

# Get ready, we'll start at 12:00

- Have your computer ready for exercises!
- GitHub repo with URLs and other resources:
  - <https://github.com/nuitrcs/CoDEx-Choose-Your-LLM>
- Log in to your Hugging Face account (optional)
- Have ready a pen and paper

# Choosing an LLM for Your Project

efrén cruz cortés  
Emilio Lehoucq

# This workshop is brought to you by:

## Northwestern IT Research Computing and Data Services

### Need help?

- AI, Machine Learning, Data Science
- Statistics
- Visualization
- Data Collection, Cleaning, Analysis, Management ...
- Scraping, Text Analysis, Computing, Reproducibility ...
- R, Python, SQL, MATLAB, Stata, SPSS, SAS, etc.

Request a **FREE** consultation at [bit.ly/rcdsconsult](https://bit.ly/rcdsconsult).

# AI Impact

- AI has an **environmental** impact. 🌱
- AI can rely on **precarious labor conditions**.
- AI can exhibit and reinforce **biases**.

# Outline



Language Tasks



Where Do I Find LLMs?



The Transformer



Recap



# Language Tasks

# Exercise 1

(5 minutes)

- What is the specific task you want to use an LLM for?
- How could your research benefit from using an LLM?

# What is your task?

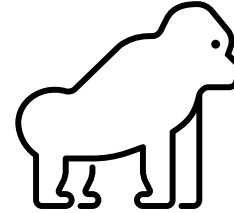
- The first step in choosing an appropriate LLM is to correctly identify the task you are working on.
- Let's look at some specific examples, before thinking of types of tasks in general.



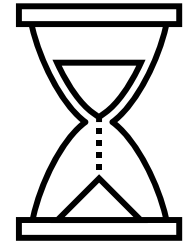
# Named Entity Recognition

Chimpanzees, gorillas, and  
orangutans have been living for  
hundreds of thousands of years  
in the forest, living fantastic lives.

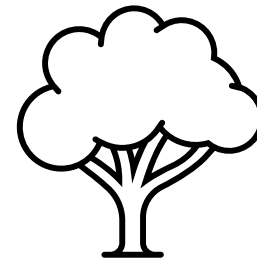
(Jane Goodall)



Great Apes

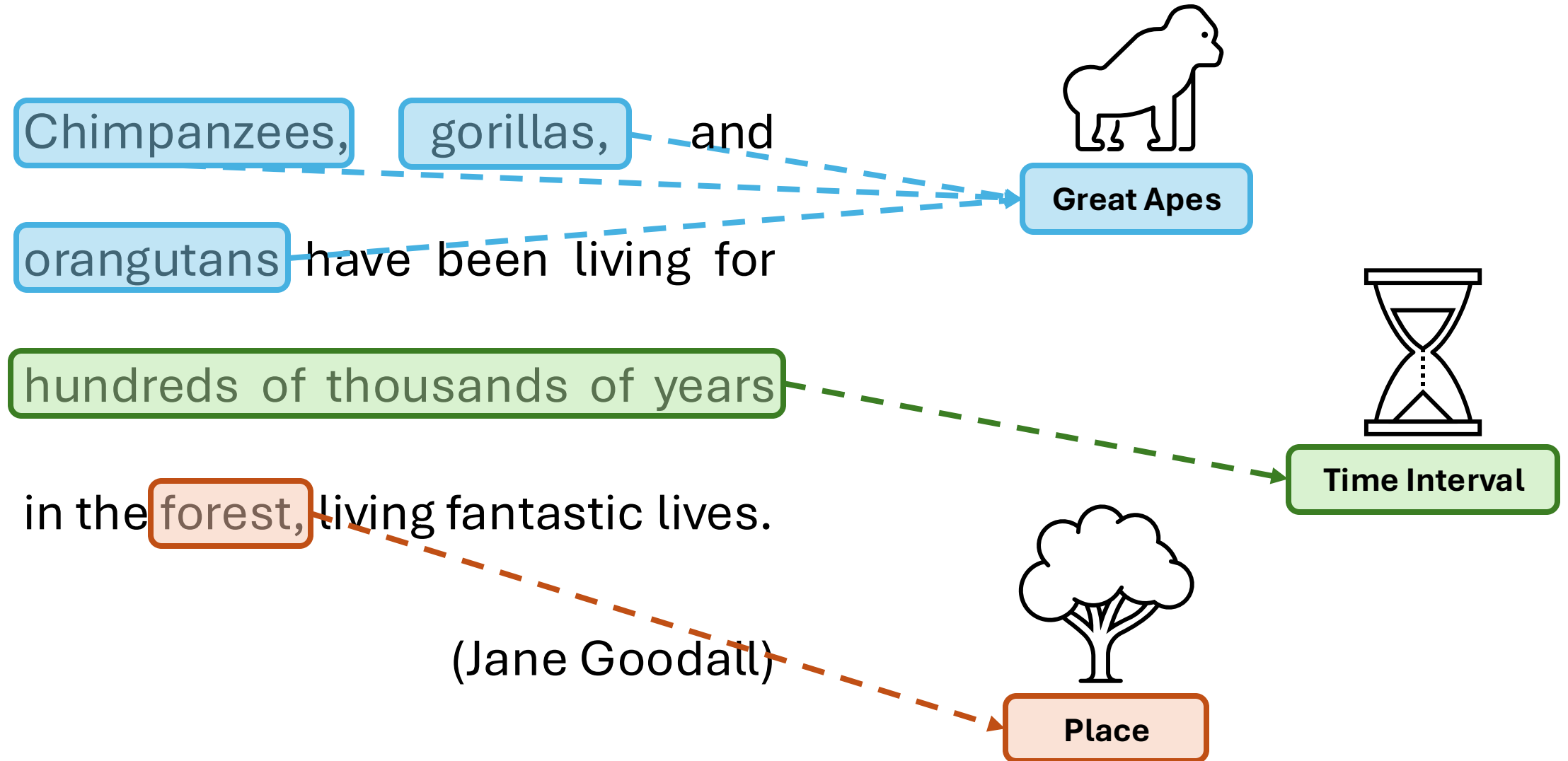


Time Interval



Place

# Named Entity Recognition



# Sentiment Analysis

Spent the day sharing stories with my grandparents :-)

Ran into an old friend today. We used to be so close; I miss that.

I'm worried about the rising rent. What if we can't afford it anymore?

Just got back from the street market—so much vibrant energy.

I'm scared we are losing the sense of community we once had.

Another historic building gone. Heartbreaking.

I love how we all come together for our Fall celebrations!

They're tearing down the park next month. Where will our kids play?



# Sentiment Analysis

Ran into an old friend today. We used to be so close; I miss that.

Just got back from the street market—so much vibrant energy.

I'm scared we are losing the sense of community we once had.

Another historic building gone. Heartbreaking.

Spent the day sharing stories with my grandparents :-)

I'm worried about the rising rent. What if we can't afford it anymore?

They're tearing down the park next month. Where will our kids play?

I love how we all come together for our Fall celebrations!



**Sadness**

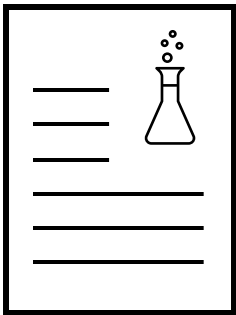
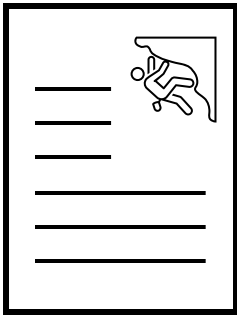
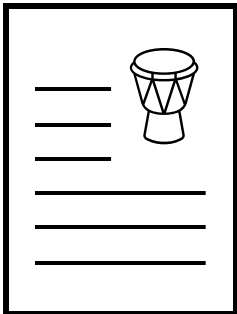
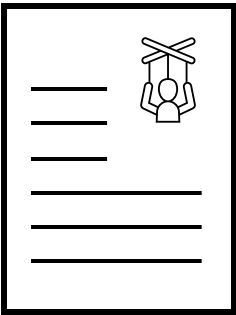
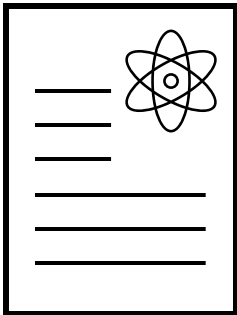
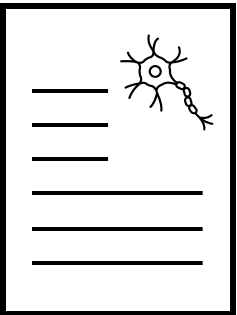
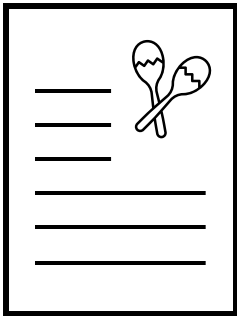
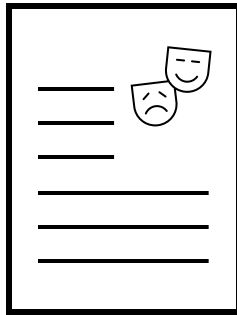
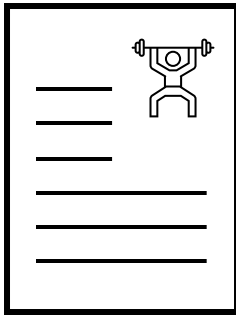
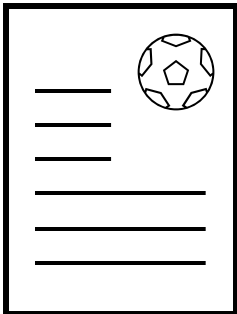
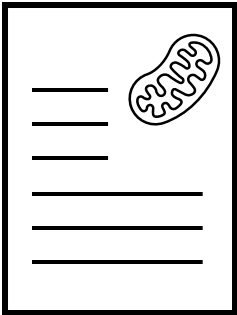
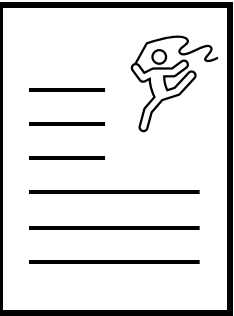


**Joy**

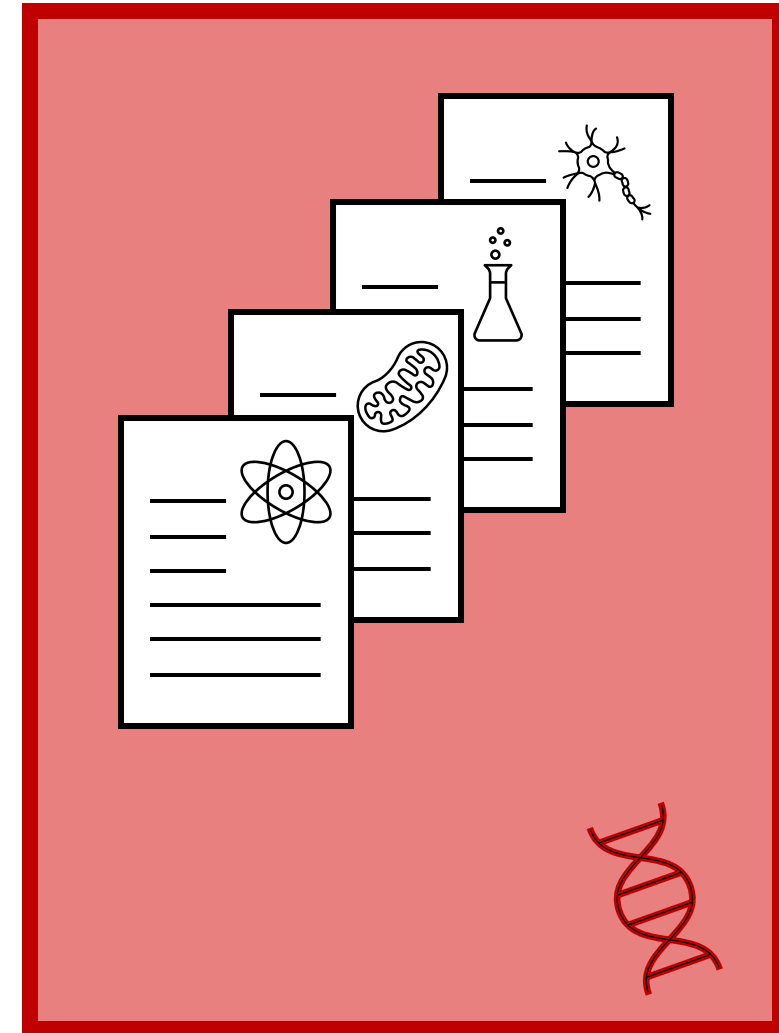
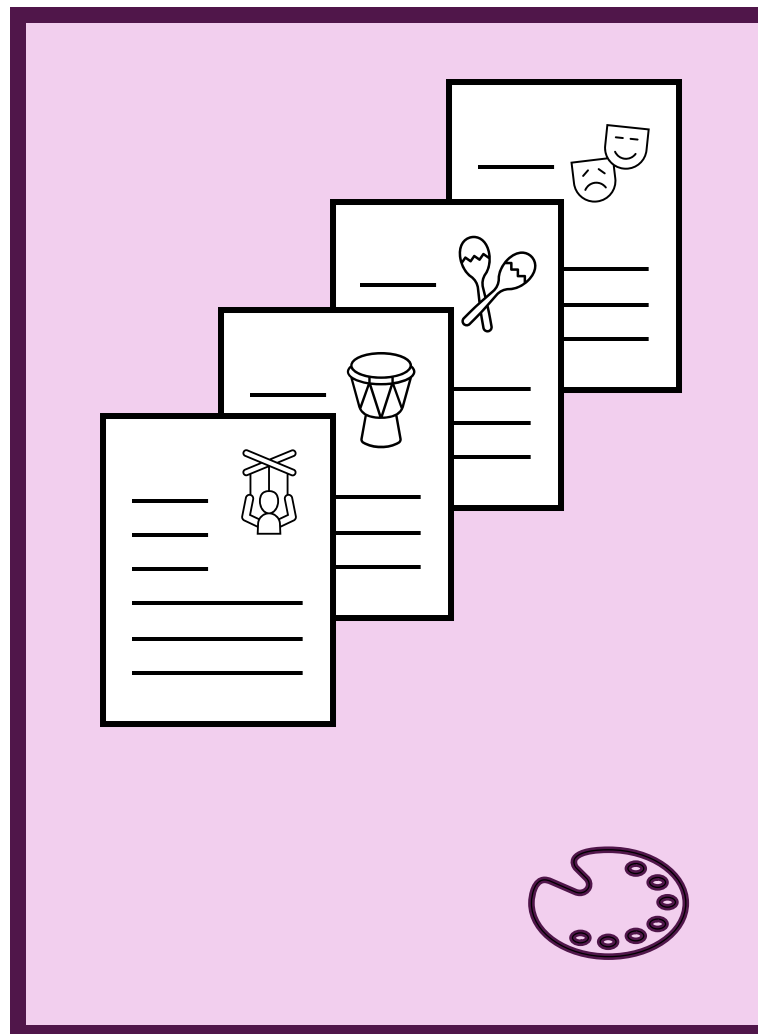
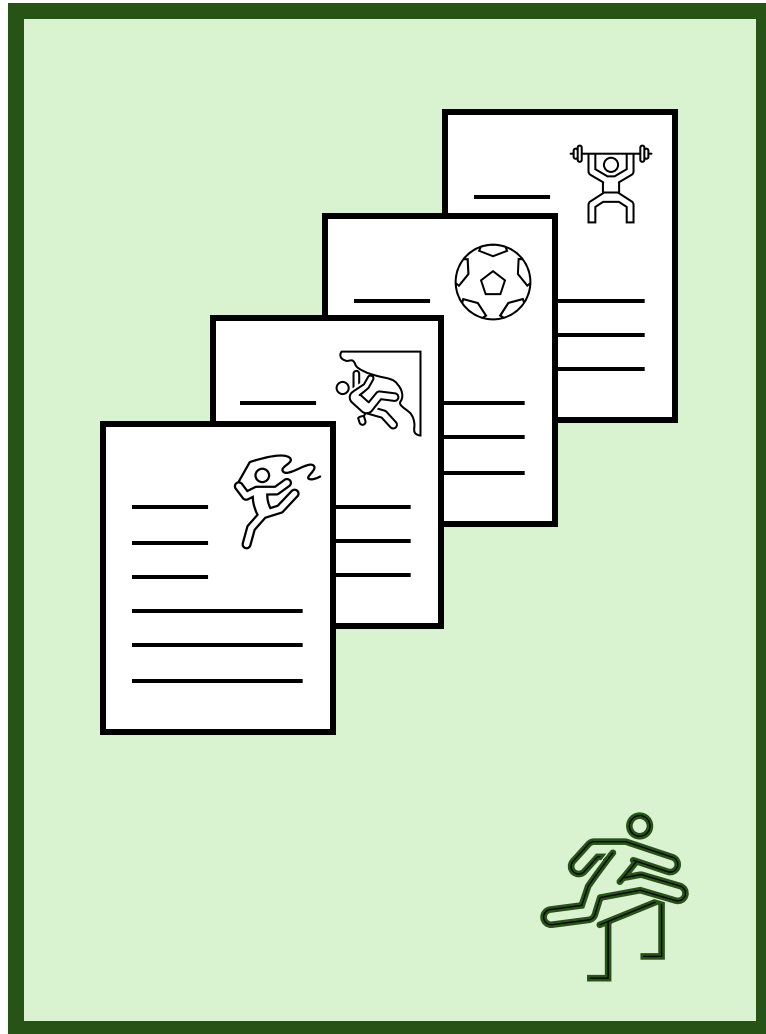


**Fear**

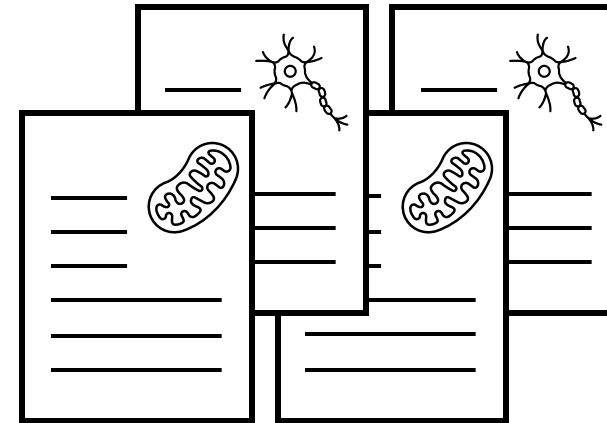
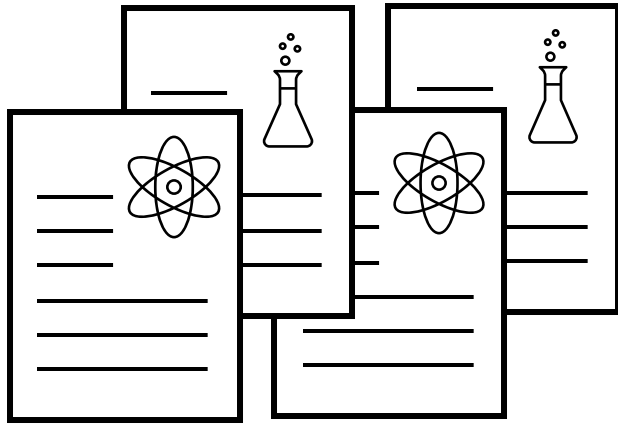
# Topic Modeling



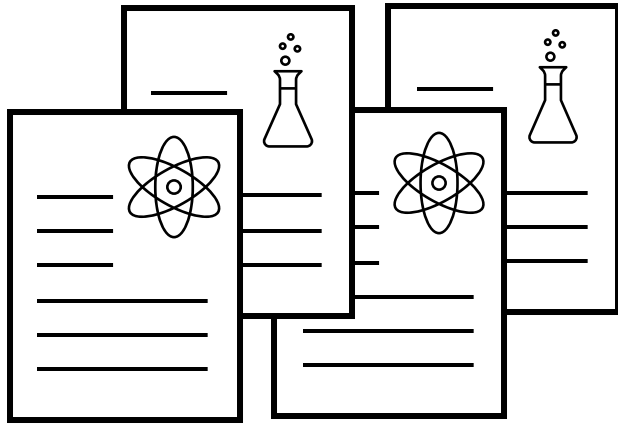
# Topic Modeling



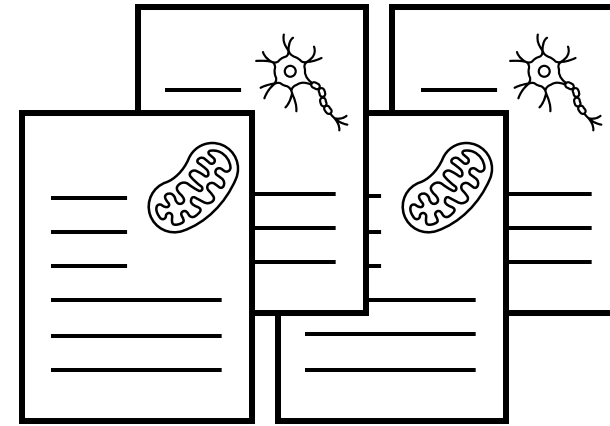
# Summarization



# Summarization



“In this series of studies, the authors have corroborated through experiments that the hypothesized law ...”



“Each of these papers studies a different intervention to alleviate the population’s challenge. Paper 1 shows that ...”



# Code Generation

I'm giving a presentation on generative AI. I need an example of how an LLM can generate code. Please provide a small snippet of code. You choose the content and the language. Be creative.

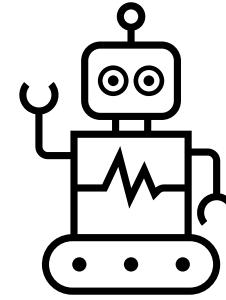
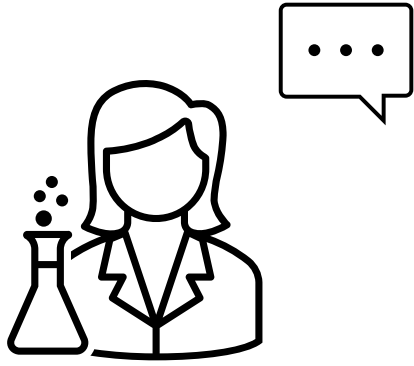
# Code Generation

I'm giving a presentation on generative AI. I need an example of how an LLM can generate code. Please provide a small snippet of code. You choose the content and the language. Be creative.

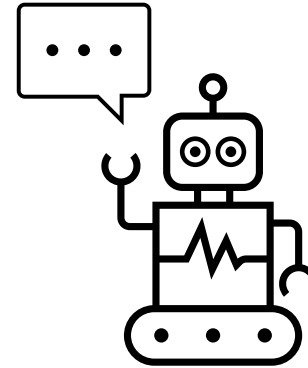
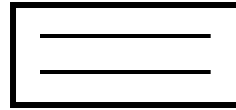
python

```
def reverse_cipher(text):  
    alphabet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'  
    reversed_alphabet = alphabet[::-1]  
    ciphered_text = ''  
  
    for char in text.upper():  
        if char.isalpha():  
            index = alphabet.index(char)  
            ciphered_text += reversed_alphabet[index]  
        else:  
            ciphered_text += char  
  
    return ciphered_text
```

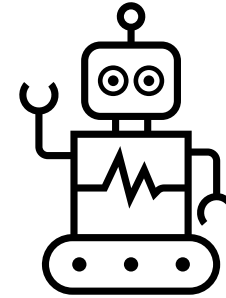
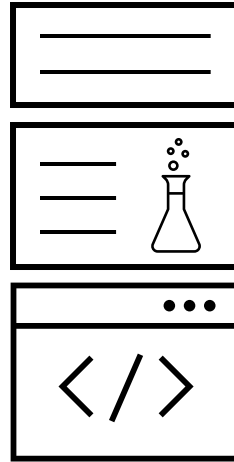
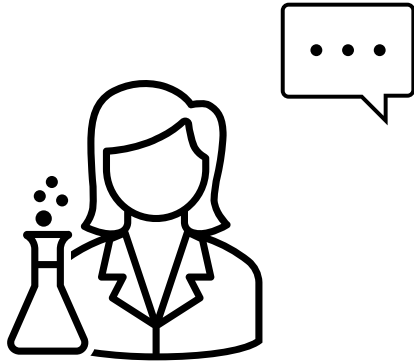
# Chatbots



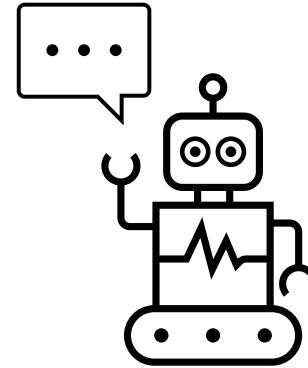
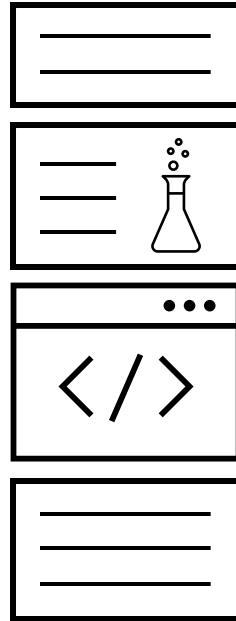
# Chatbots



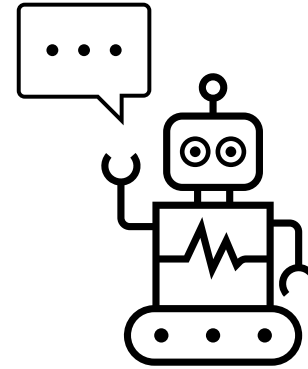
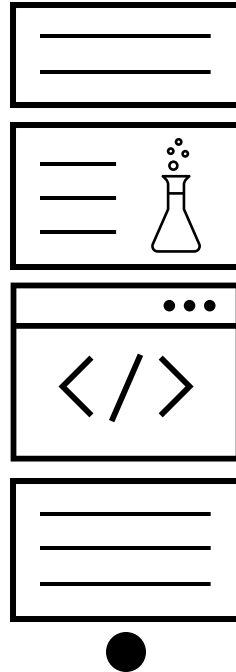
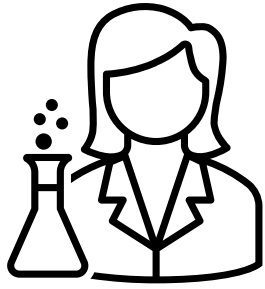
# Chatbots



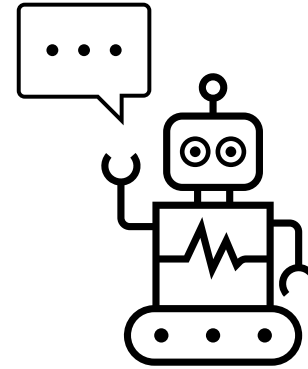
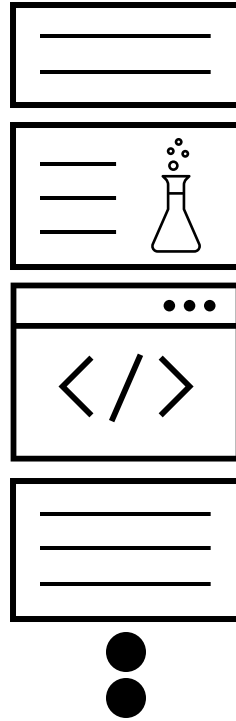
# Chatbots



# Chatbots

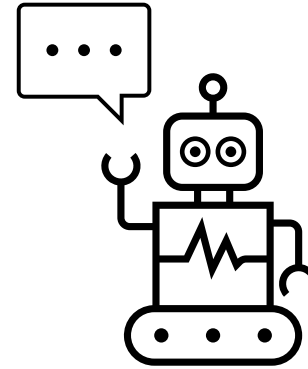
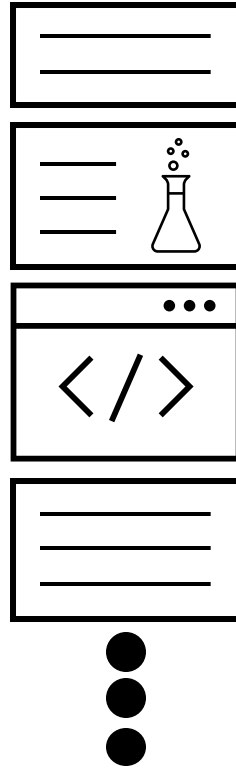
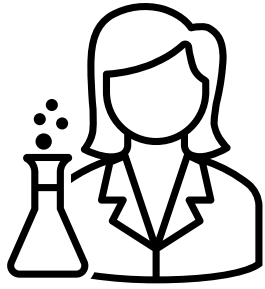


# Chatbots





# Chatbots



# Classifying Tasks

## Task types

Finding structure in the text, *as is*.

- Named entity recognition
- Sentiment Analysis
- Topic Modeling
- etc.

Generating *new text*.

- Summarization
- Code Generation
- Chatbots
- etc.

Now let's try some of these!

## Exercise 2

(6 minutes)

Now you know about some language tasks. It's time to play around and get a more concrete sense!

### **1. Named entity recognition.**

We mentioned named entity recognition. Try playing around with the Inference API Widget here: <https://huggingface.co/dslim/bert-base-NER>

### **2. Sentiment Analysis.**

We mentioned sentiment analysis. Try playing around with the Inference API Widget here: <https://huggingface.co/finiteautomata/bertweet-base-sentiment-analysis>



Where Do I Find LLMs?

## Exercise 3










(5 minute)

- What LLMs have you used?
- Where have you found them?

# Open-source vs Proprietary

- Most models can be downloaded and used locally.
- Some models can only be used through an online interface or an API.
  - GPT
  - Claude
  - Gemini

# How to access proprietary models?

Company and Models	Online	Python API
OpenAI - GPT 	chatgpt.com	<pre>from openai import OpenAI client = OpenAI()</pre> 
Microsoft - GPT 	copilot.cloud.microsoft (login with NU credentials)	N/A
Azure - GPT 	ai.azure.com (playground)	<pre>from openai import AzureOpenAI</pre> 
Anthropic - Claude 	claude.ai	<pre>import anthropic</pre> 
Google – Gemini 	gemini.google.com	<pre>import google.generativeai as genai</pre> 



# A Note on Microsoft Copilot, Azure, GPT

## Privacy:



- If you are logged in with your NU credentials, Copilot will respect your privacy, and the **privacy of your data**.
- This is not true in general for OpenAI.

## Azure:

- Azure is the name of Microsoft Cloud Services
- Your data is safe in Azure

<https://www.it.northwestern.edu/about/policies/guidance-on-the-use-of-generative-ai.html>

# How to access open source models

Company	Website	Notes
 <b>Hugging Face</b>	<a href="https://huggingface.co">https://huggingface.co</a>	<ul style="list-style-type: none"><li>• Large collection of models</li><li>• Large community</li><li>• Developed python library: transformers</li><li>• Not all models tested for safety!</li></ul>
 <b>Ollama</b>	<a href="https://ollama.com">https://ollama.com</a>	<ul style="list-style-type: none"><li>• Easier to set up</li><li>• Friendly Python library</li><li>• Fewer models</li><li>• Models generally trustworthy</li></ul>

# Security Considerations

- Apparently, some models in Hugging Face contained viruses 🤖
  - This is because anyone can upload to HF.
- Only download models from trusted developers / organizations!
- Pay attention to safe model formats:
  - *.safetensors*
  - *.GGUF*

# Navigating Hugging Face

Throughout this workshop, we are using many examples from Hugging Face. Let's get more familiar with the website:

<https://huggingface.co/>



# (Reference Slide)

## Partial checklist for choosing models on Hugging Face

- Look at models that are “liked” by the community
- Check the person/organization uploading the model
- Check the model format in “Files and version”



## Exercise 4

(5 minutes)

Earlier we gave you two model cards from Hugging Face for you to play with the Inference API Widgets. Now it's your turn to search on Hugging Face for a model that *could* be useful for your research! **Search for a model and check if it has an Inference API Widget, and if so, play around with it. (You can also use Google Colab.)**

The goal of this exercise is for you to start becoming familiar with how to search for models on Hugging Face. *You don't need to find the perfect model. It's also okay if you don't find a model with an Inference API Widget. Just play around with searching for models on Hugging Face!*

The image features a white background with several thick, dark purple diagonal stripes. These stripes are positioned in the top right and bottom right corners, extending from the edges towards the center. A solid, light purple horizontal band spans the width of the image, serving as a background for the title text.

# The Transformer

# How do we deal with text data?

Most machine learning algorithms only work with numerical data.  
So how do we deal with text?

- Convert text into numerical data (vectors).
- So as to **preserve** the original meaning of the text.



# “Embedding” Models

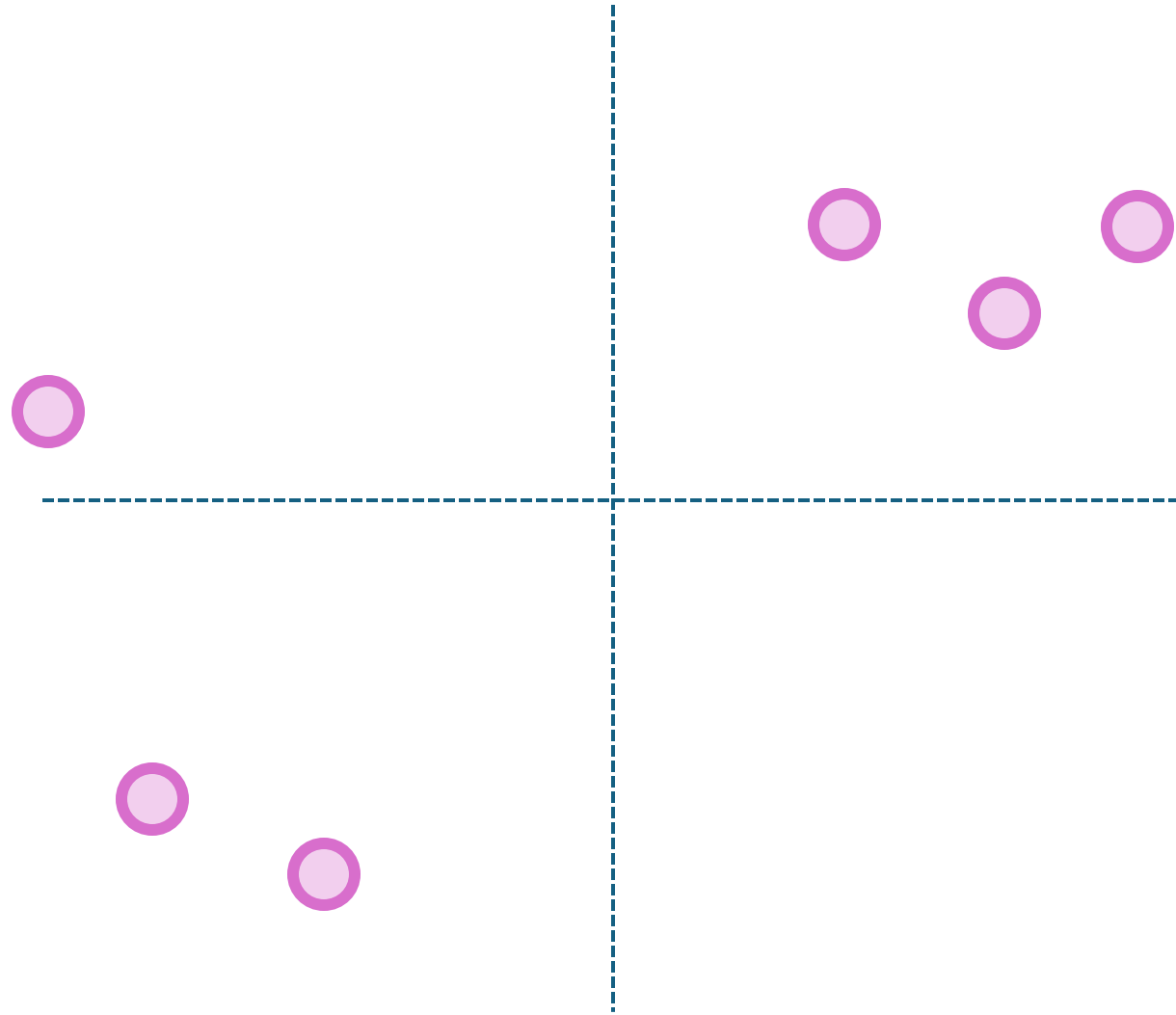
Chimpanzees, gorillas, and orangutans have been living for hundreds of thousands of years in the forest, living fantastic lives.

(Jane Goodall)

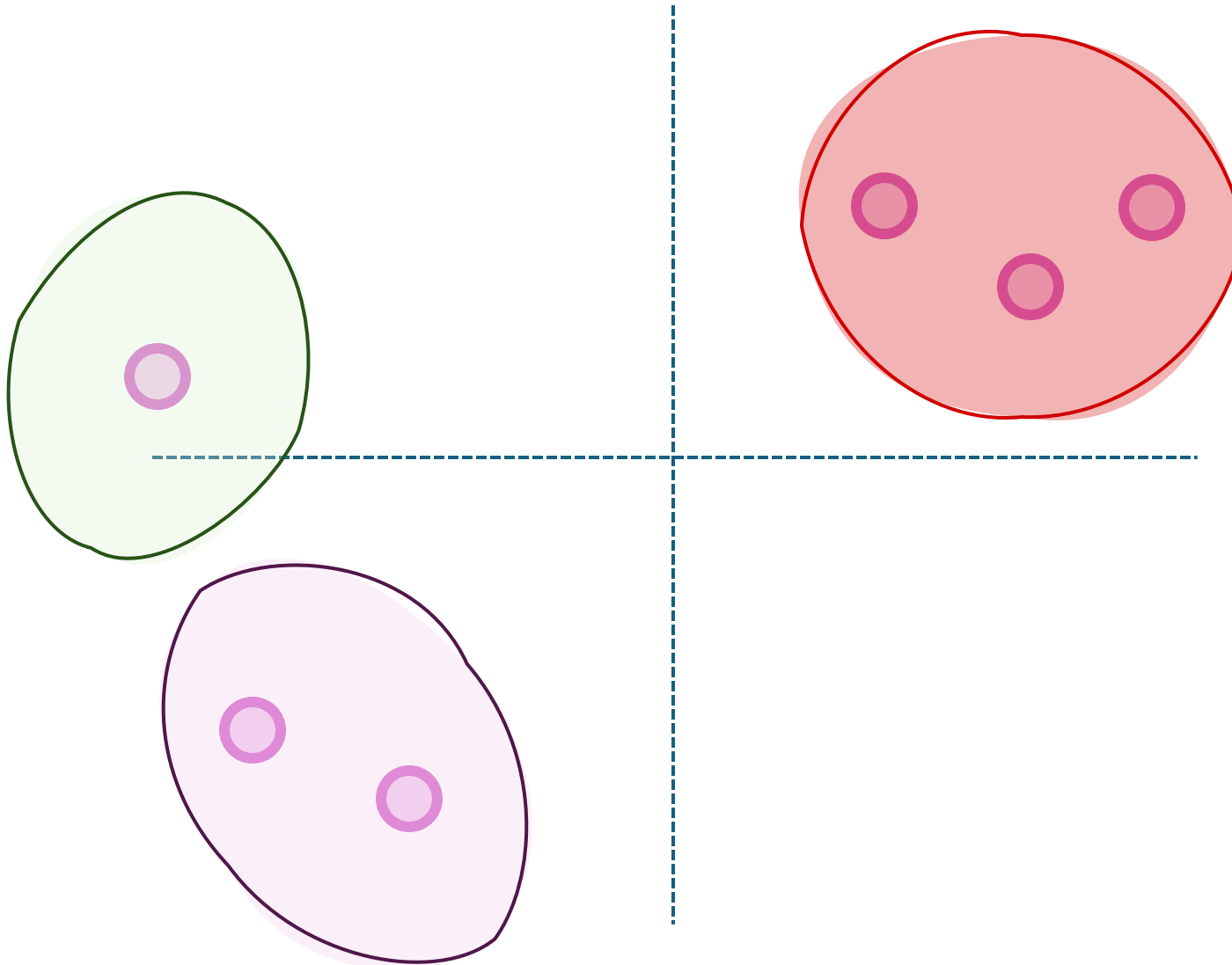
# “Embedding” Models



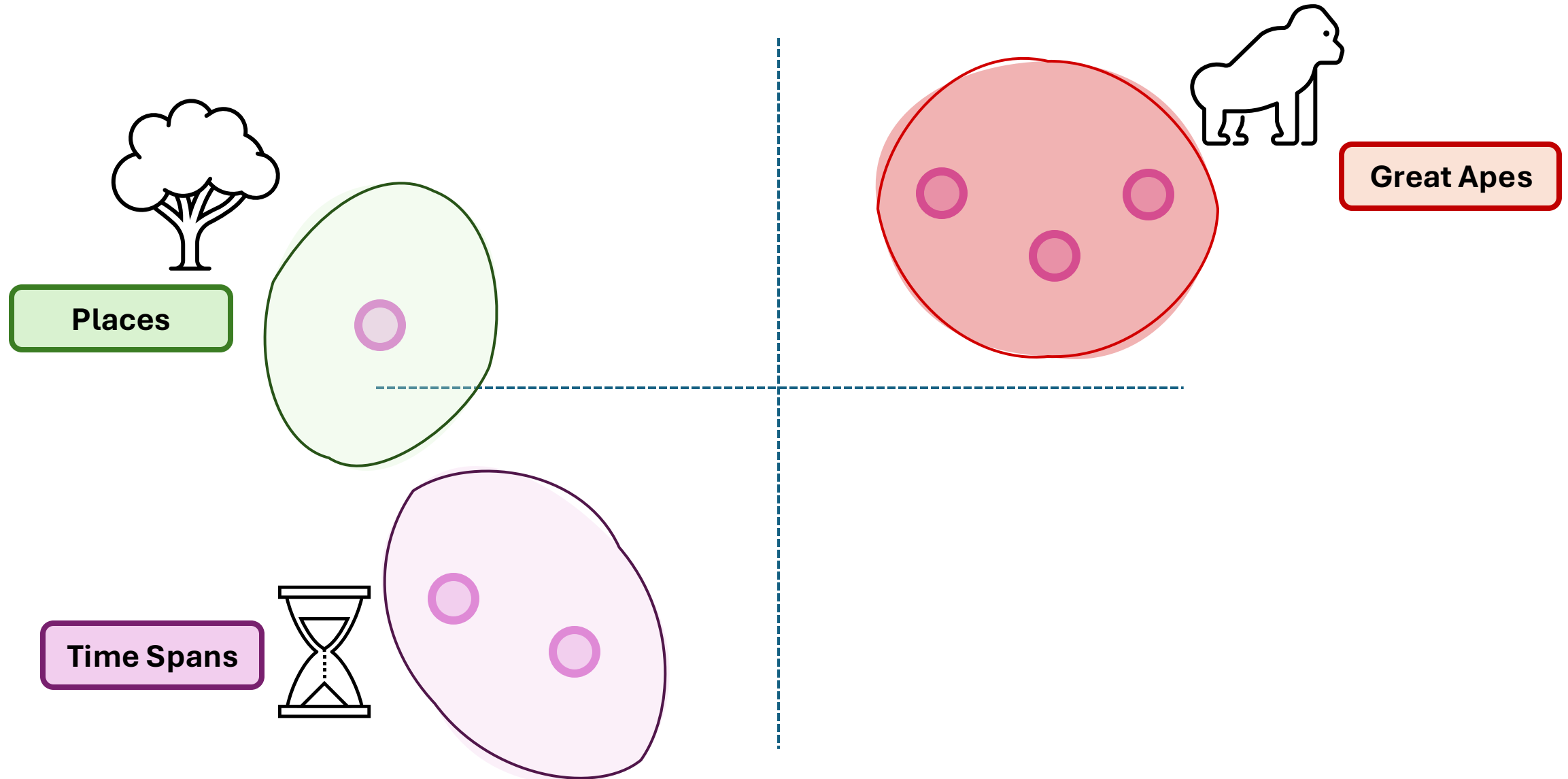
# “Embedding” Models



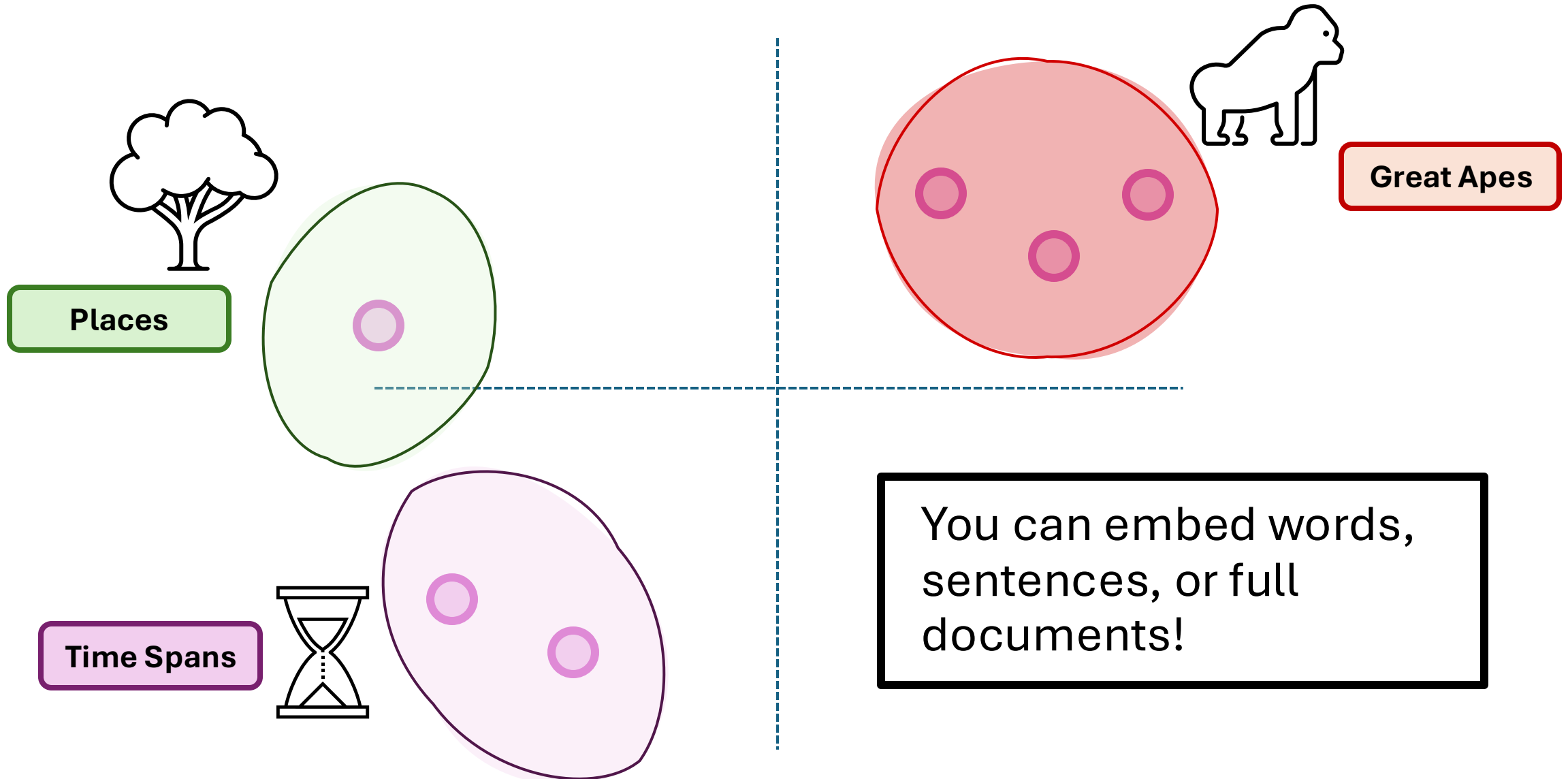
# “Embedding” Models



# “Embedding” Models



# “Embedding” Models

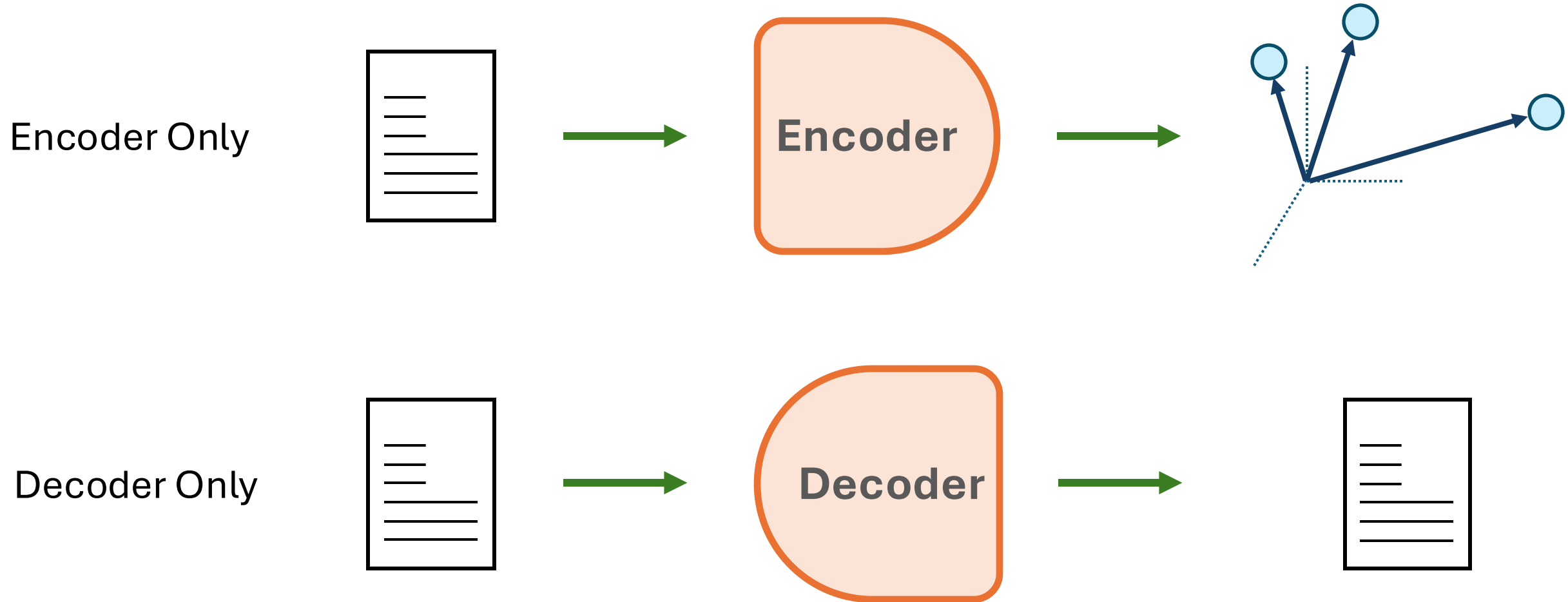


# The Transformer

The transformer architecture is the **key element** driving LLMs today.

- Context aware.
  - “The virus is bad for my computer”
  - “The virus is bad for my health”
- It’s divided in two parts:
  - Encoder → From text to vector representations.
  - Decoder → From text to text.

# The Transformer





# Common Transformer-based Models

## Decoder-Only

- GPT Series (OpenAI)
- Claude (Anthropic) \*
- Llama (Meta)
- Mistral/Mixtral (Mistral AI)
- OLMo (Allen Institute)
- BLOOM (Big Science / HF)
- Gemma (Google)

## Encoder-Only

- BERT
  - RoBERTa
- OpenAI text-embeddings

## Encoder-Decoder

- Gemini (Google) \*
- Phi-3 Vision (Microsoft)
- T5
- BART

\* I couldn't find official documentation but I'm pretty sure.

# Approximate correspondence

## Task types

Finding structure in the text, *as is*.

- Named entity recognition
- Sentiment Analysis
- Topic Modeling
- etc.

**Encoder**

Generating *new text*.

- Summarization
- Code Generation
- Chatbots
- etc.

**Decoder**

# Case Example 1

- You're investigating how narrative elements in educational materials influence responses by young audiences. For example, plot devices, literary styles, character traits, etc.
- Your LLM must create short educational stories varying across all these elements.

What type of model is best suited for this task?

# Case Example 2

- You are tracing the similarity of song lyrics across different musical genres over the years. Your dataset contains thousands of songs and spans several decades.
- Your similarity is a number between 0 and 1.

What type of model is best suited for this task?

## Exercise 5

(5 minutes)

Let's go back to two of the tasks that you're already familiar with: named entity recognition (NER) and sentiment analysis.

- Provide different texts to encoder-only and decoder-only models and see what they do and what you can get from them.
- For encoder-based NER, you can use the Inference API Widget here: <https://huggingface.co/dslim/bert-base-NER>
- For encoder-based sentiment analysis, you can use the Inference API Widget here: <https://huggingface.co/finiteautomata/bertweet-base-sentiment-analysis>
- For a decoder model, you can use ChatGPT or, better yet, you can try some of the models available here: <https://huggingface.co/chat/>



Recap

# Choosing your LLM

- What task are you performing?
- Do you need an encoder or decoder?
- What are your privacy and security constraints?
- Do you want an open-source or closed-source model?
- Do you want to use your local machine, Quest, or Azure?
- Was the LLM created following ethical guidelines?

# Bonus Exercise

(5 minutes)

Think back to exercise 1 and your original idea about what LLMs could do for your research.

- Does it fit in one of the tasks that we presented?
- Where are you planning to find the LLM?
- Are you thinking of using a decoder or an encoder (or an encoder-decoder)?



# Questions?

The image features a white background with several thick, dark purple diagonal stripes running from the top-left towards the bottom-right. These stripes are positioned in the upper and lower portions of the frame, creating a modern, geometric design. A solid, light purple horizontal band spans the width of the image, serving as a backdrop for the text.

Thank you!



# Appendix slides

# Are we done with encoders?

- Decoders can do things that encoders can't (e.g., generate new text).
- At first glance, decoders seem more versatile as they can "mimic" encoding tasks (e.g., topic modeling).
- Why use encoders? 🤖
  - Decoders are not reproducible, and don't have the same mathematical guarantees you have with encoder techniques.
  - Encoders can be more efficient since they are optimized for a particular task.
  - Encoders work directly with latent representations that you can use for tasks such as measuring similarity, while decoders always need to generate text, which may not be what you need.
  - Encoders can generate a probability distribution over a set of labels (e.g., sentiment classes), which you can use to estimate uncertainty. Decoders cannot generate probabilities for the whole text generated (only for each token).
  - Decoders can hallucinate.
- We need both!