

Important Notice

- Check privacy rules (apps may use your private data)
- Be careful: always check the results
- No liability for your use of the AI tools
- Field is rapidly evolving



Outline

- Prompt Engineering in Large Language Models (LLM)
- Using Large Language Models (LLM) to do AI
- An example of using AI in research
- Convenient platforms for using AI in research

Background



How do we train language models?

Update gradient

Original word : thank you for **inviting me to your party**

Loss

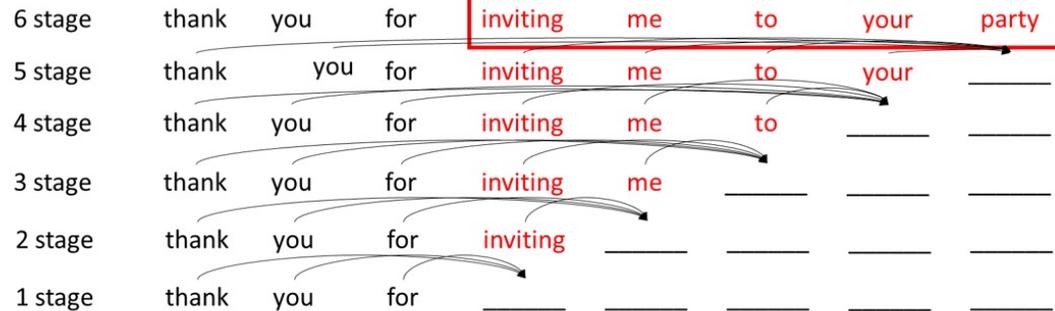


Image from stack exchange

Output

you has the highest probability

you,they, your..

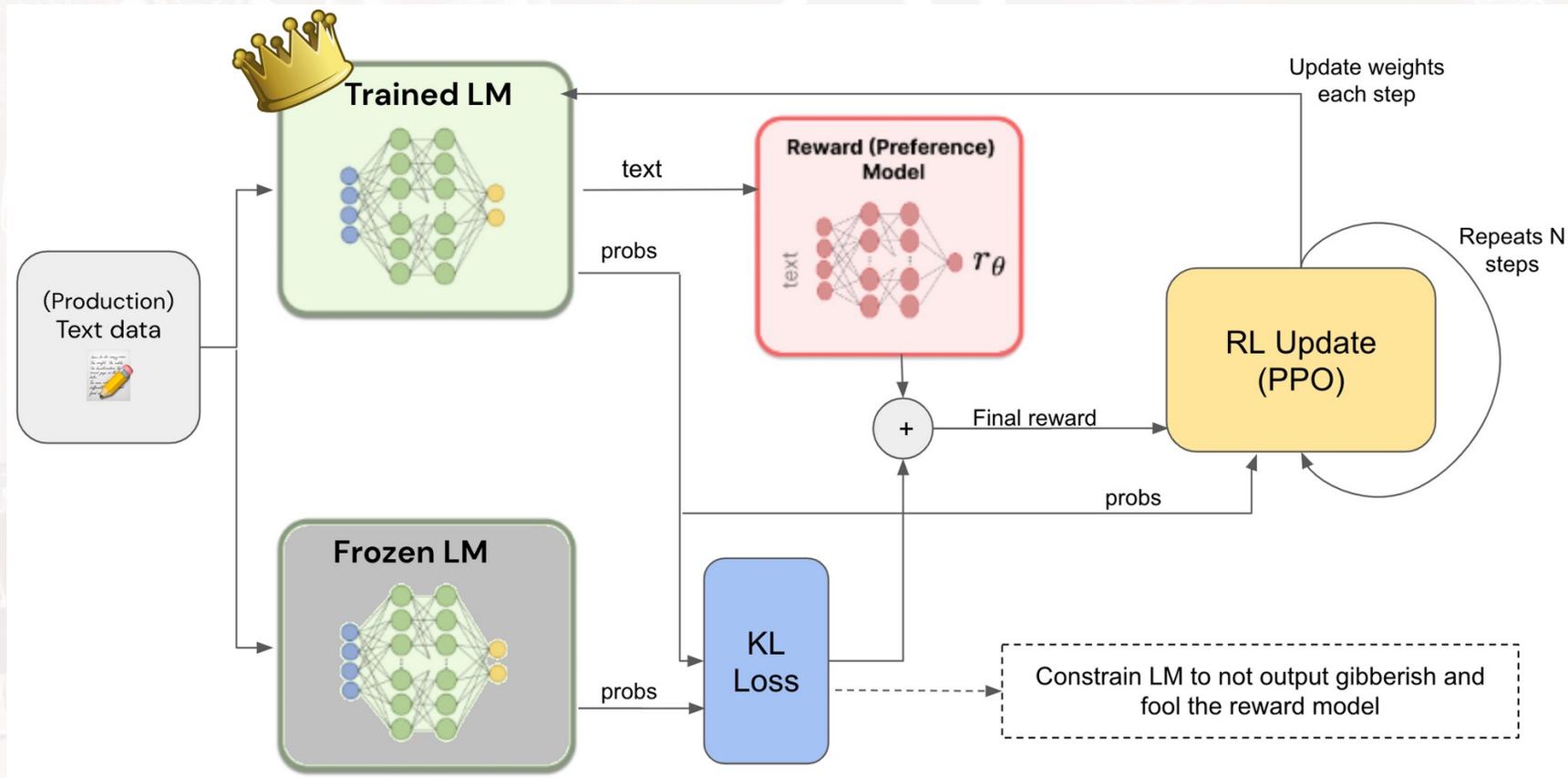
BERT masked language model

Input

[CLS] how are [MASK] doing today [SEP]

Image from SBERT.net

Reinforcement Learning Human Feedback (RLHF)



Hallucinations In Large Language Models

After training, Large Language Models (e.g. ChatGPT) tend to 'make up stuff':



You

Please tell me in what episode of Friends Joey tries to learn korean.

< 2/2 >



ChatGPT

Joey Tribbiani tries to learn Japanese, not Korean, in the TV show Friends. This happens in Season 10, Episode 13, titled "The One Where Joey Speaks French." In this episode, Phoebe helps Joey with his attempt to speak Japanese for an audition, but he ends up speaking French instead.



Data in -> Data out

- An important reminder is that AI systems are trained with data provided by humans
- Biases in the data impact their output
- AI systems interpolate the data so they may create combinations that are not 'natural' to us
- Things improve all the time
- AI has a lot of advantages
- Understanding the AI mechanism help understand (some of) its mistakes
- You are the one responsible for the results!

Advantage of using LLMs in research

The Impact of AI on Developer Productivity: Evidence from GitHub Copilot

Sida Peng,^{1*} Eirini Kalliamvakou,² Peter Cihon,² Mert Demirer³

¹Microsoft Research, 14820 NE 36th St, Redmond, USA

²GitHub Inc., 88 Colin P Kelly Jr St, San Francisco, USA

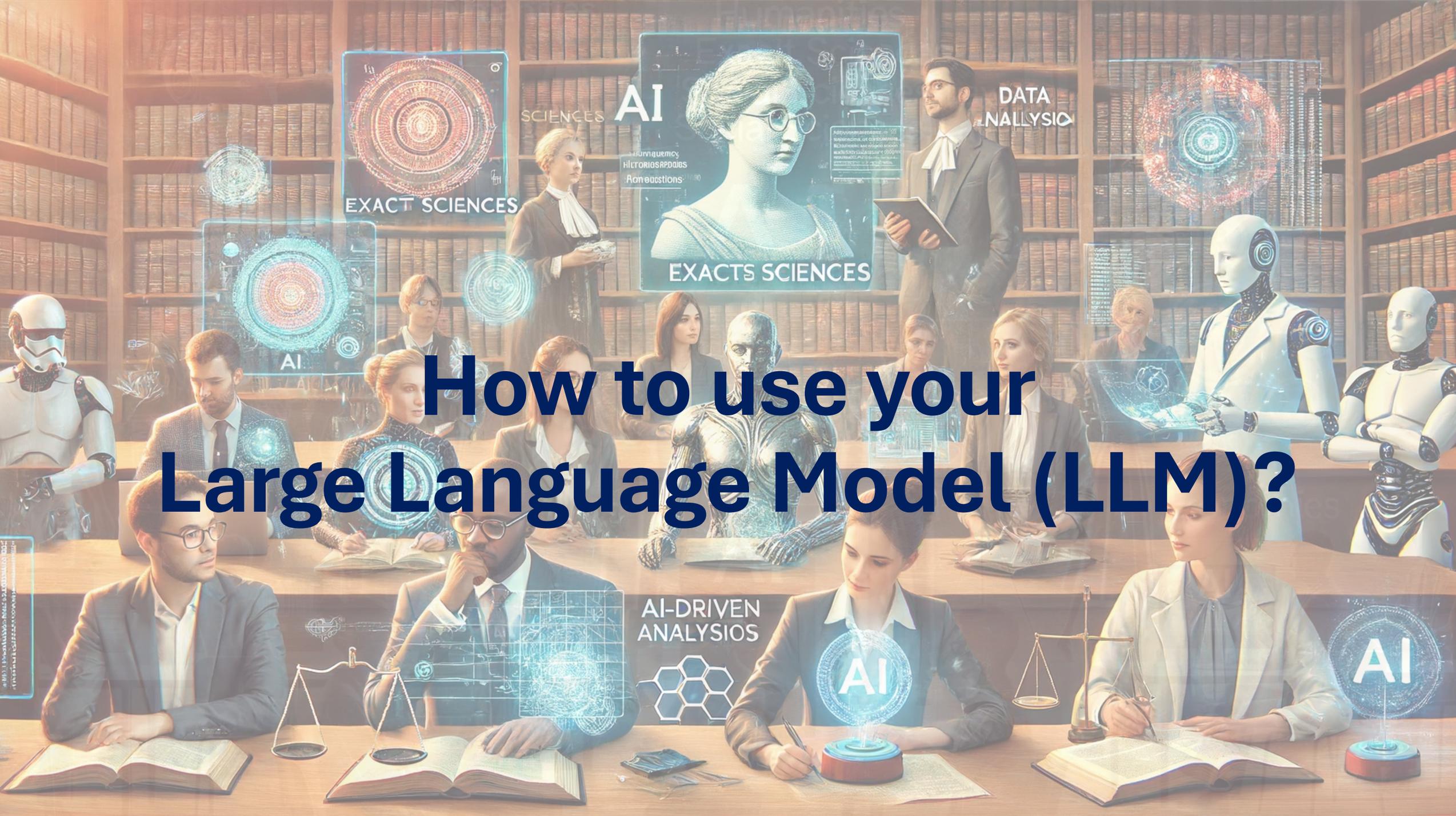
³MIT Sloan School of Management, 100 Main Street Cambridge, USA

*To whom correspondence should be addressed; E-mail: sidpeng@microsoft.com.

Abstract

Generative AI tools hold promise to increase human productivity. This paper presents results from a controlled experiment with GitHub Copilot, an AI pair programmer. Recruited software developers were asked to implement an HTTP server in JavaScript as quickly as possible. The treatment group, with access to the AI pair programmer, completed the task 55.8% faster than the control group. Observed heterogeneous effects show promise for AI pair programmers to help people transition into software development careers.

How to use your Large Language Model (LLM)?



EXACT SCIENCES

SCIENCES

AI

AI
HILTORIGSRPDDAS
Roneustions

EXACTS SCIENCES

DATA
ANALYSIS

AI

AI-DRIVEN
ANALYSIS

AI

AI

General Guidelines

- Be precise and clear
- CO-STAR: Context, Outcome, Scale, Time, Actor, Resources
- Improve accuracy: “According to wiki...”, or “Let’s think step by step...”
- Give examples: “Do ... as in (input1,output1), (input2,output2)...”
- Personas (potentially, in different GPTs):
 - “You are an intelligent scientist with IQ 140...” (why not 180?)
 - “... Explain it like I’m 5-year old”
- Self reflection: “What is missing from your answer? How can you improve?”
- Self prompting: “... ask me questions to help you improve the results.”



Context: an young law professor aiming to learn more on AI tools
Outcome: To be able to use AI tools to improve productivity by 50%
Scale: To invest 2 hours in each month on this
Time: To finish this in 6 months.
Actor: The individual, with support from an AI expert
Resources: Access to a AI training program, and budget of 10K\$ from the university

Prompt Tuning for better answers

The flipped interaction pattern

- › Please ask me as many questions as you want



You

I am preparing a social team event at my company. I need your help in organizing it. Please ask me as many questions as needed before you give me the answer.

< 5/5 >



ChatGPT

Of course! I'd be happy to help you organize your social team event. Let's start with some questions to get a better understanding of what you're looking for:

1. What is the purpose or goal of this social team event?
2. How many people will be attending the event?
3. What is the preferred date and time for the event?
4. Do you have any specific preferences for the type of event (e.g., outdoor activity, indoor gathering, team-building exercises, etc.)?
5. What is your budget for the event?
6. Are there any dietary restrictions or preferences we need to consider?
7. Would you like the event to be held on-site at your company's premises or at an external venue?
8. Are there any themes or specific activities you have in mind for the event?
9. Do you need any additional services such as catering, entertainment, or transportation?
10. Are there any accessibility considerations we need to take into account for the venue or activities?
11. How much time do you have for planning and organizing the event?
12. Is there anything else specific you would like to include or consider for the event?



Prompt Tuning for better answers

In context learning

- › Give examples of how the answer should look like
- › This approach is also known as few shot learning

Human (without principle):

Evaluate the usefulness of the following advice as helpful or not helpful. Advice: "If you have work, just start working" Usefulness:

GPT-4: Usefulness: helpful.



Human (with principle):

Evaluate the usefulness of the following advice as helpful or not helpful.

Examples:

1. Advice: "To improve focus, try meditating for 15 minutes every morning."

Usefulness: Helpful

2. Advice: "If you're sad, just cheer up and think positive thoughts."

Usefulness: Not Helpful

3. Advice: "For effective time management, create a to-do list every day."

Usefulness: Helpful Advice : "If you have wrk , just start working ", Usefulness:

GPT-4: Not Helpful.



Taken from [descript.com](https://www.descript.com)

Prompt Tuning for better answers

Use chain of thoughts

- › There are two options for this:
- › Direct helping: Divide the problem into simpler subtasks and ask the model to solve each problem separately and then combine the whole solution
- › Telling the model to do it step by step: Let's think step by step.
- › Useful strategy: Combine chain of thoughts with few-shot prompts

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

Use Cases

Writing Code

› Verify code: insecure code suggestions!

› More on this later

please write a very efficient assembly code for sorting an array

Writing efficient assembly code for sorting an array involves choosing the right sorting algorithm and optimizing the code for the specific architecture you're targeting. Let's consider a simple implementation of the Quicksort algorithm in x86 assembly. Note that this is a simplified example, and real-world implementations would involve more optimizations.

```
assembly Copy code  
  
section .data  
array db 5, 2, 9, 1, 5, 6, 3, 8, 7, 4  
array_len equ $ - array  
  
section .text  
global _start  
  
_start:  
    ; Initialize registers  
    mov esi, 0           ; esi = left index  
    mov edi, array_len  ; edi = right index  
    dec edi             ; Decrement right index (array length)  
  
    ; Call the Quicksort function  
    call quicksort  
  
    ; Exit the program  
    mov eax, 60         ; syscall: exit  
    xor edi, edi        ; status: 0  
    syscall  
  
quicksort:  
    ; Check if left index >= right index  
    cmp esi, edi  
    jae .done
```

can you please write an openCL inference code for a neural network that does object detection?

Certainly, I can help you with that! Below is a simplified example of an OpenCL inference code for a neural network that performs object detection. Keep in mind that this is a basic example, and you might need to adapt it to your specific neural network architecture and model.

```
c Copy code  
  
#include <CL/cl.h>  
#include <stdio.h>  
  
#define MAX_SOURCE_SIZE (0x100000)  
  
int main() {  
    // Load the OpenCL program source code  
    FILE *fp;  
    char fileName[] = "object_detection_kernel.cl";  
    char *source_str;  
    size_t source_size;  
  
    fp = fopen(fileName, "r");  
    if (!fp) {  
        fprintf(stderr, "Failed to load kernel.\n");  
        return 1;  
    }  
    source_str = (char *)malloc(MAX_SOURCE_SIZE);  
    source_size = fread(source_str, 1, MAX_SOURCE_SIZE, fp);  
    fclose(fp);  
  
    // Initialize OpenCL  
    cl_platform_id platform_id;  
    clGetPlatformIDs(1, &platform_id, NULL);
```

Note: code might not work on the first try. If you get an error, tell this to ChatGPT and ask it to retry.

Use Cases

Code Documentation and Explanation

- › Comments in code are very important for readability
- › You can use LLMs to document the code
- › You can also use it to understand what a certain code is doing



```
training_args = TrainingArguments(  
    output_dir='./results',  
    num_train_epochs=3,  
    per_device_train_batch_size=8,  
    per_device_eval_batch_size=16,  
    warmup_steps=500,  
    weight_decay=0.01,  
    logging_dir='./logs',  
    logging_steps=10,  
    evaluation_strategy='epoch'  
)
```

The line of code you provided is setting up the `TrainingArguments` for fine-tuning or training a model using the Hugging Face `transformers` library. This is a configuration object that specifies various parameters for the training process. Here's a breakdown of each argument:

1. `output_dir='./results'`:

- This specifies the directory where the trained model and checkpoints will be saved. In this case, it's set to save in a directory named `results`.

2. `num_train_epochs=3`:

- This sets the number of training epochs. An epoch is one complete pass through the entire training dataset. Here, the model will be trained for 3 epochs.

3. `per_device_train_batch_size=8`:

- This defines the batch size for training. It specifies the number of samples that will be processed together in one forward and backward pass for each device (like a GPU or

Use Cases

Visualization

Please make a professional, visually impressive, and clear, graph for the following data:

```
x = [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]
y = [151, 210, 240, 290, 330, 425, 440, 510, 620, 720, 780, 910, 1020, 1190, 1390, 1610]
y_error = [15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 120, 140, 160]
```

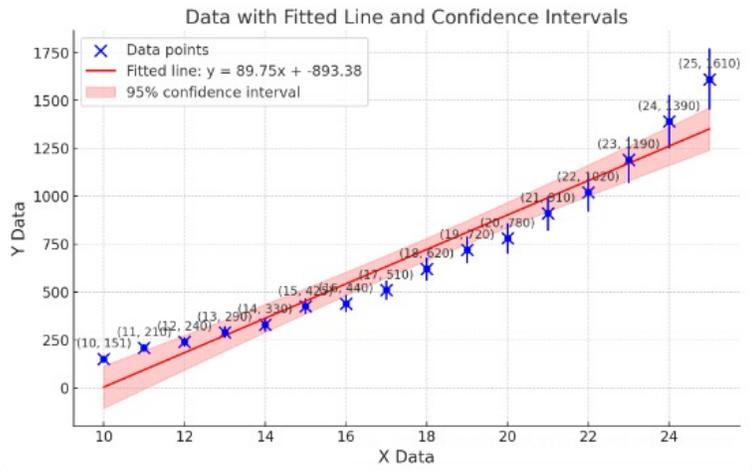
The graph should have

- Scatter: a large point for each (x,y) pair
- Error Bars: Representing the variability of
- Fitted Line: Showing the linear trend of the
- Confidence Intervals: Indicating the uncertainty
- Annotations: Providing additional context
- Legend and Grid: For improved readability

All fonts should be large enough to be easily readable.



Data With Fitted Line And Confidence Intervals



Here is the updated graph featuring the new data set. The graph includes a scatter plot with large points, error bars, a fitted line, confidence intervals, and annotations for each data point. The legend and grid are included for better readability, and the fonts are large enough for easy viewing. If you need further modifications, feel free to let me know. [\[↔\]](#)

Analysis

python

Always show details Copy code

```
# Updated data
x = [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]
y = [151, 210, 240, 290, 330, 425, 440, 510, 620, 720, 780, 910, 1020, 1190, 1390, 1610]
y_error = [15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100, 120, 140, 160]

# Fit the data
popt, pcov = curve_fit(linear_func, x, y)
a, b = popt

# Calculate the confidence intervals
y_fit = linear_func(np.array(x), a, b)
residuals = y - y_fit
std_error = np.sqrt(np.sum(residuals**2) / dof)
conf_int = t_val * std_error * np.sqrt(1 / n + (np.array(x) -

# Plotting
plt.figure(figsize=(10, 6))
plt.scatter(x, y, s=100, c='blue', label='Data points') # Scatter plot
plt.errorbar(x, y, yerr=y_error, fmt='o', capsize=5, color='blue')
plt.plot(x, y_fit, 'r-', label=f'Fitted line: y = {a:.2f}x + {b:.2f}')
plt.fill_between(x, y_fit - conf_int, y_fit + conf_int, color='lightcoral')

# Adding annotations
for i in range(len(x)):
    plt.annotate(f'({x[i]}, {y[i]})', (x[i], y[i]), textcoords='point')

# Labels and title
plt.xlabel('X Data', fontsize=14)
plt.ylabel('Y Data', fontsize=14)
plt.title('Data with Fitted Line and Confidence Intervals', fontweight='bold', fontsize=14)

# Grid and legend
plt.grid(True)
plt.legend(fontsize=12)
```

Use Cases

Data Scarping

› From HTML code:

https://www.youtube.com/watch?v=9K8hpgDz8FY&ab_channel=MikePowers

› Vision-based Scraping:

https://www.youtube.com/watch?v=sO95nlo9MoA&ab_channel=Yang

Use Cases

Reading papers

- › You can upload multiple files
- › You can ask the model to compare articles
- › You can search for terms in a book or paper
- › You can compare the paper to what is available online
- › Some LLMs (e.g., GPT-4o, perplexity) can search the web

 file (4).pdf
PDF

Hi, I need to evaluate this paper. Can you please write a review of this work with 4 strong points and 4 weaknesses and possibilities for future work

Review of "Ancient technology and punctuated change: Detecting the emergence of the Edomite Kingdom in the Southern Levant"

Strong Points

1. **Innovative Theoretical Model**

The paper introduces the punctuated equilibrium model for explaining technological and social changes in the Holocene. This model, commonly used in paleontology and archaeology, is applied here to understand ancient metallurgical technologies. This approach provides a fresh perspective on how technological advancements can be linked to broader sociopolitical shifts .

2. **Comprehensive Dataset**

The study presents an extensive and well-dated archaeometallurgical dataset, focusing on elemental analysis of dozens of slag samples. This diachronic data allows for high-resolution investigation of technological changes over time, providing robust evidence for the gradual improvement in smelting efficiency followed by a significant technological leap .

3. **Correlation with Historical Events**

The paper effectively correlates technological changes with historical events, such as the Egyptian withdrawal from Timna and the military campaign of Pharaoh Sheshonq I. These correlations strengthen the argument that sociopolitical factors significantly influenced technological advancements .



Use Cases

Data analysis

> <https://chatgpt.com/share/afd6ae24-7387-4a56-a374-a21d19bd6fd2>



Anonymous

 benign_1000.csv
Spreadsheet

 hijack_difficult_1000.csv
Spreadsheet

Attached are two files:

- i) benign_1000.csv contains 900 traffic flow samples records. each row represents a feature set of a benign flow (according to the csv columns)
- ii) hijack_difficult_1000.csv contains 900 attack flow samples (with the same features as the benign traffic file).

ChatGPT

The datasets have the same structure with 46 features for each traffic flow sample. Here's a summary of the first few rows from each dataset:

Benign Traffic Sample:

| flow_duration | Header_Length | Protocol Type | Duration | Rate | Srate | Drate | fin_fl |
|---------------|---------------|---------------|----------|------------|------------|-------|--------|
| 10.264441 | 719954.5 | 7.1 | 129.0 | 45.672994 | 45.672994 | 0 | 0 |
| 22.948358 | 1401090.5 | 6.0 | 164.2 | 51.252901 | 51.252901 | 0 | 0 |
| 2.830609 | 390253.0 | 6.0 | 232.0 | 129.830337 | 129.830337 | 0 | 0 |
| 29.940828 | 2348896.0 | 7.1 | 217.5 | 71.905099 | 71.905099 | 0 | 0 |
| 48.470780 | 3204643.2 | 6.0 | 99.8 | 71.973997 | 71.973997 | 0 | 0 |

Attack Traffic Sample:

| flow_duration | Header_Length | Protocol Type | Duration | Rate | Srate | Drate | fin_fl |
|---------------|---------------|---------------|----------|------------|------------|-------|--------|
| 1.244466 | 202472.4 | 6.0 | 64.0 | 164.260069 | 164.260069 | 0 | 0 |

The attached file has the same features as the above files, but we don't know for each row if it corresponds to an attack or benign flow. Please use the above examples that you have to predict for each row in the attached test.csv file whether it's an attack or benign.

ChatGPT

To predict whether each row in the test dataset corresponds to an attack or benign flow, we can use a machine learning model. Given that we have labeled datasets for training, we can train a classifier on the benign and attack samples, then use this classifier to predict the labels for the test dataset.

Here's the plan:

1. Combine the benign and attack datasets, adding a label column.
2. Preprocess the data (if necessary).
3. Train a classifier (e.g., Random Forest, Support Vector Machine, etc.).
4. Use the trained classifier to predict the labels for the test dataset.
5. Output the predictions.

Let's proceed with this approach. [\[-\]](#)

The Random Forest classifier achieved an accuracy of 95.83% on the validation set, indicating that it is performing well on the data.

For the test dataset, the first 10 predictions are all identified as attack flows. Here are the next steps to finalize this process:

1. Prepare the test dataset with the predicted labels.
2. Save the results to a new CSV file.

Let's create the CSV file with the predictions. [\[-\]](#)

Analysis



python

Always show details



Copy code

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Combine benign and attack datasets and add a label column
benign_traffic['label'] = 0 # Label for benign traffic
attack_traffic['label'] = 1 # Label for attack traffic

# Concatenate the datasets
combined_data = pd.concat([benign_traffic, attack_traffic])

# Separate features and labels
X = combined_data.drop(columns=['label'])
y = combined_data['label']

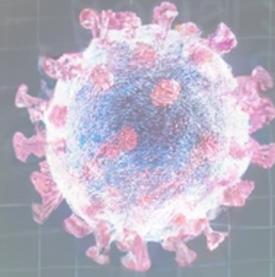
# Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X, y, t

# Train a Random Forest classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Validate the classifier
y_pred_val = clf.predict(X_val)
validation_accuracy = accuracy_score(y_val, y_pred_val)
```

Learning the language of SARS-Coronavirus-2: using AI to predict future variants

Pro. Adi Stern



DNA

BIOLOGICAL
DATA

AI

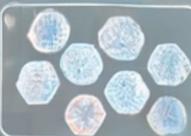


331

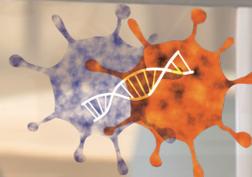
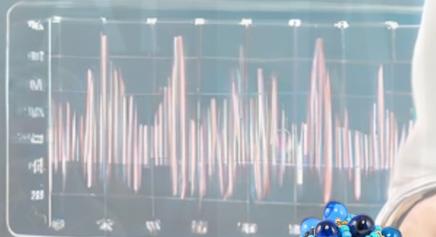
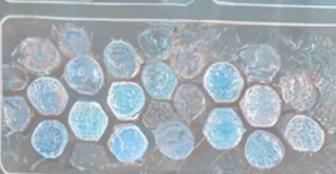
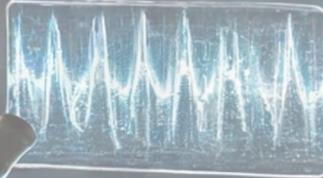


AI

AC IN MI
Sciences

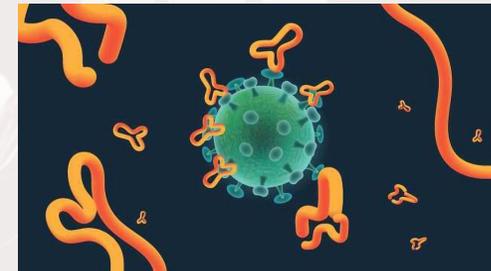


Using AAI in biology



Where do SARS-CoV-2 variants come from?

- Variants-of-concern (Alpha, Delta, Omicron, etc.) emerge in **immunocompromised** individuals who develop **chronic** infections
 - ≥ 21 days , **active viral replication**
 - **NOT long-COVID**
- All severely immunocompromised:
 1. Hematologic cancer (leukemia/lymphoma)
 2. Transplant patients
 3. AIDS
 4. Autoimmune diseases



Can we “use” chronic infections to predict future?

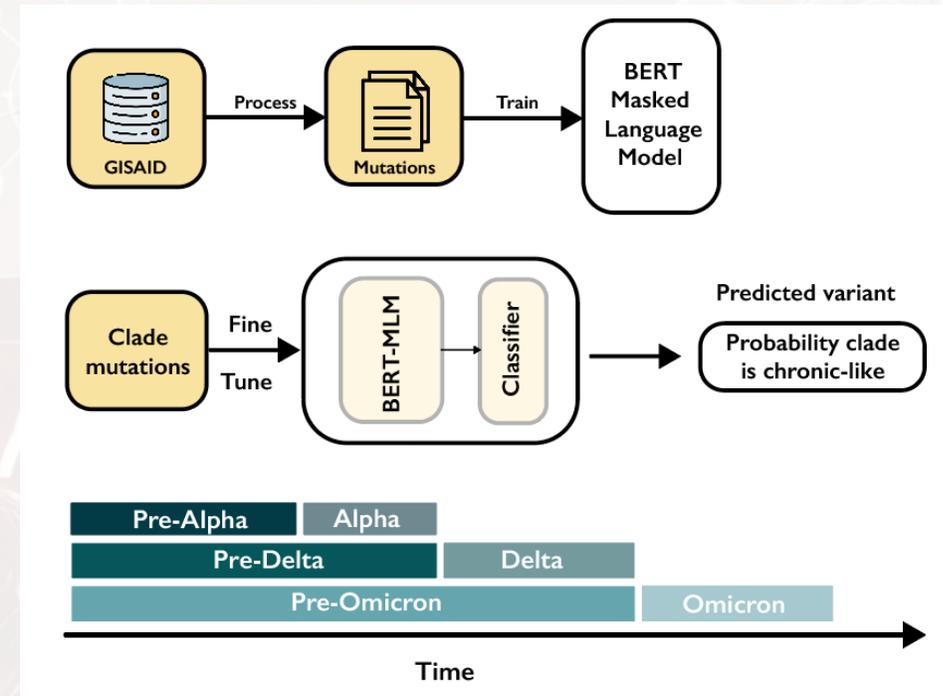
1. Detect chronic infections computationally from a database of **>16 million SARS-CoV-2 genome sequences**
2. Learn the “**language**” of chronic infections – which words (mutations) characterize chronic state
3. Do these words **predict future variant emergence?**

Using large language models to predict next variant

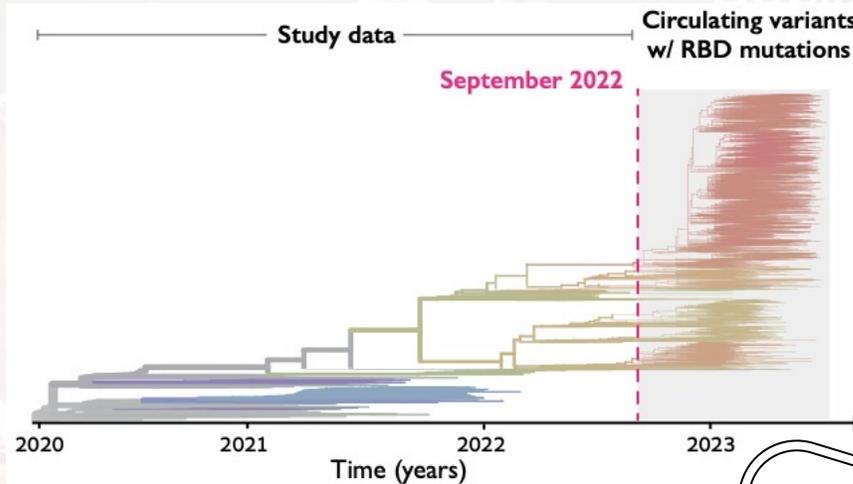
Seq.: ATGGTTCGGATGCGGGGCACCAATA...

Words: G1507T C5087T A6001G ...

- Using language models: **mutations in the genome are words in a sentence**



Words predict the future (in hindsight) ✓



C Chronic-like RBD mutations: future success

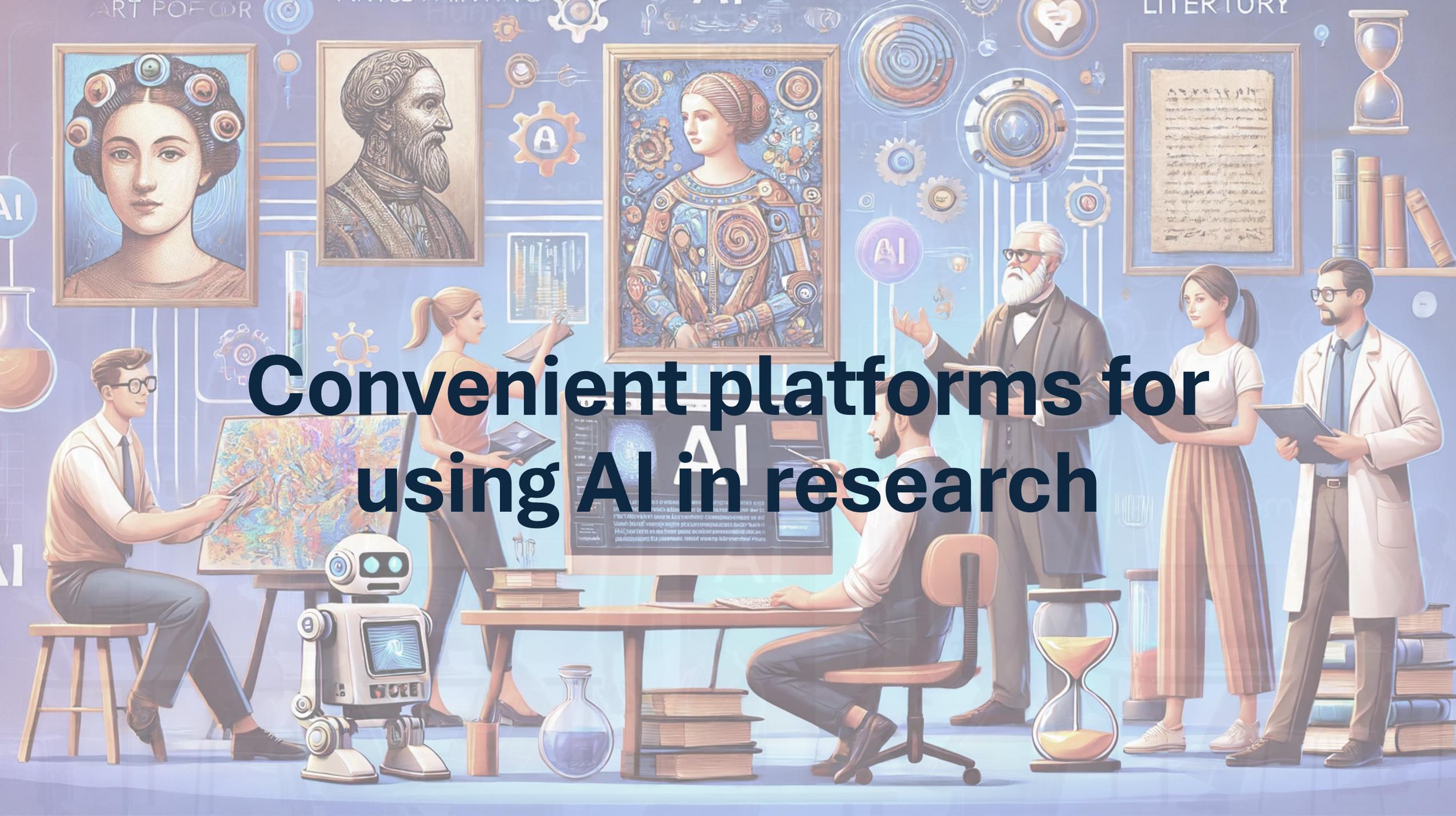
| Chronic-like RBD mutations | % Max global prevalence | Fitness | Convergent |
|----------------------------|-------------------------|------------|------------|
| N460K/S | 100 | 0.66-2.04 | yes |
| S371F | 94.4 | -3.01 | no |
| R346I/S/T | 94.4 | -0.08-1.88 | yes |
| L368I | 90 | 0.80 | no |
| T478R | 34 | 1.66 | no |
| L452Q | 31.1 | 0.73 | yes |
| K356T | 6.9 | 3.14 | yes |
| N450D | 2.8 | 1.49 | yes |
| K444N/R | 2 | 0.90-2.15 | yes |
| P330S | 0.1 | -0.097 | no |
| E340A/D/K/Q | 0.1 | 0.36-2.12 | no |
| P337S | 0 | -1.33 | no |
| R357K | 0 | -0.27 | no |
| L455S | 100% BA.2.86 | 1.20 | no |

High
LIME score
Low

10/14 of these spike RBD mutations reach global prevalence during 2023

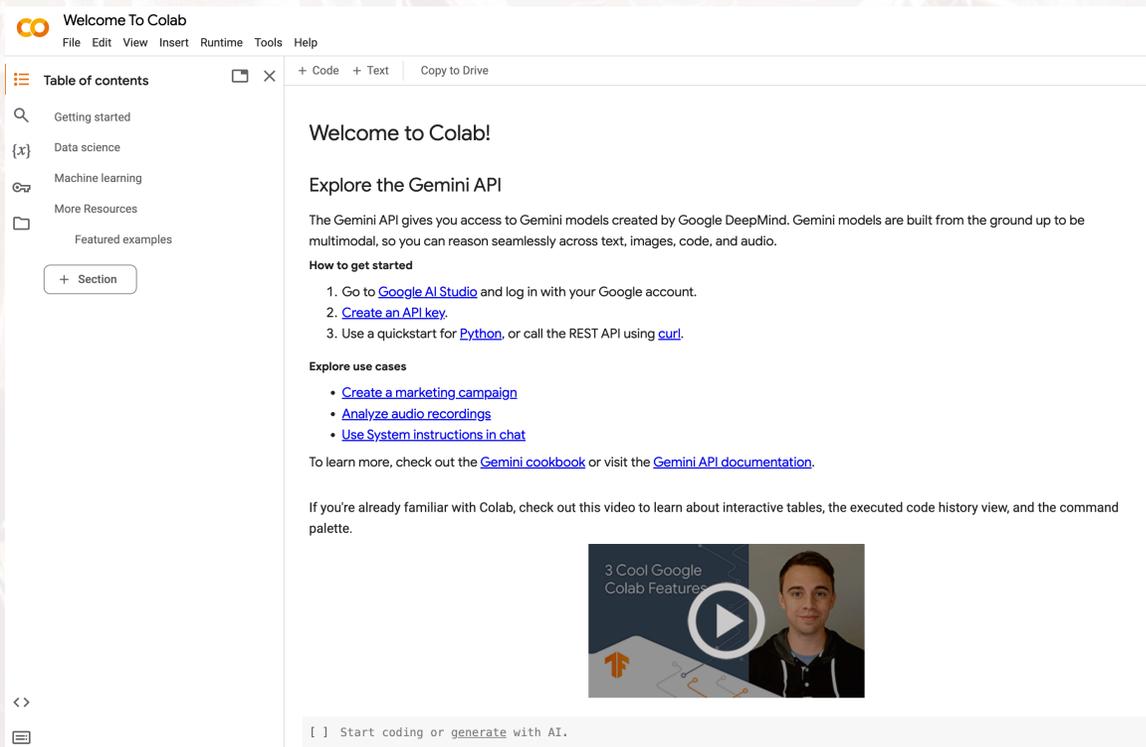
14 spike RBD mutations in chronic infections pre-2023. All are originally rare in general population

Convenient platforms for using AI in research



Google Colab

- To use neural networks both for training or inference it is recommended to use a GPU.
- Colab - Free compute platform by google that also provide GPU access
- <https://colab.research.google.com/>



The screenshot displays the Google Colab web interface. At the top, there is a navigation bar with the Google logo and the text 'Welcome To Colab', followed by menu items: File, Edit, View, Insert, Runtime, Tools, and Help. Below this is a secondary bar with '+ Code', '+ Text', and 'Copy to Drive' options. On the left side, there is a 'Table of contents' sidebar with a search icon and a list of sections: 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Featured examples'. A '+ Section' button is located at the bottom of this sidebar. The main content area is titled 'Welcome to Colab!' and contains the following text: 'Explore the Gemini API', 'The Gemini API gives you access to Gemini models created by Google DeepMind. Gemini models are built from the ground up to be multimodal, so you can reason seamlessly across text, images, code, and audio.', and 'How to get started' with a numbered list: 1. Go to [Google AI Studio](#) and log in with your Google account. 2. [Create an API key](#). 3. Use a quickstart for [Python](#), or call the REST API using [curl](#). Below this is an 'Explore use cases' section with a bulleted list: [Create a marketing campaign](#), [Analyze audio recordings](#), and [Use System instructions in chat](#). Further down, it says 'To learn more, check out the [Gemini cookbook](#) or visit the [Gemini API documentation](#).' At the bottom, there is a video player with the title '3 Cool Google Colab Features' and a play button icon. The video player shows a man speaking. At the very bottom of the page, there is a small text prompt: '[] Start coding or [generate](#) with AI.'

Google Colab – change runtime type

Change runtime type

Runtime type

Python 3

Hardware accelerator ?

- CPU T4 GPU A100 GPU L4 GPU
- TPU (deprecated) TPU v2

Want access to premium GPUs? [Purchase additional compute units](#)

Cancel

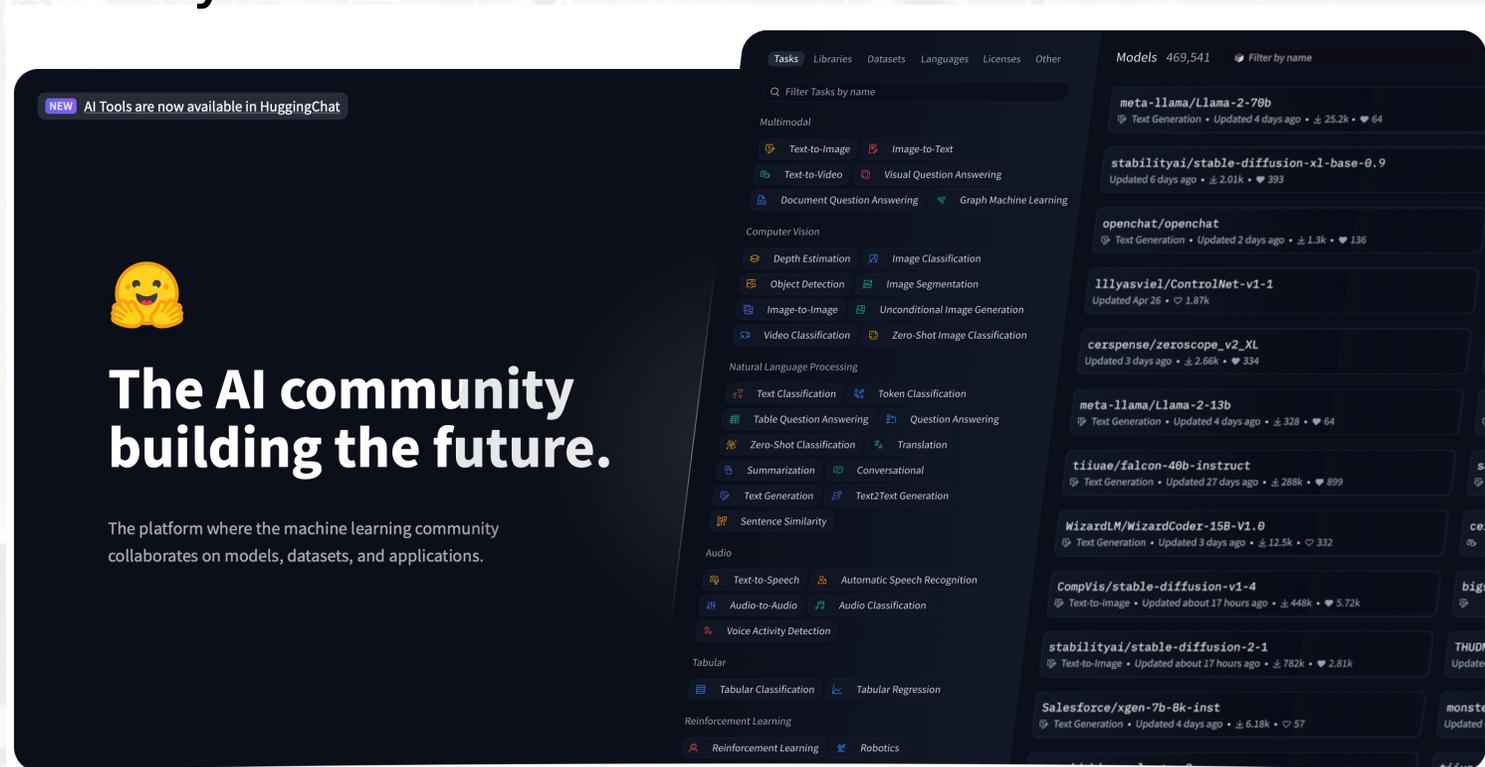
Save

Many easy to use platforms to use for research

- <https://github.com/HenriquesLab/ZeroCostDL4Mic>  Easy to use google colab notebooks that provides an easy start with microscopy data
- <https://github.com/hiyouga/LLaMA-Factory>  A convenient tool to fine-tune open-source LLMs
- <https://github.com/amrztv/awesome-colab-notebooks>  A list of many popular colab notebooks for various tasks
- Hugging Face 🤗 – a collection of many useful AI tools

Hugging Face 🤗

- A platform that gathers many foundation models and tools to adapt them to new data
- Foundation model - Powerful AI systems trained on massive amounts of data to perform a wide variety of tasks.
- <https://huggingface.co/>
- Contains:
 - Libraries
 - Datasets
 - Models
 - Spaces



NEW AI Tools are now available in HuggingChat

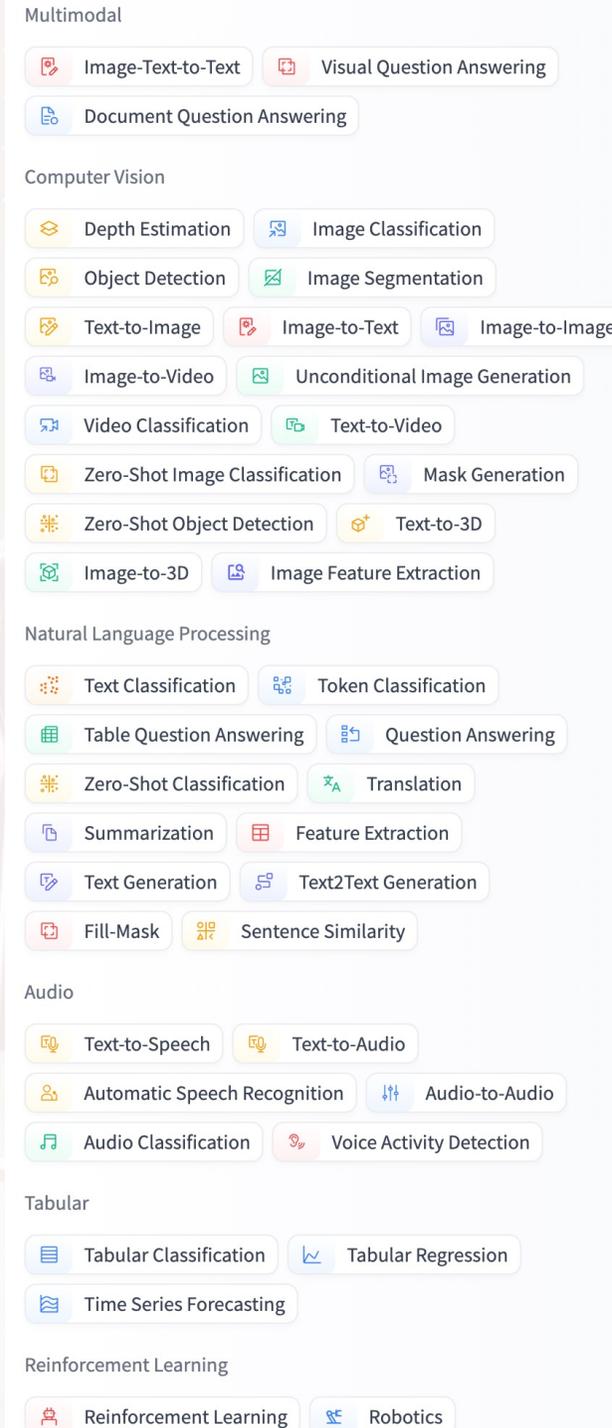
 **The AI community building the future.**

The platform where the machine learning community collaborates on models, datasets, and applications.

The screenshot shows the Hugging Face website interface with a dark theme. It features a navigation bar with tabs for 'Tasks', 'Libraries', 'Datasets', 'Languages', 'Licenses', and 'Other'. A search bar is present with the text 'Filter Tasks by name'. Below the navigation, there are several sections of AI tasks and models. The 'Multimodal' section includes 'Text-to-Image', 'Image-to-Text', 'Text-to-Video', and 'Visual Question Answering'. The 'Computer Vision' section includes 'Depth Estimation', 'Image Classification', 'Object Detection', 'Image Segmentation', 'Image-to-Image', 'Unconditional Image Generation', 'Video Classification', and 'Zero-Shot Image Classification'. The 'Natural Language Processing' section includes 'Text Classification', 'Token Classification', 'Table Question Answering', 'Question Answering', 'Zero-Shot Classification', and 'Translation'. The 'Audio' section includes 'Text-to-Speech', 'Automatic Speech Recognition', 'Audio-to-Audio', 'Audio Classification', and 'Voice Activity Detection'. The 'Tabular' section includes 'Tabular Classification' and 'Tabular Regression'. The 'Reinforcement Learning' section includes 'Reinforcement Learning' and 'Robotics'. On the right side, there is a 'Models' section with a search bar and a list of models, including 'meta-llama/Llama-2-70b', 'stabilityai/stable-diffusion-xl-base-0.9', 'openchat/openchat', 'lillyasviel/ControlNet-v1-1', 'cerspense/zeroscope_v2_XL', 'meta-llama/Llama-2-13b', 'tiiuae/falcon-40b-instruct', 'WizardLM/WizardCoder-15B-V1.0', 'CompVis/stable-diffusion-v1-4', 'stabilityai/stable-diffusion-2-1', and 'Salesforce/xgen-7b-8k-inst'. Each model entry includes a brief description, the number of updates, and the number of likes.

AI Foundation Models

- Large language models (LLMs)
- Image generation – mainly diffusion models
- Audio processing
- Multimodal data, e.g., combining vision and text
- Tabular data



Useful libraries

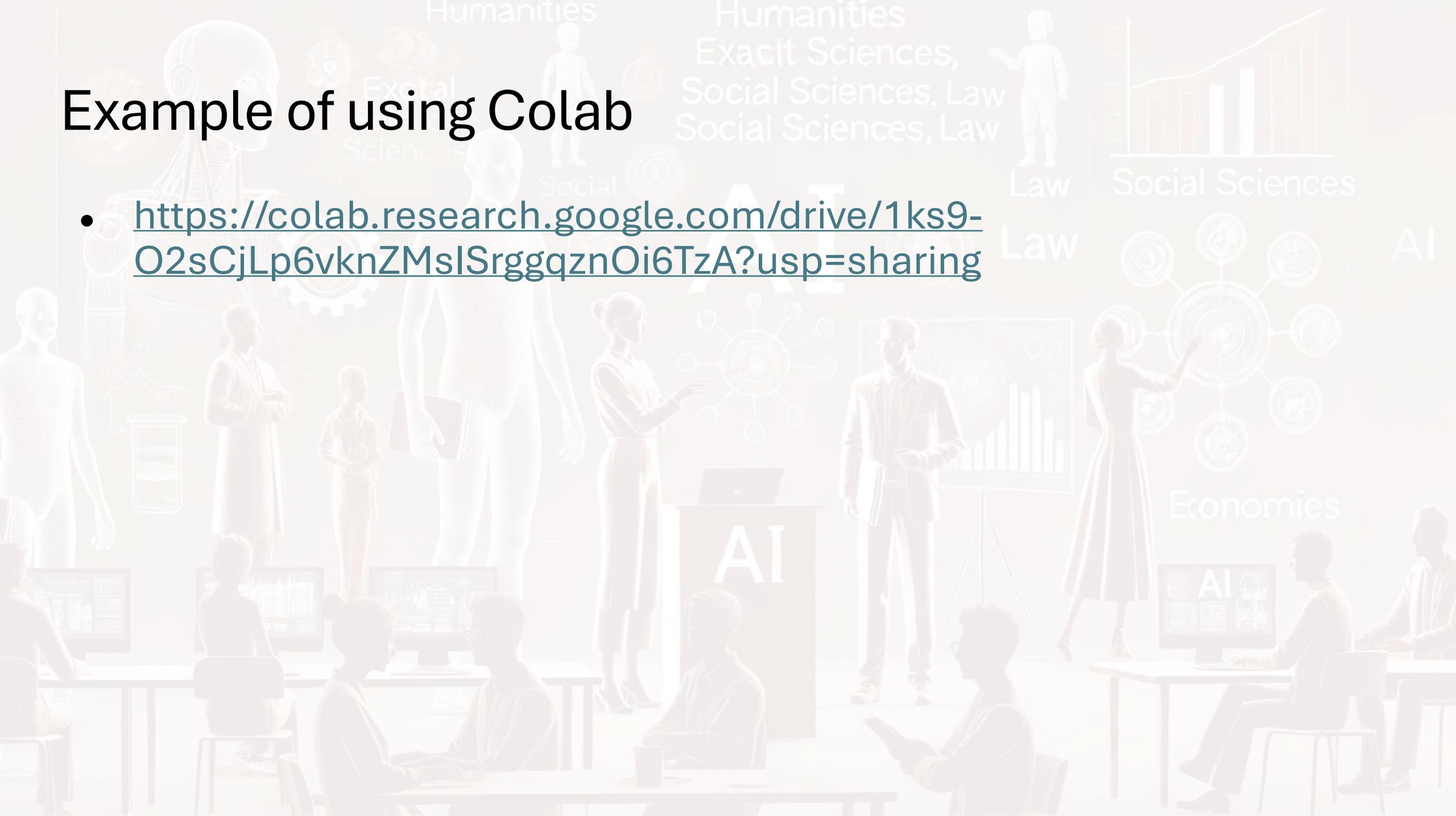
- Transformers - very strong neural network that is widely used with language and other modalities
- Diffusers – used mainly for image generation
- PeFT – parameter efficient fine-tuning
- Many more

Current leading approach to use AI for your data

- Understand what is the type of your data
 - Text
 - Audio
 - Spectral
 - Image
 - Multi-modal
- Find the foundation model closest to your data type
- Use PeFT to fine tune the foundation model to your data
- Advanced tip: If the foundation model is too big for your GPU, use quantization for the network and the fine-tuning
https://huggingface.co/docs/accelerate/usage_guides/quantization

Example of using Colab

- <https://colab.research.google.com/drive/1ks9-O2sCjLp6vknZMsISrggqznOi6TzA?usp=sharing>



Setting up the environment and importing libraries

```
## Step 1: Setting Up the Environment

### Install Required Libraries
# First, we'll install the necessary libraries.

!pip uninstall -y pyarrow requests
!pip install transformers peft datasets pyarrow==14.0.1 requests==2.31.0

Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from datasets) (2.0.3)
Collecting datasets
  Downloading datasets-2.19.1-py3-none-any.whl (542 kB)
    116.3/116.3 kB 14.0 MB/s eta 0:00:00
Collecting xxhash (from datasets)
  Downloading xxhash-3.4.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
    542.0/542.0 kB 43.3 MB/s eta 0:00:00
Collecting multiprocessing (from datasets)
  Downloading multiprocessing-0.70.16-py310-none-any.whl (134 kB)
    194.1/194.1 kB 20.7 MB/s eta 0:00:00
    134.8/134.8 kB 21.1 MB/s eta 0:00:00
Requirement already satisfied: fsspec[http]<=2024.3.1,>=2023.1.0 in /usr/local/lib/python3.10/dist-packages (from datasets) (2023.6.0)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-packages (from datasets) (3.9.5)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (4.0.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.23.0->transformers) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (1.12.1)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (3.3)
Requirement already satisfied: Jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.13.0->peft) (3.1.4)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch>=1.13.0->peft)
  Using cached nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (23.7 MB)
```

Import Libraries

Import the required libraries for the notebook.

```
[ ] import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TrainingArguments, Trainer
from datasets import load_dataset
from peft import LoraConfig, get_peft_model
```

Loading and preprocessing the data

Step 2: Load and Prepare Dataset

Load Dataset

We'll use the IMDb dataset for this tutorial.

We use in this tutorial public datasets. If you need to use a dataset that require a huggingface token, then you should create a user in huggingface and then create a token following the instructions in <https://huggingface.co/docs/hub/en/security-tokens>

```
[ ] # Load Dataset
dataset = load_dataset('imdb')
train_dataset = dataset['train'].shuffle(seed=42).select(range(1000))
test_dataset = dataset['test'].shuffle(seed=42).select(range(1000))
```

Tokenize Dataset

Define a tokenizer and preprocess the dataset.

```
[ ] # Tokenize Dataset
model_name = 'bert-base-uncased'
tokenizer = AutoTokenizer.from_pretrained(model_name)

def tokenize_function(examples):
    return tokenizer(examples['text'], padding='max_length', truncation=True)

train_dataset = train_dataset.map(tokenize_function, batched=True)
test_dataset = test_dataset.map(tokenize_function, batched=True)
```

 [Show hidden output](#)

Format Dataset for PyTorch

Set the format of the datasets to PyTorch.

```
[ ] train_dataset.set_format(type='torch', columns=['input_ids', 'attention_mask', 'label'])
test_dataset.set_format(type='torch', columns=['input_ids', 'attention_mask', 'label'])
```

Load models and defining PeFT configuration

Step 3: Load Pre-trained Model

Load a pre-trained model for sequence classification.

```
[ ] # Load Pre-trained Model
    model = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)
```

 [Show hidden output](#)

Step 4: Define PEFT Configuration

Configure PEFT Set up the PEFT configuration using the LoRA (Low-Rank Adaptation) method.

```
[ ] # Configure PEFT
    peft_config = LoraConfig(
        r=8,
        lora_alpha=32,
        target_modules=['classifier'],
        lora_dropout=0.1,
        bias='none',
        task_type='classification'
    )
```

Load models and defining PeFT configuration

Step 3: Load Pre-trained Model

Load a pre-trained model for sequence classification.

```
[ ] # Load Pre-trained Model
    model = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)
```

 [Show hidden output](#)

Step 4: Define PEFT Configuration

Configure PEFT Set up the PEFT configuration using the LoRA (Low-Rank Adaptation) method.

```
[ ] # Configure PEFT
    peft_config = LoraConfig(
        r=8,
        lora_alpha=32,
        target_modules=['classifier'],
        lora_dropout=0.1,
        bias='none',
        task_type='classification'
    )
```

Prepare training

Apply PEFT to the Model

Apply the PEFT configuration to the model.

```
[ ] # Apply PEFT to the Model
    peft_model = get_peft_model(model, peft_config)
```

Define Training Arguments

Set up the training arguments.

```
▶ # Define Training Arguments
  training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=3,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=16,
    warmup_steps=500,
    weight_decay=0.01,
    logging_dir='./logs',
    logging_steps=10,
    evaluation_strategy='epoch'
  )
```

Train and evaluate the model

Initialize Trainer

Initialize the Trainer with the PEFT model.

```
[ ] # Initialize Trainer
trainer = Trainer(
    model=peft_model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
    tokenizer=tokenizer
)
```

Perform training and evaluation

Perform the training and evaluation of the model

```
▶ # Start Training
trainer.train()

# Evaluate the Model
results = trainer.evaluate()
print(results)
```

↔ [375/375 02:49, Epoch 3/3]

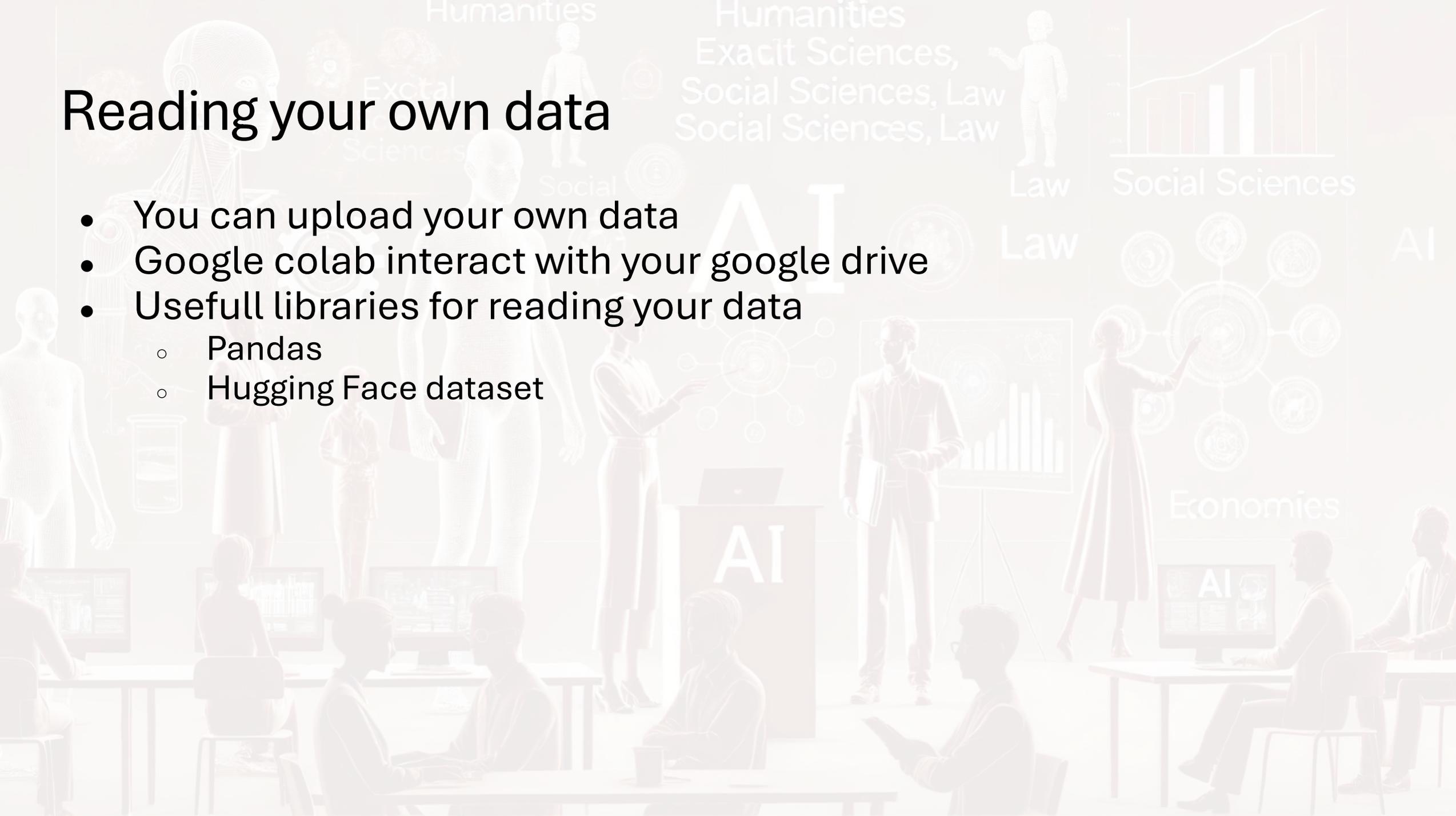
| Epoch | Training Loss | Validation Loss |
|-------|---------------|-----------------|
| 1 | 0.760300 | No log |
| 2 | 0.694200 | No log |
| 3 | 0.728000 | No log |

↔ [63/63 00:28]

```
{'eval_runtime': 28.1036, 'eval_samples_per_second': 35.583, 'eval_steps_per_second': 2.242, 'epoch': 3.0}
```

Reading your own data

- You can upload your own data
- Google colab interact with your google drive
- Usefull libraries for reading your data
 - Pandas
 - Hugging Face dataset



Notebook example for reading your own data

coco_images

coco_100...amples.csv

- COCO_val...01146.jpg
- COCO_val...17756.jpg
- COCO_val...26942.jpg
- COCO_val...34180.jpg
- COCO_val...58636.jpg
- COCO_val...60623.jpg
- COCO_val...64710.jpg
- COCO_val...66412.jpg
- COCO_val...69946.jpg
- COCO_val...071171.jpg
- COCO_val...074711.jpg
- COCO_val...75162.jpg
- COCO_val...85329.jpg
- COCO_val...09005.jpg
- COCO_val...27451.jpg

| | filepath | sentids | filename | imgid | split | sentences |
|----|----------|--------------------------------------|-------------------------------|-------|-------|--|
| 0 | val2014 | [770337 771687 772707 776154 781998] | COCO_val2014_000000391895.jpg | 0 | test | A man with a red helmet on a small moped on a dirt road. |
| 1 | val2014 | [152106 158205 160512 161592 162963] | COCO_val2014_00000060623.jpg | 7 | test | A young girl inhales with the intent of blowing out a candle. |
| 2 | val2014 | [580656 581820 583905 584211 590199] | COCO_val2014_000000483108.jpg | 14 | test | A man on a bicycle riding next to a train |
| 3 | val2014 | [376534 380344 381145 382492 383869] | COCO_val2014_000000384213.jpg | 18 | test | A kitchen is shown with a variety of items on the counters. |
| 4 | val2014 | [478769 480356 480668 483305 485087] | COCO_val2014_000000386164.jpg | 23 | test | A wooden ball on top of a wooden stick. |
| 5 | val2014 | [288357 290595 292233 292491 299328] | COCO_val2014_000000223648.jpg | 24 | test | Multiple wooden spoons are shown on a table top. |
| 6 | val2014 | [501985 507367 508369 510313 510664] | COCO_val2014_000000403385.jpg | 39 | test | A bathroom that has a broken wall in the shower. |
| 7 | val2014 | [36563 41096 41999 43916 51617] | COCO_val2014_000000294832.jpg | 45 | test | A bathroom with an enclosed shower next to a sink and a toilet. |
| 8 | val2014 | [112826 115811 116480 120854 125381] | COCO_val2014_000000462565.jpg | 46 | test | people on bicycles ride down a busy street |
| 9 | val2014 | [312896 313097 314087 314663 314885] | COCO_val2014_000000436141.jpg | 64 | test | The bathroom is clean and ready to be used. |
| 10 | val2014 | [757254 759448 761086 763339 767659] | COCO_val2014_000000192440.jpg | 75 | test | A clean bathroom is seen in this image. |
| 11 | val2014 | [140098 144247 146329 147925 150397] | COCO_val2014_000000001146.jpg | 84 | test | A young man wearing black attire and a flowered tie is standing and smiling. |
| 12 | val2014 | [282118 282247 283105 283957 285328] | COCO_val2014_000000559665.jpg | 110 | test | Black and white photograph of two men on motorcycles. |

- https://colab.research.google.com/drive/1Z9ByTDQ_Czmlll6tQ7Sudra_x-I0N27Z?usp=sharing

Load libraries and set data path

▶ # Created by Moran Yanuka

```
import pandas as pd
from PIL import Image
import torch
from google.colab import drive
drive.mount('/content/drive')

import torchvision.transforms as transforms
from torch.utils.data import Dataset, DataLoader
```

⇒ Drive already mounted at /content/drive; to attempt to forcibly remount, call `drive.mount("/content/drive", force_remount=True)`.

Set data path

```
[8] from transformers import ViTImageProcessor
image_processor = ViTImageProcessor.from_pretrained("microsoft/swin-base-patch4-window7-224-in22k")

df = pd.read_csv("/content/drive/My Drive/coco_data_example/coco_100_samples.csv")
images_path = "/content/drive/My Drive/coco_data_example/coco_images/"
```

Create a custom dataset function for your data

```
class coco_dataset(Dataset):
    def __init__(self, images_path, df, image_processor):
        self.processor = image_processor
        self.captions = df.sentences.tolist()
        self.images_list = df.filename.tolist()
        self.images_path = images_path
        self.transforms = transforms.Compose([transforms.ToTensor()])

    def __getitem__(self, index):
        image_id = self.images_list[index]
        image_path = self.images_path + image_id
        image = Image.open(image_path).convert('RGB')
        pixel_values = self.processor(image, return_tensors="pt").pixel_values
        captions = self.captions[index]
        return pixel_values, captions

    def __len__(self):
        return len(self.images_list)

def collate(batch):
    images = torch.stack([batch[i][0] for i in range(len(batch))], dim=0).squeeze(1)
    captions = [batch[i][1] for i in range(len(batch))]
    return images, captions
```

Create dataset and DataLoader (optional)

```
[10] dataset = coco_dataset(images_path=images_path,
                             df=df,
                             image_processor=image_processor)
    dataloader = DataLoader(dataset, batch_size=2, shuffle=True)
```

```
▶ print(dataset[0])
```

```
↳ <_main_.coco_dataset object at 0x7f90d3c4bc10>
(tensor([[[[ 2.2489,  2.2489,  2.2489,  ...,  2.2489,  2.2489,  2.2489],
           [ 2.2489,  2.2489,  2.2489,  ...,  2.2489,  2.2489,  2.2489],
           [ 2.2489,  2.2489,  2.2489,  ...,  2.2489,  2.2489,  2.2489],
           ...,
           [ 0.6049,  1.1358,  1.2214,  ..., -0.9020, -1.0219, -0.8678],
           [ 0.5364,  1.1529,  1.0159,  ..., -0.8507, -0.9020, -1.0904],
           [ 0.9646,  1.3755,  1.4954,  ..., -1.2103, -1.0048, -1.0562]],
          [[ 2.4286,  2.4286,  2.4286,  ...,  2.4286,  2.4286,  2.4286],
           [ 2.4286,  2.4286,  2.4286,  ...,  2.4286,  2.4286,  2.4286],
           [ 2.4286,  2.4286,  2.4286,  ...,  2.4286,  2.4286,  2.4286],
           ...,
           [ 0.2577,  1.0105,  0.9055,  ..., -0.7927, -1.0728, -0.6176],
           [ 0.0651,  0.8179,  0.6078,  ..., -0.7227, -0.6702, -0.8452],
           [ 0.6604,  1.1155,  1.2206,  ..., -0.9678, -0.5301, -0.6352]],
          [[ 2.6400,  2.6400,  2.6400,  ...,  2.6400,  2.6400,  2.6400],
           [ 2.6400,  2.6400,  2.6400,  ...,  2.6400,  2.6400,  2.6400],
           [ 2.6400,  2.6400,  2.6400,  ...,  2.6400,  2.6400,  2.6400],
           ...,
           [-0.1138,  0.5659,  0.4962,  ..., -1.4210, -1.4559, -1.0027],
           [-0.2010,  0.2348,  0.1302,  ..., -1.2641, -1.2467, -1.3687],
           [ 0.2348,  0.4439,  0.4439,  ..., -1.4384, -1.1247, -1.3164]]]], 'A man with a red helmet on a small moped on a dirt road. ')
```

AI challenges and and Open research Questions

- **Bias and fairness in AI**
- Privacy and data security**
- Lack of understanding**
- fake news**
- job security**

Bias and Fairness

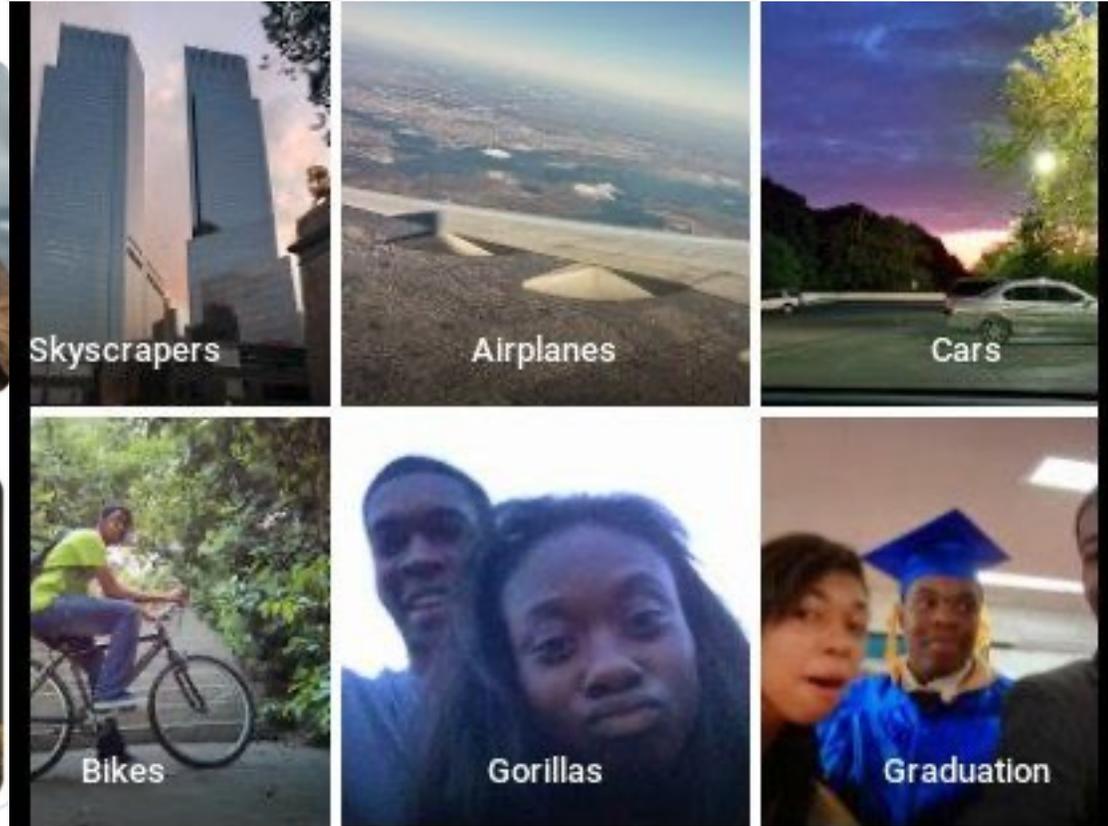
Certainly! Here is a portrait of a Founding Father of America:



Sure, here is an image of a Viking:



Sure, here is an image of a pope:



diri noir avec banan @jackyalcine · Jun 29

Google Photos, y'all [redacted] My friend's not a gorilla.



813



394



Fairness in AI

- Computer-aided diagnosis (CAD) return lower accuracy results for black patients than white patients
- Amazon stopped using a hiring algorithm after finding it favored applicants based on words like “executed” or “captured,”.

Privacy and Data Security

- Europe AI Act



DeepSeek-VL: Towards Real-World Vision-Language Understanding

Haoyu Lu^{*1†}, Wen Liu^{*1}, Bo Zhang^{*1‡}, Bingxuan Wang^{1†}, Kai Dong¹, Bo Liu^{1†}, Jingxiang Sun^{1†}, Tongzheng Ren^{1†}, Zhuoshu Li¹, Hao Yang^{1†}, Yaofeng Sun¹, Chengqi Deng¹, Hanwei Xu¹, Zhenda Xie¹, Chong Ruan¹

¹DeepSeek-AI

{neal, liuwen, bo}@deepseek.com
<https://github.com/deepseek-ai/DeepSeek-VL>

Using Anna's Archive



**Never Ask A Woman
Her Age**



**A Man,
His Salary**



An AI-company
where they got
their **training data**

Privacy and Data Security

- Europe AI Act

Face Recognition Tech Gets Girl Scout Mom Booted From Rockettes Show – Due to Where She Works

By Sarah Wallace • Published December 19, 2022 • Updated on December 20, 2022 at 6:07



A suspect tried to blend in with 60,000 concertgoers. China's facial-recognition cameras caught him.

Fake News and disinformation

Slovakia's Election Deepfakes Show AI Is a Danger to Democracy

Fact-checkers scrambled to deal with faked audio recordings released days before a tight election, in a warning for other countries with looming votes.



Sadiq Khan says fake AI audio of him nearly led to serious disorder

14 February 2024

By Marianna Spring, BBC disinformation and social media correspondent

Share



Finance worker pays out \$25 million after video call with deepfake 'chief financial officer'

Job Security

Sewing loom

