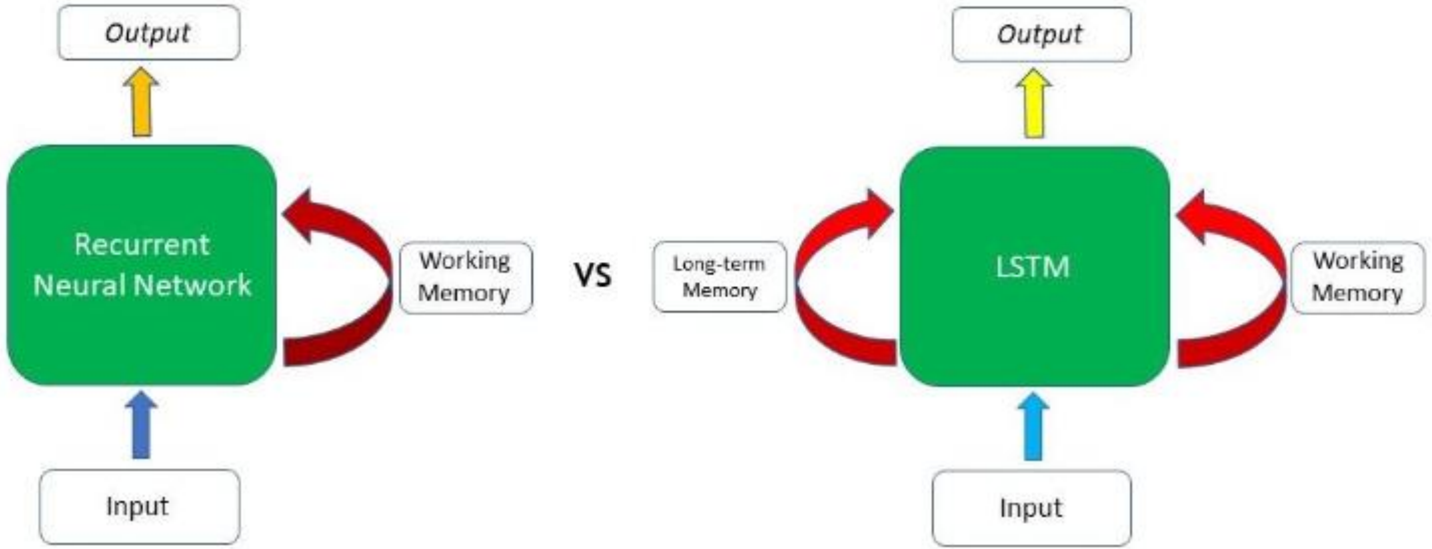




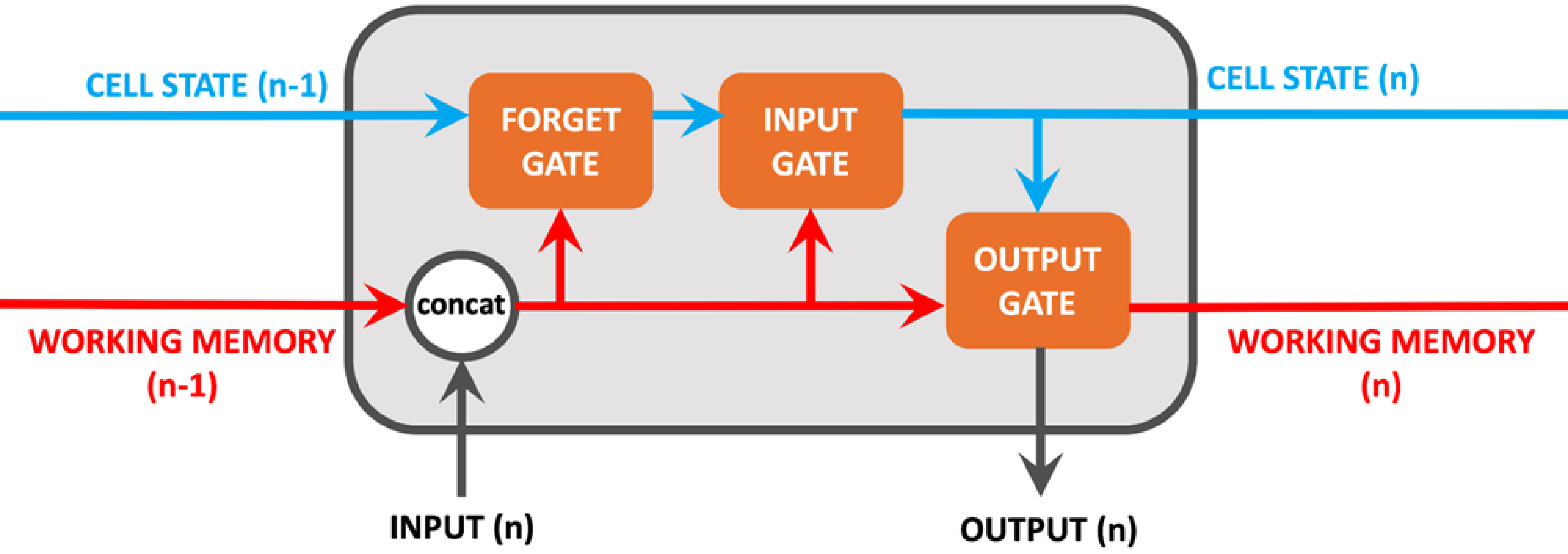
Simple Neural Network VS RNN


RNNs
An important shortcoming of the typical RNN is vanishing/exploding gradients.
Back Propagation
poses this problem, particularly for networks with deeper layers.
Vanishing
Exploding
Vanishing
The gradients shrink exponentially
Exploding
The gradients blow up exponentially
Vanishing
Having a gradient that is too small prevents the weights from updating and learning
Exploding
Having extremely large gradients cause the model to be unstable
No Learning
Unstable

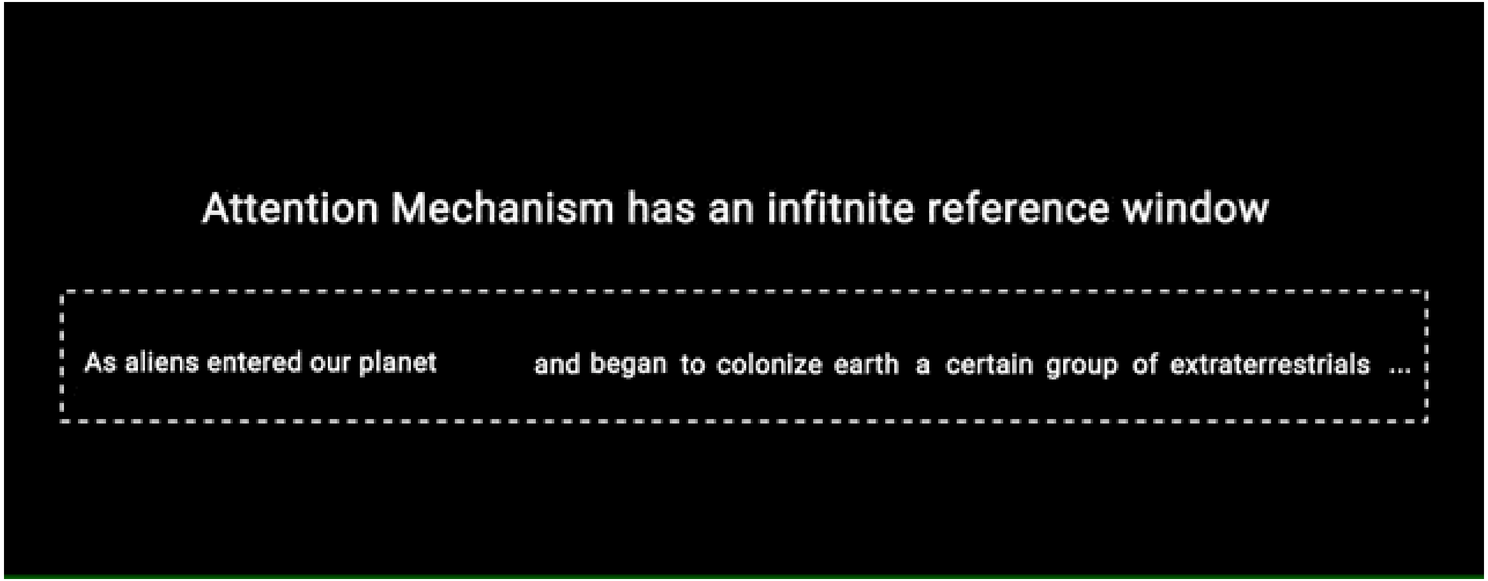
Long Short-Term Memory



Gate Mechanism



How Do Attention Mechanisms Solve RNN and LSTM Challenges?



Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

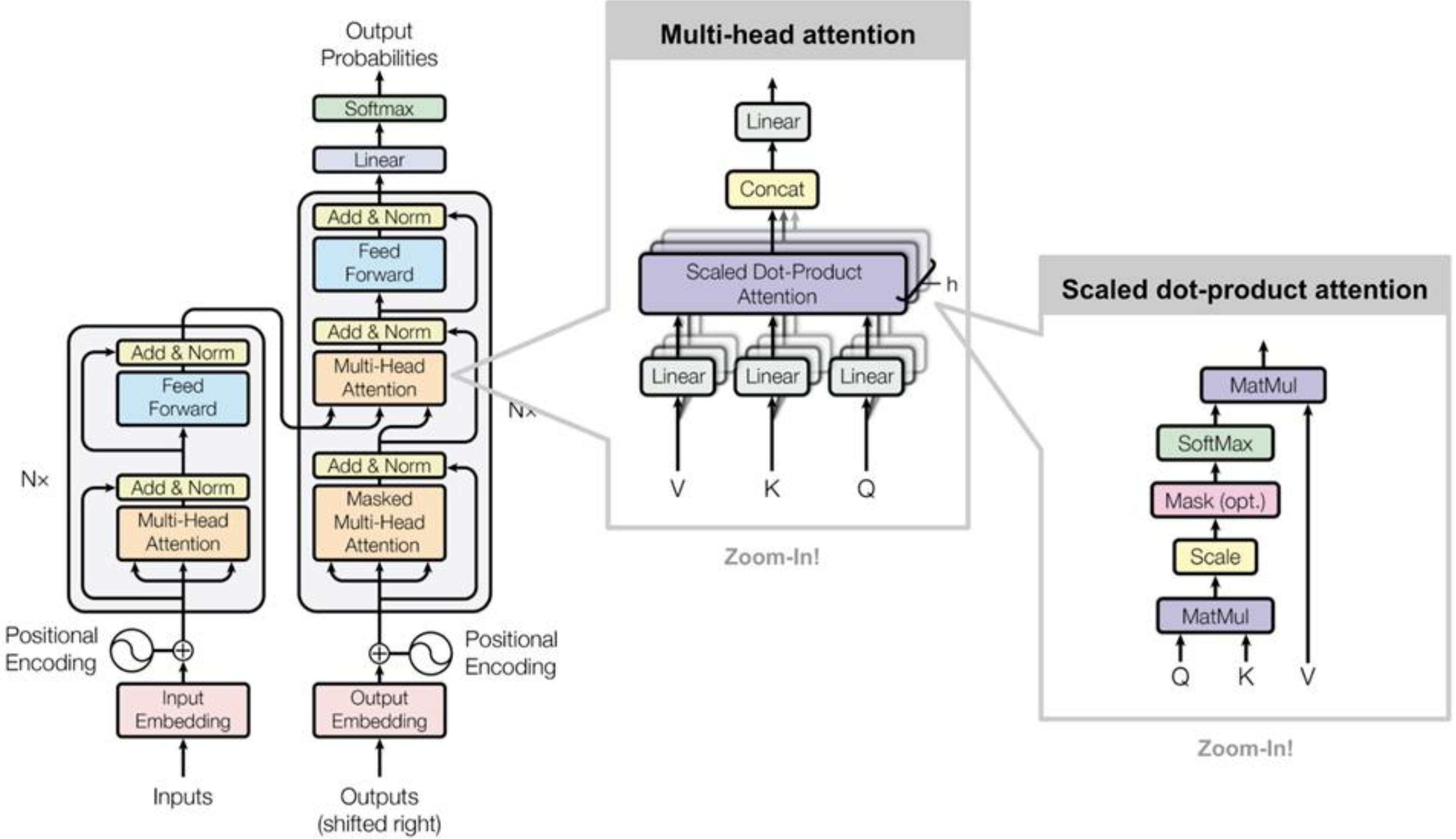
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Multi-head attention
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q
Zoom-In!
Scaled dot-product attention
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Zoom-In!

HOMOGRAPH

SAME SPELLING, DIFFERENT MEANINGS. MAY SOUND DIFFERENT.

- MAY BE PRONOUNCED THE SAME OR DIFFERENTLY

lead
to guide

tear
a drop from the eye

ESTIMATED AMOUNT:
Estimates vary. Studies suggest hundreds of common pairs, with many words having multiple meanings.

HOMONYM

SAME SPELLING, SAME SOUND, DIFFERENT MEANINGS.

BAT
an animal

BAT
used in sports

ESTIMATED AMOUNT:
Broadly defined, over 6,000. Strictly defined (same spelling & sound), common pairs are in the hundreds.

@Phonics Tutor

HOMOPHONE

SAME SOUND, DIFFERENT SPELLING & MEANING.

TWO
2

TOO
also

FLOWER
a plant

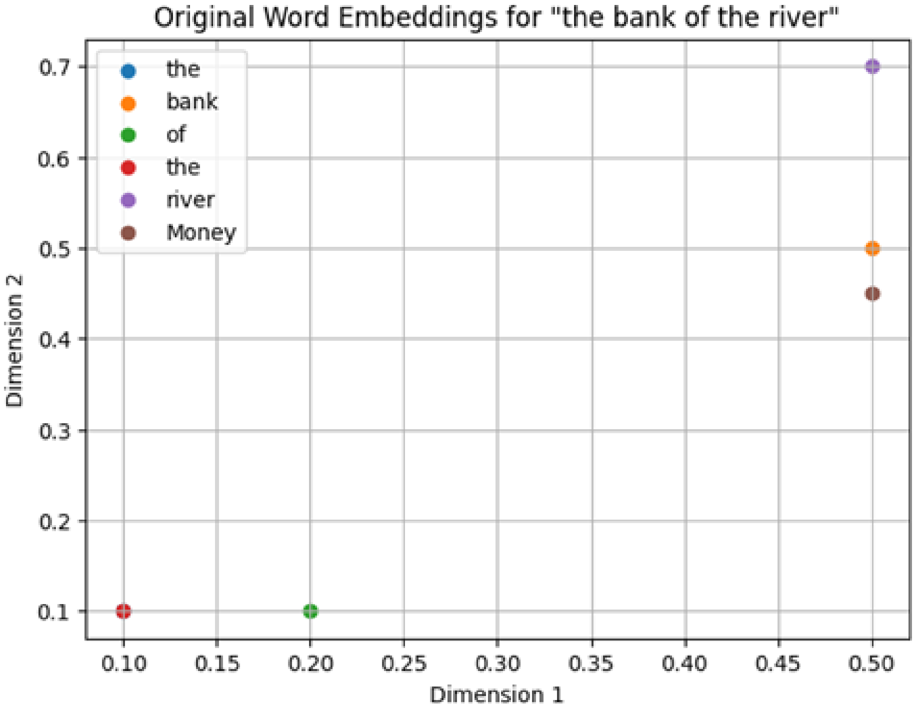
FLOUR

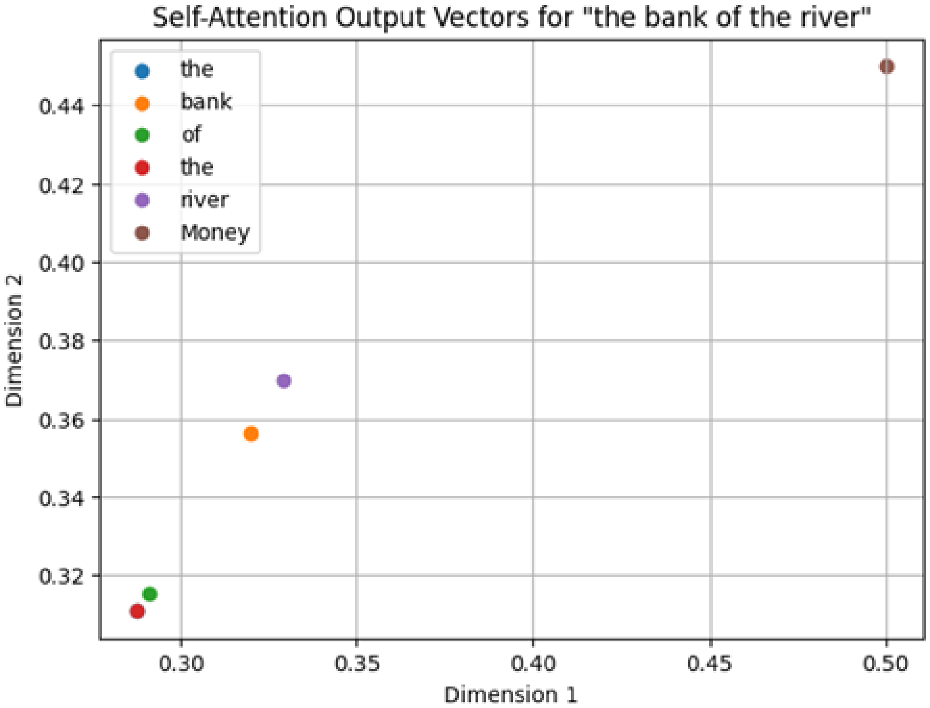
ESTIMATED AMOUNT:
Estimated over 6,000. Homophony is common in English.

Attention:

Telling context in words

Attention Is All You Need!





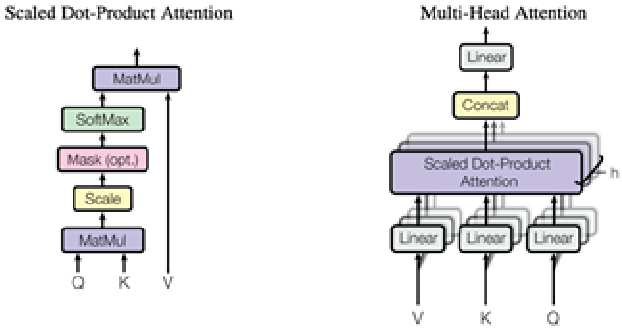


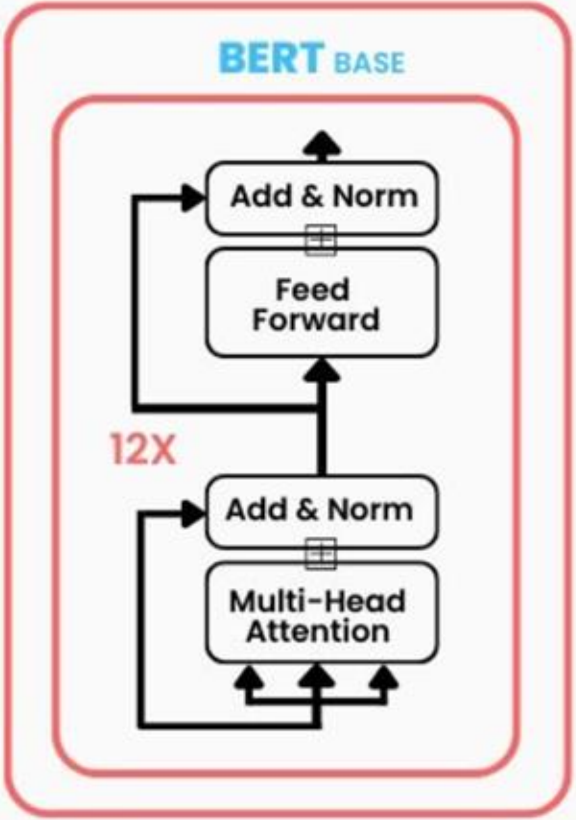
Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values.

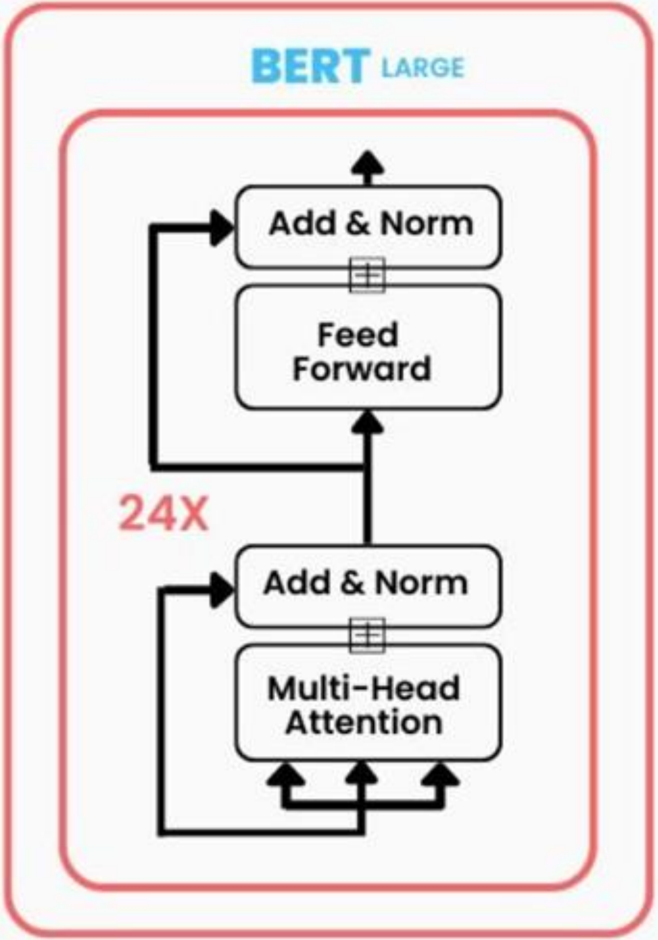
In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. We compute the matrix of outputs as:

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

The two most commonly used attention functions are additive attention [2], and dot-product (multi-plicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of $\frac{1}{\sqrt{d_k}}$. Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is

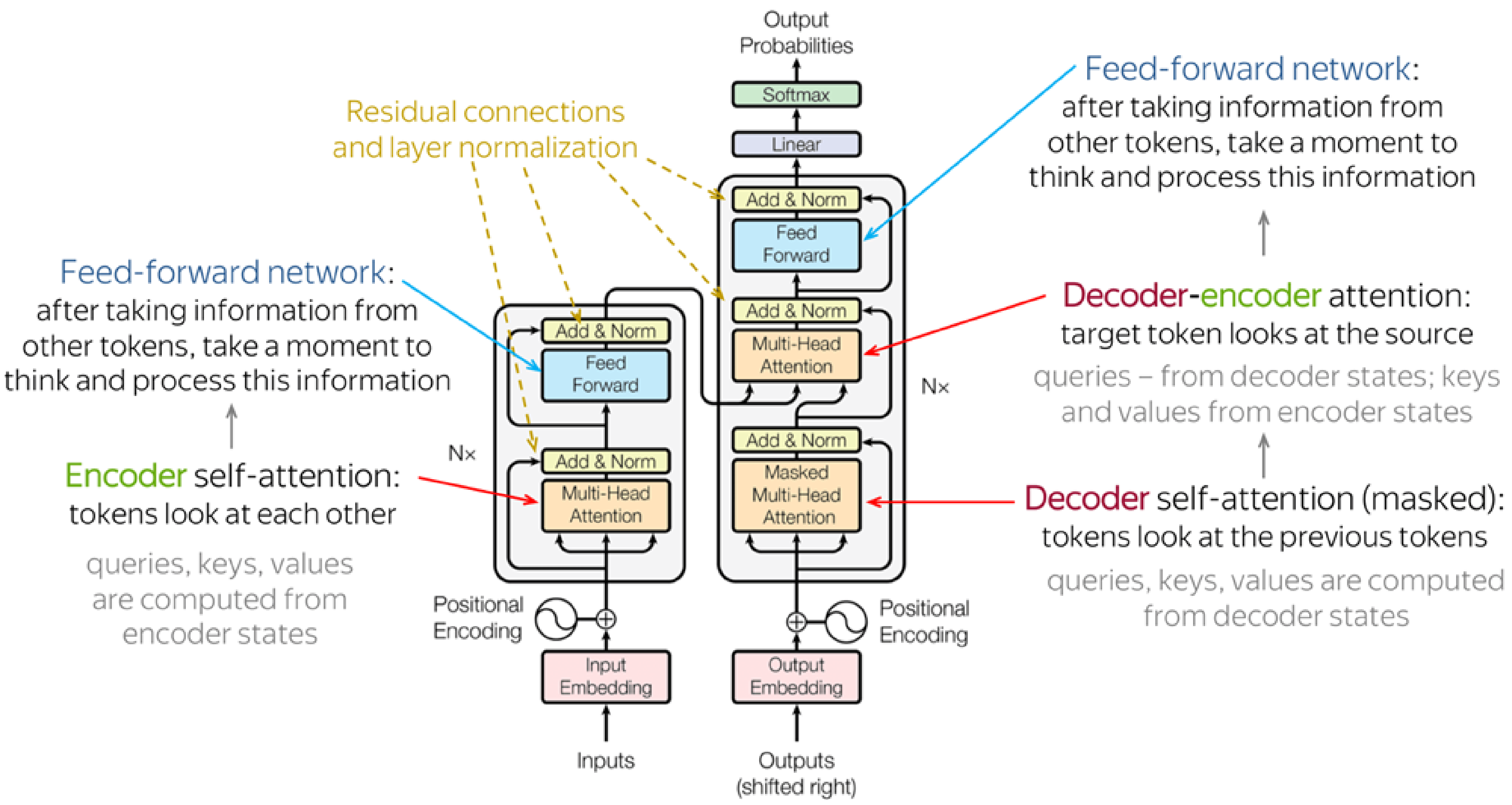


110M Parameters



340M Parameters

Comparison	BERT October 11, 2018	RoBERTa July 26, 2019	DistilBERT October 2, 2019	ALBERT September 26, 2019
Parameters	**Base:** 110M **Large:** 340M	**Base:** 125 **Large:** 355	**Base:** 66	**Base:** 12M **Large:** 18M
Layers / Hidden Dimensions / Self-Attention Heads	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16	**Base:** 6 / 768 / 12	**Base:** 12 / 768 / 12 **Large:** 24 / 1024 / 16
Training Time	**Base:** 8 x V100 x 12d **Large:** 280 x V100 x 1d	1024 x V100 x 1 day (4-5x more than BERT)	**Base:** 8 x V100 x 3.5d (4 times less than BERT)	[not given] **Large:** 1.7x faster
Performance	Outperforming SOTA in Oct 2018	88.5 on GLUE	97% of BERT-base's performance on GLUE	89.4 on GLUE
Pre-Training Data	BooksCorpus + English Wikipedia = 16 GB	BERT + CCNews + OpenWebText + Stories = 160 GB	BooksCorpus + English Wikipedia = 16 GB	BooksCorpus + English Wikipedia = 16 GB
Method	Bidirectional Transformer, MLM & NSP	BERT without NSP, Using Dynamic Masking	BERT Distillation	BERT with reduced parameters & SOP (not NSP)


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Positional Encoding
Output Embedding
Outputs (shifted right)
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Input Embedding
Inputs
Residual connections and layer normalization
Feed-forward network: after taking information from other tokens, take a moment to think and process this information
Decoder-encoder attention: target token looks at the source
queries – from decoder states; keys and values from encoder states
Decoder self-attention (masked): tokens look at the previous tokens
queries, keys, values are computed from decoder states
Feed-forward network: after taking information from other tokens, take a moment to think and process this information
Encoder self-attention: tokens look at each other
queries, keys, values are computed from encoder states