Welcome Here!!

NATURAL LANGUAGE PROCESSING (NLP)

TRANSFORMERS, ATTENTION AND PRETRAINED NLP MODELS (HUGGING FACE, GPT)

DSN LEKKI-AJAH

BY ABEREJO HABEEBLAH O.

X (TWITTER): @HABEREJO

Day 4

Tuesday, 25th March, 2025

Why are we here

To equip learners with the skills to understand how Artificial Intelligence models are built, explore Machine Learning and Deep Learning, and focus on Natural Language Processing (NLP).

Project Class

Classes Structure

Every Tuesday throughout the month of March.

4 classes in total. This is the fourth class

Each class for about 1:45 minutes

Yupppppp, Last Class

Who should be here

- You're curious and ready to learn something new
- You dream of becoming a Data Scientist.
- You're passionate about building the future as a Machine Learning Engineer.
- You have a basic understanding of Python programming.
- You've worked with data and want to take your skills to the next level.
- You're a researcher exploring the exciting world of AI.

which of these are you?

Who should be here ✓

- You're familiar with the fundamentals of Machine Learning and Deep Learning.
- You love problem solving and are excited to tackle real world problems with AI.
- You want to understand the technology that is shaping the future.
- You're driven by a desire to learn and grow in the field of AI.
- You enjoy collaborating and learning from others.

Which of these are you?????
You ticked any of these boxes?
Let's Goooooooo

Before we begin

Who I am?

- A Machine Learning Engineer
- A young MAN eager to master AI

FUN FACT:
I've taught over 100 people Python, but
I'm still learning new things every day.
(Especially how to avoid typos when
coding late at night!)

Scan to visit my portfolio
or
https://bheez.netlify.app

Before we begin

Quick Recap (First Class)

1️⃣ **Understanding AI, ML, DL, and NLP**

2️⃣ **Introduction to NLP**

3️⃣ **NLP Use Cases:** Translation Apps 🌍, Spam Filters 📧, Search Engines 🔍, Sentiment Analysis 😀 😡, Voice Assistants 🎙️, Chatbots 💬, Autocorrect & Predictive Text 📱.

4️⃣ **How Computer Understands Text:** 1. Text Preprocessing 2. Tokenization, 3. Numerical Representation (Word Embedding)

5️⃣ **Text Preprocessing Techniques:** Lowercasing, Removing Punctuation Marks, Removing Stopwords (i.e. like A, of, in, etc), Stemming / Lemmatization.

Before we begin

Quick Recap (Second Class)

1 **How Computer Understands Text:** 1. Text Preprocessing

2 **Text Preprocessing Techniques:** Lowercasing, Removing Punctuation Marks, Removing Stopwords (i.e. like A, of, in, etc), Stemming / Lemmatization.

3 **How Computer Understands Text:** 2. Tokenization

4 Tokenization: **Sentence Tokenization** (Sentence-Level Tokenization)

Word Tokenization (Word-Level Tokenization)

Subword Tokenization

Character-level Tokenization

5 **How Computer Understands Text:** 3. Numerical Representation

6 **Numerical Representation:** Vocabulary and Integer Encoding

7 **Numerical Representation:** One Hot Encoding

Before we begin

Quick Recap (Third Class)

1. Numerical Representation: Word Embedding
2. **Word2Vec**
3. **Glove**
4. **t-distributed stochastic neighbor embedding, t-SNE (pronounced tee-snee)**
5. **FastText**
6. **BERT** (bi-directional encoder representations from transformers)
7. Recurrent Neural Networks (RNNs), Transformers

Class Structure

1. Address RNNs Limitations
2. **Transformers**: Attention is all you Need
3. **Attentions**
4. **Attention mechanism:** Queries, Keys and Value
5. Encode and Decoder
6. Evaluating models with metrics
7. Questions and Answer

Bottleneck of RNNs

RNNs have trouble dealing with very long sequences, as they tend to progressively forget about the past.

By the time you’ve reached the 100th token in either sequence, little information remains about the start of the sequence.

That means RNN-based models can’t hold onto long-term context, which can be essential for translating long documents.

Addressing Limitations:

1. **Attention Mechanisms** which allow the model to focus on relevant parts of the input sequence.
2. **Transformers** which replaces **recurrence** with **attention mechanisms**, enabling parallel processing.

   Don't get confused, and Transformers are a model, Attention is a component that can be used in any Neural Network like CNN.
3. **Using LSTMs and GRUs** which uses gating mechanisms to control information flow.

This limitation is what has led the ML community to embrace the Transformer architecture for sequence-to-sequence problems

The Transformer

1. Starting in 2017, a new model architecture started overtaking recurrent neural networks across most natural language processing tasks: **the Transformer.**

2. Transformers were introduced in the seminal paper **"Attention is all you need"** by Vaswani et al.

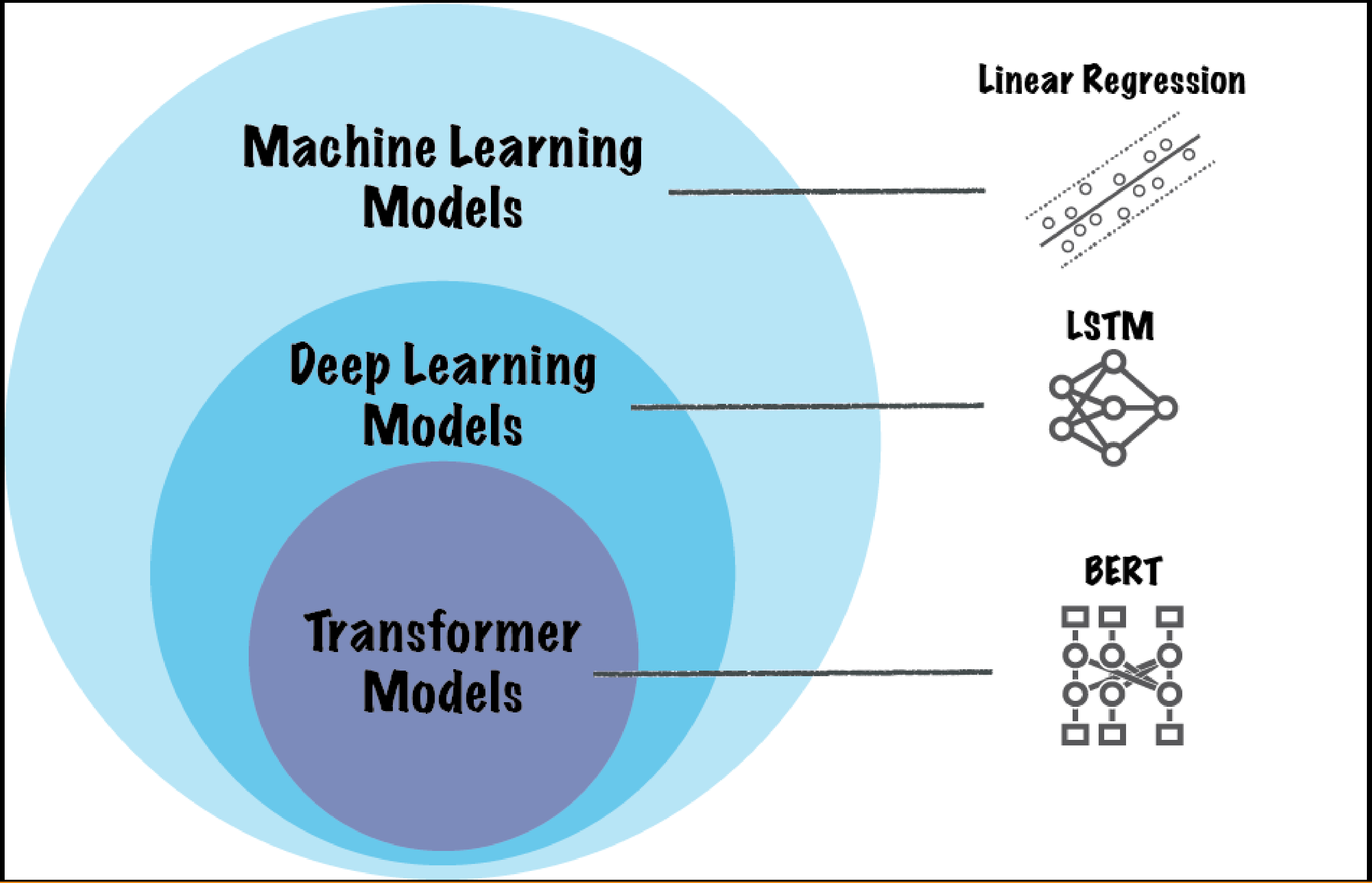
3. The gist of the paper is right there in the title: as it turned out, a simple mechanism called **"neural attention"** could be used to build powerful sequence models that didn't feature any recurrent layers or convolution layers.

Read the paper via this link: https://arxiv.org/abs/1706.03762. or a Google search

Transformers: Attention is All You Need

So now, let’s redefine Transformers,

Transformers is a type of **Neural Network** architecture that is built on **ATTENTION** as its foundational mechanism. **e.g BERT, GPT.**



Transformers: Attention is All You Need

Transformers are a revolutionary type of **neural network** that rely entirely on something called **attention**.

Transformers are based entirely on **attention mechanisms**, eliminating **recurrence** like in RNN.

Unlike RNN that analyze words one by one, **Transformers** read all the words at the same time.

Time to now explain Attention better

Attention

Attention is a technique that allows a neural network to focus on specific parts of an input sequence when processing it.

They enable the model to weigh the importance of different input elements, giving more "**attention**" to the most relevant ones.

Think of it as a way for the model to selectively focus on the most important information.

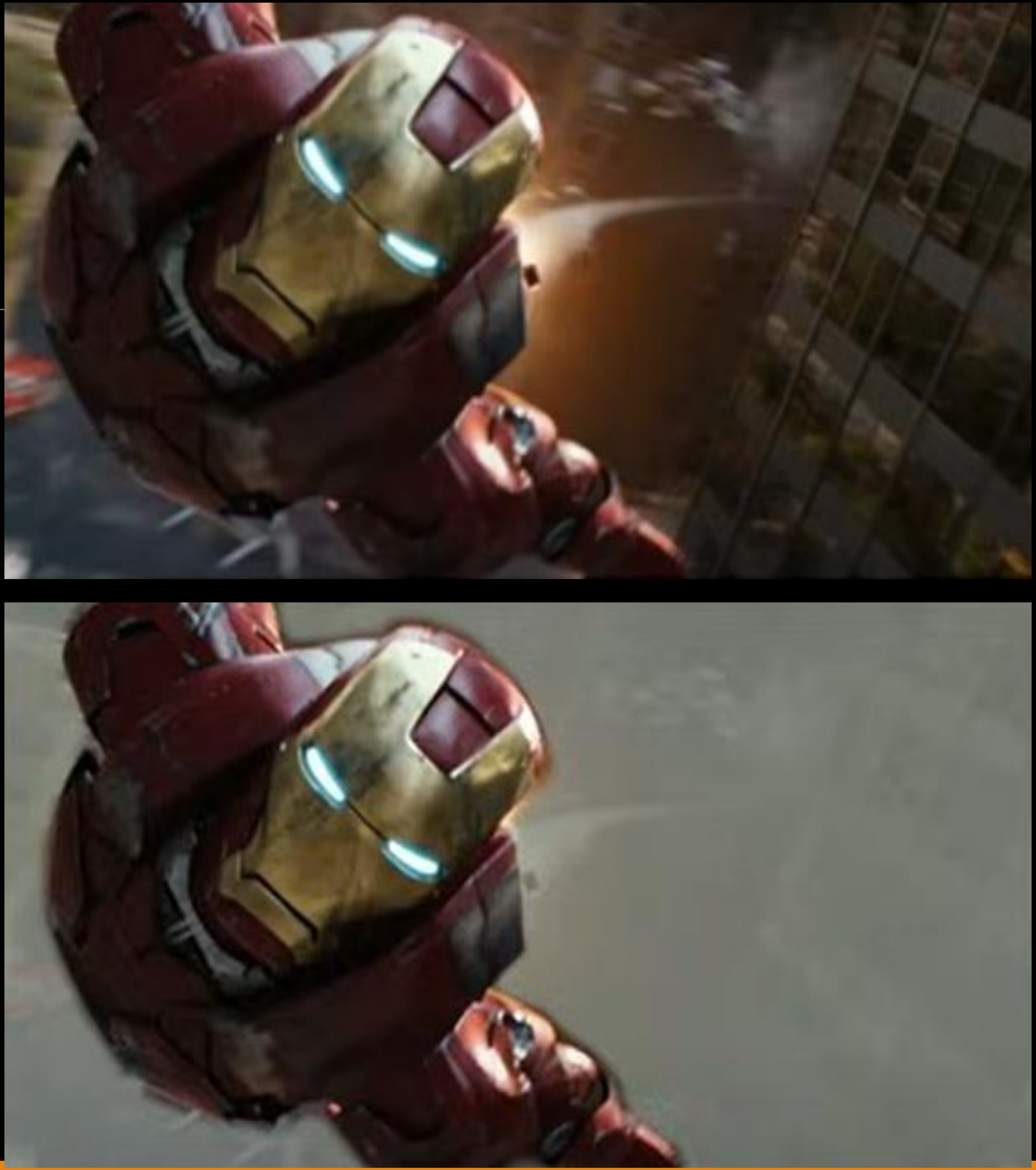
Attention

Attention (or Neural Attention) is a simple yet powerful idea, Because not all input information seen by a model is equally important to the task at hand, so models should **“pay more attention”** to some features and **“pay less attention”** to other features.

Attention

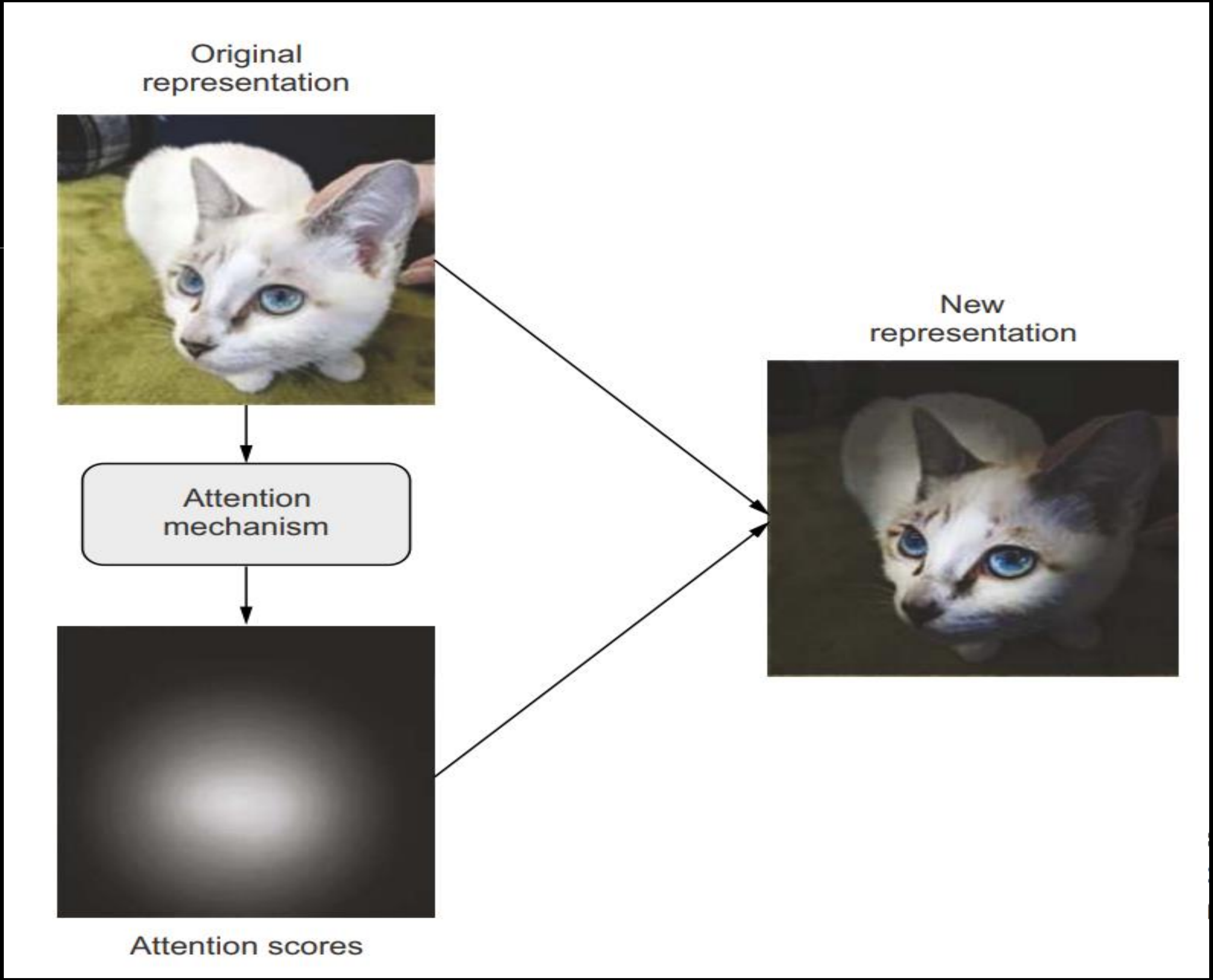
Because not all input information seen by a model is equally important to the task at hand, so models should **"pay more attention"** to some features and **"pay less attention"** to other features.

For a task, What/who is moving?



Attention

This idea looks similar to that of Max pooling in CNN, Max pooling in convnets looks at a pool of features in a spatial region and selects just one feature to keep. That's an "all or nothing" form of attention: keep the most important feature and discard the rest



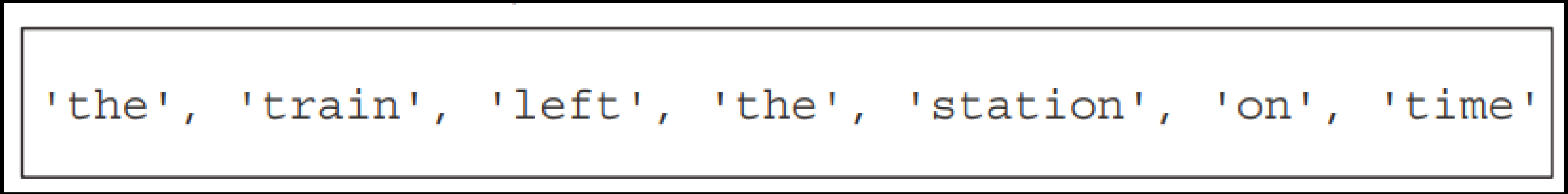
Attention

Let's consider an example sentence:

"the train left the station on time."

Now, consider one word in the sentence: **station**.

What kind of station are we talking about? Could it be a radio station? Maybe the International Space Station? Let's figure it out algorithmically via self-attention



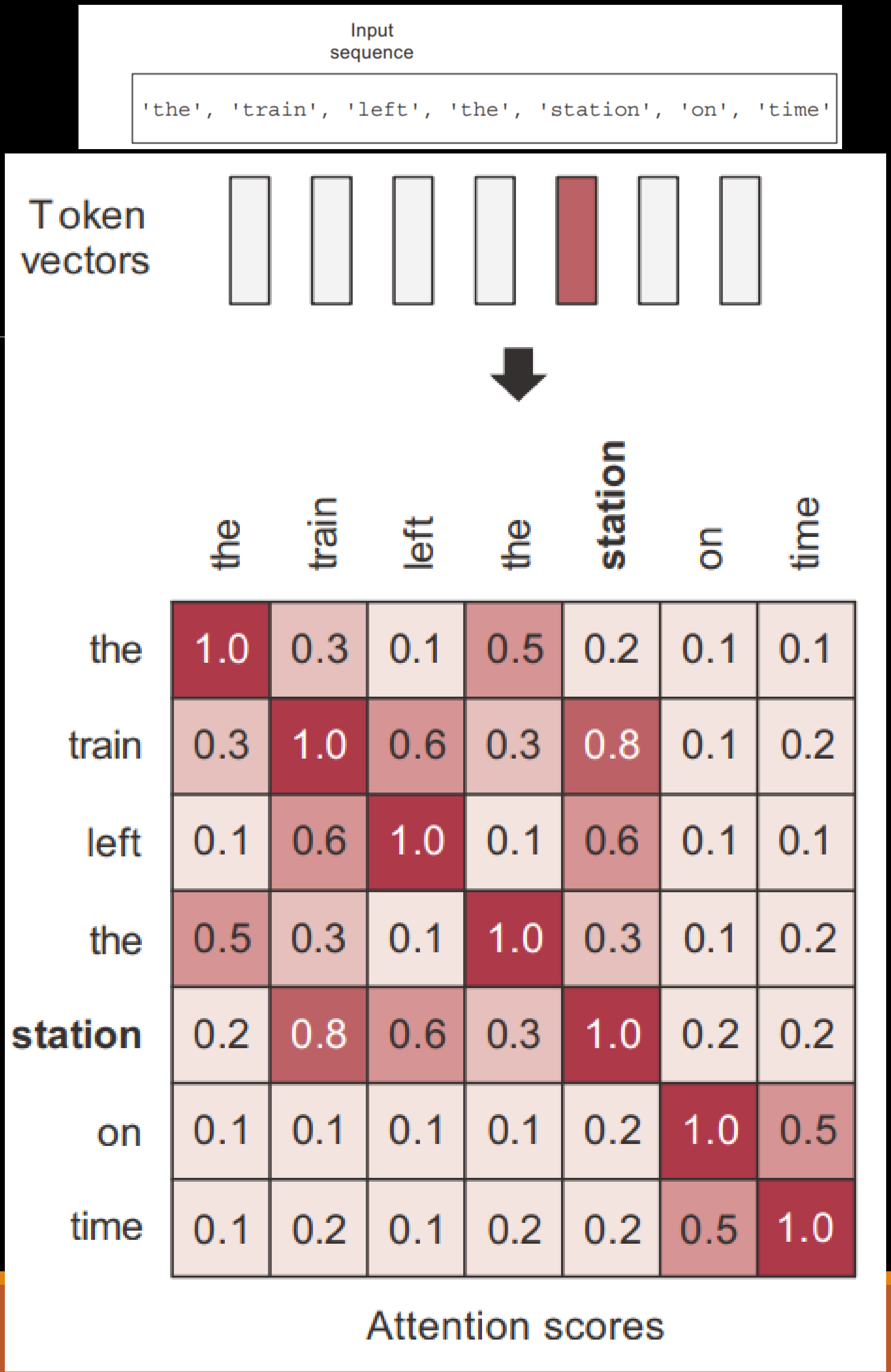
```
'the', 'train', 'left', 'the', 'station', 'on', 'time'
```

Attention

Step 1 is to compute relevancy scores between the vector for “**station**” and every other word in the sentence.

These are our **“attention scores.”** We’re simply going to use the dot product between two-word vectors as a measure of the strength of their relationship.

It’s a very computationally efficient distance function, and it was already the standard way to relate two-word embeddings to each other long before **Transformers**.

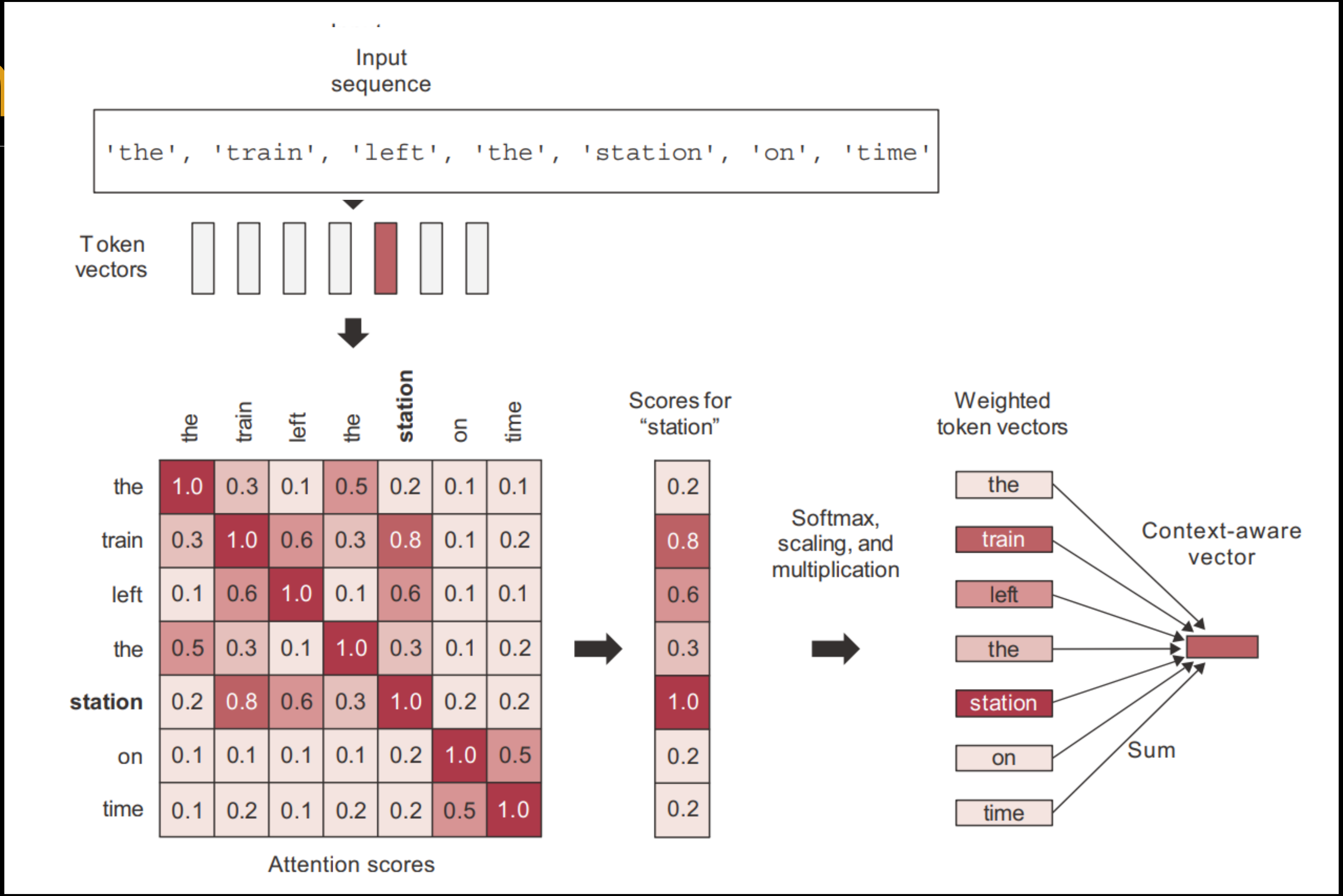


```
'the', 'train', 'left', 'the', 'station', 'on', 'time'
```

	the	train	left	the	**station**	on	time
the	1.0	0.3	0.1	0.5	0.2	0.1	0.1
train	0.3	1.0	0.6	0.3	0.8	0.1	0.2
left	0.1	0.6	1.0	0.1	0.6	0.1	0.1
the	0.5	0.3	0.1	1.0	0.3	0.1	0.2
station	0.2	0.8	0.6	0.3	1.0	0.2	0.2
on	0.1	0.1	0.1	0.1	0.2	1.0	0.5
time	0.1	0.2	0.1	0.2	0.2	0.5	1.0

Attention

Right?



	the	train	left	the	station	on	time
the	1.0	0.3	0.1	0.5	0.2	0.1	0.1
train	0.3	1.0	0.6	0.3	0.8	0.1	0.2
left	0.1	0.6	1.0	0.1	0.6	0.1	0.1
the	0.5	0.3	0.1	1.0	0.3	0.1	0.2
station	0.2	0.8	0.6	0.3	1.0	0.2	0.2
on	0.1	0.1	0.1	0.1	0.2	1.0	0.5
time	0.1	0.2	0.1	0.2	0.2	0.5	1.0

Scores for "station": 0.2, 0.8, 0.6, 0.3, 1.0, 0.2, 0.2

Weighted token vectors: the, train, left, the, station, on, time

Attention

Helps us attend to the most important parts of an input by

1. Identify which parts to attend to

2. Extract the features with high attention

Attention Mechanism (Queries, Keys, Values)

Attention mechanisms helps to fine the right information, using **QUERIES**, **KEYS**, **VALUES.**

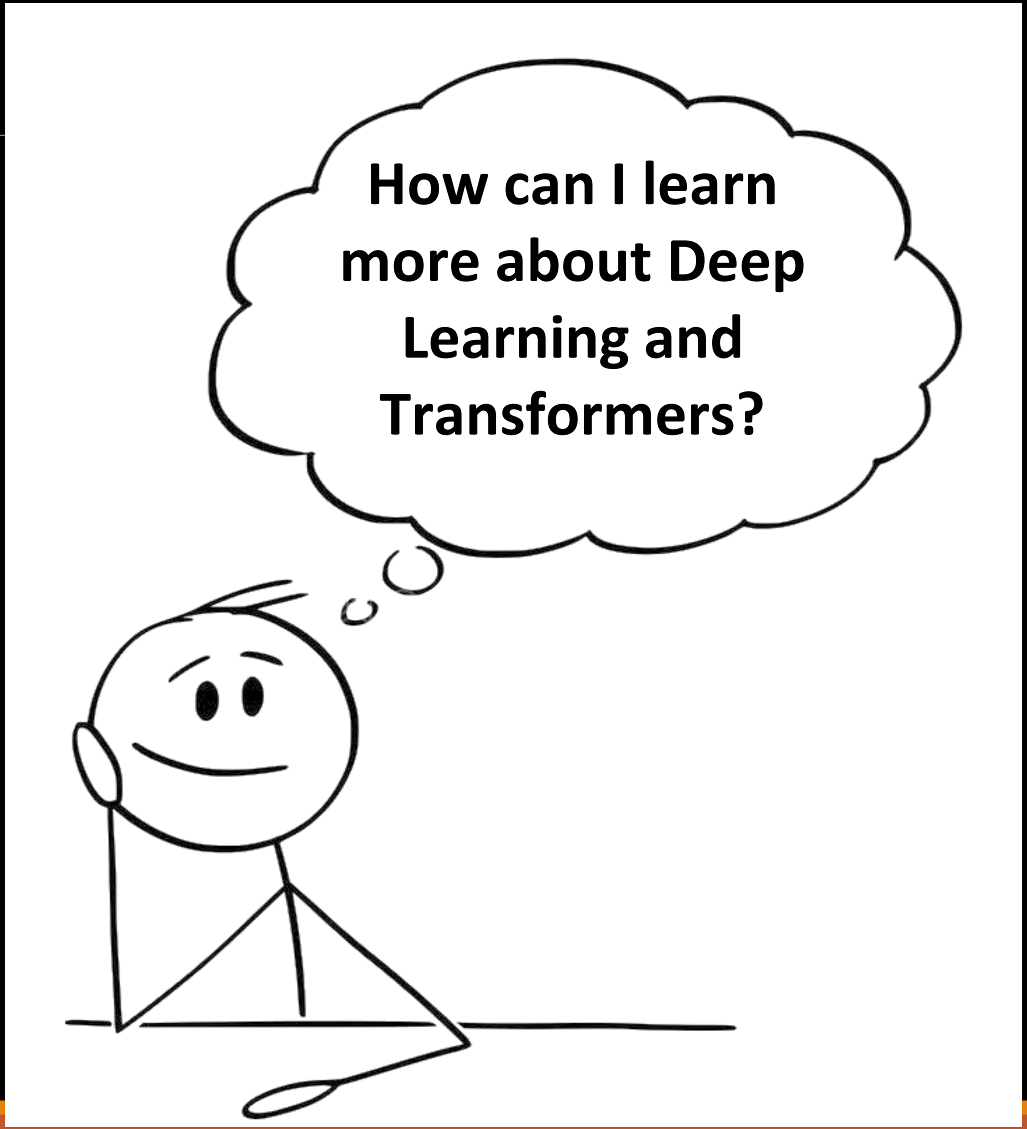
Let me help you understand **Attention Mechanism** with **SEARCH.** (i.e. using a SEARCH example)

Attention Mechanism (Queries, Keys, Values)

Attention mechanisms helps to fine the right information, using **Queries, Keys, Values**

Let's explain like this………

Because you are here to **Learn and Understand NLP and Deep Learning**, you therefore have the question:



Attention Mechanism (Queries, Keys, Values)

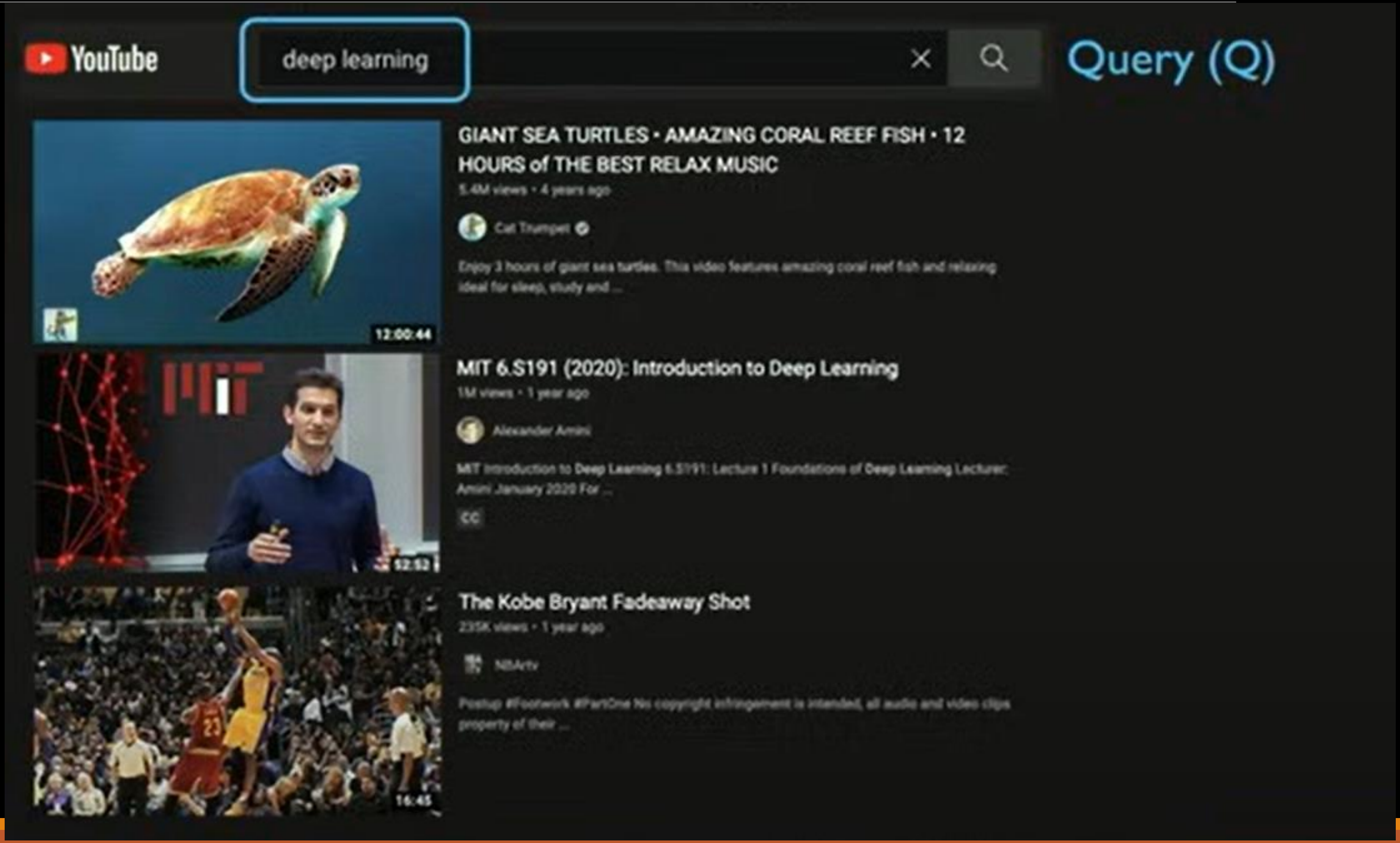
In addition to joining this class, what you may do again is

1. Go search online, YouTube Precisely

Attention Mechanism (Queries, Keys, Values)

As you get to YouTube, how do you find the needed video from the giant database?

- With a search, by supplying the **Query, Deep Learning.**



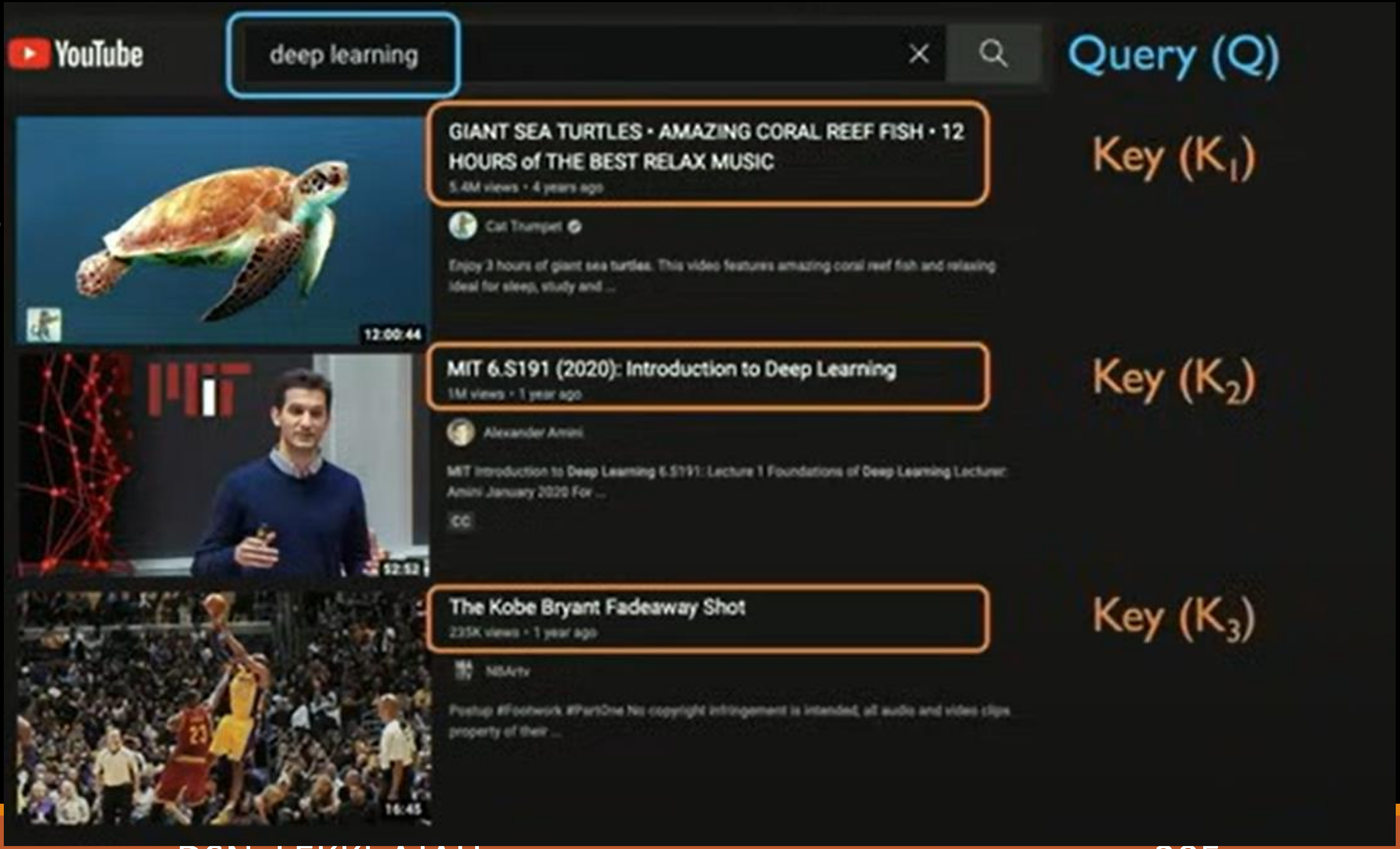
GIANT SEA TURTLES • AMAZING CORAL REEF FISH • 12 HOURS of THE BEST RELAX MUSIC
5.4M views • 4 years ago
Cat Trumpet
Enjoy 3 hours of giant sea turtles. This video features amazing coral reef fish and relaxing ideal for sleep, study and ...

MIT 6.S191 (2020): Introduction to Deep Learning
1M views • 1 year ago
Alexander Amini
MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...
CC

The Kobe Bryant Fadeaway Shot
235K views • 1 year ago
NBAtv
Postup #Footwork #PartOne No copyright infringement is intended, all audio and video clips property of their ...

Attention Mechanism (Queries, Keys, Values)

Now, the submitted query needs to be compared with **title** of videos on YouTube. All the titles are considered **Keys**



GIANT SEA TURTLES • AMAZING CORAL REEF FISH • 12 HOURS of THE BEST RELAX MUSIC

5.4M views • 4 years ago

Cat Trumpet

Enjoy 3 hours of giant sea turtles. This video features amazing coral reef fish and relaxing ideal for sleep, study and ...

MIT 6.S191 (2020): Introduction to Deep Learning

1M views • 1 year ago

Alexander Amini

MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...

CC

The Kobe Bryant Fadeaway Shot

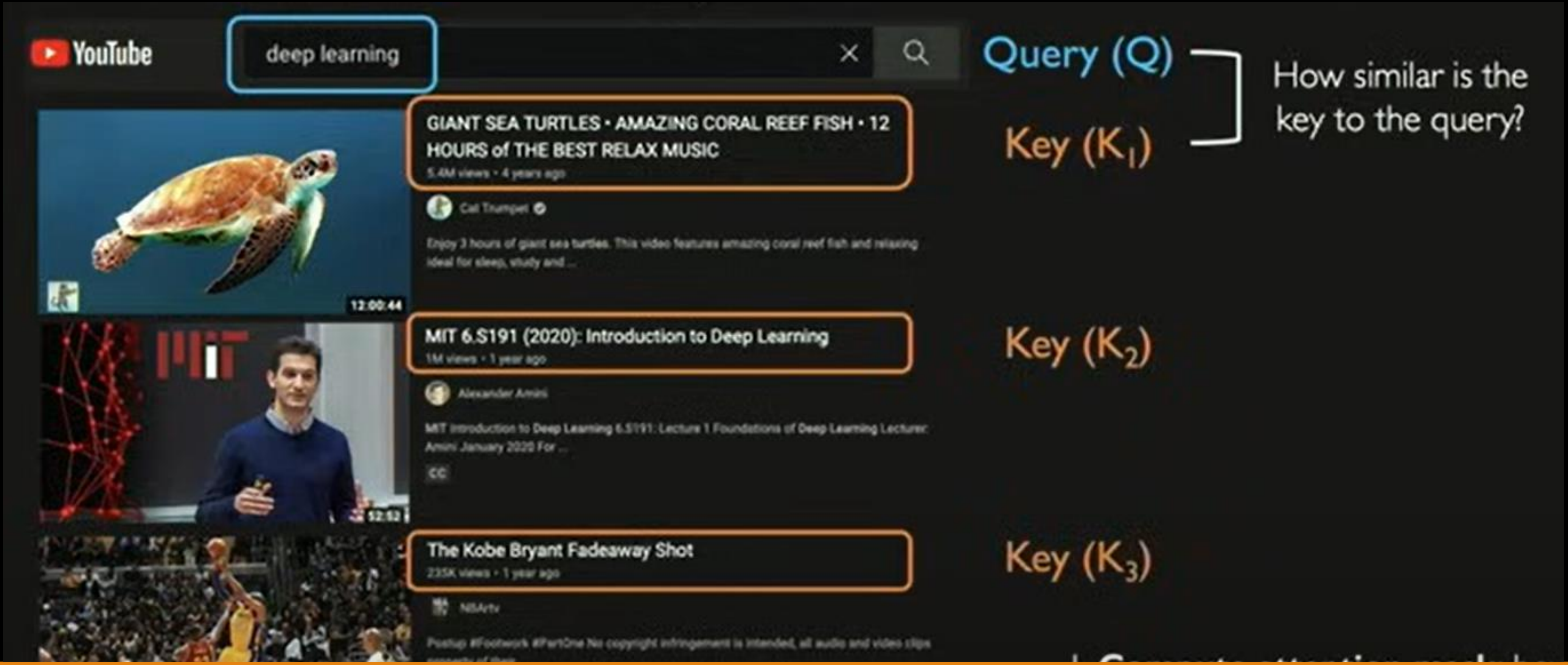
235K views • 1 year ago

NBAtv

Postup #Footwork #PartOne No copyright infringement is intended, all audio and video clips property of their ...

Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,



YouTube

deep learning

GIANT SEA TURTLES • AMAZING CORAL REEF FISH • 12 HOURS of THE BEST RELAX MUSIC

5.4M views • 4 years ago

Cat Trumpet

Enjoy 3 hours of giant sea turtles. This video features amazing coral reef fish and relaxing ideal for sleep, study and ...

12:00:44

MIT 6.S191 (2020): Introduction to Deep Learning

1M views • 1 year ago

Alexander Amini

MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...

CC

52:52

The Kobe Bryant Fadeaway Shot

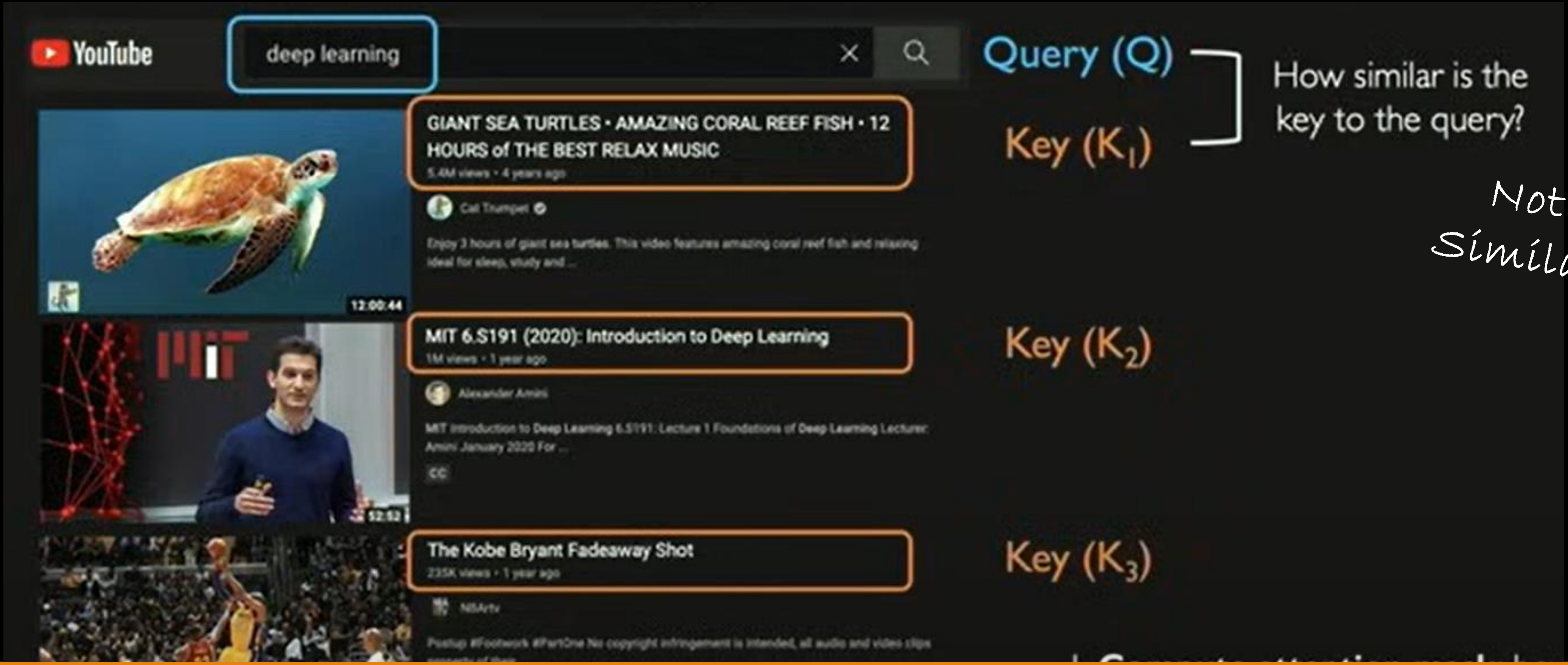
235K views • 1 year ago

NBArtv

Postup #Footwork #PartOne No copyright infringement is intended, all audio and video clips

Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,



YouTube

deep learning

GIANT SEA TURTLES • AMAZING CORAL REEF FISH • 12 HOURS of THE BEST RELAX MUSIC
5.4M views • 4 years ago
Cat Trumpet
Enjoy 3 hours of giant sea turtles. This video features amazing coral reef fish and relaxing ideal for sleep, study and ...

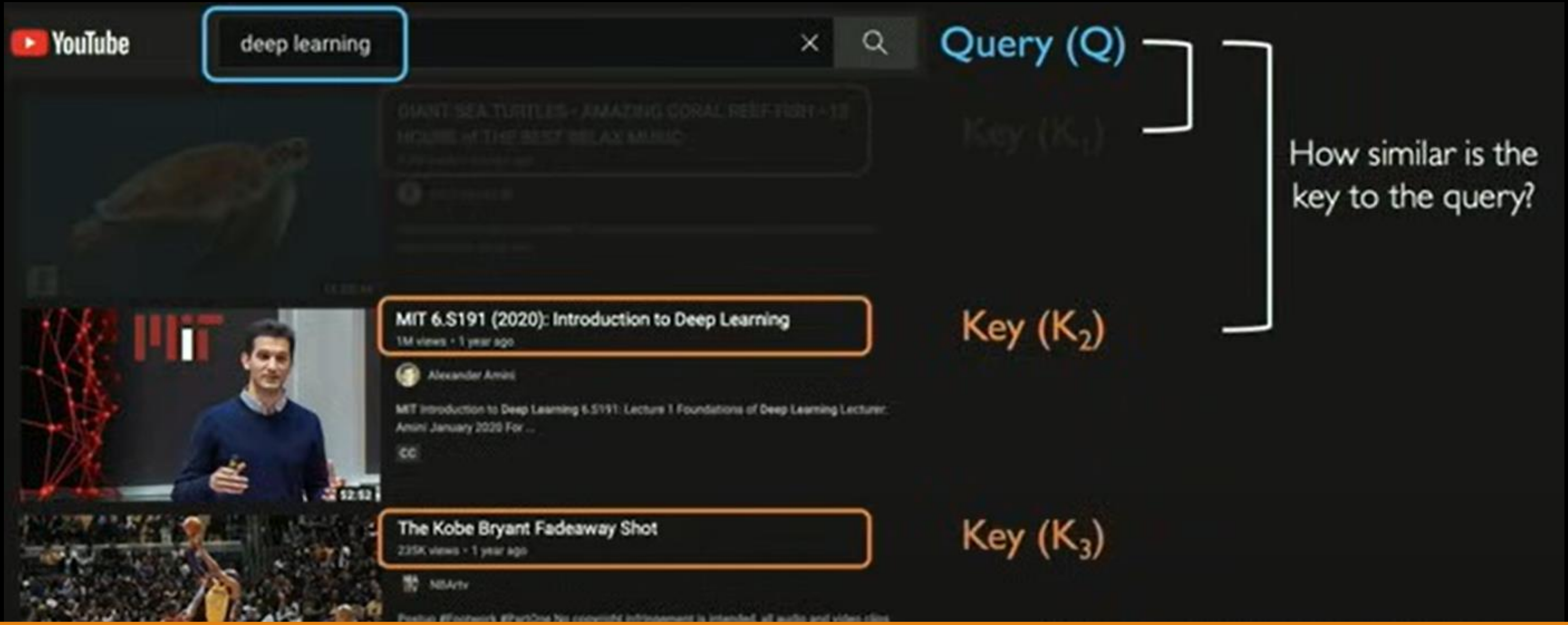
MIT 6.S191 (2020): Introduction to Deep Learning
1M views • 1 year ago
Alexander Amini
MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...
CC

The Kobe Bryant Fadeaway Shot
235K views • 1 year ago
NBArtv
Postup #Footwork #PartOne No copyright infringement is intended, all audio and video clips

Not Similar

Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,

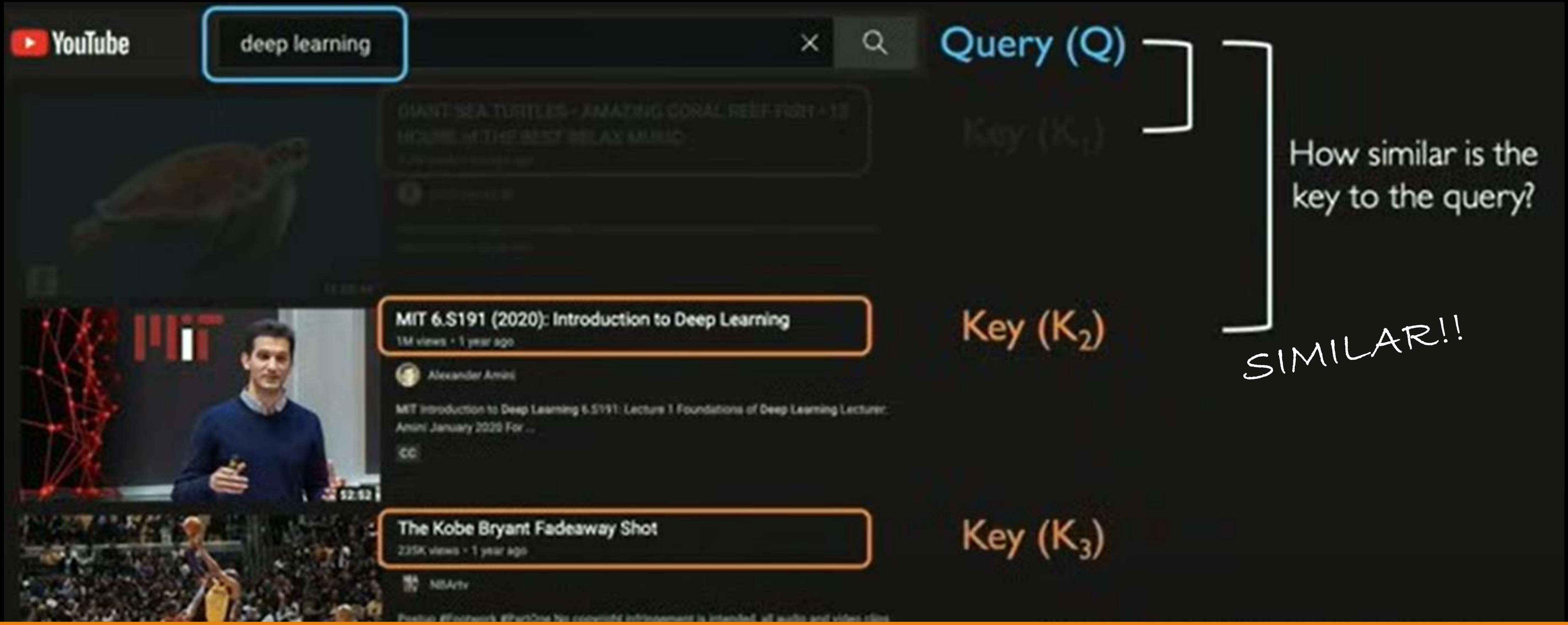


MIT 6.S191 (2020): Introduction to Deep Learning
1M views • 1 year ago
Alexander Amini
MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...

The Kobe Bryant Fadeaway Shot
235K views • 1 year ago
NBArtv

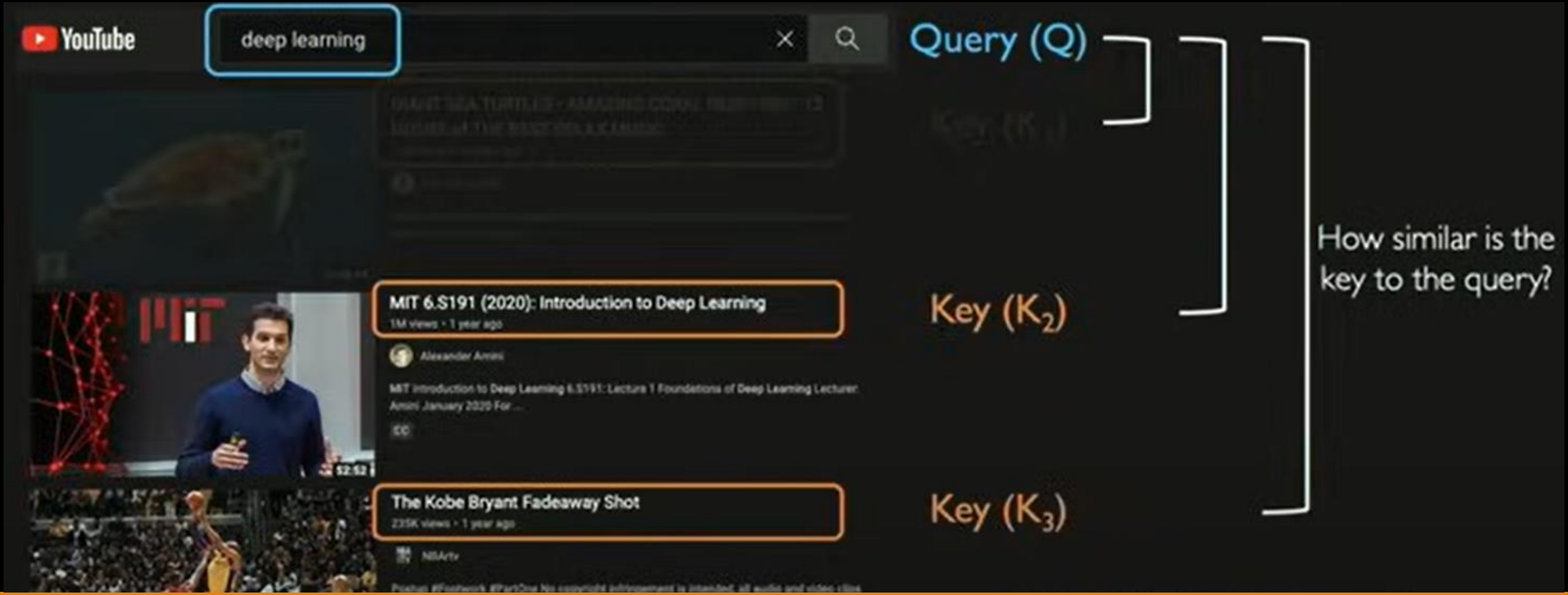
Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,


YouTube
deep learning
Query (Q)
GIANT SEA TURTLES - AMAZING CORAL REEF FISH - 12 HOURS of THE BEST RELAX MUSIC
Key (K₁)
How similar is the key to the query?
MIT 6.S191 (2020): Introduction to Deep Learning
1M views · 1 year ago
Alexander Amini
MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...
Key (K₂)
SIMILAR!!
The Kobe Bryant Fadeaway Shot
235K views · 1 year ago
NBA.tv
Key (K₃)

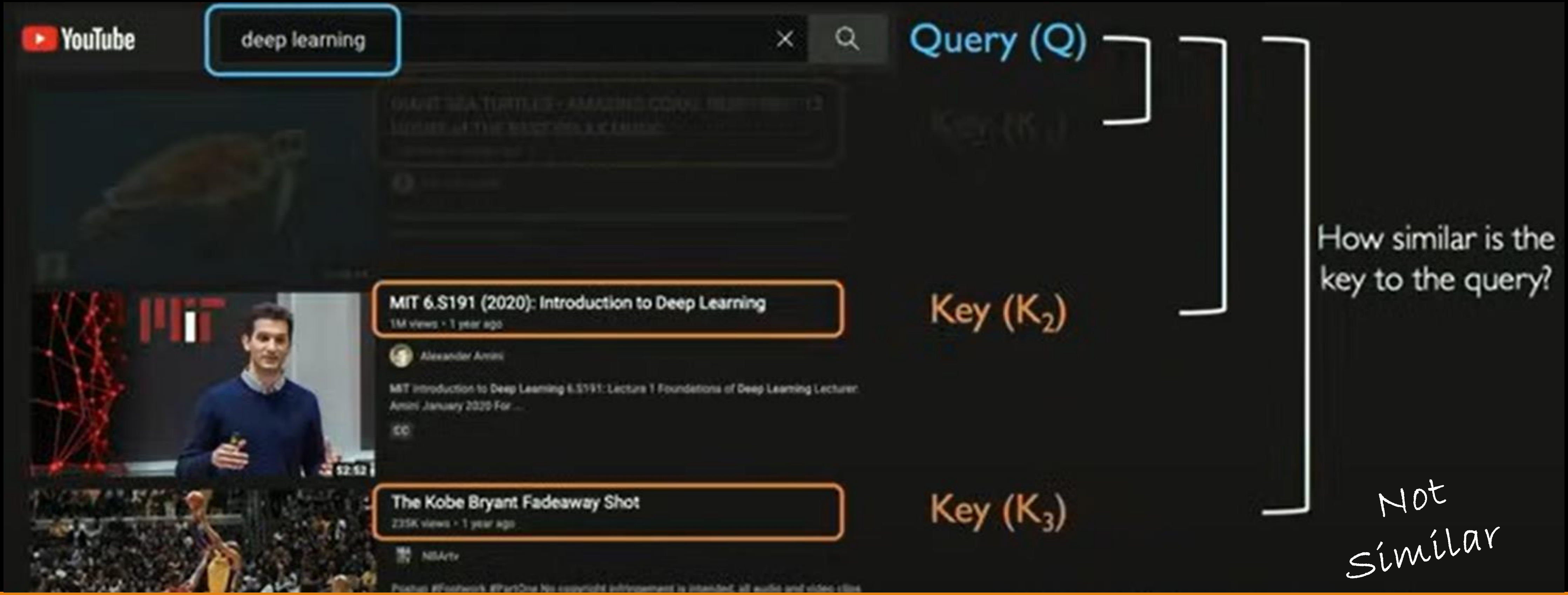
Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,


YouTube
deep learning
Query (Q)
Key (K_1)
Key (K_2)
Key (K_3)
How similar is the key to the query?
MIT 6.S191 (2020): Introduction to Deep Learning
The Kobe Bryant Fadeaway Shot

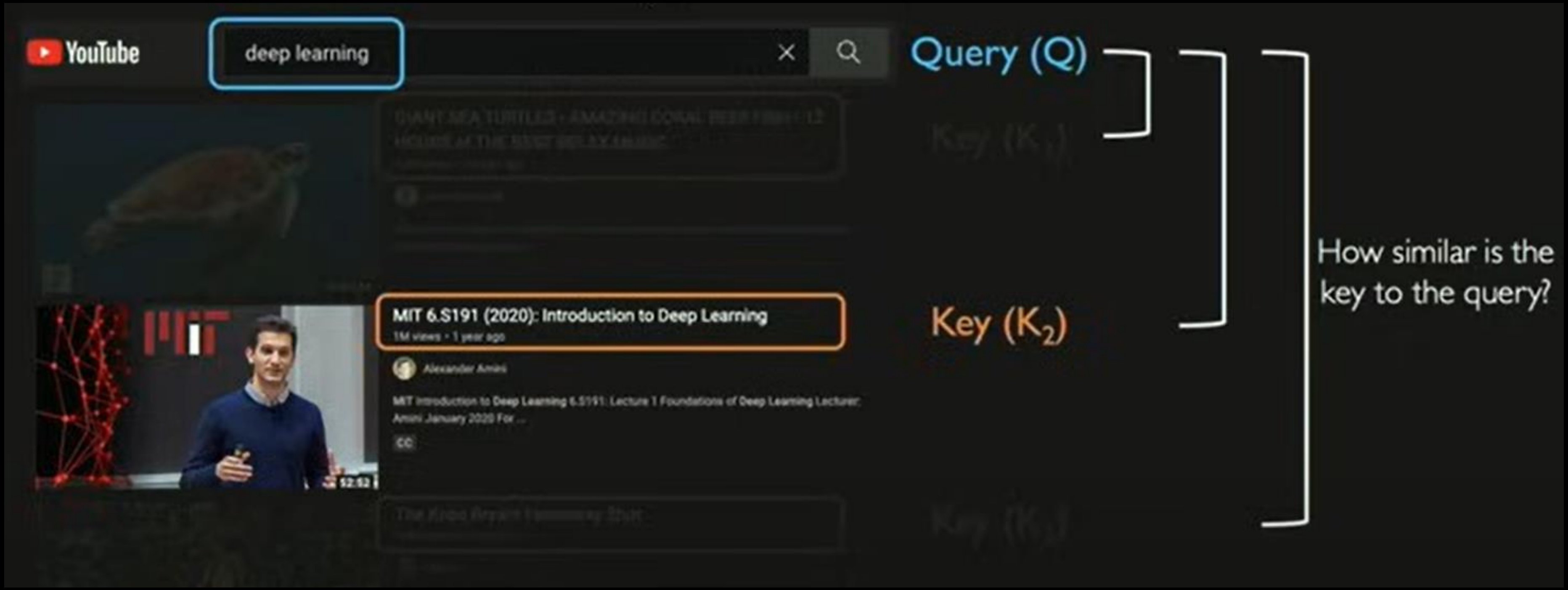
Attention Mechanism (Queries, Keys, Values)

Next is to Take my query and check for Keys with closest meaning,



Attention Mechanism (Queries, Keys, Values)

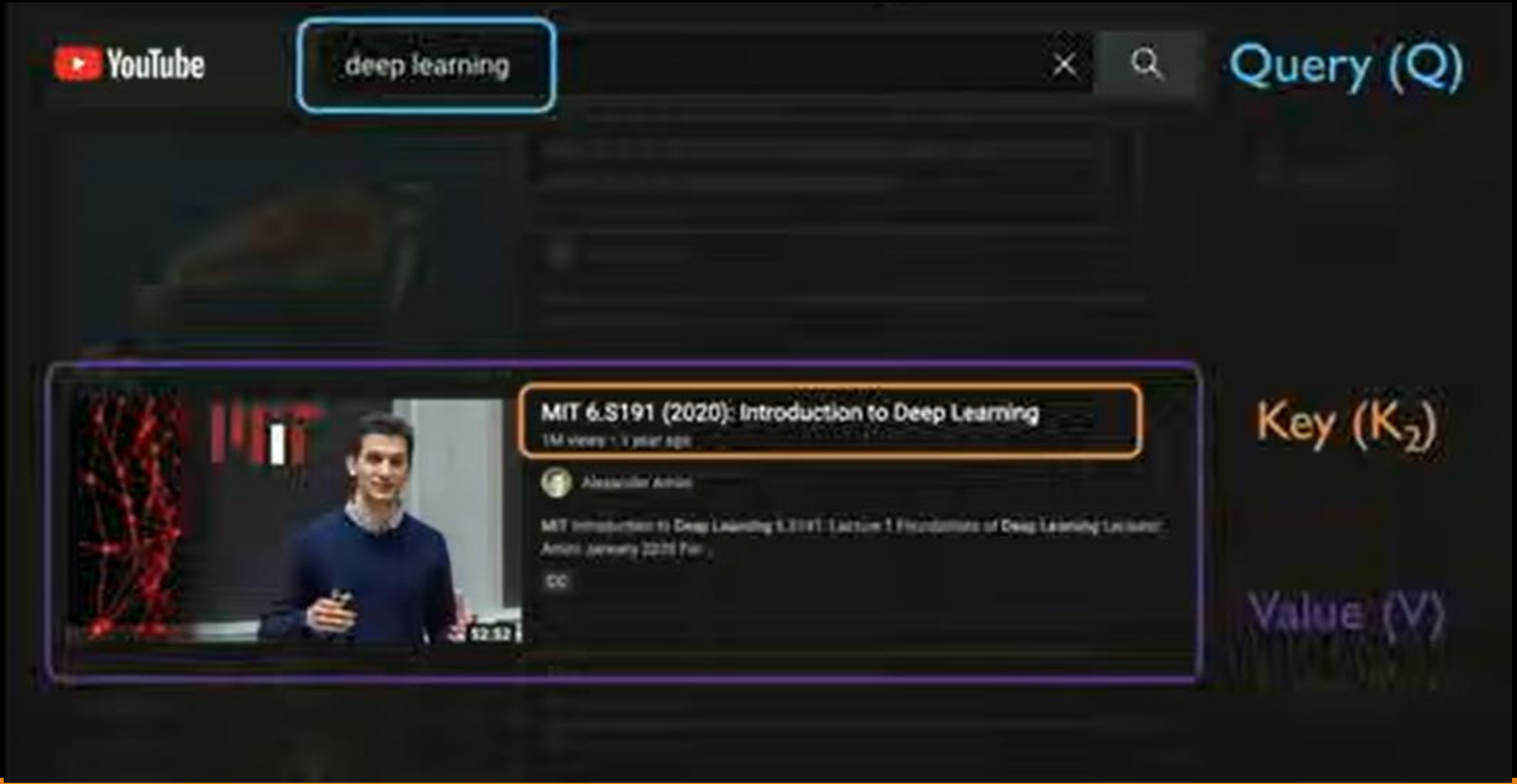
The check of how similar is each key to the desired Query is a process called **“Computing Attention Mask”**


YouTube
deep learning
Query (Q)
Key (K_1)
MIT 6.S191 (2020): Introduction to Deep Learning
1M views • 1 year ago
Alexander Amini
MIT Introduction to Deep Learning 6.S191: Lecture 1 Foundations of Deep Learning Lecturer: Amini January 2020 For ...
CC
52:52
Key (K_2)
Key (K_3)
How similar is the key to the query?

Attention Mechanism (Queries, Keys, Values)

After Explaining **Queries and Keys,** how about Value?? Let's move on.........

Value, basically the key with the **highest Attention Number**



MIT 6.S191 (2020): Introduction to Deep Learning

Attention Mechanism (Queries, Keys, Values)

Attention mechanisms helps to find the right information, using Queries, Keys, Values

Let’s use another analogy: **Searching for a book in a library**

Query (Question): What you're looking for **(e.g., 'books about animals’).**

Keys (Labels): The labels on the books (e.g., titles, keywords like 'science’, ‘animals’, ‘physics’ etc).

Values (Information): The actual books (for this example, books about animals).

The computer compares your 'Query' with the 'Keys' to find the most relevant 'Values'.

What's Next after Attention Mechanism 😀 😀

Let's briefly discuss another important layer and component of a Model Architecture

Encoder

and

Decoder

What’s Encoder-Decoder

The Encoder-Decoder architecture is a neural network design pattern that's particularly effective for sequence-to-sequence tasks. It's used in various applications, including machine translation, text summarization, and question answering.

Models that used **Encoder** alone are tagged **Encode-Only Model, eg BERT**

Models that used **Decoder** alone are tagged **Decode-Only Model, eg GPT**

Models that use both an **Encoder** and a **Decoder** are tagged **Encoder-Decoder Models, e.g: T5 by Google, BART by Facebook AI**

Encoder-Decoder

The Encoder-Decoder architecture though, isn't exclusive to Transformers. While Transformers popularized it and significantly improved its performance, the general concept has been used with RNNs and, to a lesser extent, with CNNs as well.

A typical **Seq2Seq model** consists of an RNN encoder and an RNN decoder.

But: Transformers have significantly improved the performance of Encoder-Decoder models, particularly for NLP tasks.

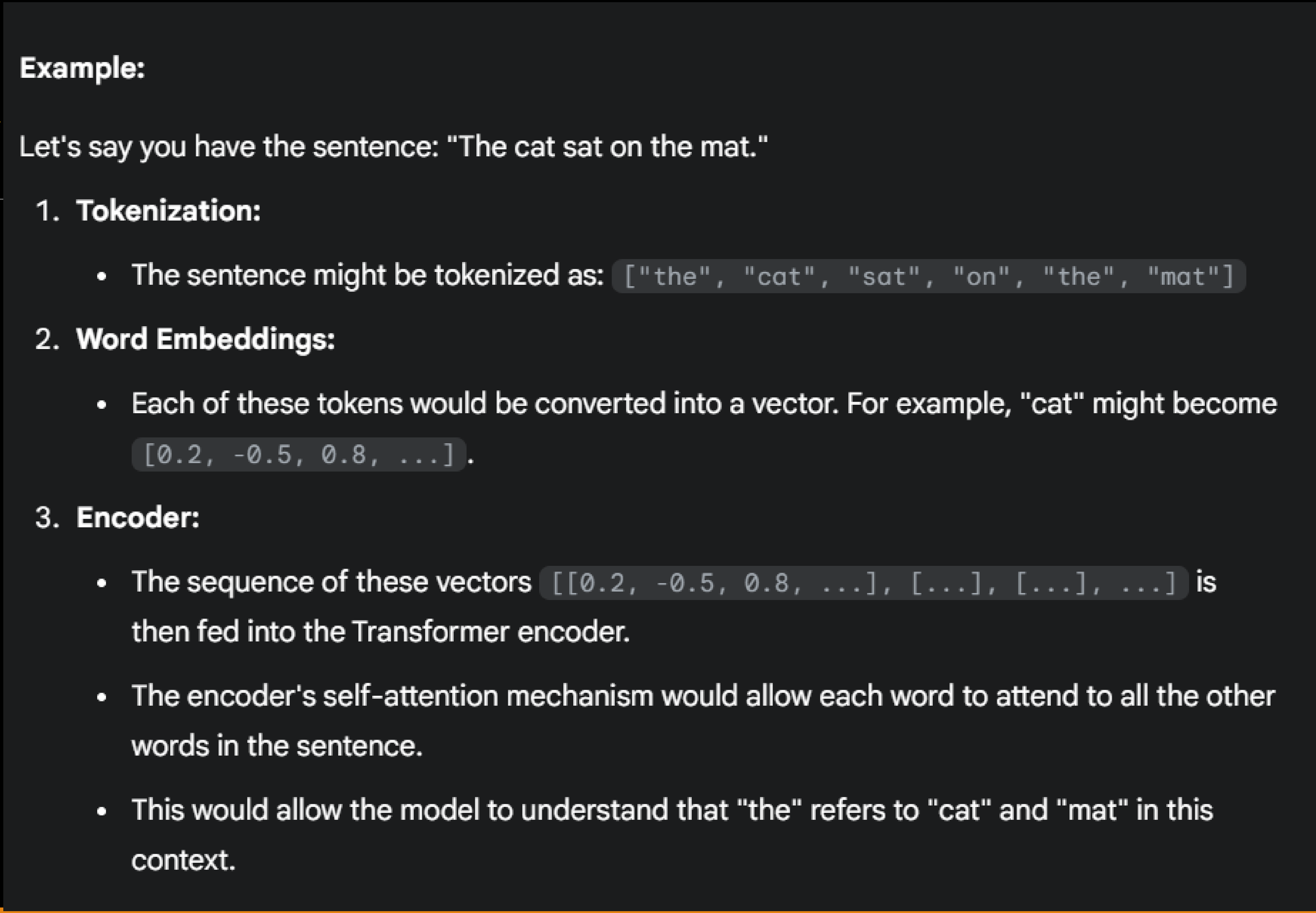
Encoder

The Encoder: Helps to understand the Input. The encoder's job is to take an input sequence of work embeddings (e.g., the token of an English sentence) processes these embeddings, using **self-attention** and **feed-forward networks**, to create contextualized representations of the input sequence.

i.e The encoder then takes those numerical representations and builds a deeper understanding of the relationships between the tokens in the context of the entire input sequence.

Encoder

E.g.:



Example:

Let's say you have the sentence: "The cat sat on the mat."

1. **Tokenization:**
   - The sentence might be tokenized as: `["the", "cat", "sat", "on", "the", "mat"]`
2. **Word Embeddings:**
   - Each of these tokens would be converted into a vector. For example, "cat" might become `[0.2, -0.5, 0.8, ...]`.
3. **Encoder:**
   - The sequence of these vectors `[[0.2, -0.5, 0.8, ...], [...], [...], ...]` is then fed into the Transformer encoder.
   - The encoder's self-attention mechanism would allow each word to attend to all the other words in the sentence.
   - This would allow the model to understand that "the" refers to "cat" and "mat" in this context.

Encoder-Only Models

- **BERT (Bidirectional Encoder Representations from Transformers):** Its bidirectional training and **masked language modeling** focus on deep contextual understanding.
- **RoBERTa (Robustly Optimized BERT Pretraining Approach):** An optimized variant of BERT, with improved training procedures and larger datasets.
- **ALBERT (A Lite BERT for Self-supervised Learning of Language Representations):** A parameter-efficient version of BERT, using parameter sharing and sentence-order prediction.
- **DistilBERT:** A distilled version of BERT, that is much smaller and faster, but retains much of the performance.

Decoder

The Decoder: The decoder's primary function is to generate output sequences, such as translated sentences, summaries, or chatbot responses. It does this step-by-step, taking into account the encoded representation of the input and the previously generated tokens.

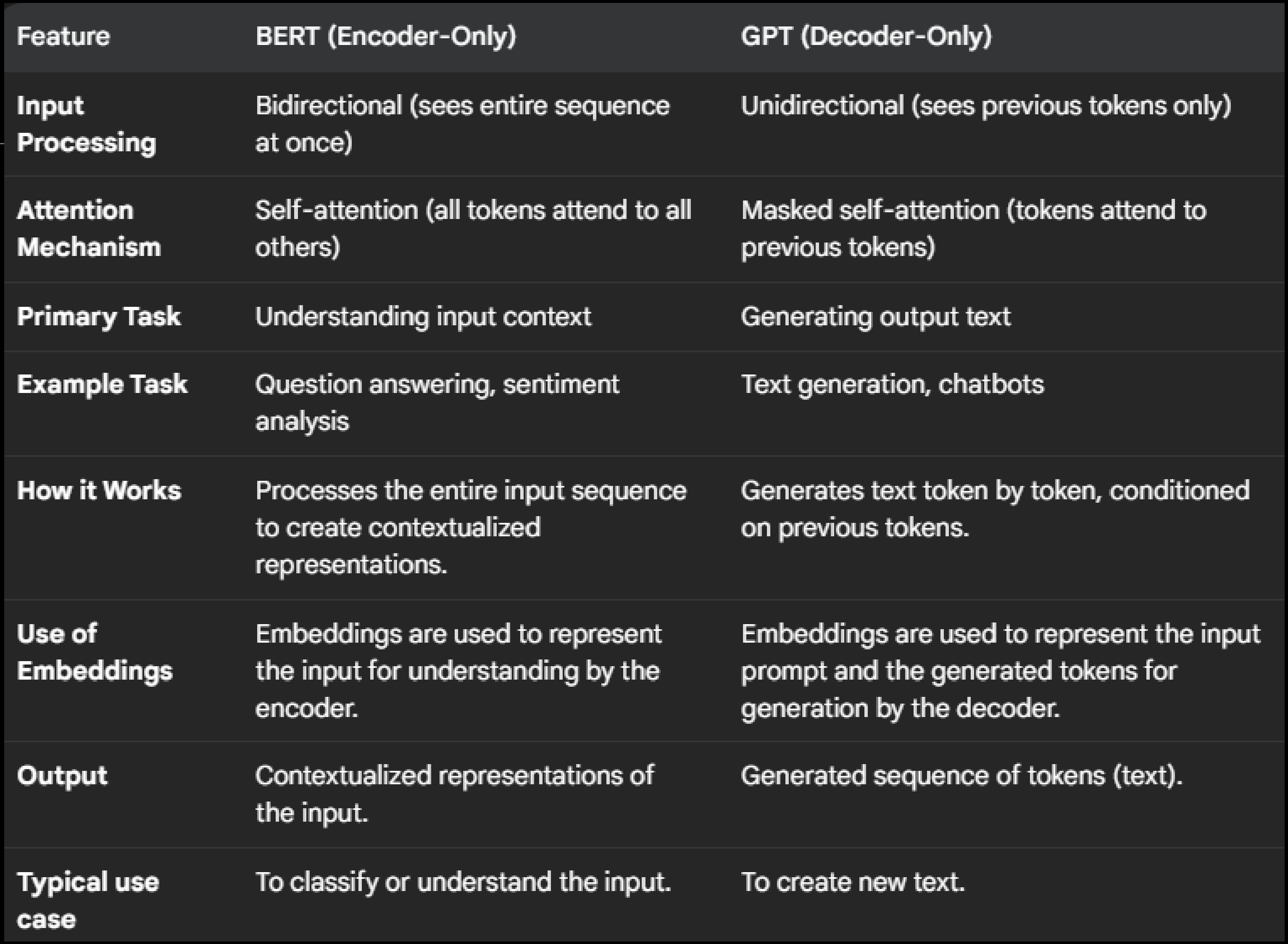
The **decoder** receives the encoded representations from the encoder (in encoder-decoder architectures) or the context from previous decoder layers (in decoder-only architectures like GPT)

Decoder-Only Models

- **GPT (Generative Pre-trained Transformer) Family (GPT, GPT-2, GPT-3, GPT-4):** These models are the prime examples of decoder-only architectures. Their autoregressive nature and causal language modeling are geared towards text generation.
- **CTRL (Conditional Transformer Language Model):** Allows for fine-grained control over text generation through control codes.
- **Transformer-XL (Transformer Extra Long):** An enhanced version of the transformer decoder that is designed to handle much longer context windows.

Comparison

Encoder-Only (e.g BERT) vs. Decoder-Only (e.g GPT)



Feature	BERT (Encoder-Only)	GPT (Decoder-Only)
Input Processing	Bidirectional (sees entire sequence at once)	Unidirectional (sees previous tokens only)
Attention Mechanism	Self-attention (all tokens attend to all others)	Masked self-attention (tokens attend to previous tokens)
Primary Task	Understanding input context	Generating output text
Example Task	Question answering, sentiment analysis	Text generation, chatbots
How it Works	Processes the entire input sequence to create contextualized representations.	Generates text token by token, conditioned on previous tokens.
Use of Embeddings	Embeddings are used to represent the input for understanding by the encoder.	Embeddings are used to represent the input prompt and the generated tokens for generation by the decoder.
Output	Contextualized representations of the input.	Generated sequence of tokens (text).
Typical use case	To classify or understand the input.	To create new text.

Can a model? 😀 😀

Let's attend to this

Can a model have both Encode and Decoder Architecture?

Can a model? 😀 😀

Let's attend to this

They are called Encode-Decoder Models

Encoder-Decoder Transfer Learning Models:

- **T5 (Text-to-Text Transfer Transformer):** Trained to treat all NLP tasks as text-to-text problems. Available on Hugging Face: t5-small, t5-base, t5-large, etc. It's Highly versatile for various NLP tasks.

T5 Model links:

- t5-small: https://huggingface.co/t5-small
- t5-base: https://huggingface.co/t5-base
- t5-large: https://huggingface.co/t5-large

Encoder-Decoder Transfer Learning Models:

- **BART (Bidirectional and Auto-Regressive Transformers):** Excellent for sequence-to-sequence tasks like summarization and translation. Available on Hugging Face: facebook/bart-base, facebook/bart-large, etc.

BART Models link:

- facebook/bart-base: https://huggingface.co/facebook/bart-base
- facebook/bart-large: https://huggingface.co/facebook/bart-large

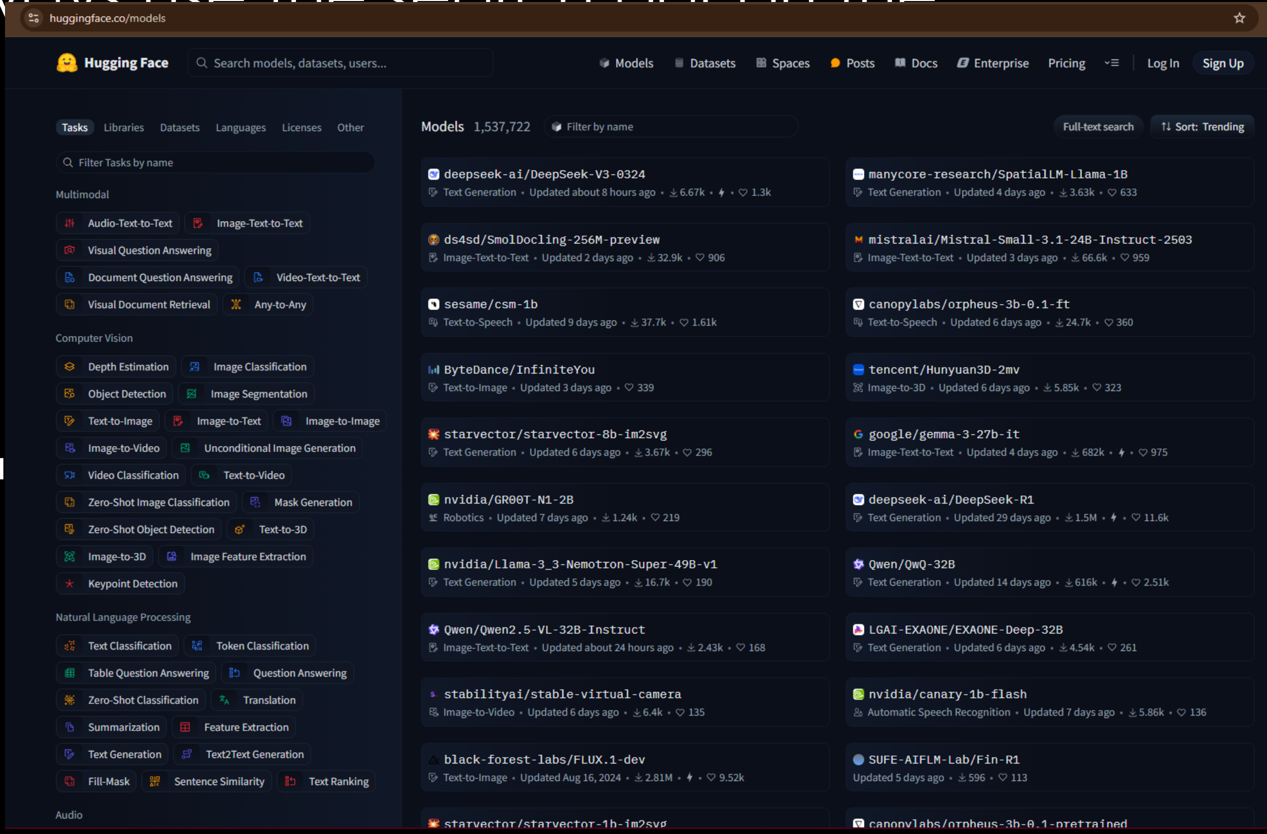
Encoder-Decoder Transfer Learning Models:

- **MarianMT:** Specifically designed for machine translation. Available on Hugging Face: Helsinki-NLP/opus-mt-*. A large number of language pairs are available. I can find all MarianMT models by searching "Helsinki-NLP/opus-mt" on Hugging Face. https://huggingface.co/Helsinki-NLP
- **Pegasus:** Designed for abstractive summarization. Available on hugging face: google/pegasus-*. https://huggingface.co/google/pegasus-large
- **Nb: the '*' means that there are many models that follow the pattern: e.g Helsinki-NLP/opus-mt-en-fr, Helsinki-NLP/opus-mt-fr-en, etc.**

Hugging Face Hub Search:

To find other models, you can always use the search bar on the Hugging Face Hub:

https://huggingface.co/models.



Models 1,537,722

deepseek-ai/DeepSeek-V3-0324
Text Generation • Updated about 8 hours ago • 6.67k • • 1.3k

manycore-research/SpatialLM-Llama-1B
Text Generation • Updated 4 days ago • 3.63k • 633

ds4sd/SmolDocling-256M-preview
Image-Text-to-Text • Updated 2 days ago • 32.9k • 906

mistralai/Mistral-Small-3.1-24B-Instruct-2503
Image-Text-to-Text • Updated 3 days ago • 66.6k • 959

sesame/csm-1b
Text-to-Speech • Updated 9 days ago • 37.7k • 1.61k

canopylabs/orpheus-3b-0.1-ft
Text-to-Speech • Updated 6 days ago • 24.7k • 360

ByteDance/InfiniteYou
Text-to-Image • Updated 3 days ago • 339

tencent/Hunyuan3D-2mv
Image-to-3D • Updated 6 days ago • 5.85k • 323

starvector/starvector-8b-im2svg
Text Generation • Updated 6 days ago • 3.67k • 296

google/gemma-3-27b-it
Image-Text-to-Text • Updated 4 days ago • 682k • • 975

nvidia/GR00T-N1-2B
Robotics • Updated 7 days ago • 1.24k • 219

deepseek-ai/DeepSeek-R1
Text Generation • Updated 29 days ago • 1.5M • • 11.6k

nvidia/Llama-3_3-Nemotron-Super-49B-v1
Text Generation • Updated 5 days ago • 16.7k • 190

Qwen/QwQ-32B
Text Generation • Updated 14 days ago • 616k • • 2.51k

Qwen/Qwen2.5-VL-32B-Instruct
Image-Text-to-Text • Updated about 24 hours ago • 2.43k • 168

LGAI-EXAONE/EXAONE-Deep-32B
Text Generation • Updated 6 days ago • 4.54k • 261

stabilityai/stable-virtual-camera
Image-to-Video • Updated 6 days ago • 6.4k • 135

nvidia/canary-1b-flash
Automatic Speech Recognition • Updated 7 days ago • 5.86k • 136

black-forest-labs/FLUX.1-dev
Text-to-Image • Updated Aug 16, 2024 • 2.81M • • 9.52k

SUFE-AIFLM-Lab/Fin-R1
Updated 5 days ago • 596 • 113

starvector/starvector-1b-im2svg

canopylabs/orpheus-3b-0.1-pretrained

NB: When using the Hugging Face hub, it is always a good idea to read the model card that is attached to each model, as it contains important information about the model, such as intended usage, limitations, and ethical considerations.

Evaluating models with metrics

It is impossible to compare one transformer model to another transformer model (or any other NLP model) without a universal measurement system that uses metrics.

1. Accuracy score
2. F1-score
3. Matthews Correlation Coefficient (MCC)

The basic function is:

Accuracy score

$$Accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i)$$

The accuracy score, in whatever variant you use, is a practical evaluation. The score function calculates a straightforward true or false value for each result. Either the model's outputs, yyy, match the correct predictions, yy, for a given subset, samples$_i$, of a set of samples or not.

We will obtain 1 if the result for the subset is correct and 0 if it is false.

Let's now examine the more flexible F1-score.

F1-score

The F1-score introduces a more flexible approach that can help when faced with datasets containing uneven class distributions.

The F1-score uses the weighted values of precision and recall. It is a weighted average of precision and recall values:

$$P = \frac{T_p}{T_p + F_p}$$

$$R = \frac{T_p}{T_p + F_n}$$

F1score= 2* (precision * recall)/(precision + recall)

The equation is:

$$F1\ score = 2 \times \frac{P \times R}{P + R}$$

In its equation, true *(T)* positives *(p)*, false *(F)* positives *(p)*, and false *(F)* negatives *(n)* are plugged into the precision *(P)* and recall *(R)* equations

Matthews Correlation Coefficient (MCC)

The Matthews Correlation Coefficient (MCC) was initially designed to measure the quality of binary classifications and can be modified to be a multi-class correlation coefficient.

A two-class classification can be made with four probabilities at each prediction:

• TP = True Positive, • TN = True Negative, • FP = False Positive, • FN = False Negative

Brian W. Matthews, a biochemist, designed it in 1975, inspired by his predecessors' phi function. Since then, it has evolved into various formats such as the following one:

$$MCC = \frac{TP \ x \ TN - FP \ x \ FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

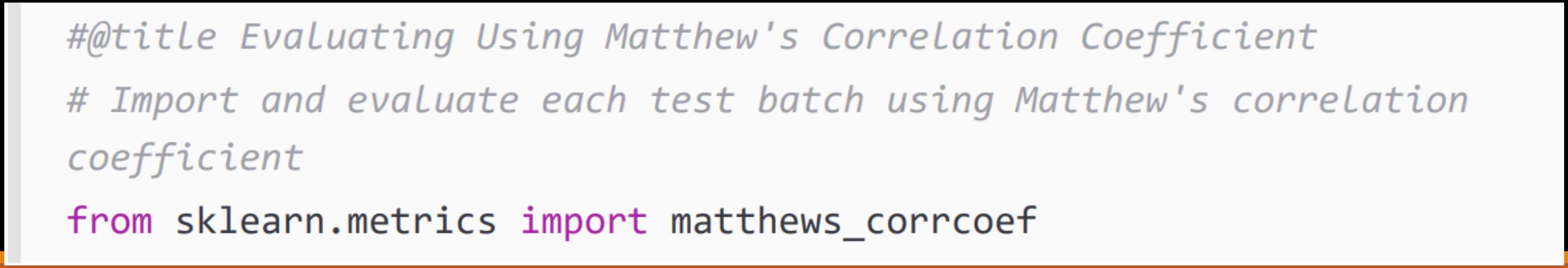
The value produced by MCC is between -1 and +1. +1 is the maximum positive value of a prediction. -1 is an inverse prediction. 0 is an average random prediction.

Matthews Correlation Coefficient (MCC)

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

The value produced by MCC is between -1 and +1. +1 is the maximum positive value of a prediction. -1 is an inverse prediction. 0 is an average random prediction.

To Use, MCC is imported from sklearn.metrics:



```
#@title Evaluating Using Matthew's Correlation Coefficient
# Import and evaluate each test batch using Matthew's correlation
coefficient
from sklearn.metrics import matthews_corrcoef
```

Matthews Correlation Coefficient (MCC)

MCC computes a measurement with true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

$$P = \frac{T_p}{T_p + F_p}$$

The MCC can be summarized by the followin

$$R = \frac{T_p}{T_p + F_n}$$

The equation is:

$$F1\ score = 2 \times \frac{P \times R}{P + R}$$

Links to materials:

https://jalammar.github.io/illustrated-transformer/

Hands-On Large Language Models: Language Understanding and Generation: https://www.amazon.com/Hands-Large-Language-Models-Understanding/dp/1098150961

https://www.deeplearning.ai/short-courses/how-transformer-llms-work/?utm_campaign=handsonllm-launch&utm_medium=partner

QUESTIONS AND ANSWERS

what's next??

PRACTICE AND ATTEND OTHER SESSIONS ON

Model Development and Deployment

Attribution & Acknowledgment

While diligent effort has been made to attribute sources, some content and images within this presentation may be the property of others. This presentation is for educational use and does not claim ownership of all materials.

i.e.:

A few things in here might be someone else's. I am ONLY using them to teach, and I respect the original creators

The END

Day 4

DSN LEKKI-AJAH

BY ABEREJO HABEEBLAH O.

X: @HABEREJO

Next Class should be on LLM and it's advanced techniques and Model Deployment

This is a PLUS +

Takeaway:

Next page!

A Glossary of Essential AI/ML/DL/NLP Terminologies

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

General AI Terms are:

1. **Artificial Intelligence (AI):** The broad concept of creating machines that can perform tasks that usually require human intelligence. **i.e.**: Making computers smart like us. **Scenario**: Imagine a robot that can cook dinner, do laundry, and even hold a conversation.

2. **Agent:** An entity (software or hardware) that perceives its environment and acts to achieve a goal **i.e.**: Something that can see what's around it and make decisions to do something. **Scenario**: Think of a video game character. It sees the game world (environment) and you control it to jump, shoot, etc. (actions) to win the game (goal).

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

3. **Machine Learning (ML):** A subset of AI where machines learn from data without being explicitly programmed. **i.e.**: Teaching a computer to learn by showing it examples, like training a dog with treats. **Scenario**: A music app recommending songs you might like. It learns your taste based on what you've listened to before.

4. **Deep Learning (DL):** A subfield of ML using artificial neural networks with many layers to learn complex patterns. **i.e.**: A really complex way of teaching a computer, like how our brains learn with lots of interconnected parts. **Scenario**: Image recognition software that can tell the difference between a cat and a dog, even if they look similar.

5. **Natural Language Processing (NLP):** A branch of AI focused on enabling computers to understand, interpret, and generate human language. **i.e.**: Teaching computers to understand and talk like us. **Scenario**: Google Translate, which can translate text from one language to another.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

Machine Learning Terms:

1. **Supervised Learning:** Learning from labeled data (input-output pairs). **i.e.**: Teaching a computer with answer keys. **Scenario**: Training a program to predict house prices. You give it examples of houses with their sizes, locations, and prices (the "answer key").
2. **Unsupervised Learning:** Learning from unlabeled data, finding patterns. **i.e.**: Giving a computer a bunch of stuff and asking it to sort it into groups. **Scenario**: A website suggesting groups you might like to join based on your interests.
3. **Reinforcement Learning:** An agent learns by interacting with an environment and getting rewards or penalties. **i.e.**: Training a pet with positive and negative reinforcement. **Scenario**: An AI playing a video game. It gets points for winning and loses points for losing, learning to play better over time.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

4. **Training Data:** The data used to train a model. **i.e.**: The examples you show the computer to learn from. **Scenario**: All the photos of cats and dogs you use to train the image recognition software.

5. **Test Data:** Data used to check how well the trained model performs on new data. **i.e.**: Showing the computer pictures, it hasn't seen before to see if it learned correctly. **Scenario**: Using a different set of cat and dog photos to test the image recognition software.

6. **Feature:** A measurable characteristic. **i.e.**: A detail about something. **Scenario**: In predicting house prices, features are size, location, number of rooms, etc.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

7. **Model:** A mathematical representation of learned patterns. **i.e.**: The rules the computer learns. **Scenario**: The formula the computer uses to predict house prices.

8. **Algorithm:** The specific set of rules or instructions used for learning. Simple: The method the computer uses to learn. Scenario: Different ways of training a computer, like different study techniques.

9. **Overfitting:** A model that learns the training data too well (including noise) and performs poorly on test data. **i.e.**: Memorizing the answers to practice questions but failing the real test. **Scenario**: A model that can perfectly predict prices of houses it has seen, but it can't predict prices of houses it has not seen.

This is a PLUS +

A Glossary of Essential AI/ML/DL/NLP Terminologies

10. **Underfitting:** A model that is too simple to capture the patterns and performs poorly on both training and test data. **i.e.**: Not studying enough and failing the test. Scenario: A model that is too simple to predict house prices.

11. **Accuracy:** How often the model makes correct predictions. **i.e.**: How often the computer gets the answer right.

12. **Precision:** Out of all positive predictions, how many were actually positive? **i.e.**: Of all the things the computer said were cats, how many were actually cats?

13. **Recall:** Out of all actually positive cases, how many did the model correctly identify? **i.e.**: Of all the actual cats, how many did the computer correctly identify?

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

14. **Bias**: Systematic error in the model due to flawed assumptions. **i.e.**: A prejudice in the computer's thinking. Scenario: A model trained only on pictures of sunny days will be biased and might not recognize cloudy days.

15. **Variance**: Sensitivity of the model to fluctuations in the training data. **i.e.**: The computer getting confused by small changes in the examples. Scenario: A model that works perfectly on one set of data but fails on another very similar set of data.

16. **Hyperparameters**: Parameters set before training that control the learning process. **i.e.**: Settings you adjust before you start training. **Scenario**: Like adjusting the focus and brightness on a camera before taking a picture.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

Deep Learning Terms:

1. **Neural Network:** Interconnected layers of neurons that process information. **i.e.**: A network of connected "brains" working together.
2. **Neuron (Node):** A basic processing unit in a neural network. **i.e.**: A single "brain" in the network.
3. **Layer:** A group of neurons (input, hidden, output layers). **i.e.**: A level in the network.
4. **Activation Function:** Introduces non-linearity to neurons. **i.e.**: A way for the "brains" to make more complex decisions.
5. **Backpropagation:** Adjusting connection weights to minimize prediction errors. **i.e.**: A way for the network to learn from its mistakes.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

6. **Epoch:** One complete pass through the entire training dataset. **i.e.**: Showing the computer all the examples once.

7. **Batch Size:** Number of training examples processed at once. **i.e.**: How many examples the computer looks at a time.

8. **Loss Function:** Measures the model's prediction error. **i.e.**: How the computer knows it made a mistake.

9. **Weight**: The strength of the connection between two neurons. **i.e.**: How strongly two "brains" influence each other.

10. **Learning Rate:** How quickly the model updates its weights. **i.e.**: How quickly the computer learns.

This is a PLUS +

A Glossary of Essential AI/ML/DL/NLP Terminologies

Natural Language Processing (NLP) Terms:

1. **Tokenization:** Breaking down text into tokens (words, phrases). **i.e.**: Splitting a sentence into words.
2. **Stop Words:** Common words (e.g., "the," "a," "is") often removed. **i.e.**: Words that don't carry much meaning.
3. **Stemming:** Reducing words to their root form. **i.e.**: Chopping off the endings of words.
4. **Lemmatization:** Reducing words to their dictionary form (lemma). **i.e.**: Finding the base word.

6. **Corpus:** A large collection of text. **i.e.**: A big library of text.

A Glossary of Essential AI/ML/DL/NLP Terminologies

This is a PLUS +

6. **Word Embeddings:** Representing words as vectors capturing semantic relationships. **i.e.**: Giving each word a set of coordinates in "meaning-space".
7. **Sentiment Analysis:** Determining the emotional tone of text. **i.e.**: Figuring out if someone is happy or sad based on what they wrote.
8. **Machine Translation:** Translating text from one language to another.
9. **Text Summarization:** Generating a concise summary of text.
10. **Named Entity Recognition (NER):** Identifying and classifying named entities. **i.e.**: Finding and labeling names of people, places, and organizations.

The END Day 4

DSN LEKKI-AJAH

BY ABEREJO HABEEBLAH O.

X: @HABEREJO

Next Class should be on LLM and it's advanced techniques and Model Deployment