

AI Bootcamp

---

# Tokenizers and Large Language Models

Module 21 Day 1



# Class Objectives

By the end of class, you will be able to:

---

1 Analyze the development of NLP to the present.

2 Interpret the principles of converting words to arrays or vectors.

3 Explain the how LLMs work.

4 Describe how tokenizers process sentences into tokens and numerical values.

5 Explain the benefits of Hugging Face tokenizers.

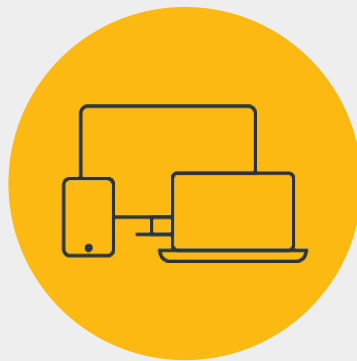
6 Convert sentences into tokens and numerical values.

7 Explain similarity measures and why they are important.

8 Assess the similarity between sentences.



Welcome



# Instructor **Demonstration**

A Brief History of NLP

# 1950s–1970s: Symbolic and Rule-Based NLP

The 1950s, 60s, and 70s were characterized by symbolic and rule-based NLP tasks.

1950s

Alan Turing introduces the “Imitation Game” as a measure of a machine’s ability to exhibit intelligent behavior indistinguishable from that of a human.

- His publication “Computing Machinery and Intelligence” caused a lot of interest in whether computers were able to exhibit intelligent behavior or not.

1960s

ELIZA, an early chatbot developed by the MIT Artificial Intelligence Laboratory, was released in 1966.

- ELIZA was an early example of the Turing Test. It used pattern-matching and substitution to respond to users’ questions or input as a therapist.
- ELIZA was able to make users feel as though they were conversing with a therapist.
- NLP programs such as ELIZA, which were predominantly rule-based, were not scalable.

1970s

Rule-based systems dominate early NLP efforts.

# 1990s: Statistical NLP and Machine Learning

In the 1990s, computational power increased significantly and the advent of ML made it possible for NLP tasks to be performed with statistical rather than rule-based methods.

1990s

Statistical NLP models still use rules to perform tasks but they use statistical methods to learn the rules instead of needing the rules to be coded manually by developers.

Other statistical methods that are now familiar to us in this course also emerged during this time, such as n-grams and hidden Markov models used to recognize speech.

With statistical NLP algorithms, analyzing and understanding human language has become easier.

- POS and NER algorithms were able to make predictions on unseen data.

In 1998, the TF-IDF approach to weighting word frequency was introduced, extending the statistical capabilities of the basic bag-of-words approach from 1957.

During this time, machine learning techniques, such as support vector machines and decision trees, gained traction in NLP, enabling data-driven approaches to language processing.

# 2000s–2010s: Deep Learning and Neural Networks

Deep learning and neural networks emerged, revolutionizing NLP. RNNs and CNNs came on the scene, allowing for significant advances in machine translation and sentiment analysis, amongst other NLP tasks.

The widely used RNNs and CNNs have numerous disadvantages.

- High computational expenses
- A need for large amounts of training data, and slow computation
- Training RNNs can be difficult, especially over long sequences or sentences, leading to the “exploding or vanishing gradient problem”.

2000s

In 2007, the LSTM networks emerged as a solution to the problem of long-term dependencies.

- LSTM networks solved the vanishing and exploding gradient problem of RNNs.
- However, they were more complicated than traditional RNNs and required more training data to learn effectively.
- They are not well-suited for online learning tasks such as prediction or classification tasks where input data is not a sequence.

# 2000s–2010s: Deep Learning and Neural Networks

Deep learning and neural networks emerged, revolutionizing NLP. RNNs and CNNs came on the scene, allowing for significant advances in machine translation and sentiment analysis, amongst other NLP tasks.

2010s

In 2014, a turning point in the development of NLP models was the introduction of **attention**, which allows the model to develop an internal representation of how each word is related to the next.

In 2017, the transformer model, introduced in the paper “Attention is All You Need” by Vaswani et al., adopted attention mechanisms as a core component.

- Since this paper, transformers have become the backbone of many state-of-the-art NLP models, showcasing the enduring impact of attention mechanisms on the field.



# 2010s–2020s: Modern LLMs and GPT

A transformative period marked by the advent of Large Language Models, capable of generating human-like text and performing a wide range of language tasks.

In 2018, OpenAI introduced the Generative Pre-trained Transformer (GPT) models, starting with GPT-1.

In 2020, GPT-3 further pushed the boundaries with 175 billion parameters. It performed tasks that it wasn't specifically trained for.

In December 2023, Google introduced Gemini, an evolution of Bard, with new innovations in contextual understanding and efficiency.

2020s

In 2019, GPT-2 made headlines for its ability to generate coherent and contextually relevant text at a scale never seen before.

- With 1,5 billion parameters, it showcased the power of scaling up model size for training.

In March 2023, Google launched Bard, a next-generation LLM designed to focus on generating creative and contextually aware content, adding a new dimension to the capabilities of LLMs.



# Instructor **Demonstration**

Introduction to Large Language Models

# Bag-of-Words

The “bag” keeps a tally of how many times a word occurs in the text, developing a “vocabulary”.

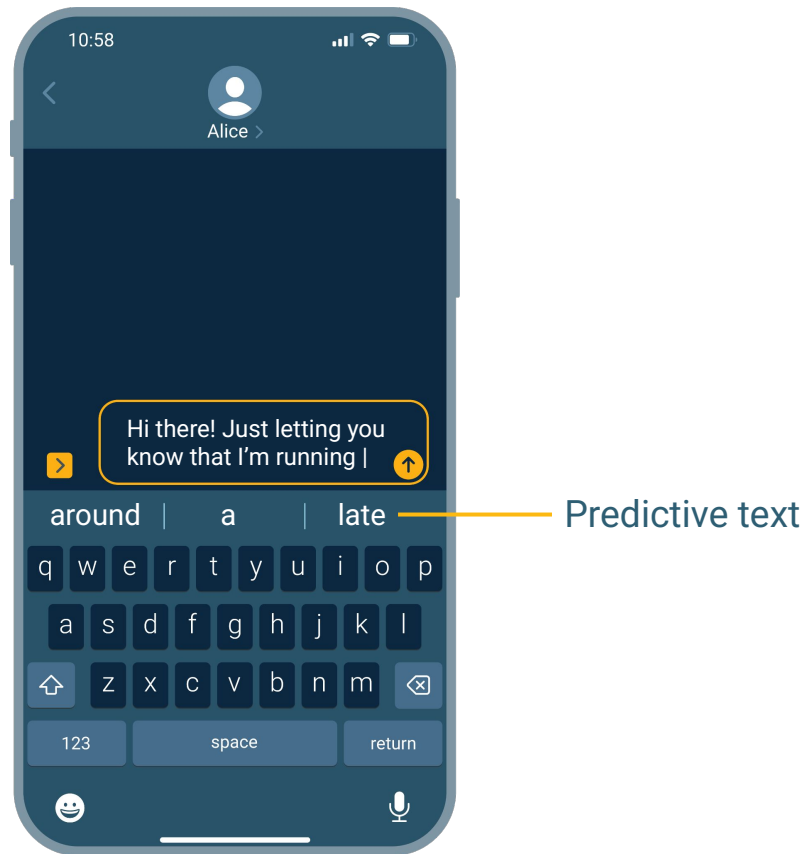
I love this movie! It's sweet but with satirical humor. The dialogue is great and the adventure scenes are fun...It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



It	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1

# NLP Applications

Predictive text



# Language models

Overview

01

Preprocessing

02

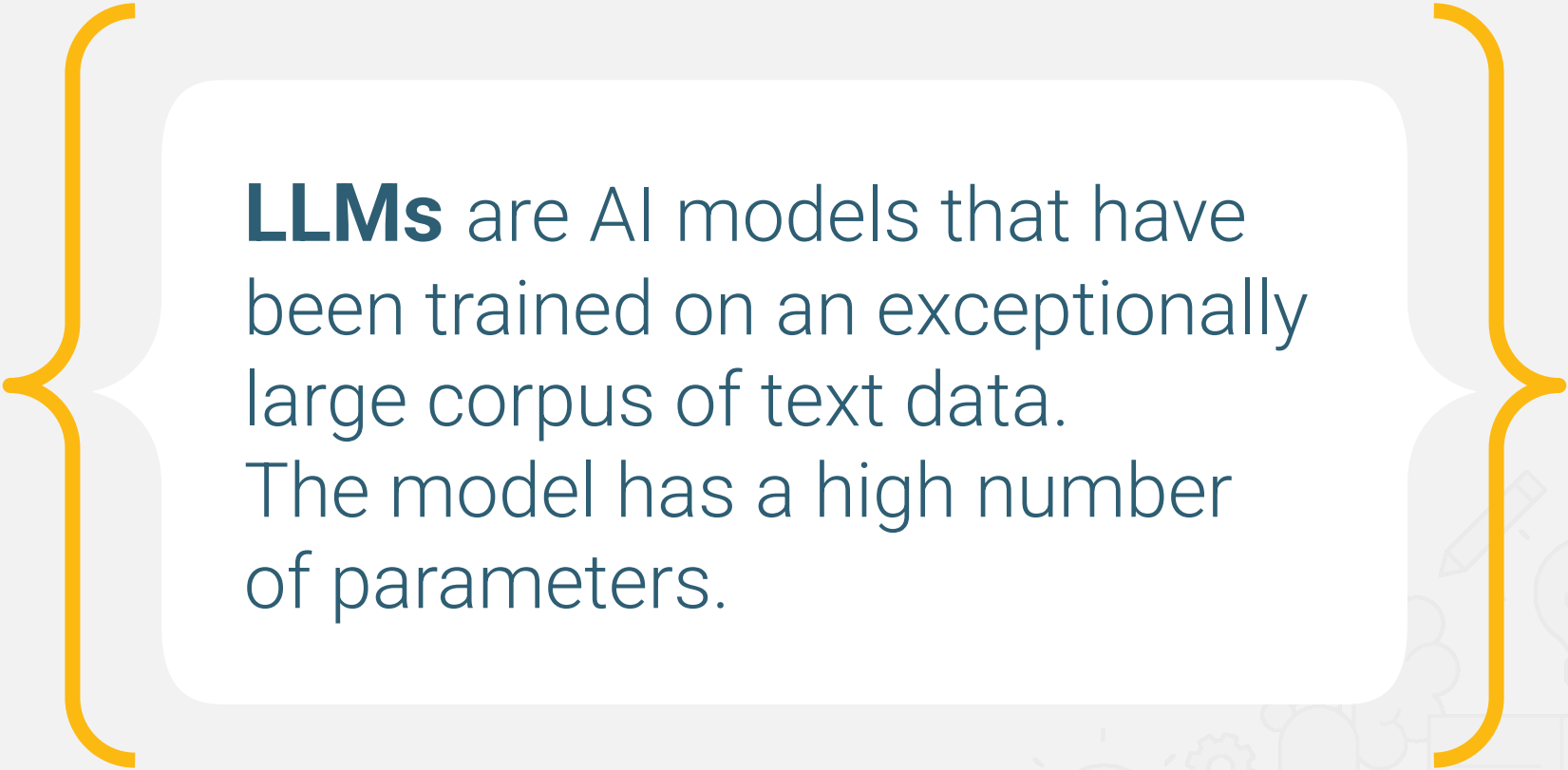
Input data

03


Fitting and  
evaluation

04

Optimization



**LLMs** are AI models that have been trained on an exceptionally large corpus of text data. The model has a high number of parameters.





# Activity:

## LLM Tokenizers

---

In this activity, you'll be code along with the instructor to understand how pre-trained LLMs perform subword tokenization.

**Suggested Time:**  
15 Minutes





**Time's up!**  
Let's review





# Questions?





# Activity:

## Index Encoding Review

---

In this activity, you'll code along with the instructor to review index encoding uses the Keras tokenizer as a prelude to using Hugging Face tokenizers in the next activity.

**Suggested Time:**

10 Minutes





**Time's up!**  
Let's review



**Questions?**





**Break**

15 mins



# Instructor **Demonstration**

Hugging Face Tokenizers and Similarity Measures

# Embeddings

Embeddings are vector representations of text sequences that capture their meaning.

**A list of sentences to tokenize.**

```
["I love my dog.", "I love my family.", "My dog is a lab"]
```

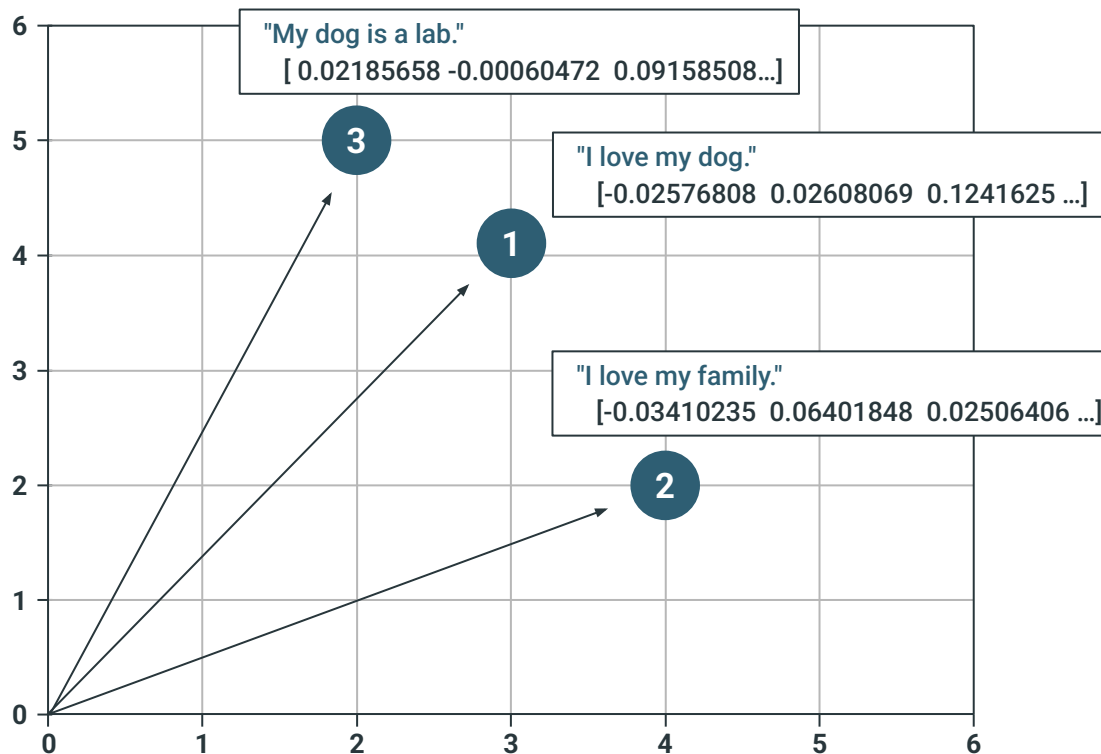


**After tokenization we get an array of numbers, where each number corresponds to a word in the vocabulary.**

```
[[2, 3, 1, 4], [2, 3, 1, 5], [1, 4, 6, 7, 8]]
```

# Similarity measures

The similarity values are a measure of how close in geometric space the vectors are to each other.







## Activity:

### Similarity Search Engine

---

In this activity, you will use a pre-trained sentence-transformer LLM to determine the most likely category for a given headline by comparing similarity measures between a random news headline and known headlines.

**Suggested Time:**

20 Minutes





**Time's up!**  
Let's review



# Questions?





# Activity:

## SMS Text Classification

---

In this activity, you will work in pairs to encode SMS text messages. This activity aims to simulate the process of classifying unlabelled SMS texts by comparing them with labeled ones. After you encode the text messages, you'll identify the top 5 similar classified SMS text messages for each unclassified SMS text message.

**Suggested Time:**

25 Minutes



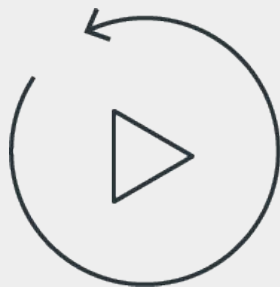


**Time's up!**  
Let's review



**Questions?**





Let's **recap**



# Recap

After today's lesson you are able to:

---

1 Analyze the development of NLP to the present.

2 Interpret the principles of converting words to arrays or vectors.

3 Explain the how LLMs work.

4 Describe how tokenizers process sentences into tokens and numerical values.

5 Explain the benefits of Hugging Face tokenizers.

6 Convert sentences into tokens and numerical values.

7 Explain similarity measures and why they are important.

8 Assess the similarity between sentences.





# Next

In the next lesson, you will explore pre-trained transformers models and how to use the pre-trained models for specific NLP tasks, such as language translation, text generation, question and answer, and text summarization.



**Questions?**





## Question 1:

Consider the following sentence:

'I walked along the riverbank to ruminate upon a solution.'

Now look at each of the tokenizer outputs for this sentence below. For each output, identify what kind of tokenizer was used:

- Word tokenization
- Character tokenization
- Subword tokenization

1 ['I', 'walked', 'along', 'the', 'riverbank', 'to', 'ruminate', 'upon', 'a', 'solution.']

2 ['I', 'w', 'a', 'l', 'k', 'e', 'd', 'a', 'l', 'o', 'n', 'g', 't', 'h', 'e', 'r', 'i', 'v', 'e', 'r', 'b', 'a', 'n', 'k', 't', 'o', 'r', 'u', 'm', 'i', 'n', 'a', 't', 'e', 'u', 'p', 'o', 'n', 'a', 's', 'o', 'l', 'u', 't', 'i', 'o', 'n', '.']

3 ['i', 'walked', 'along', 'the', 'river', '##bank', 'to', 'rum', '##inate', 'upon', 'a', 'solution', '.']



# Question 1: Answers

Consider the following sentence:

'I walked along the riverbank to ruminate upon a solution.'

1 ['I', 'walked', 'along', 'the', 'riverbank', 'to', 'ruminate', 'upon', 'a', 'solution.']

Word tokenization

2 ['I', 'w', 'a', 'l', 'k', 'e', 'd', 'a', 'l', 'o', 'n', 'g', 't', 'h', 'e', 'r', 'i', 'v', 'e', 'r', 'b', 'a', 'n', 'k', 't', 'o', 'r', 'u', 'm', 'i', 'n', 'a', 't', 'e', 'u', 'p', 'o', 'n', 'a', 's', 'o', 'l', 'u', 't', 'i', 'o', 'n', '.']

Character tokenization

3 ['i', 'walked', 'along', 'the', 'river', '##bank', 'to', 'rum', '##inate', 'upon', 'a', 'solution', '.']

Subword tokenization



## Question 2:

---

Which two NLP challenges are addressed using tokenization?

- 1 NLP models require large training datasets before they perform well consistently.
- 2 NLP problems are classified as AI-hard problems.
- 3 Natural language is translated into equivalent numerical representations that computers can interpret and models can use.
- 4 Variable text lengths can be made into fixed-length vectors that make up sparse matrices.



## Question 2: Answers

Which two NLP challenges are addressed using tokenization?

- 1

NLP models require large training datasets before they perform well consistently.

Incorrect
- 2

NLP problems are classified as AI-hard problems.

Incorrect
- 3

Natural language is translated into equivalent numerical representations that computers can interpret and models can use.

Correct
- 4

Variable text lengths can be made into fixed-length vectors that make up sparse matrices.

Correct



## Question 3:

Imagine that you've used the similarity search model in this lesson's demonstration on four new sentences. The similarity scores comparing each of the four sentences to the search query in order are:

0.5678, 0.6383, 0.1256, and 0.8760.

Which sentence is least relevant to the search query?

1

Sentence 1

2

Sentence 2

3

Sentence 3

4

Sentence 4



## Question 3: Answers

Imagine that you've used the similarity search model in this lesson's demonstration on four new sentences. The similarity scores comparing each of the four sentences to the search query in order are:

0.5678, 0.6383, 0.1256, and 0.8760.

Which sentence is least relevant to the search query?

1

Sentence 1

Incorrect

2

Sentence 2

Incorrect

3

Sentence 3

**Correct**

4

Sentence 4

Incorrect





**The End**