jax LONDON
by devmio

From Zero to Kotlin GPT

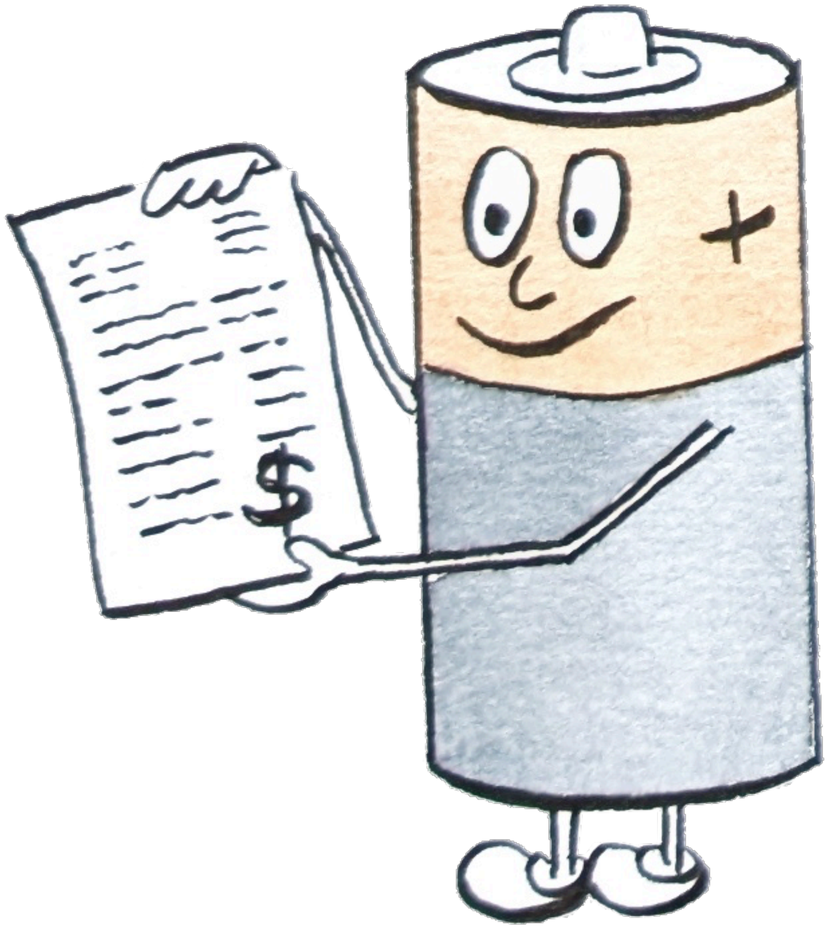
in 45 Minutes

Michal Harakal

Deutsche Telekom AG, 2024

Let's talk about ChatGPT & Co first

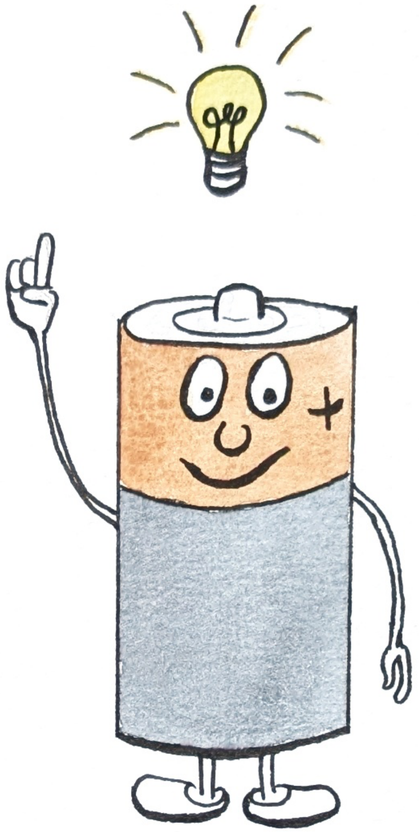
Let's talk about ChatGPT & Co first



(C) Adriana Harakalova, 2024

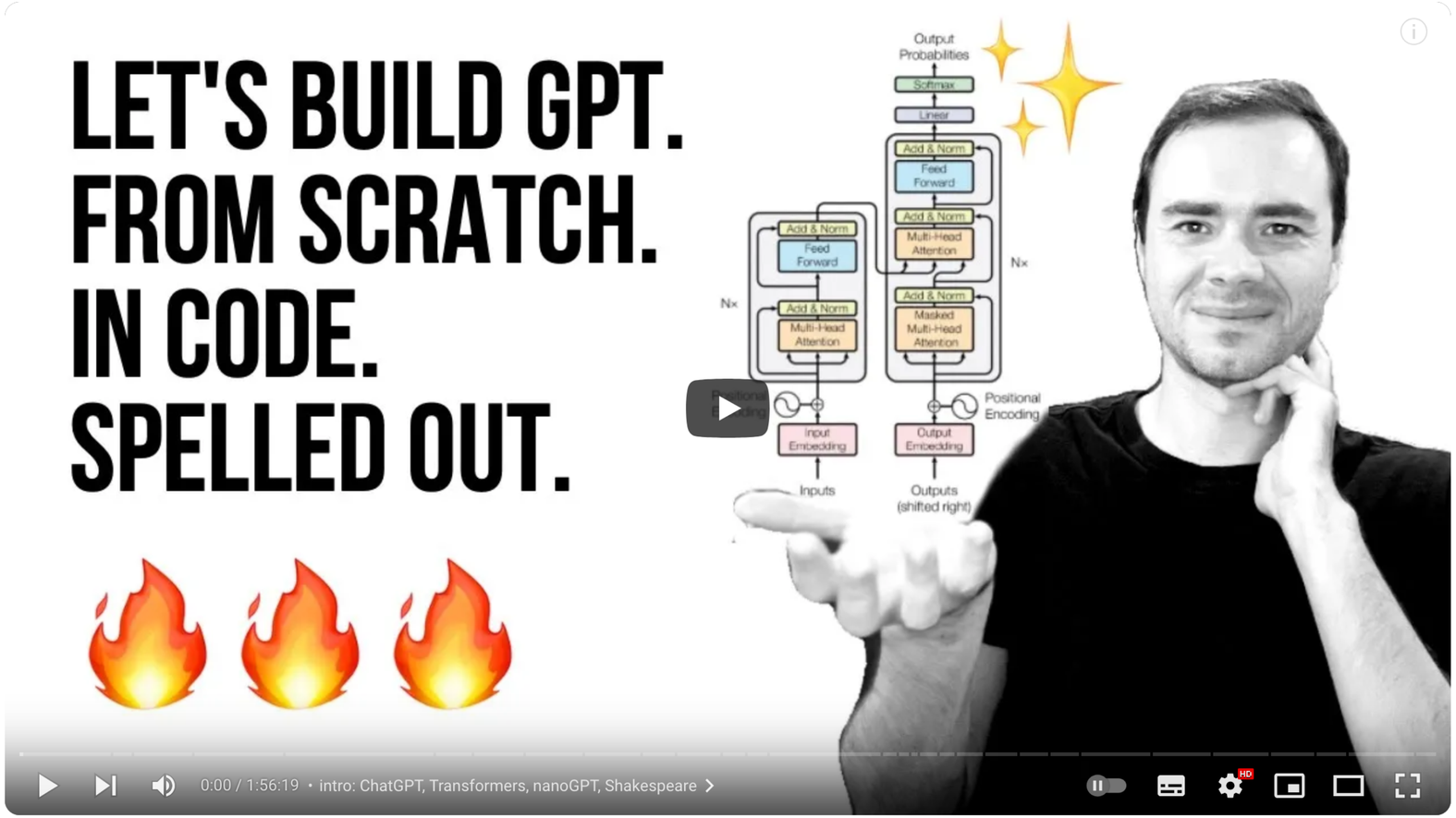
What I cannot create,
I do not understand.
Know how to solve every problem that has been solved.

Richard Feynman, 1988



YouTube DE

Suchen



Let's build GPT: from scratch, in code, spelled out.

Andrej Karpathy
540.000 Abonnenten

Abonnieren

107.486 Teilen Speichern

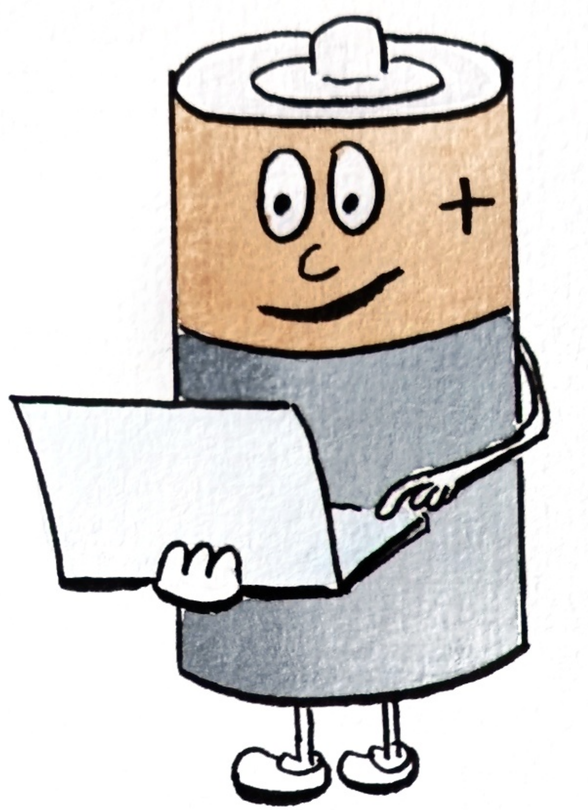
https://www.youtube.com/watch?v=kCc8FmEb1nY

Let's build GPT: from scratch, in code, spelled out.

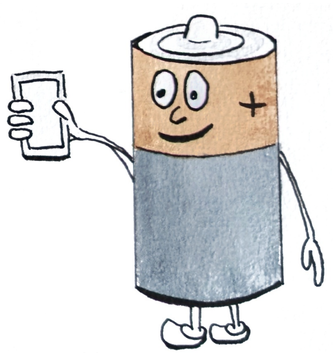
Andrej Karpathy

- Simplified GTP-2 model with approx. 10 million parameters
- 250 lines of Python code
- uses PyTorch
- decoder only

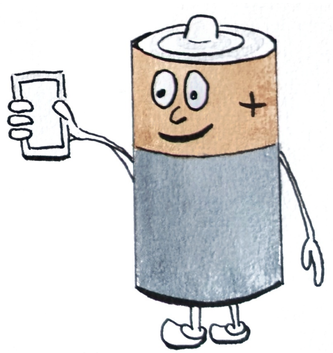
Kotlin



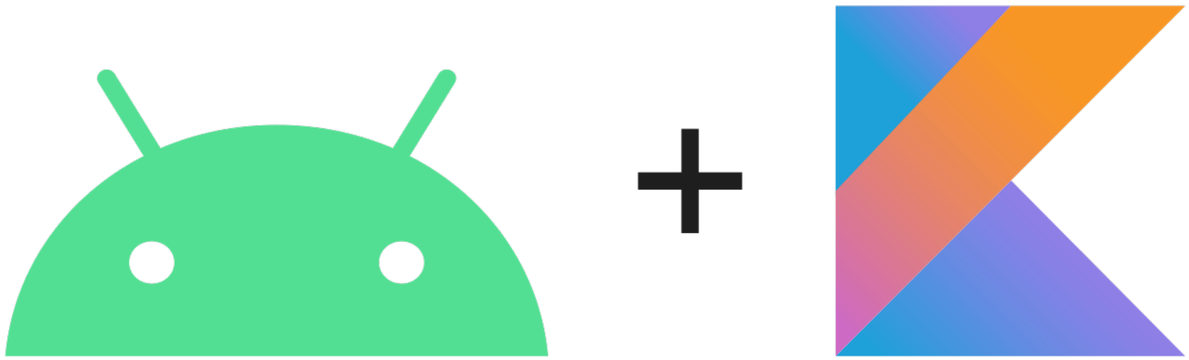
(C) Adriana Harakalova, 2024

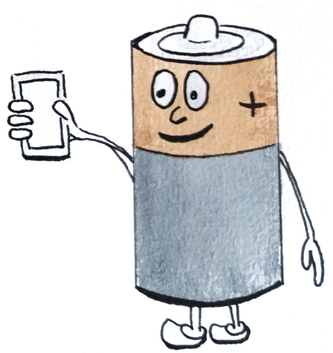


(C) Adriana Harakalova, 2024



(C) Adriana Harakalova, 2024





(C) Adriana Harakalova, 2024

Kotlin Multiplatform

Source code sharing

- Cross-platform development
- Support for native performance - Kotlin Native
- Cross-platform UI with Compose Multiplatform

AI/ML with Java and JVM

Eclipse DeepLearning4J

- Comprehensive Deep Learning Framework: Provides tools like SameDiff, **ND4J**, and DataVec for machine learning on the JVM.
- **Integration Support**: Compatible with models from TensorFlow, Keras, and more.
- Build upon native c++ and CUDA libraries for speed
- Open-Source & Community-Driven: Licensed under Apache 2.0 with active community contributions. open governance at the **Eclipse foundation**.

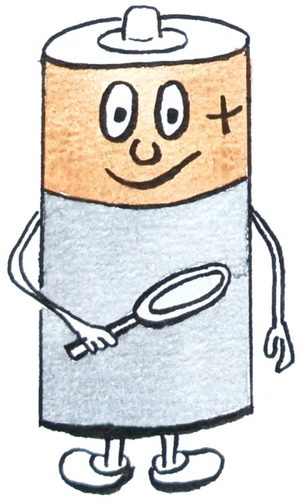
https://deeplearning4j.konduit.ai/

AI/ML with Java and JVM

Forecast

- **Project Babylon**
  - https://openjdk.org/projects/babylon/articles/auto-diff
- **Project Panama**
  - https://openjdk.org/projects/panama/
- **HAT**- Heterogeneous Accelerator Toolkit (memory structuces off the heap) using project panama for memory and functional API

Generative
Pre-trained
Transformers

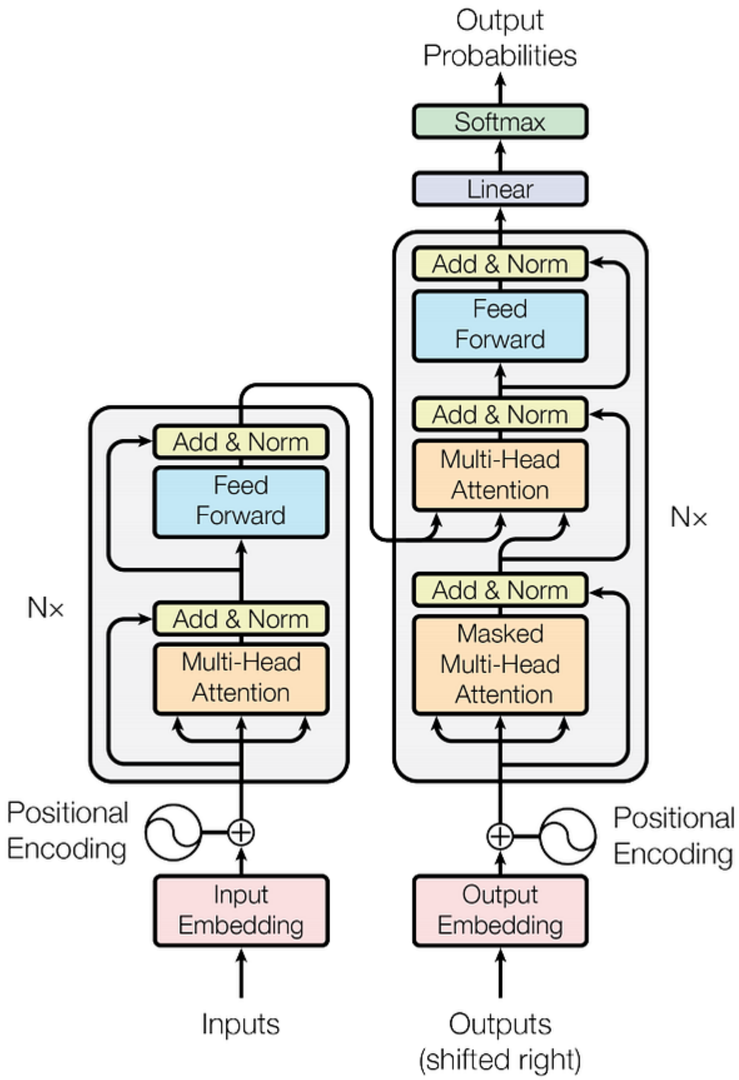


(C) Adriana Harakalova, 2024

Transformers

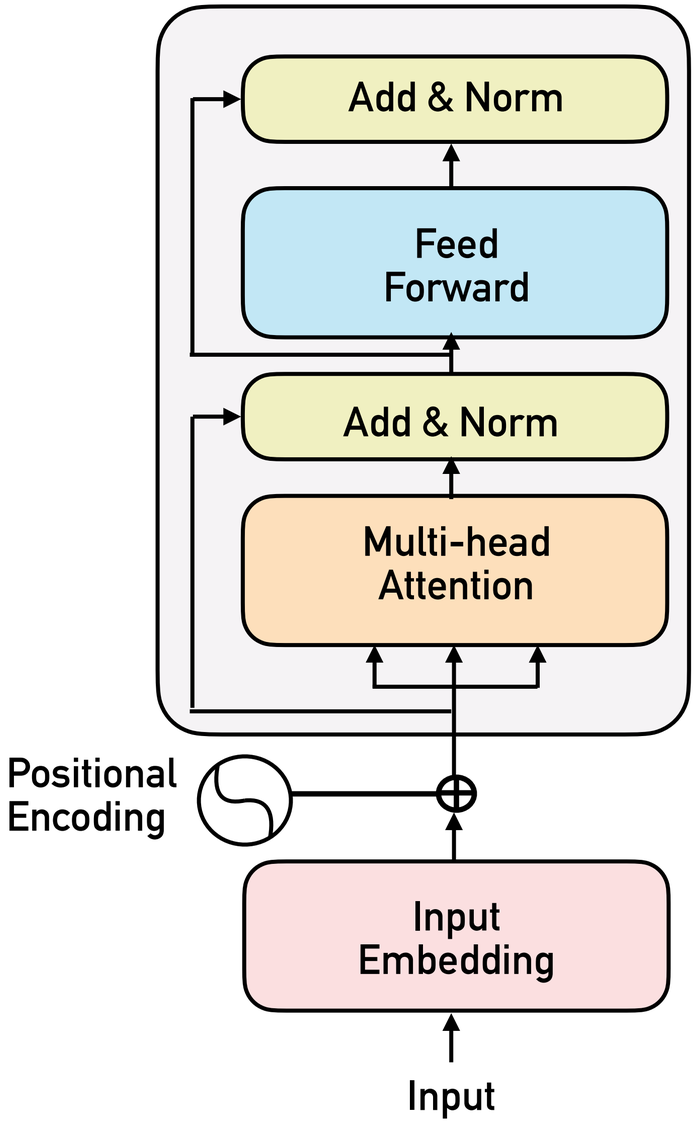
Attention Is All You Need

- Published 2017
- Google Brain, Google Research et al.
- Transformer model architecture
- Efficient computation as goal
- Translations

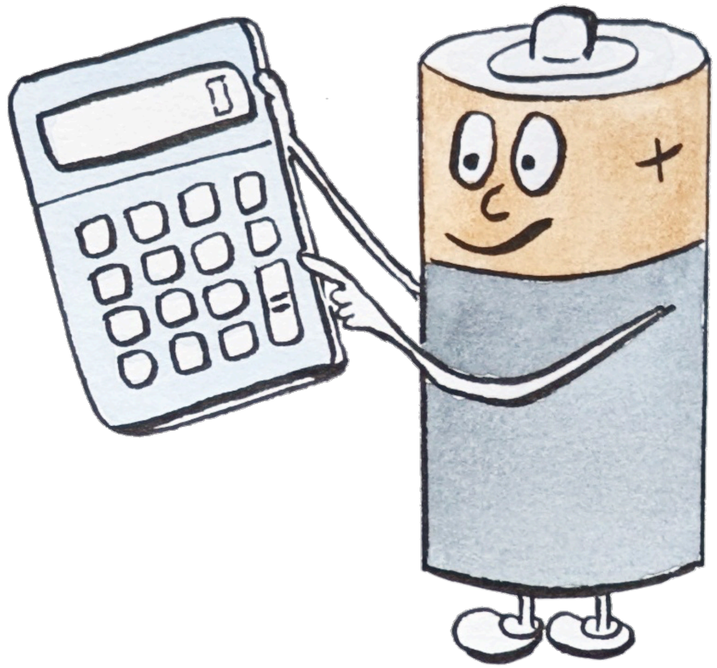


Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

Decoder only

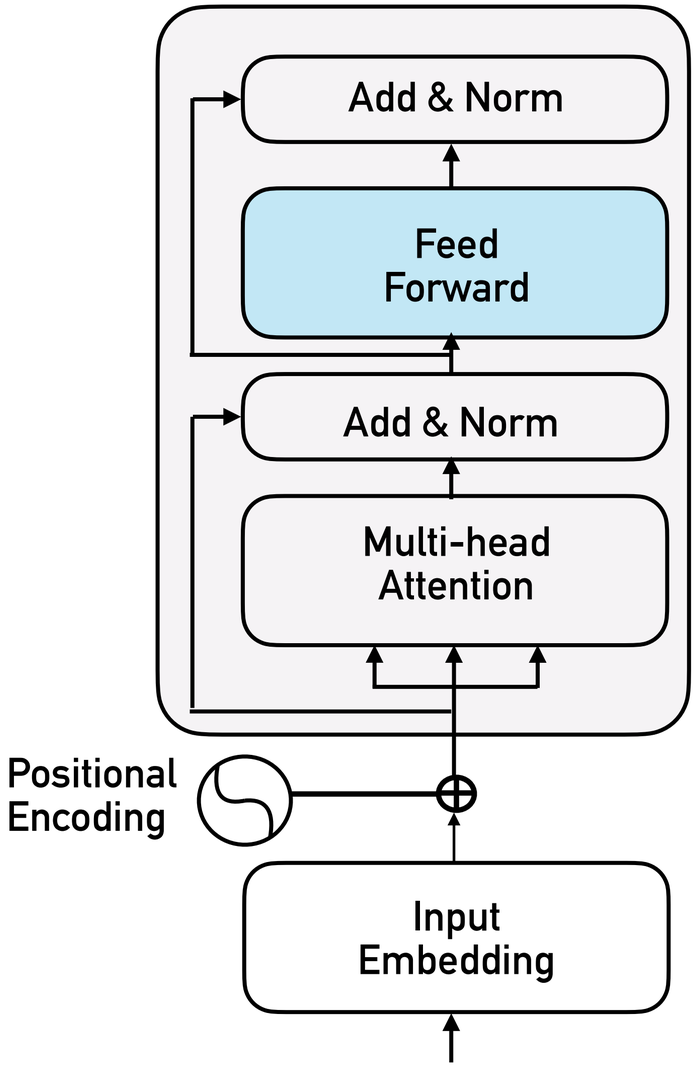


Pre-trained

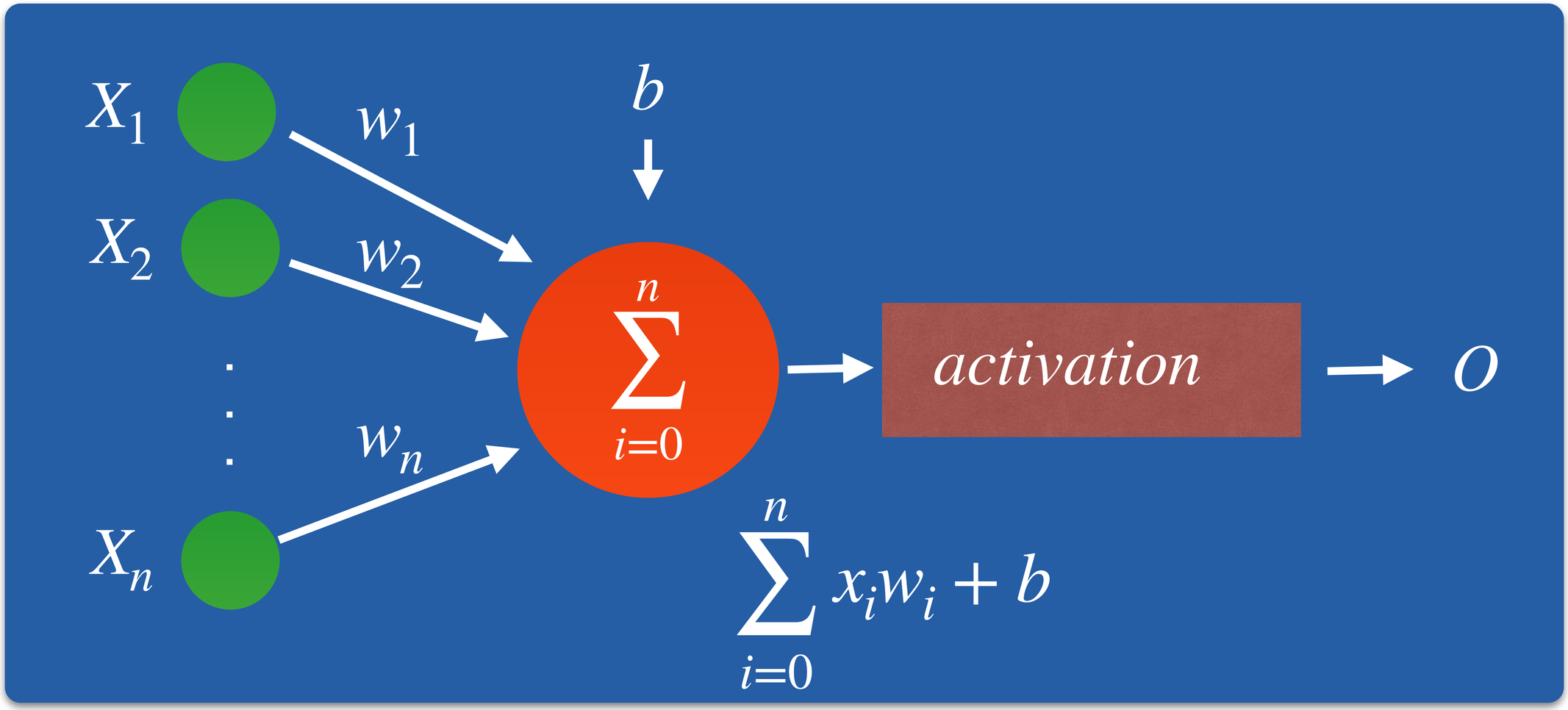


(C) Adriana Harakalova, 2024

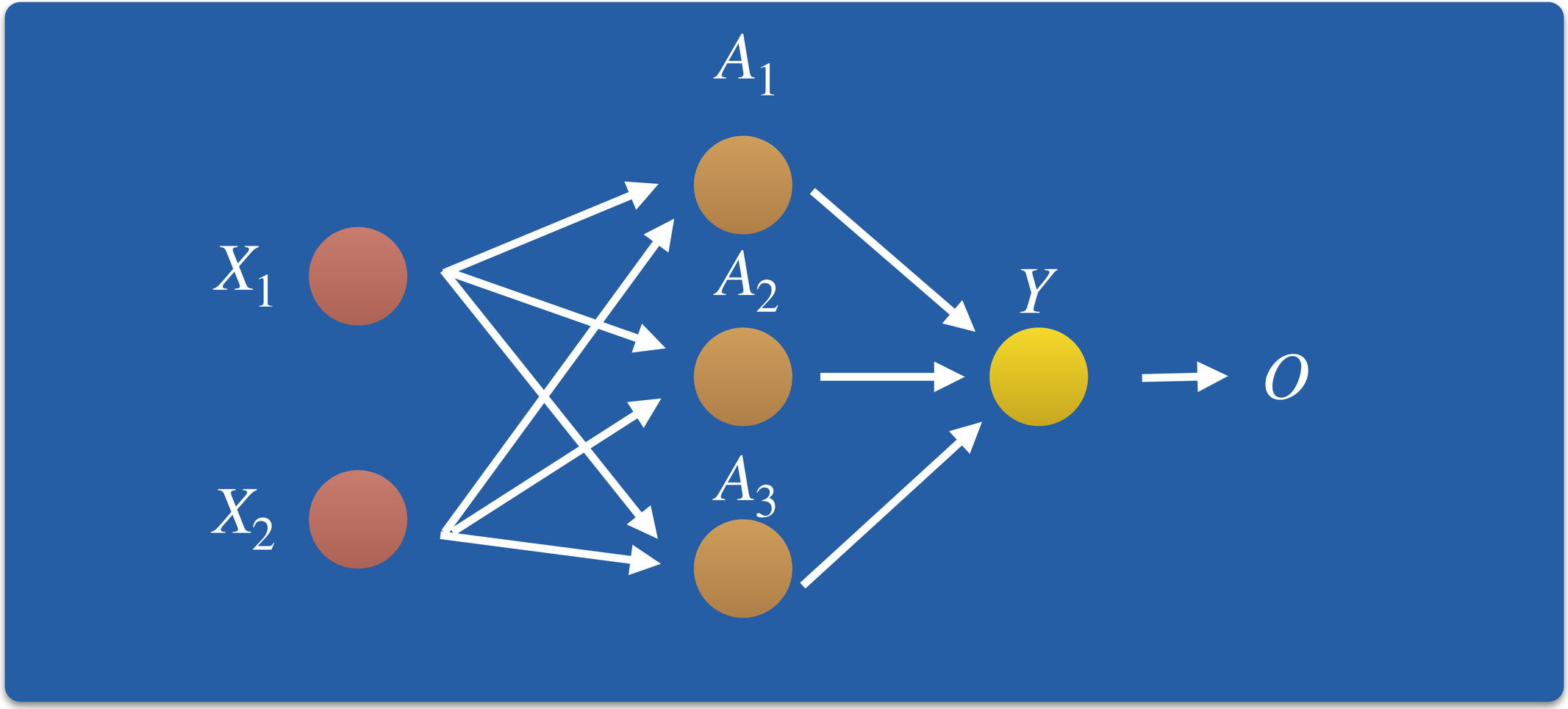
Decoder only



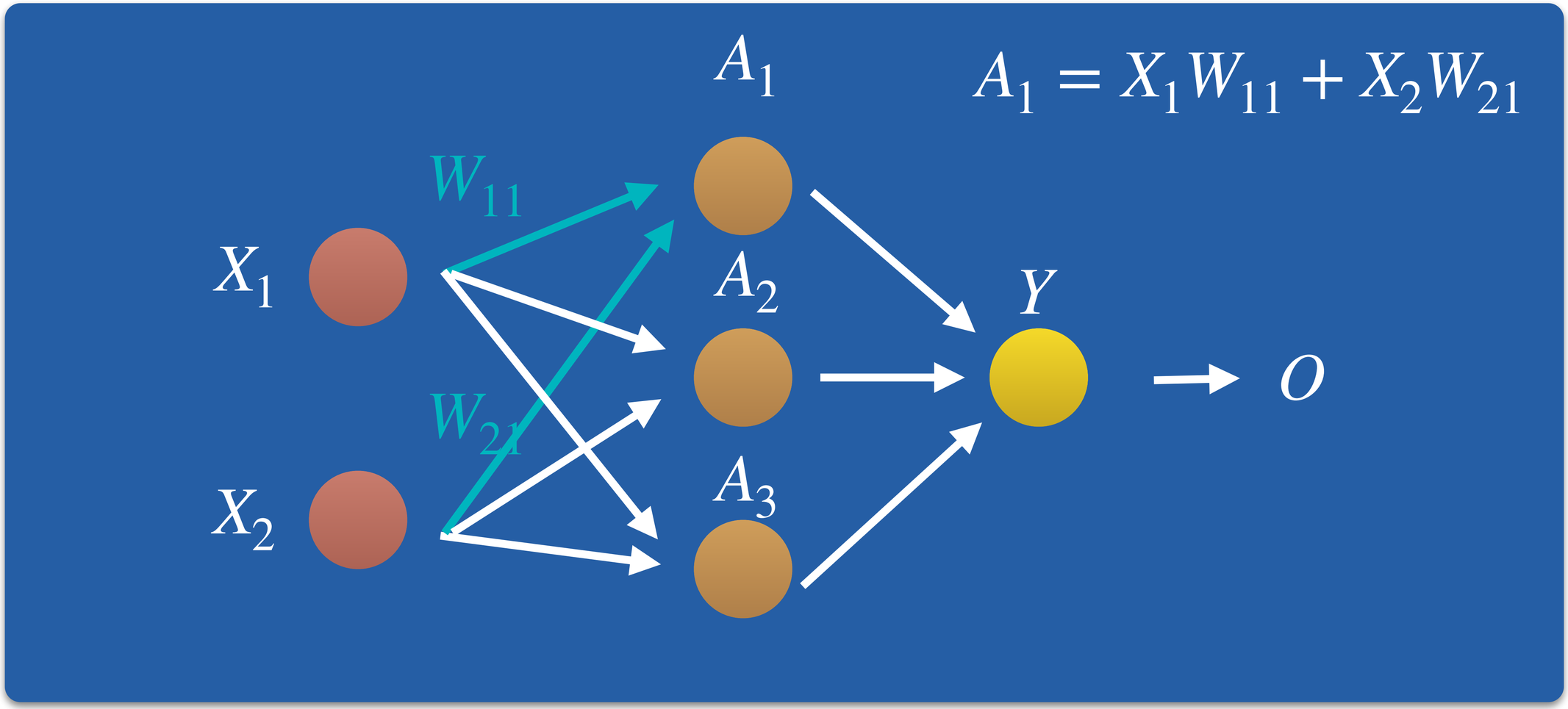
Neuron



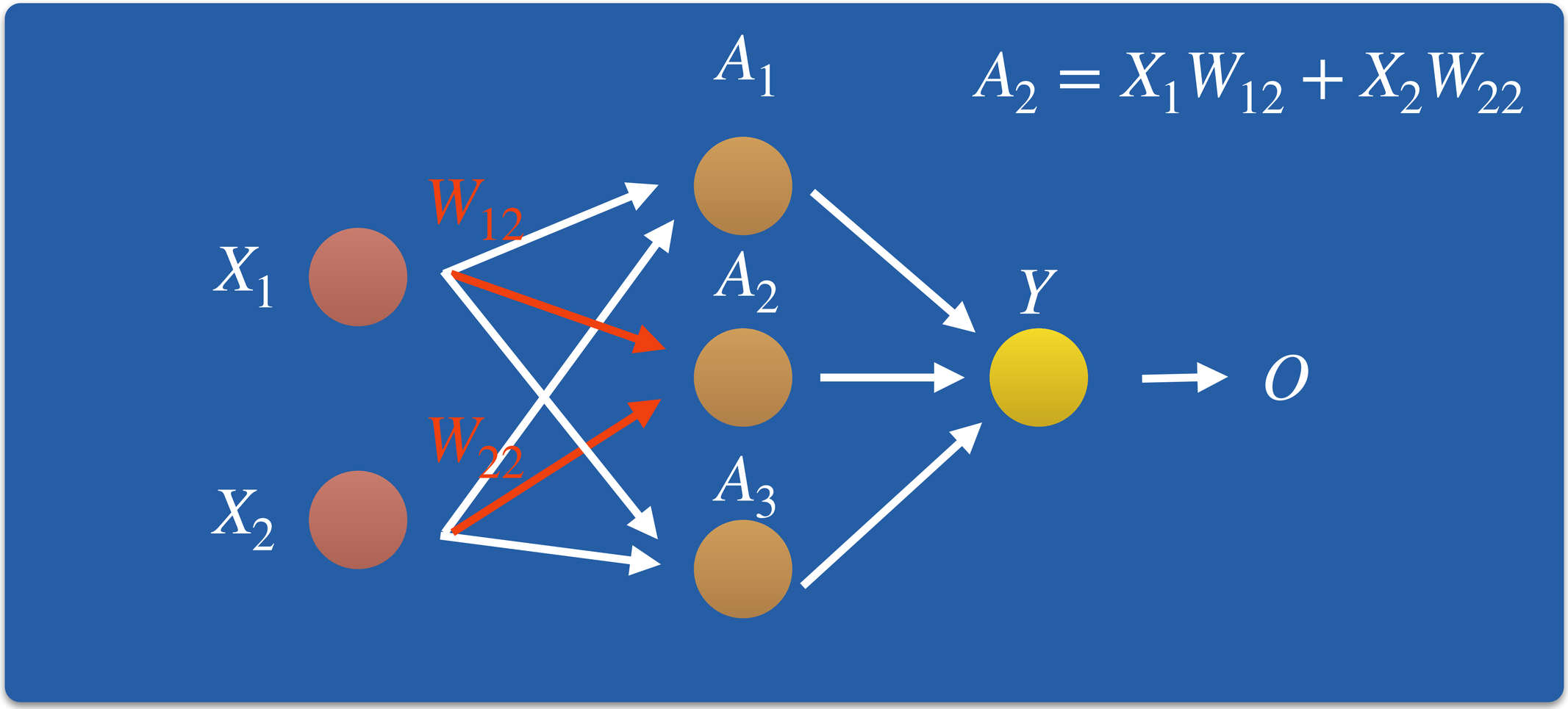
MLP - Multilayer Perceptron



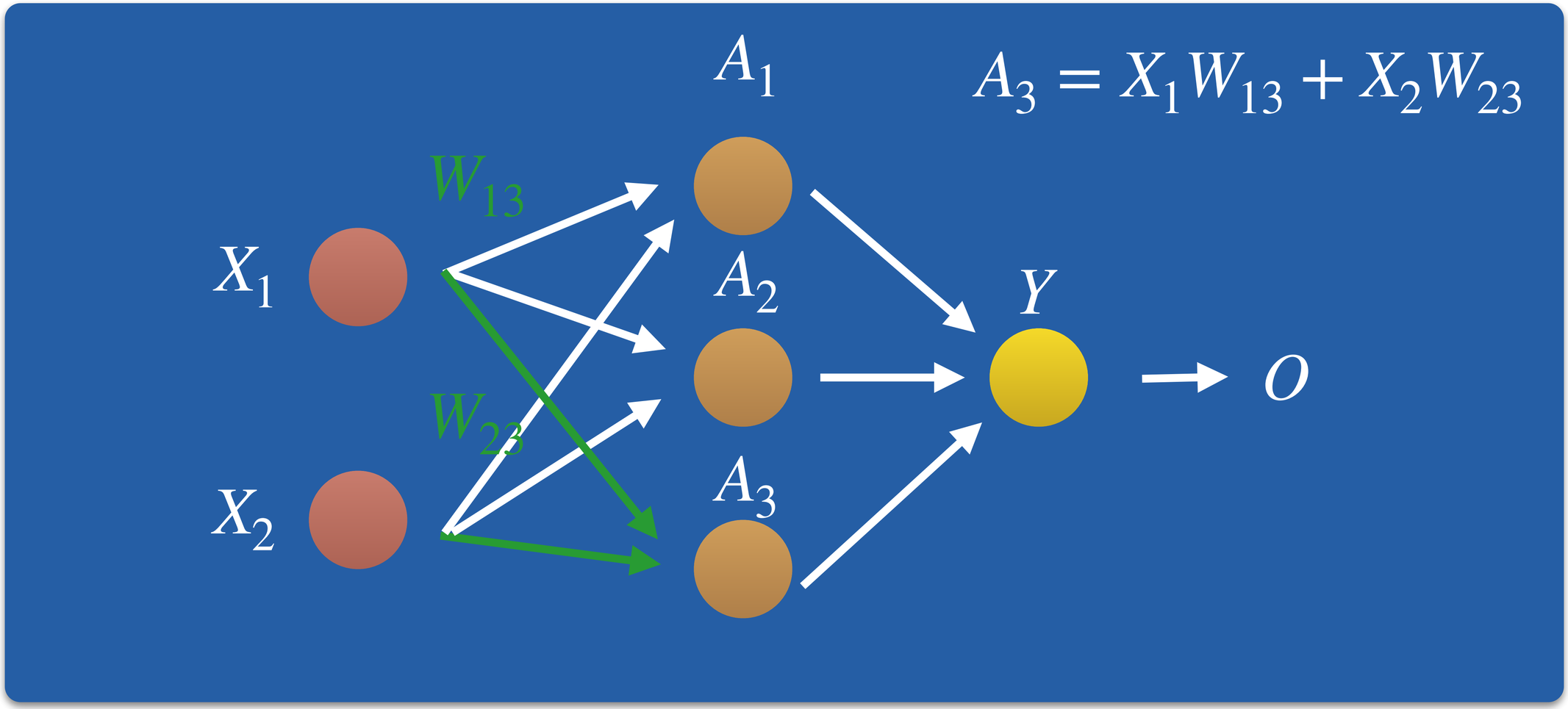
Forward propagation



Forward propagation



Forward propagation



Matrix Multiplication

$$A = X \cdot W$$

$$A_{11} = X_1 W_{11} + X_2 W_{21}$$

$$A_{12} = X_1 W_{12} + X_2 W_{22}$$

$$A_{13} = X_1 W_{13} + X_2 W_{23}$$

$$W = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \end{bmatrix}$$

$$X = \begin{bmatrix} X_1 & X_2 \end{bmatrix} \quad A = \begin{bmatrix} A_{11} & A_{12} & A_{12} \end{bmatrix}$$

$$A = X \cdot W$$

Matrices multiplication

In Kotlin with lists of doubles

$$A = X \cdot W$$

```kotlin
// Define a typealias for better readability
typealias Matrix = List<List<Double>>

class Matrix(private val data: List<List<Double>>) {
    private val numRows = data.size
    private val numCols = data.firstOrNull()?.size ?: 0

    fun matmul(other: Matrix): Matrix {
        require(numCols == other.numRows) {
            "Number of columns of A ($numCols) must equal number of rows of B (${other.numRows})."
        }

        val resultData = List(numRows) { i ->
            List(other.numCols) { j ->
                (0 until numCols).sumOf { k ->
                    data[i][k] * other.data[k][j]
                }
            }
        }
        return Matrix(resultData)
    }
}
```

ReLU

Activation function

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

```
// ReLU Activation Function
fun Matrix.relu(): Matrix {
    val reluData = data.map { row ->
        row.map { value ->
            max(0.0, value)
        }
    }
    return Matrix(reluData)
}
```

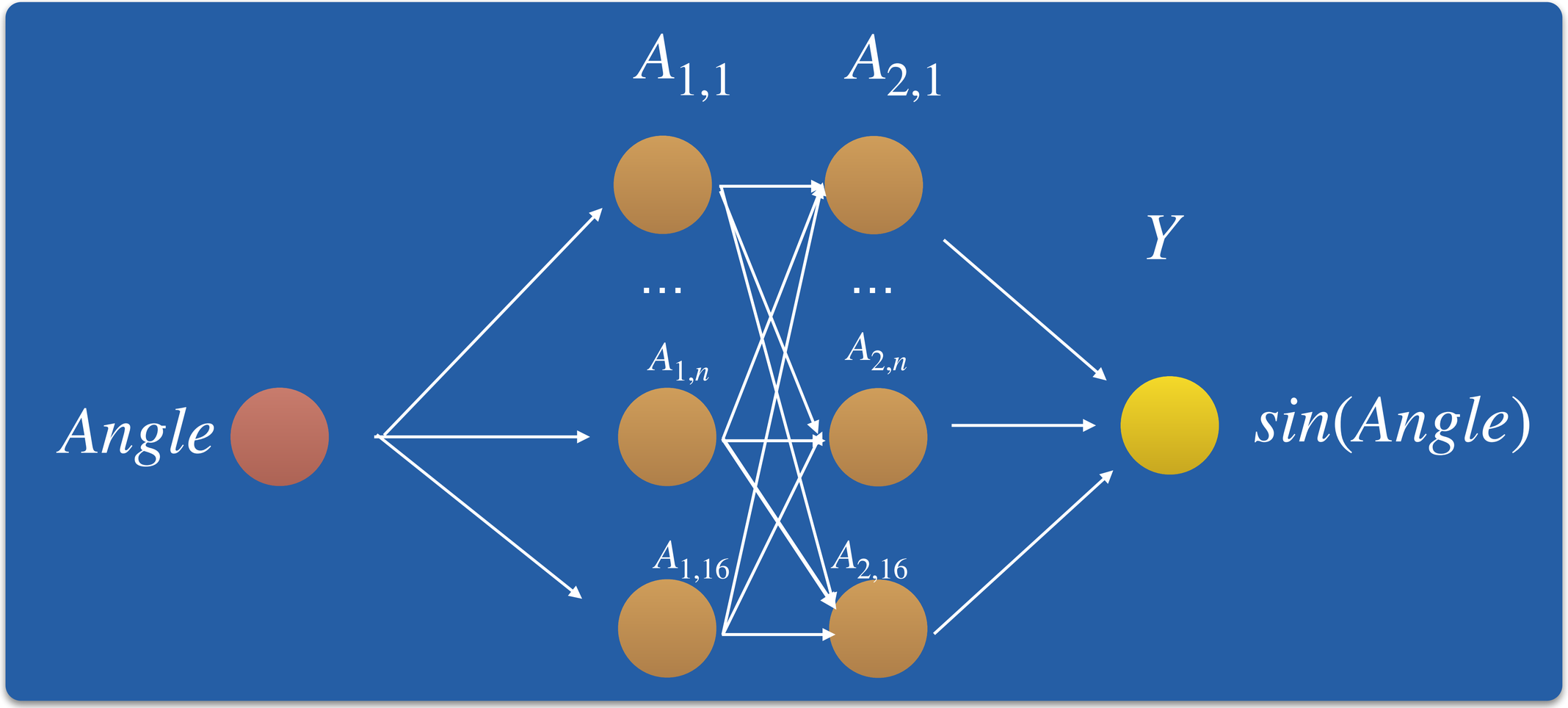
Forward Propagation in Kotlin

Hidden layer

```
class Linear() {
    val weights:Tensor
    val bias:Tensor

     fun forward(input: Tensor): Tensor {
       val output = input.matmul(weight.value.t()) + bias.value
       return output
     }
}
```

Sinus calculation with neuronal network



Forward Propagation in Kotlin

MLP

```
class SineMLP : Module() {
    private val inputLayer: Module = Linear(1, 16)
    private val hiddenLayer1: Module = Linear(16, 16)
    private val hiddenLayer2: Module = Linear(16, 16)
    private val outputLayer: Module = Linear(16, 1)
    private val relu: Module = ReLU()

    override fun forward(input: Tensor): Tensor {
        var x = input
        x = inputLayer.forward(x)
        x = relu.forward(x)
        x = hiddenLayer1.forward(x)
        x = relu.forward(x)
        x = hiddenLayer2.forward(x)
        x = relu.forward(x)
        x = outputLayer.forward(x)
        return x
    }
}
```

Forward Propagation in Kotlin

MLP with DSL

```
class SineNN(override val name: String="sin") : Module() {

    private val sineModule =  sequential{
        input(1)
        linear(16, "layer1") {
            activation = relu
        }
        linear(16, "layer2") {
            activation = relu
        }
        linear(1, "output_layer")
    }
    override fun forward(input: Tensor): Tensor =
        sineModule.forward(input)
}
```

Forward Propagation in Kotlin

1 input, 2 hidden layers with 16 neurons, 1 output

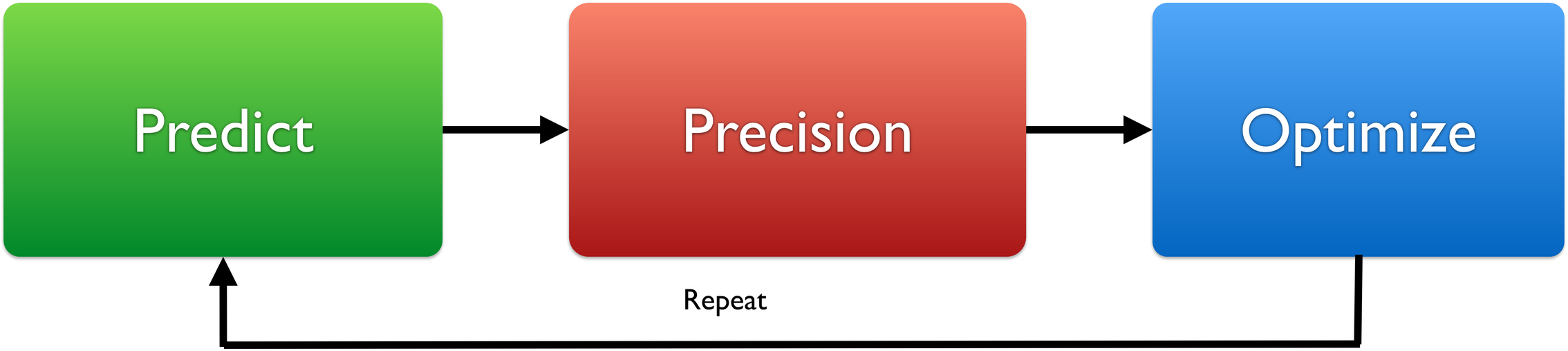
	Weight	Biases	Sum
Input	16	16	32
1.layer => 2.layer	256	16	272
2.layer => output	16	1	17
Sum	288	33	321

Trained

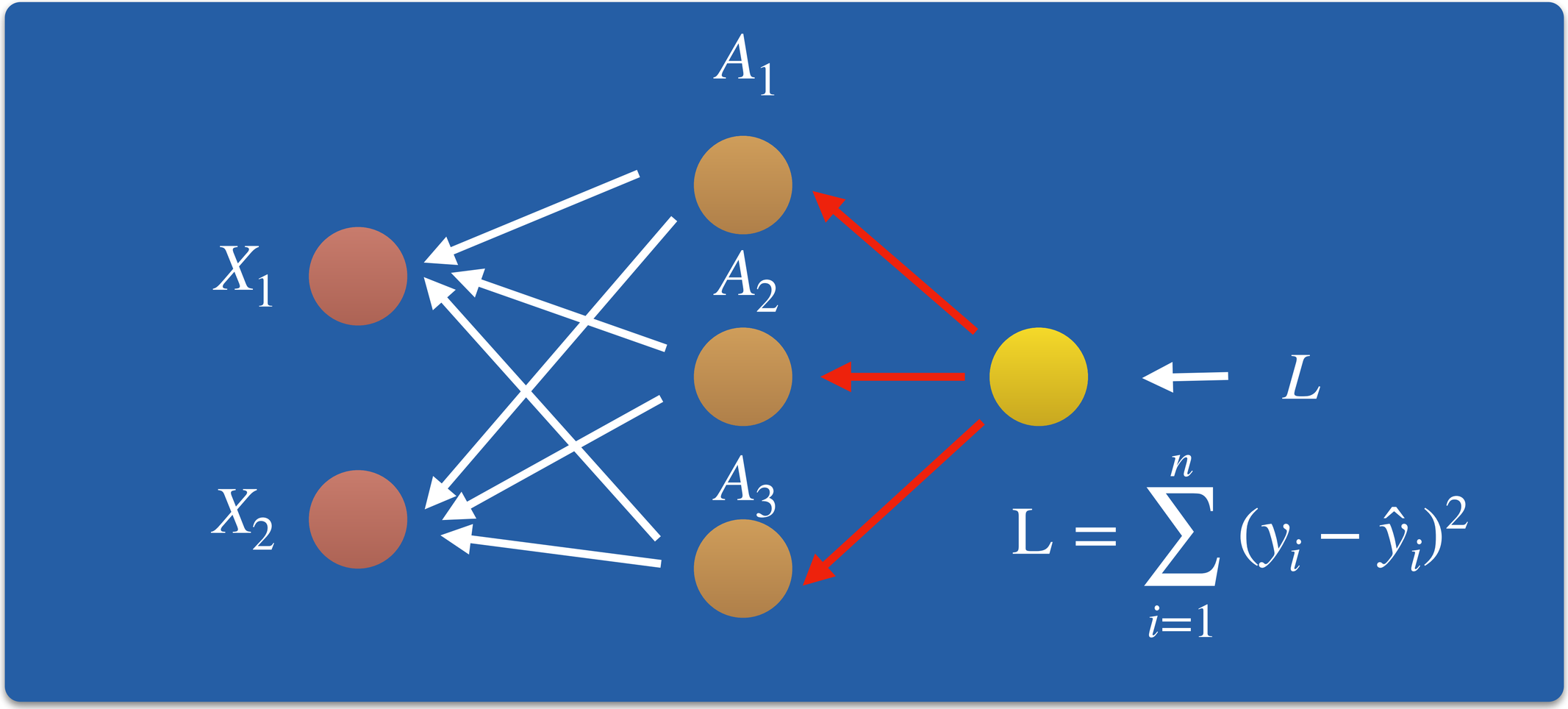
(C) Adriana Harakalova, 2024

Supervised learning

PARADIGM OF MACHINE LEARNING



Backpropagation



Loss function

Mean squared error

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

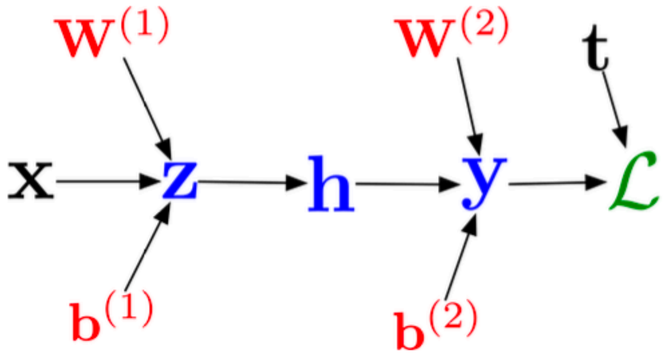
```
// Extension function for MSE calculation
fun Tensor.meanSquaredError(other: Tensor): Double {
    // Check if the tensors have the same size
    if (this.size != other.size) {
        throw IllegalArgumentException("Tensors must have the same size.")
    }

    // Calculate the squared errors and the mean
    var sum = 0.0
    for (i in 0 until this.size) {
        val error = this[i] - other[i]
        sum += error * error
    }
    return sum / this.size
}
```

Deep neural networks

Math

MLP example in vectorized form:



Forward pass:

$$\mathbf{z} = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}$$
$$\mathbf{h} = \sigma(\mathbf{z})$$
$$\mathbf{y} = \mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}$$
$$\mathcal{L} = \frac{1}{2}\|\mathbf{t} - \mathbf{y}\|^2$$

Backward pass:

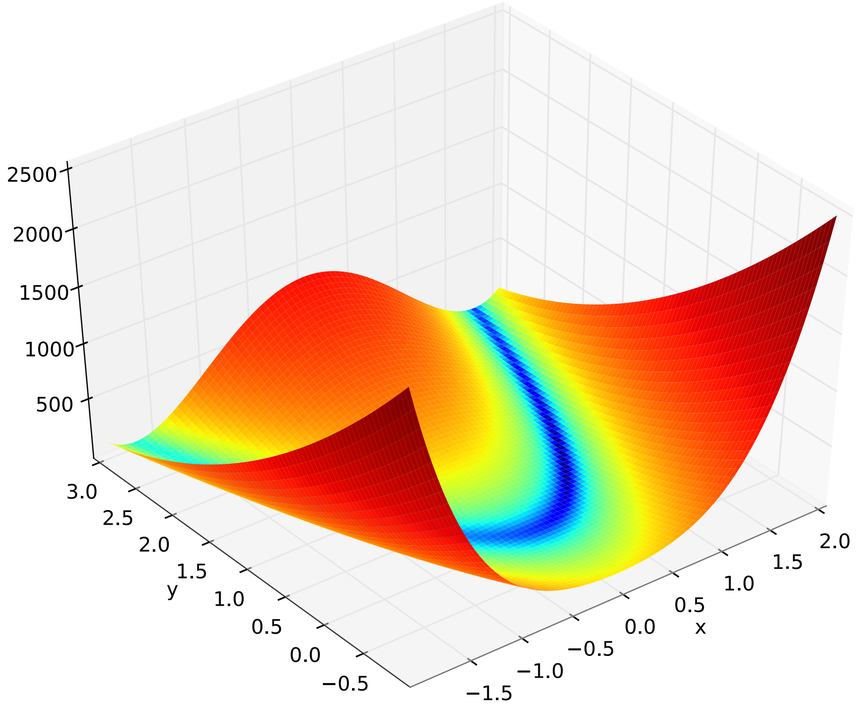
$$\overline{\mathcal{L}} = 1$$
$$\overline{\mathbf{y}} = \overline{\mathcal{L}}\,(\mathbf{y} - \mathbf{t})$$
$$\overline{\mathbf{W}^{(2)}} = \overline{\mathbf{y}}\mathbf{h}^\top$$
$$\overline{\mathbf{b}^{(2)}} = \overline{\mathbf{y}}$$
$$\overline{\mathbf{h}} = \mathbf{W}^{(2)\top}\overline{\mathbf{y}}$$
$$\overline{\mathbf{z}} = \overline{\mathbf{h}} \circ \sigma'(\mathbf{z})$$
$$\overline{\mathbf{W}^{(1)}} = \overline{\mathbf{z}}\mathbf{x}^\top$$
$$\overline{\mathbf{b}^{(1)}} = \overline{\mathbf{z}}$$

Stochastic gradient descent

Optimization method

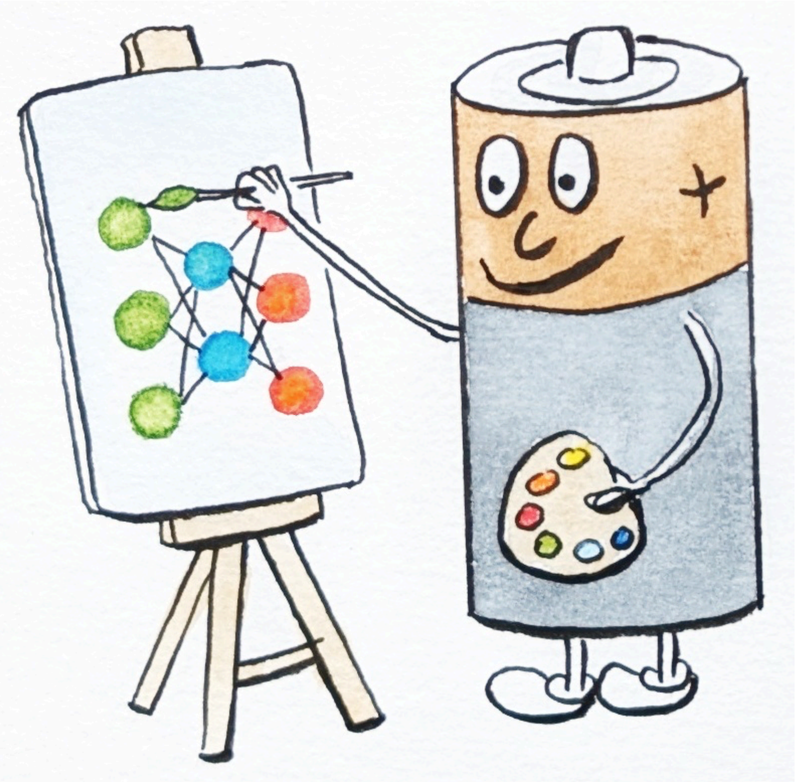
Gradient descent can write code better than you. I'm sorry.

Andrej Karpathy, 4.8.2017



https://commons.wikimedia.org/wiki/File:Rosenbrock_function.svg

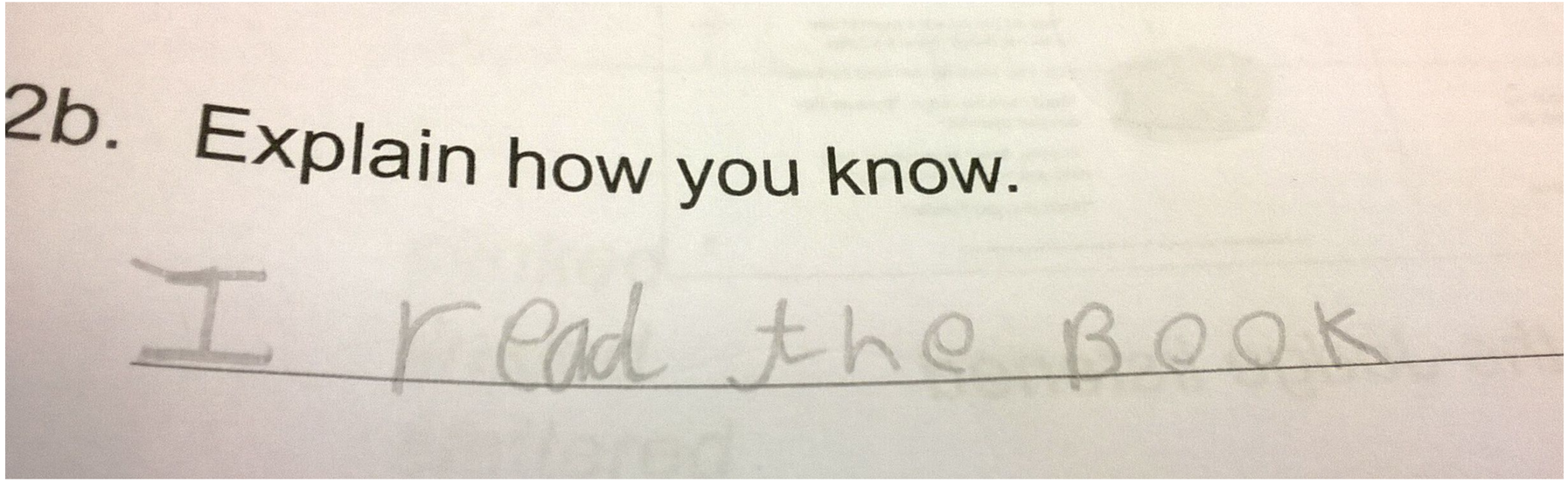
Generative



(C) Adriana Harakalova, 2024

Generative

Pattern recognition and prediction



Input Text Processing

- **Tokenization -** a process in which a text is broken down into smaller units such as words, sentences, or characters, which are referred to as "tokens."
- **Encoding -** Assigning a unique numerical identifier to each token**.**
- small GPT-2 has approximately 50,257 tokens **(Byte Pair Encoding).**

Tokenization and Encoding

Character-based encoding example

H A L L O

Text

Tokenization and Encoding

Character-based encoding example

H A L L O Text

‘H‘ ‘A‘ L‘ L‘ ‘O‘ Tokens

Tokenization and Encoding

Character-based encoding example

H A L L O — Text

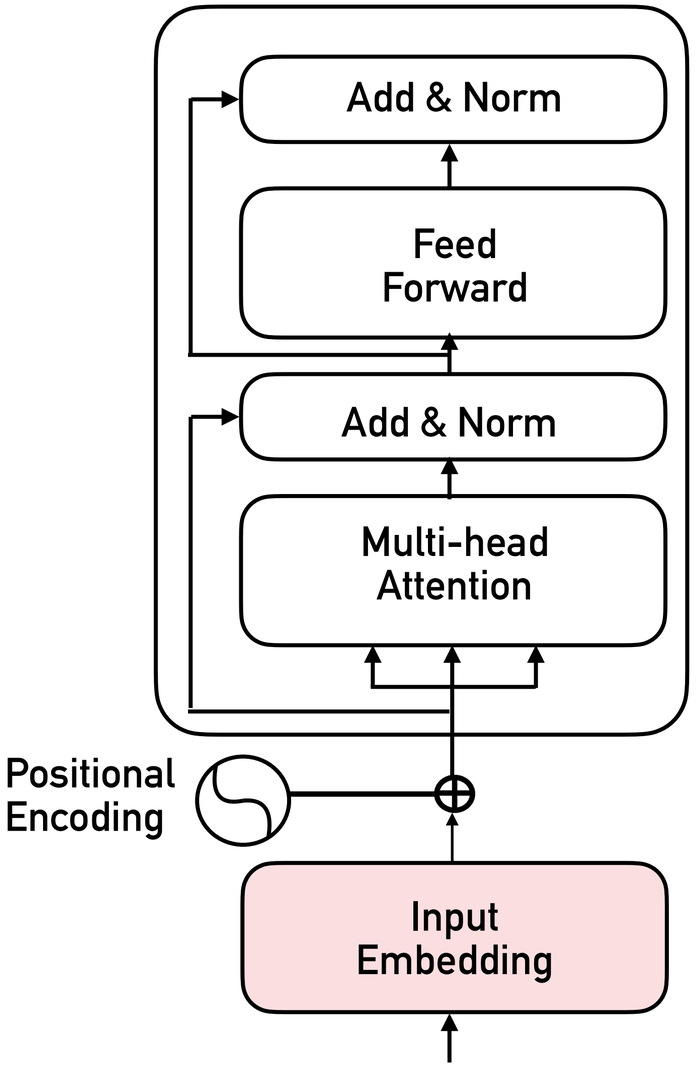
‘H‘ ‘A‘ L‘ L‘ ‘O‘ — Tokens

$$\vec{v} = (8 \quad 1 \quad 12 \quad 12 \quad 15)$$ — Numerical Sequence

Level 1

Input Embedding

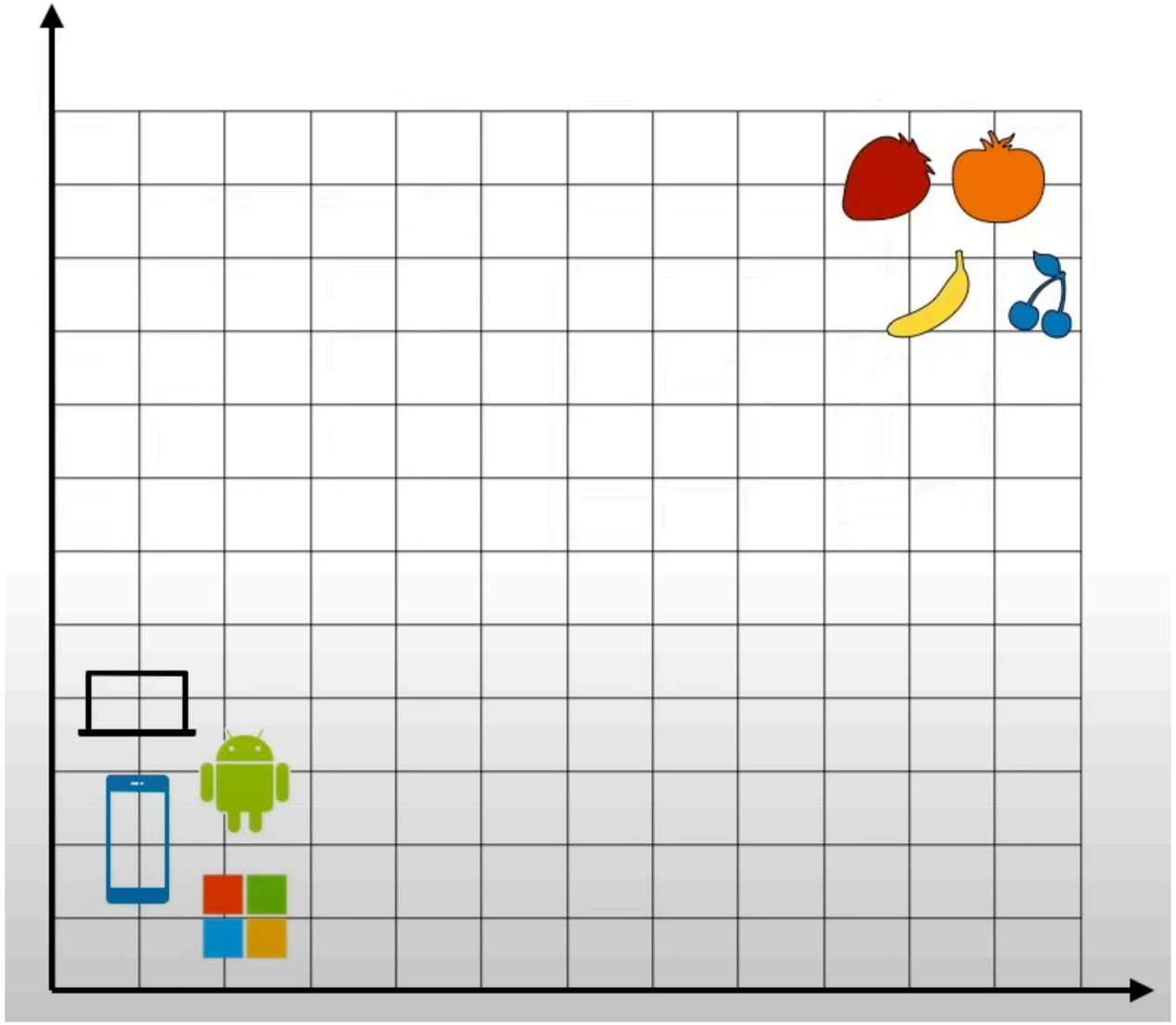
- Every token processed in the **same time** - efficient
- every token represented as a vector in **multidimensional** space
- Values represent **semantic meaning**



Embedding

Text as numbers in multidimensional space

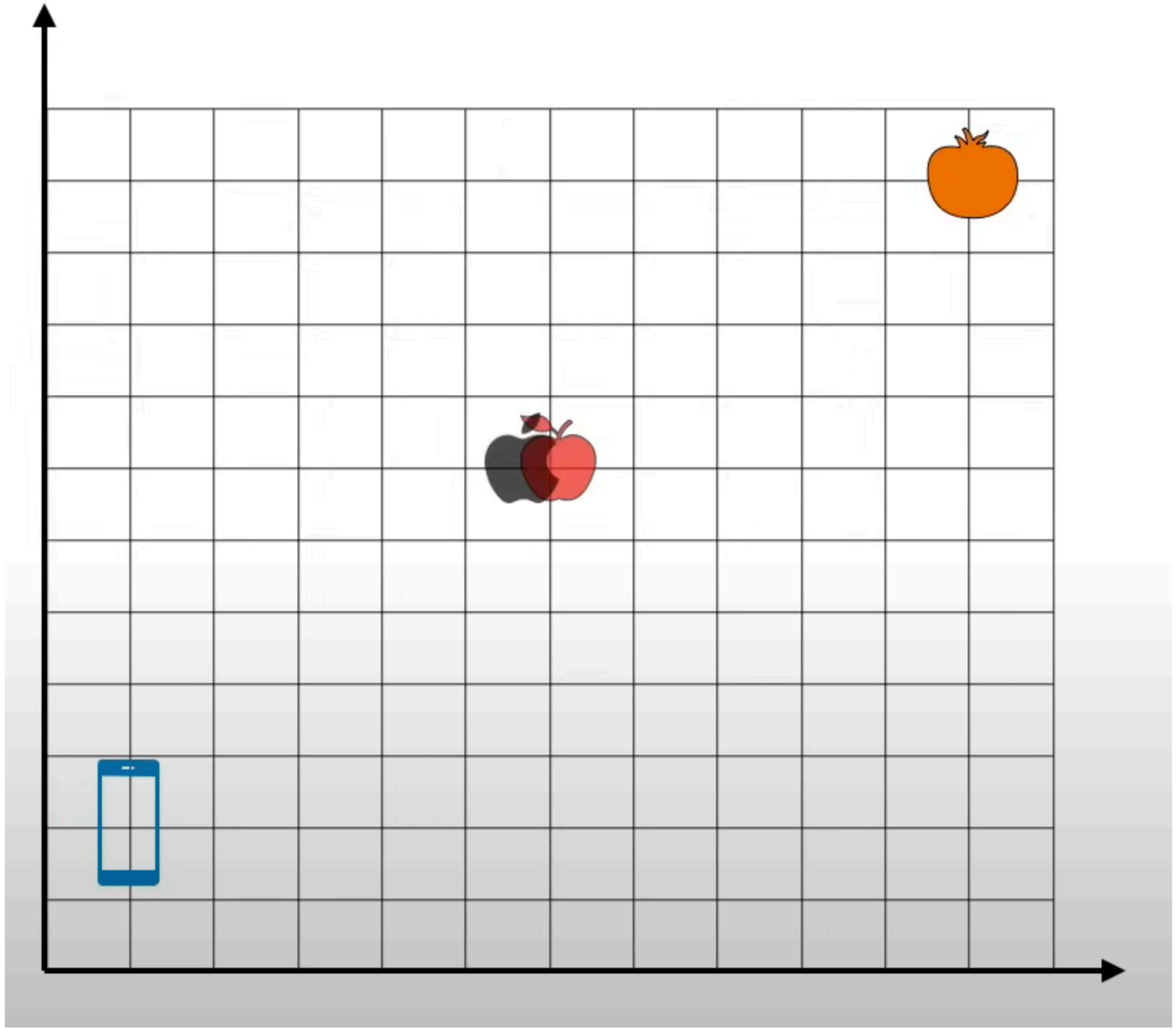
GTP2-768 dimensions



https://www.youtube.com/watch?v=OxCpWwDCDFQ&list=PLs8w1Cdi-zvYskDS2icIItfZgxclApVLv

Positional encoding

- Please buy an apple and an orange
- Apple unveiled the new phone
- **CONTEXT**

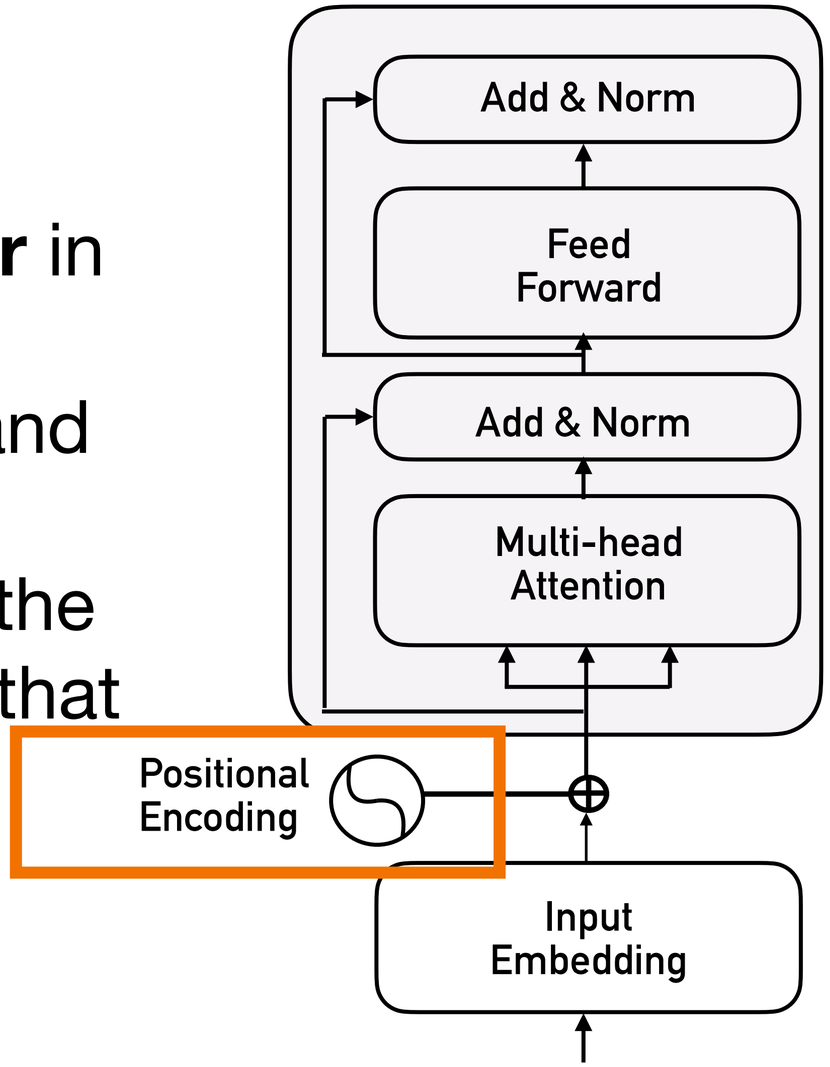


https://www.youtube.com/watch?v=OxCpWwDCDFQ&list=PLs8w1Cdi-zvYskDS2icIItfZgxclApVLv

Level 2

Positional Encoding

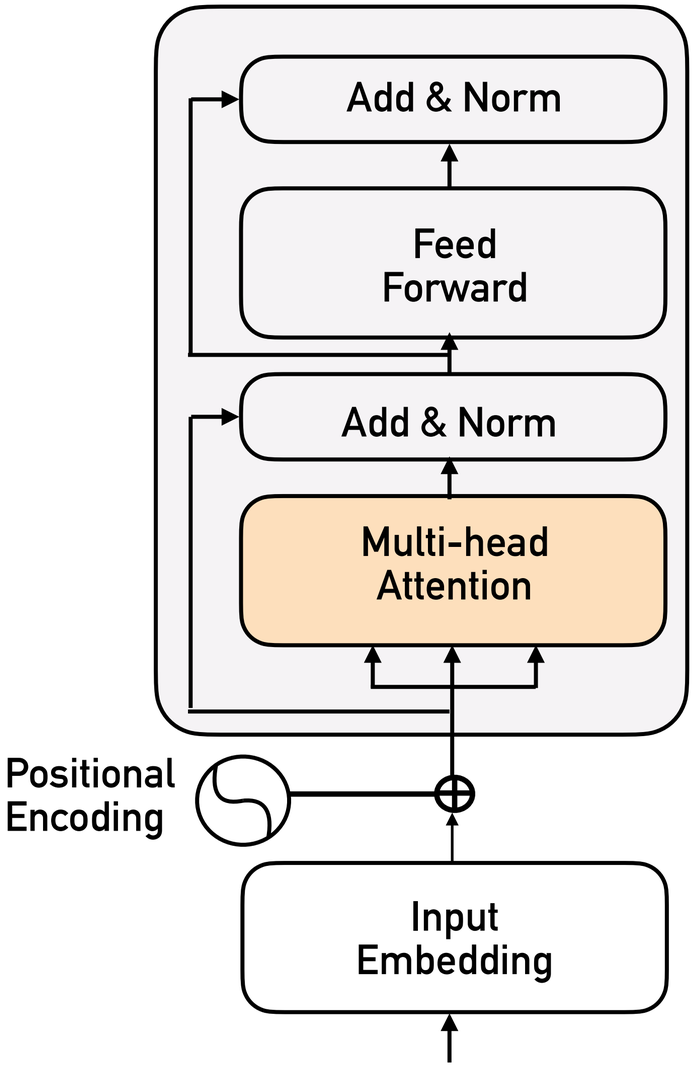
- provides information about the **order** in which words appear.
- Uses e.g sinusoidal functions (sine and cosine) to generate the positional encodings based on the position of the word in the sequence. This ensures that **words at different positions have different encodings**



Level 3

Self-attention mechanism

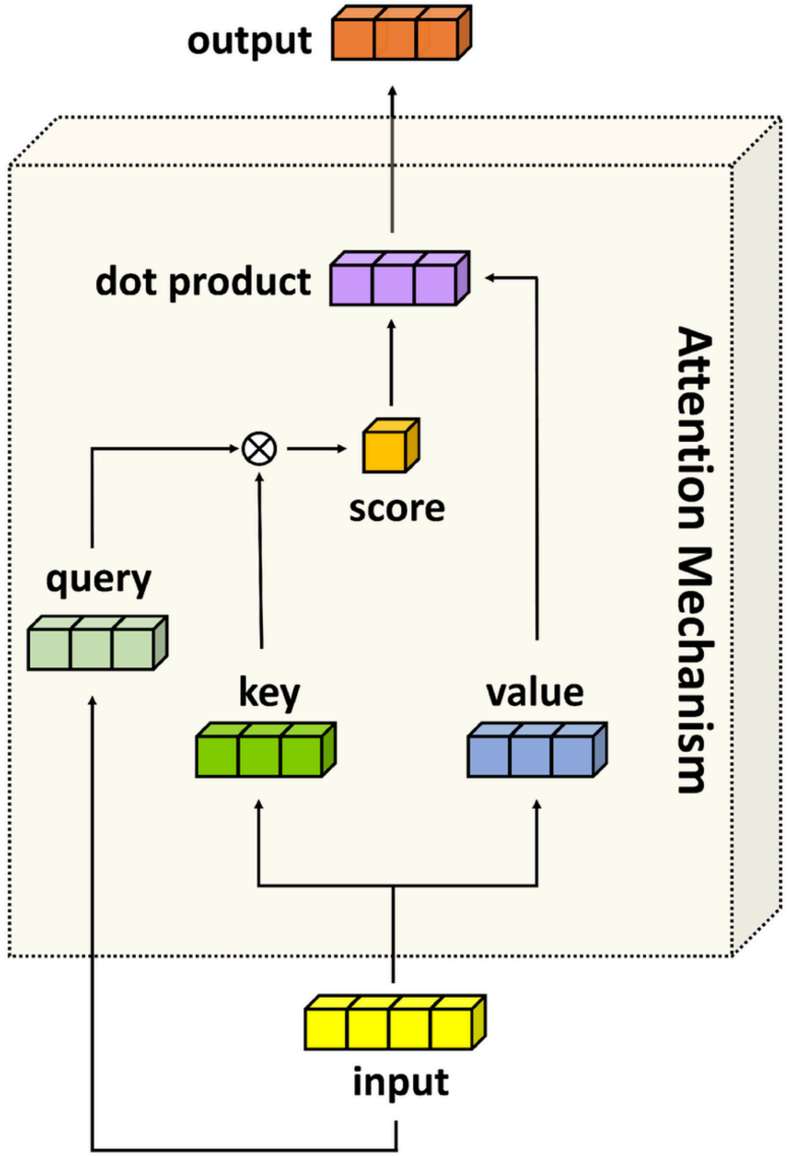
- Context rich
- Attention captures contextual meaning



Level 3

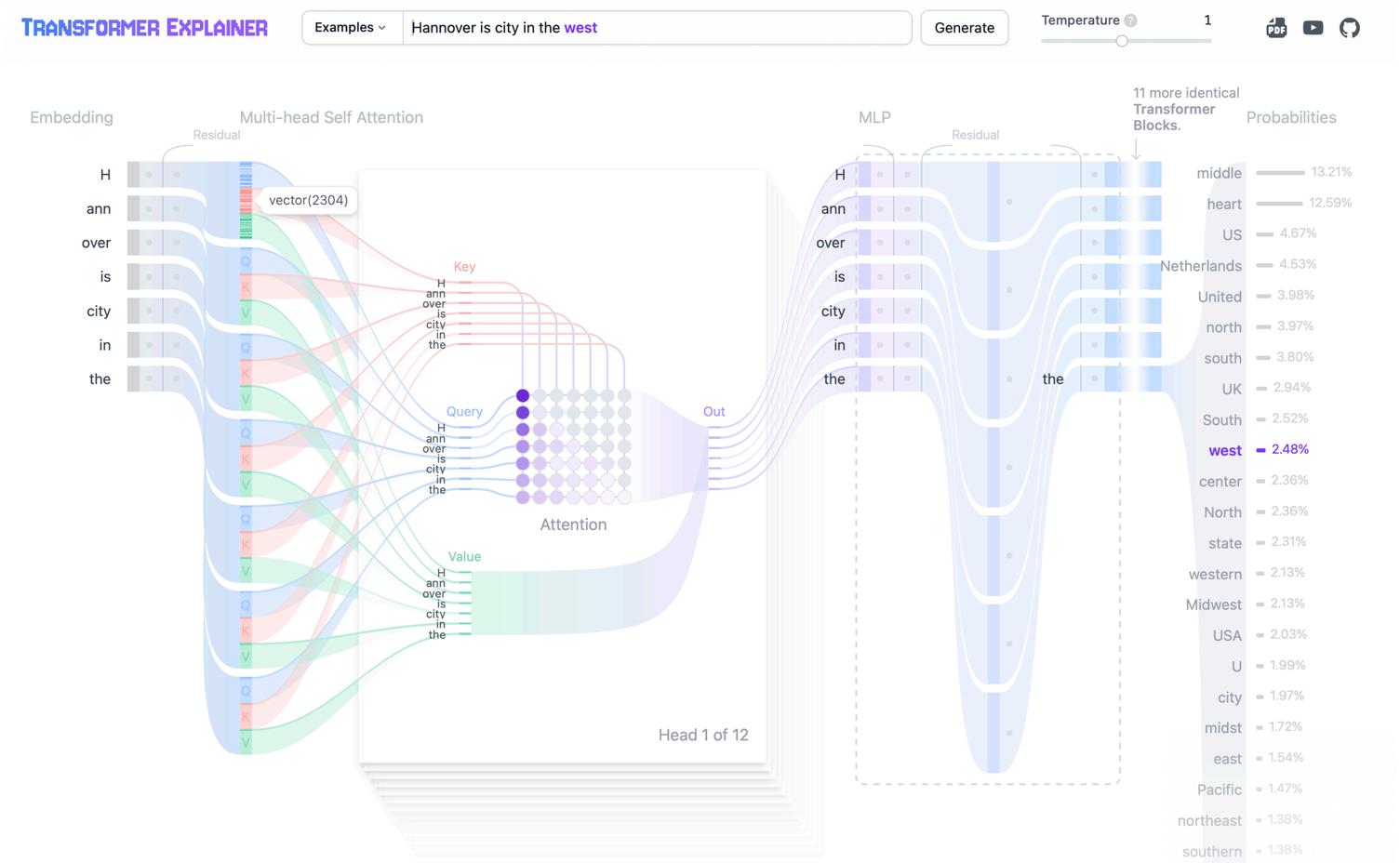
Self-attention mechanism

- **query (Q)** vector determines **what** the model is focusing on
- **key (K)** vector represents **potential matches** for the query
- **value (V) hoilds** information to be retrieved based on the attention score between the query and key



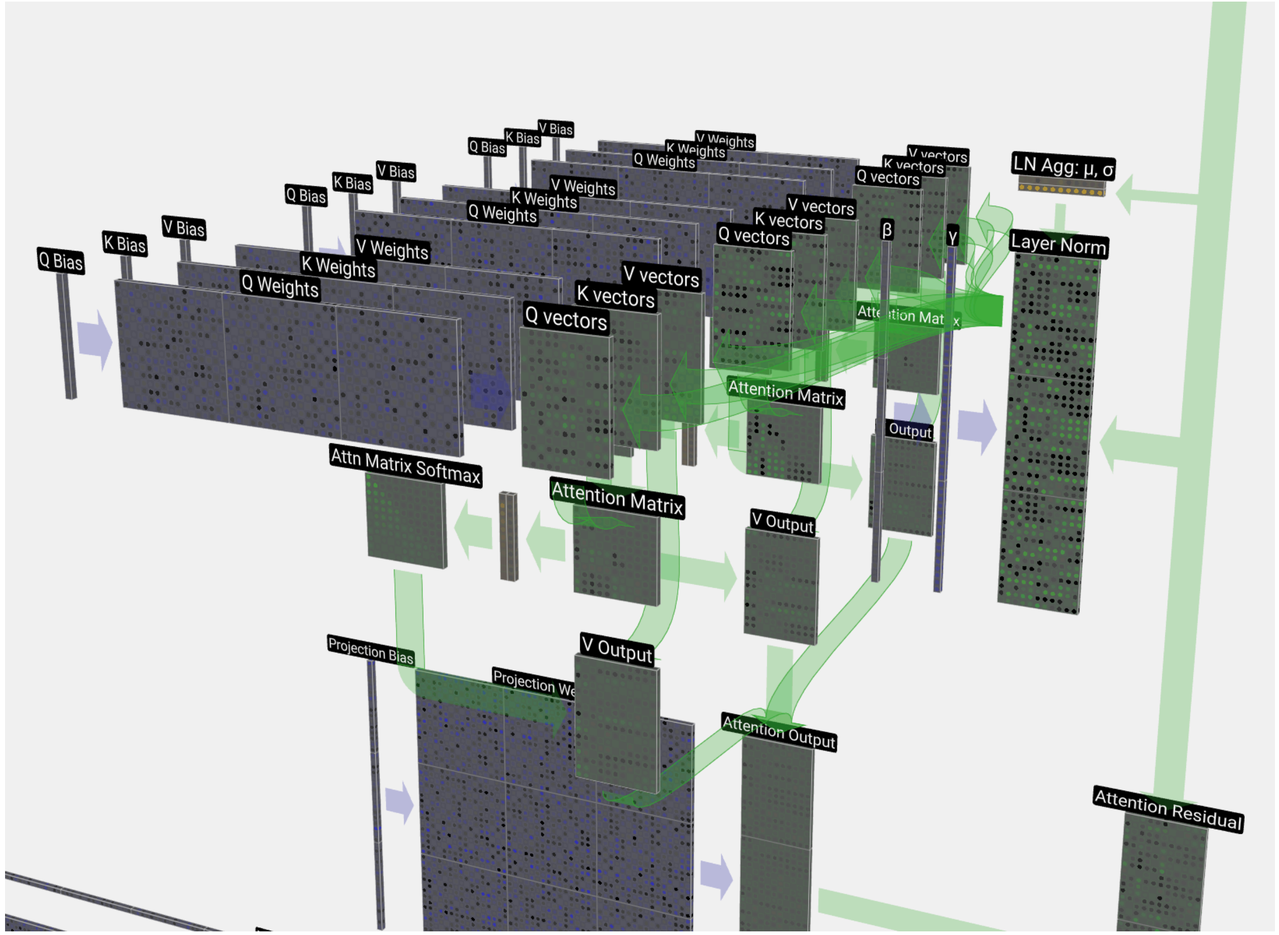
2D GTP-2 Visualization

https://poloclub.github.io/transformer-explainer/

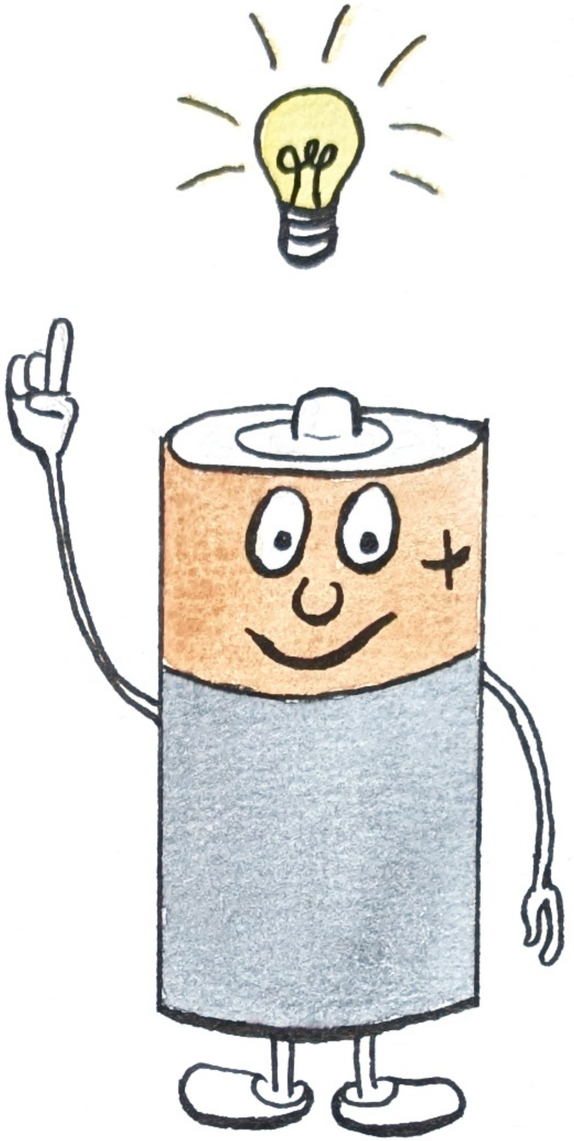


3D nanoGTP Visualization

https://bbycroft.net/llm



sKaiNET



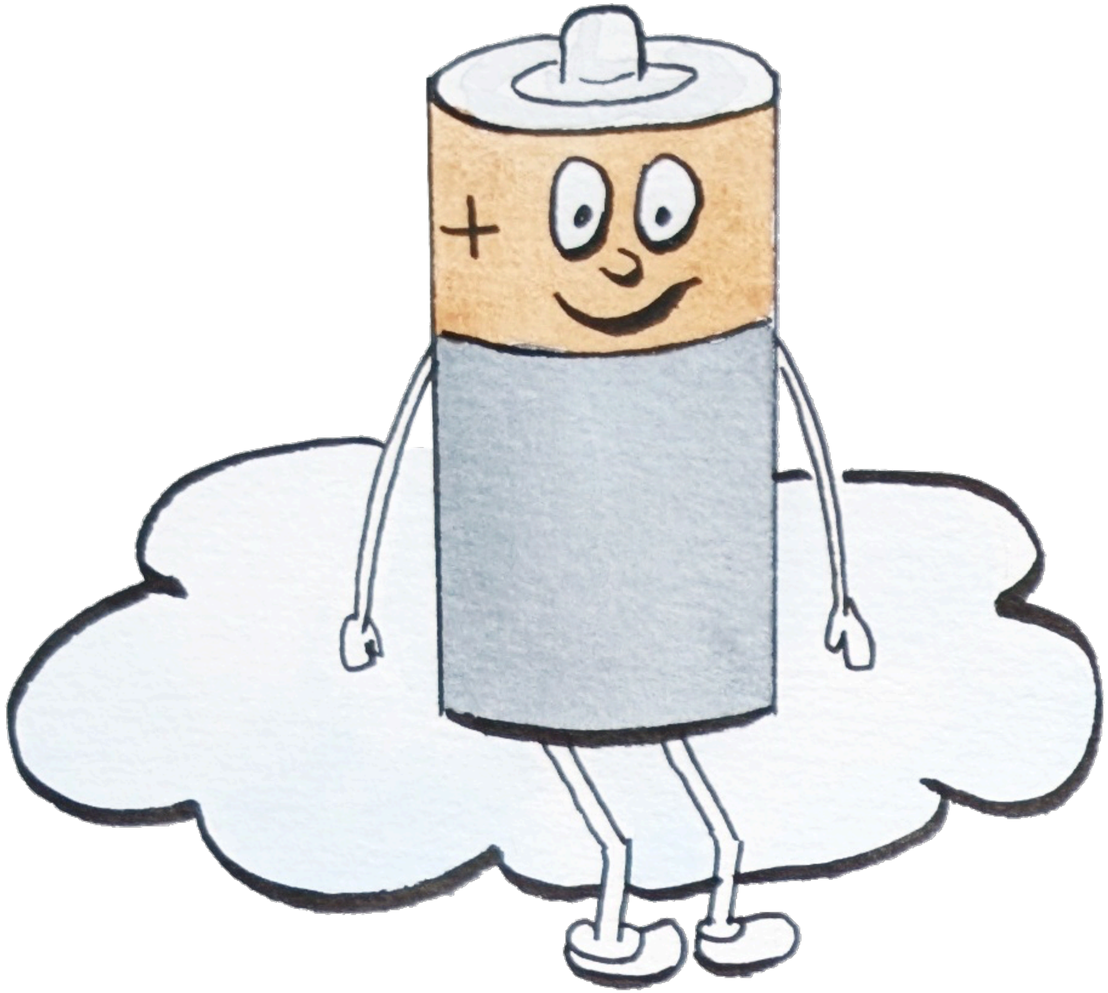
(C) Adriana Harakalova, 2024

ML in Kotlin from scratch

Kotlin libraries

- Standing on the shoulders of giants
  - Kotlin port of nanoGPT -> 80%
  - Now as Kotlin Multiplatform
  - https://github.com/michalharakal/KPTChat/tree/feature/kmp
- Kotlin port of micrograd -> 90%
  - With its own DSL language for graphviz DOT language
  - https://github.com/michalharakal/miKrograd
- Testing framework for validation with Pytorch calculations

Demo



(C) Adriana Harakalova, 2024

Thank you


+
Batterie

Source code

https://github.com/michalharakal/from-zero-to-kotlin-gpt/tree/jax-london

Links

Sources and links

- Attention Is All You Need - https://arxiv.org/pdf/1706.03762
- Let's build GPT: from scratch, in code, spelled out (https://www.youtube.com/watch?v=kCc8FmEb1nY)
- Pytorch logo https://commons.wikimedia.org/wiki/File:Pytorch_logo.png
- https://www.neuromedia.ca/how-many-neurons-are-in-the-human-brain/
- The math behind Attention: Keys, Queries, and Values matrices https://www.youtube.com/watch?v=UPtG_38Oq8o&list=PLs8w1Cdi-zvYskDS2icIItfZgxclApVLv&index=2
- https://commons.wikimedia.org/wiki/File:Rosenbrock_function.svg https://github.com/michalharakal/KPTChat/tree/feature/kmp
- https://github.com/michalharakal/miKrograd
- https://lmos-ai.github.io/arc/
- https://paperswithcode.com/paper/kan-kolmogorov-arnold-networks
- https://github.com/kindxiaoming/pykan
- https://commons.wikimedia.org/wiki/File:The-Transformer-model-architecture.png
- https://github.com/michalharakal/gradienttracer
- https://deeplearning4j.konduit.ai/