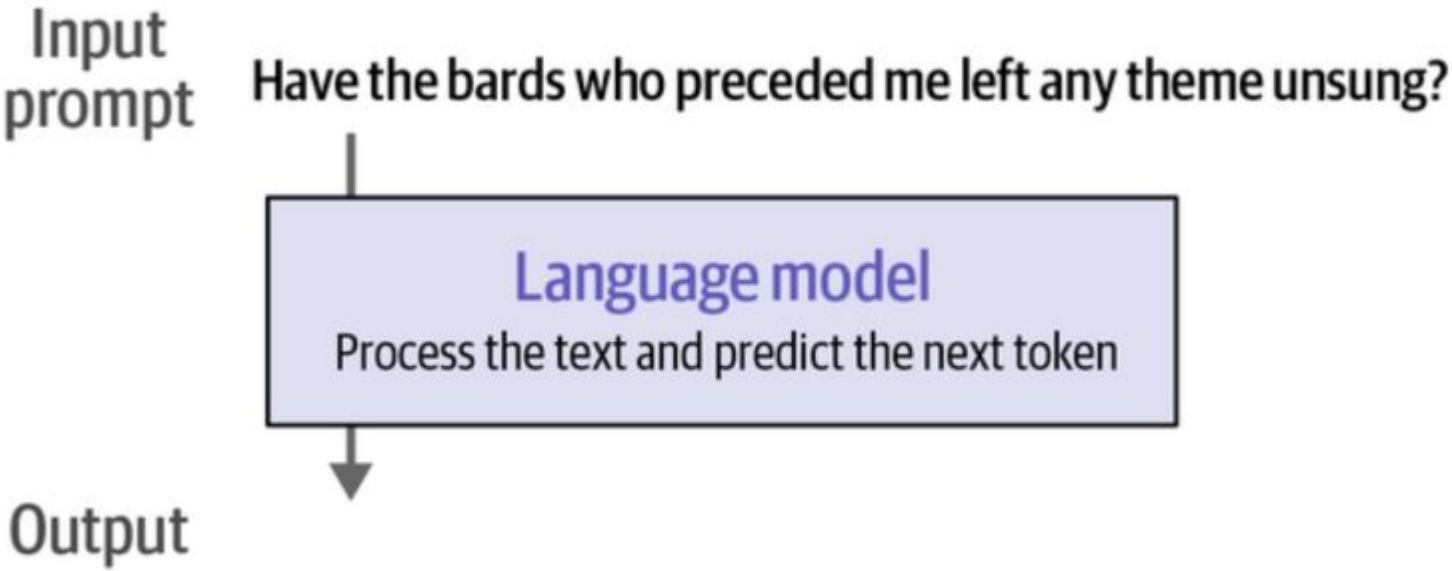
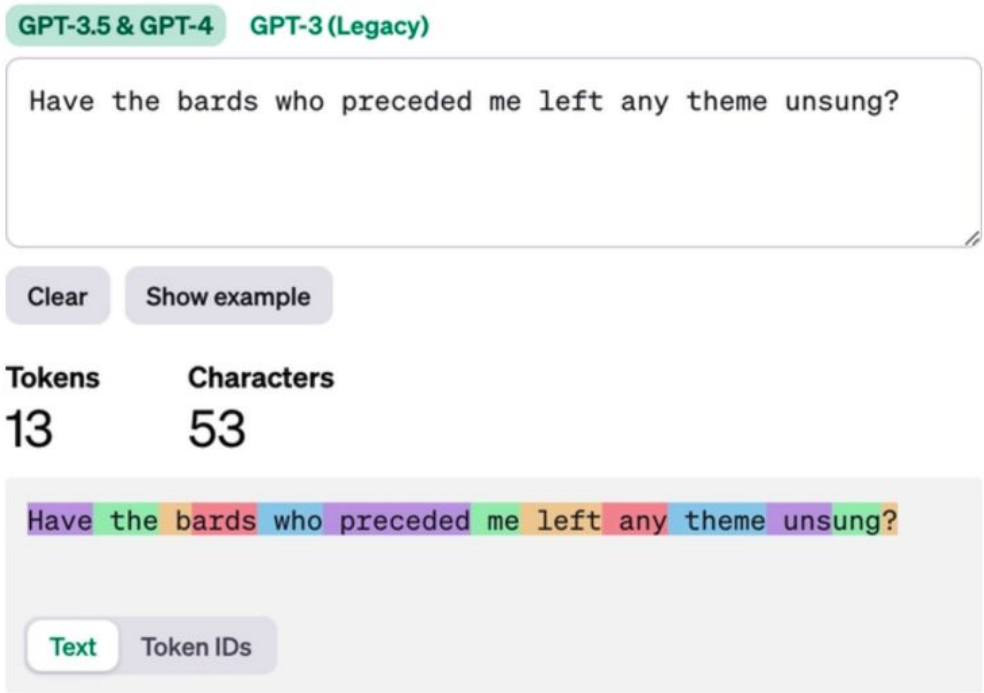
Everything about Tokenization

High Level Working of an LLM



https://platform.openai.com/tokenizer

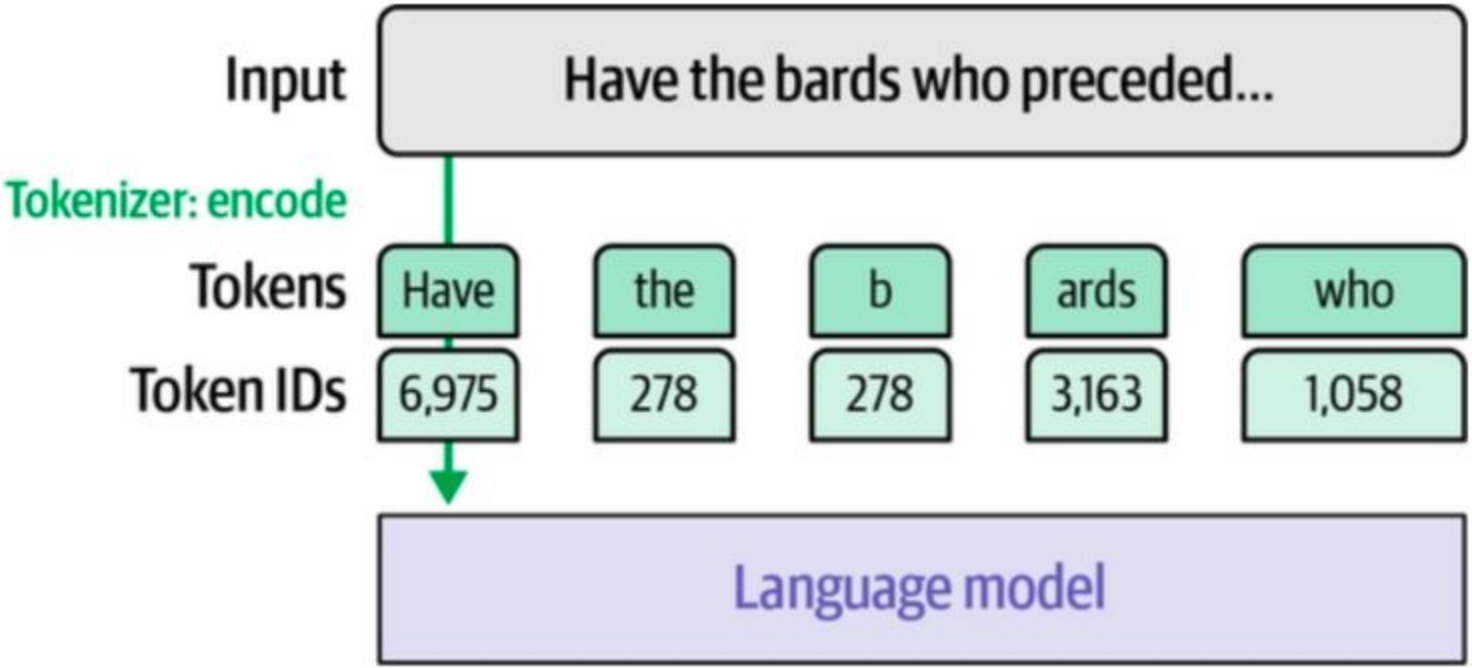


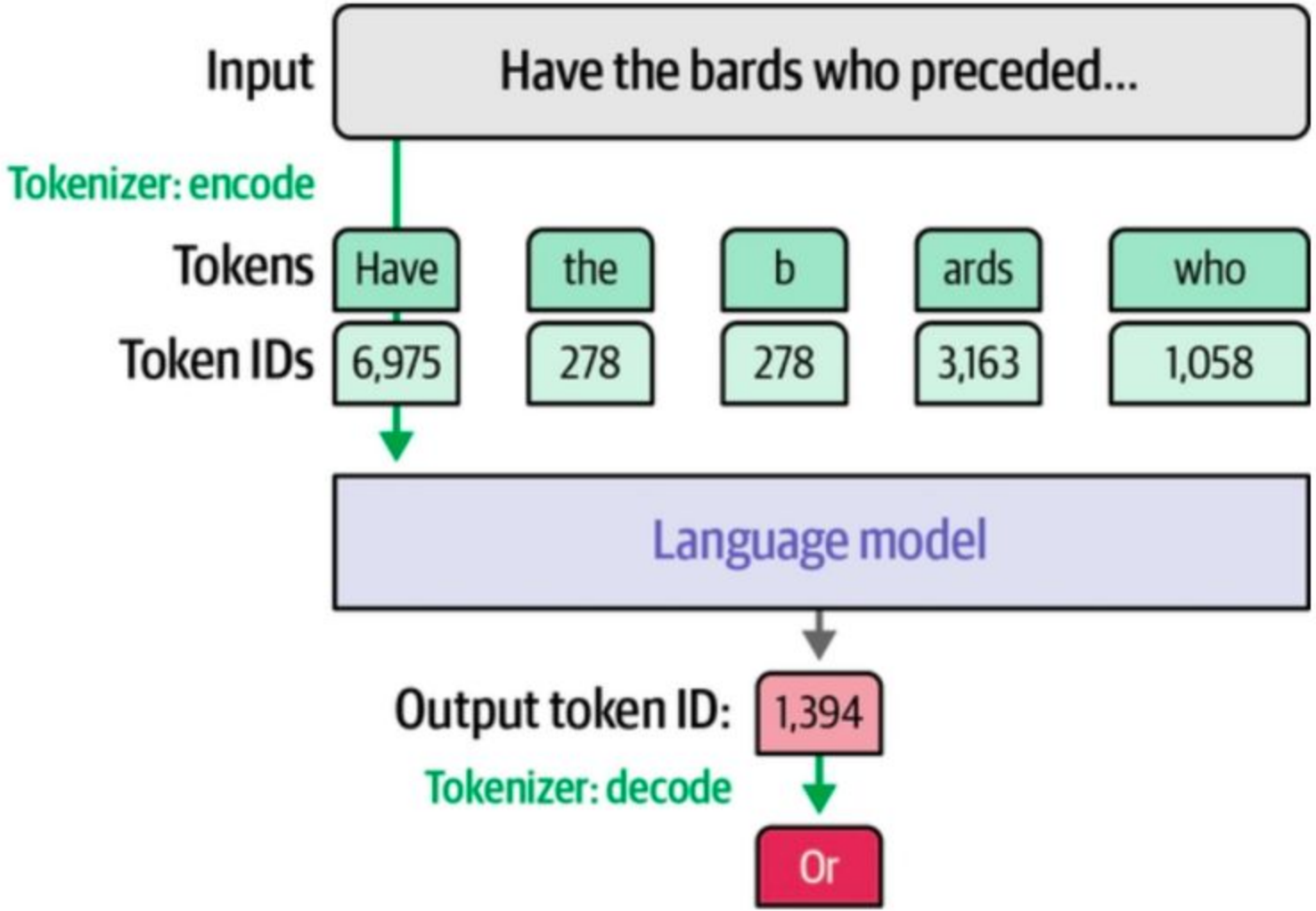
Downloading and Loading an LLM

```python
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load model and tokenizer
model = AutoModelForCausalLM.from_pretrained(
  "microsoft/Phi-3-mini-4k-instruct",
  device_map="cuda",
  torch_dtype="auto",
  trust_remote_code=True,
)
tokenizer = AutoTokenizer.from_pretrained("microsoft/Phi-3-mini-4k-instruct")

prompt = "Write an email apologizing to Sarah for the tragic gardening mishap. Explain how it happened.<|
assistant|>"

# Tokenize the input prompt
input_ids = tokenizer(prompt, return_tensors="pt").input_ids.to("cuda")

# Generate the text
generation_output = model.generate(
 input_ids=input_ids,
 max_new_tokens=20
)

# Print the output
print(tokenizer.decode(generation_output[0]))
```

Output:

```
<s> Write an email apologizing to Sarah for the tragic gardening mishap. Explain how it happened.<|
assistant|> Subject: My Sincere Apologies for the Gardening Mishap

Dear
```


Input
Have the bards who preceded...
Tokenizer: encode
Tokens
Have
the
b
ards
who
Token IDs
6,975
278
278
3,163
1,058
Language model


Input
Have the bards who preceded...
Tokenizer: encode
Tokens
Have
the
b
ards
who
Token IDs
6,975
278
278
3,163
1,058
Language model
Output token ID:
1,394
Tokenizer: decode
Or

Word-level Tokenization

Splits text into words based on **spaces and punctuation.**

Example:

Text: "I love machine learning!"

Tokens: ["I", "love", "machine", "learning", "!"]

Subword-level Tokenization

Breaks text into smaller units like **prefixes, suffixes, or common subwords** using algorithms like **BPE or WordPiece.**

Text: "unhappiness"

Tokens: ["un", "happi", "ness"]

Character-level Tokenization

Splits text into **individual characters.**

Example:

Text: "hello"

Tokens: ["h", "e", "l", "l", "o"]

Byte-level Tokenization

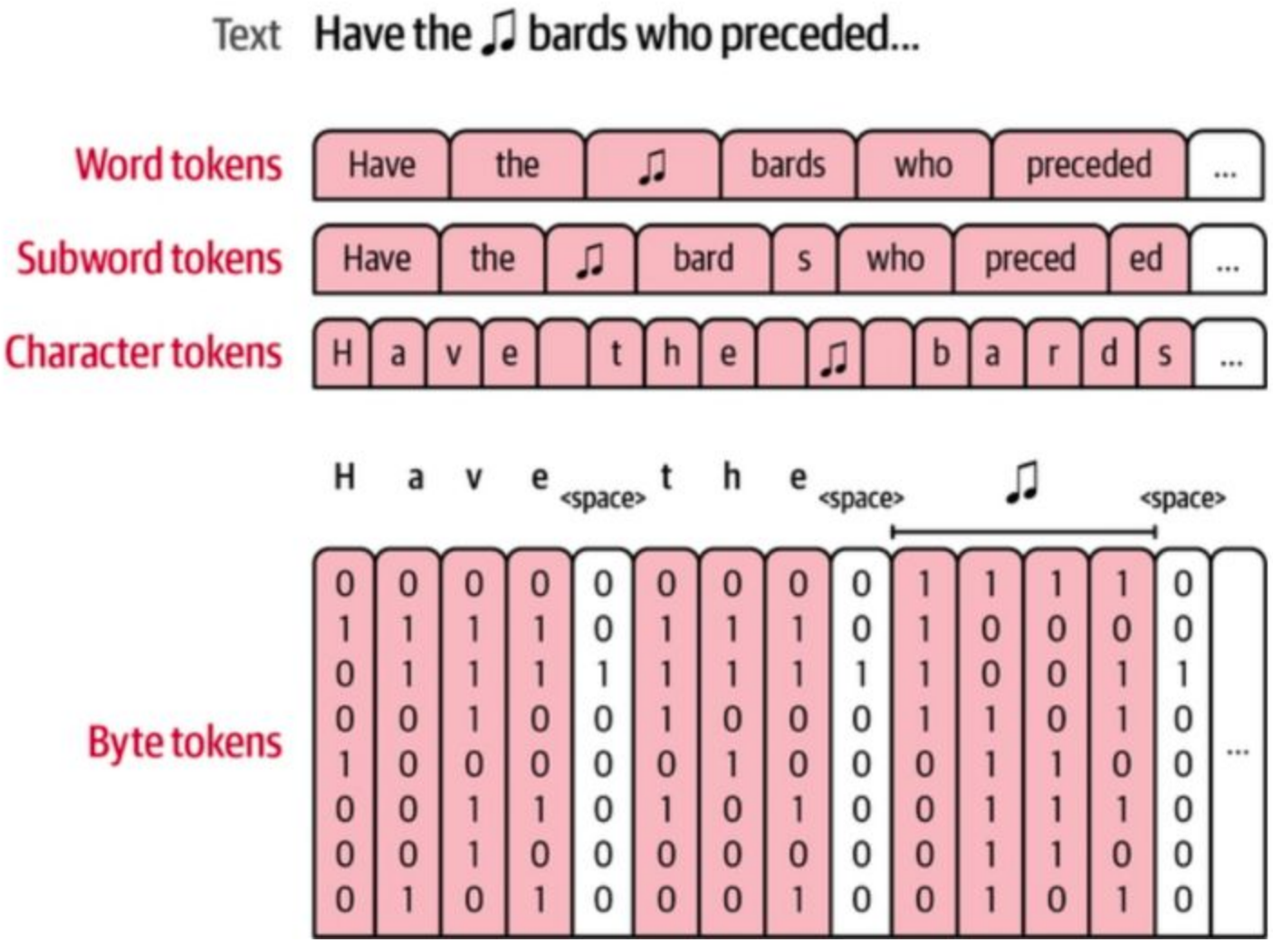
Converts text into **raw bytes** (e.g., UTF-8 encoded), often used in models like GPT-2.

Example (UTF-8 bytes):

Text: "Hi!"

Bytes: [72, 105, 33]

Tokens: ["H", "i", "!"] (via byte-to-char mapping)


Text
Have the ♫ bards who preceded...
Word tokens
Have
the
♫
bards
who
preceded
...
Subword tokens
Have
the
♫
bard
s
who
preced
ed
...
Character tokens
H a v e t h e ♫ b a r d s ...
Byte tokens
H a v e <space> t h e <space> ♫ <space>
...

BERT base model (uncased)	[CLS] english and capital ##ization [UNK] [UNK] show _ token ##s false none eli ##f = = > = else : two tab ##s : " " three tab ##s : " " 12 . 0 * 50 = 600 [SEP]
BERT base model (cased)	[CLS] English and CA ##PI ##TA ##L ##I ##Z ##AT ##ION [UNK] [UNK] show _ token ## s F ##als ##e None el ##if = = > = else : two ta ##bs : " " Three ta ##bs : " " 12 . 0 * 50 = 600 [SEP]
GPT-2	English and CAP ITAL IZ ATION 🏳 🏳 🏳 🏳 🏳 🏳 show _ t ok ens False None el if == >= else : two tabs :" " Three tabs : " " 12 . 0 * 50 = 600
FLAN-T5	English and CA PI TAL IZ ATION <unk> <unk> show _ to ken s Fal s e None e l if = = > = e lse : two tab s : " " Three tab s : " " 12. 0 * 50 = 600 </s>
GPT-4	English and CAPITAL IZATION 🏳 🏳 🏳 🏳 🏳 🏳 show _tokens False None elif == >= else : two tabs :" " Three tabs : " " 12 . 0 * 50 = 600
StarCoder	English and CAPITAL IZATION 🏳 🏳 🏳 🏳 🏳 show _ tokens False None elif == >= else : two tabs :" " Three tabs : " " 1 2 . 0 * 5 0 = 6 0 0
Galactica	English and CAP ITAL IZATION 🏳 🏳 🏳 🏳 🏳 🏳 🏳