

Fundamentals of Artificial Intelligence

Support Libraries for AI in Python & Coding Automation

**Prof. E. Adetiba,
(PI, ASPMIR and FEDGEN @ CApIC-ACE, Covenant University)**

@

**The 3rd Google TensorFlow College Outreach Bootcamp and
FEDGEN Mini-Workshop**

Date: 11th to 13th December, 2023

Sponsored

By



Outline

1.1 Artificial Intelligence: An Introduction

1.2 Machine Learning Basics

1.3 Machine Learning Tasks

2.1 ML Datasets

2.2 Python Standard Libraries for AI/ML Programming Support

2.3 Key ML Libraries/Frameworks in Python

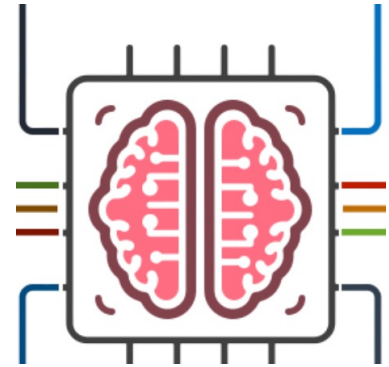
3.1 Coding Automation with AI for Catalyzing STEM Research

3.2 Prompt Engineering for Coding Automation

1.1 Artificial Intelligence: An Introduction

- Intelligence is defined as general cognitive problem solving skill.

Intelligent behaviors include: reasoning, learning, problem solving, common sense, perception, coordination, interaction, planning, expertise etc



- The word Artificial Intelligence (AI) was first used by John McCarthy at Dartmouth Conference in 1956.

John McCarthy defines AI as “The Science and Engineering of making intelligent Machines”

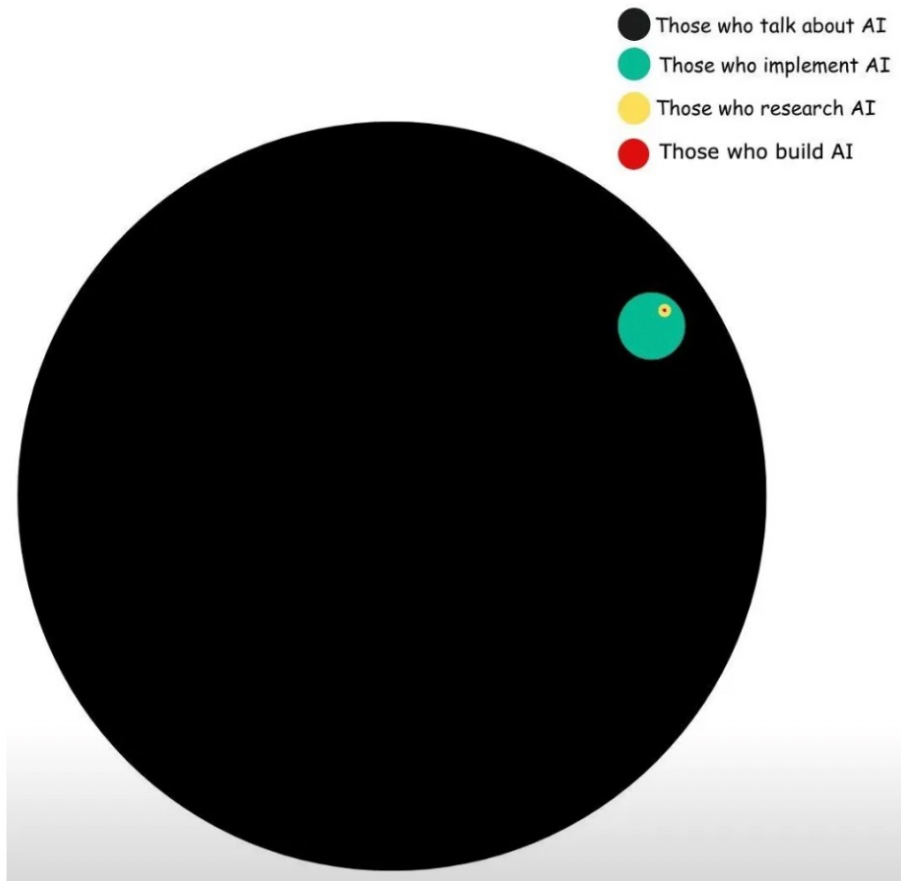


John McCarthy

1.1 Cont'd

“Just as electricity transformed almost everything 100 years ago, today I actually have a hard time thinking of an industry that I don’t think AI will transform in the next several years”

- **Andrew Ng**

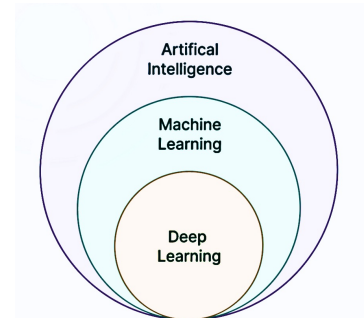


Source: www.medium.com

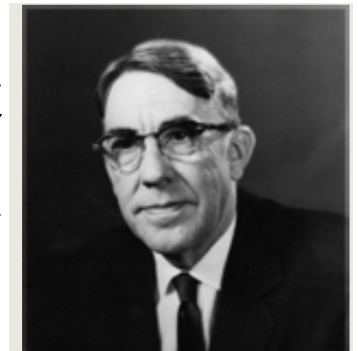
1.2 Machine Learning Basics

- Machine Learning(ML) is a subfield of Artificial Intelligence (AI).

A ML algorithm is an algorithm that is able to learn from data.



- Formally, **Arthur Samuel(1959)** defined ML as “the field of study that gives computers the ability to *learn* without being explicitly programmed.”



Arthur Samuel

1.3 Machine Learning Tasks

i) Binary and Multiclass Classifications

- *Binary classification(e.g. spam detection, gender categorisation etc).*
- *Multiclass classification(e.g. animal identification, disease diagnosis, sentiment analysis, text classification etc).*

ii) Regression

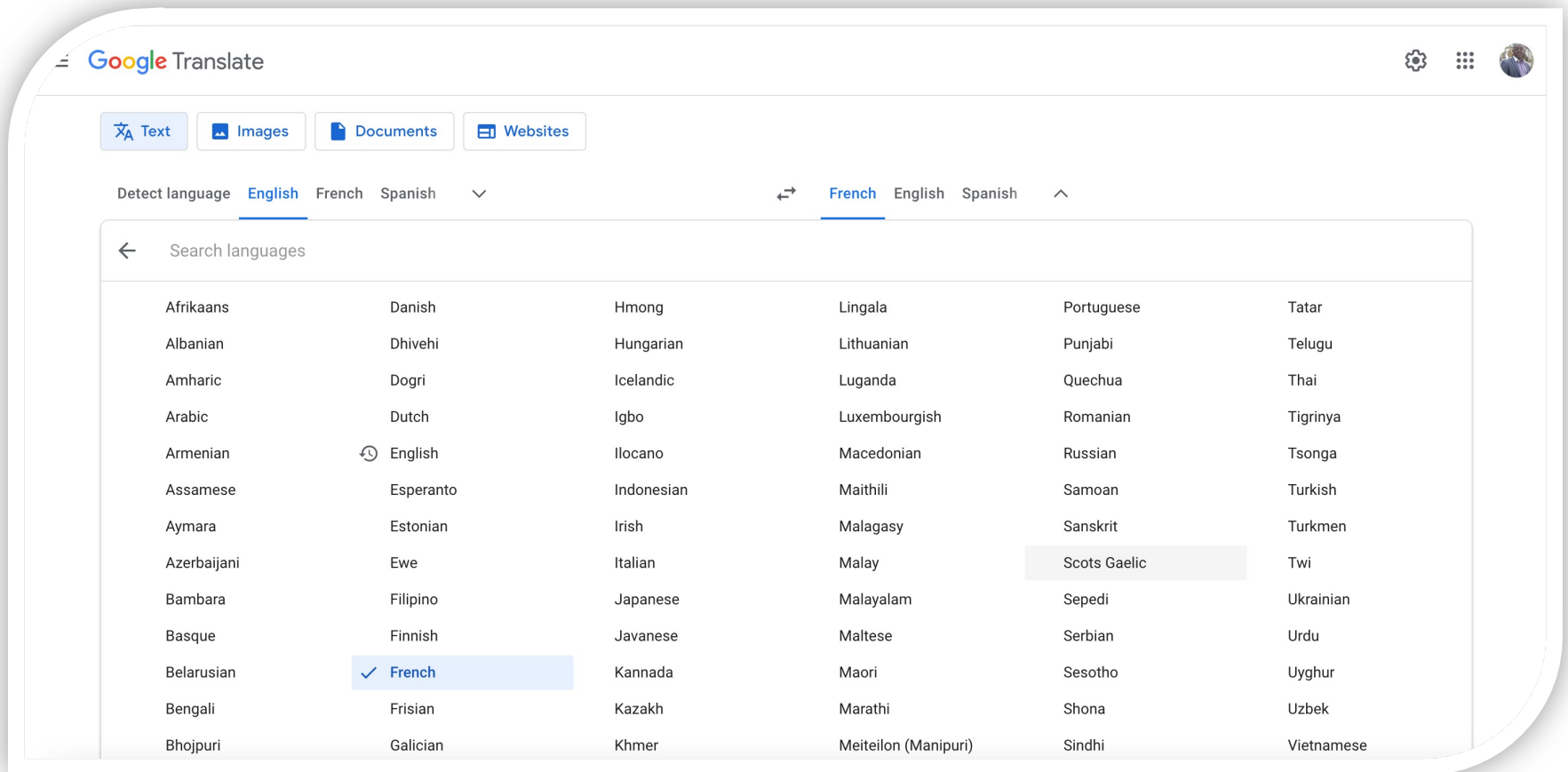
- *Given some inputs, the ML model predicts a numerical value.
(e.g. weather forecasting, stock price prediction).*

iii) Transcription

- *The ML model transcribes relatively unstructured data into discrete textual form (e.g. optical character recognition, speech recognition). A sample online platforms for OCR is <https://www.deepl.com> while a robust platform for speech recognition is <https://www.rev.ai/>).*

v) Language Translation

*This involves the conversion of sequence of symbols in **one language**(e.g. English) to sequence of symbol **other languages** (e.g. French, Yoruba, isiZulu etc). Google translate(<https://translate.google.com>) currently supports translation among 133 languages.*



Selected Languages in Google Translate

vi) Image or Video Synthesis

This involves the generation of new examples that are similar to those in the training data by the ML model. (e.g. Image or Video Synthesis). <https://deepfakesweb.com/>, is a web app for creating Deepfakes.



Fake Donald Trumps Images/Videos Generated with DeepFake AI

(Source: <https://www.youtube.com/watch?v=7ZVdb86upR4>)

2.1 ML Datasets

- Understanding the **dataset** you are working with and how it relates to the ML task you want to solve is critical.
- *Dataset collection and archiving are major components of ML model development for different tasks.*
- *Few of the repositories where datasets are archived for ML tasks are hereafter presented.*

Table 1.0: Selected ML Datasets

ML Data Repositories	Description	URL
<i>IEEE DataPort</i>	By the Institute of Electrical and Electronics Engineers (IEEE) that offers researchers and data scientists a place to store, share, and access datasets across various domains.	https://iee-dataport.org
<i>Google Dataset Search</i>	By Google to help researchers, scientists, and data enthusiasts to discover datasets from the internet.	https://datasetsearch.research.google.com

Table 1.0: Selected ML Datasets – Cont'd

ML Data Repositories	Description	URL
<i>UCI Machine Learning Repository</i>	A collection of datasets developed and maintained by the University of California, Irvine.	https://archive.ics.uci.edu/ml/index.php
<i>Kaggle</i>	It is a platform for AI competitions with datasets contributed by the community. Contains over 13 million registered users	https://www.kaggle.com/datasets
<i>Data.gov</i>	U.S. government's open data portal with access to huge public datasets.	https://data.gov/
<i>Awesome Public Datasets GitHub Repository</i>	A high-quality datasets on GitHub for various ML tasks.	https://github.com/awesomedata/awesome-public-datasets
<i>European Data Portal:</i>	Contains published European public administrations dataset.	https://data.europa.eu/en
<i>Microsoft Research Open Data:</i>	Provided by Microsoft Research.	https://www.microsoft.com/en-us/research/tools/
<i>AWS Public Datasets</i>	Amazon Web Services (AWS) public datasets	https://registry.opendata.aws/

2.2 Python Standard Libraries for AI/ML Programming Support

i) NumPy

A primary library for scientific computing in Python. It contains functionality for multidimensional arrays, mathematical operation, statistical operation and etc.

Example 2.1

***#An Example to Demonstrate Creation and Array Operations
#with NumPy***

```
import numpy as np  
  
# Create and print a 2 x 3 array  
  
arrX = np.array([[1, 2, 3], [4, 5, 6]])  
  
print("\narrX = :\n",format(arrX))
```

```
# Create and print a 4 x 4 Identity Matrix using numpy array  
arrEye = np.eye(4)  
print("\nA 4x4 Identity Matrix:\n",format(arrEye))
```

```
# Create and print a 5 element array filled with 1s  
arrY = np.ones(5)  
print("\narrY = :\n",format(arrY))
```

```
#Print the shape of arrX  
arrX_Shape = np.shape(arrX)  
print("\nThe shape of arrX is: ", arrX_Shape)
```

```
#Print the shape of arrY  
arrY_Shape = np.shape(arrY)  
print("\nThe shape of arrY is: ", arrY_Shape)
```

```
#Print the shape of arrEye  
arrEye_Shape = np.shape(arrEye)  
print("\nThe shape of arrY is: ", arrEye_Shape)
```

Example 2.2

#An Example to Demonstrate Mathematical and Statistical #Operations with NumPy

#Define 2 1D Arrays

```
arrA = np.array([10, 20, 30]); arrB = np.array([40, 50, 60])
```

#Add and Print the Two Arrays

```
arrSum = np.add(arrA,arrB)
```

```
print("The Sum of arrA and arrB is: ", arrSum)
```

#Subtract arrA from arrB and Print

```
arrDiff = np.subtract(arrA,arrB)
```

```
print("\nThe Difference of arrA and arrB is: ", arrDiff)
```

#Compute the Sum, Mean and Standard Deviation of arrA Elements

```
arrASum = np.sum(arrA); arrAMean = np.mean(arrA); arrAStd = np.std(arrA);
```

```
print("\nThe Sum of arrA elements is: ", arrASum)
```

```
print("\nThe Mean of arrA elements is: ", arrAMean)
```

```
print("\nThe STD of arrA elements is: ", arrAStd)
```

ii) Matplotlib

This is the primary scientific plotting library in Python. It contains functions for line charts, histograms, scatterplots and etc. You can show figures and other plots directly in Jupyter notebook using this library.

Example 2.3

#An example to demonstrate the use of Matplotlib

import matplotlib.pyplot as plt
import numpy as np

#Generate a sequence of numbers from -20 to 20 with 100 steps

x = np.linspace(-20,20,100)

#Create a second array y as the sine of x

y = np.sin(x)

#Use the plot function to plot array x against array y

plt.plot(x,y,marker='x',color='red')

#Create a third array y as the cosine of x

$z = \text{np.cos}(x)$

#Use the plot function to plot array z against array y

$\text{plt.plot}(x, z, \text{marker}='o', \text{color}='green')$

Axes Labels

$\text{plt.xlabel}('X\text{-axis Label}', \text{fontsize}=12, \text{color}='blue')$

$\text{plt.ylabel}('Y\text{-axis Label}', \text{fontsize}=12, \text{color}='blue')$

iii) Pandas

This is a Python library for data manipulation and analysis, which provides strong support for ML tasks.

- Pandas DataFrame is a table similar to Excel spreadsheet.*
- Pandas provides different methods to operate on the table.*
- It allows each column to have unique data type.*
- It can also ingests data from varieties of file formats and databases such as Excel file, CSV and SQL.*

Example 2.4

#An example to demonstrate the use of Pandas

import pandas as pd

from IPython.display import display

#Create a Simple Dataset of People using Disctionary type

```
data = {'Name': ["Uche", "John", "Matthew", "Dorcas", "Joy"],  
        'State': ["Ondo", "Lagos", "Abia", "Niger", "Kogi"],  
        'Age': [30, 40, 32, 51, 33]  
}
```

data_pandas = pd.DataFrame(data)

#IPython.display allows fine printing of dataframes in Jupyter

display(data_pandas)

#Select all rows that have an age column greater than or equal to 35

display(data_pandas[data_pandas.Age >= 35])

2.3 Key ML Libraries/Frameworks in Python

i) Scikit-learn

- *Scikit-learn is a popular tool and one of the most prominent open source Python libraries for Machine Learning.*

Example 2.5

Loading and Visualising Wine Dataset with Scikit Learn

from sklearn.datasets import load_wine

import pandas as pd

from IPython.display import display

Fetch the Wine Dataset

wine_data = load_wine()

```
# Create a DataFrame for better visualization  
print("Tabular Presentation of the Wine Dataset:")  
wine_df=pd.DataFrame(data=wine_data.data,  
columns=wine_data.feature_names)  
#Use IPython to Display the first few rows of the DataFrame  
display(wine_df.head(100))
```

ii) TensorFlow

- *TensorFlow(TF) is a free ML framework developed by Google.*
- *It offers a powerful library, tools, and resources for for large scale machine learning projects such as **deep learning, reinforcement learning, Computer Vision (CV), Natural Language Processing (NLP)** etc.*
- *With TF, ML developers can build and deploy ML applications efficiently through a high-level Keras API,*
- *It also provides capability to run existing ML models using the TensorFlow.js.*
- *TensorFlow Lite is a mobile library for deploying models on mobile, microcontrollers and other edge devices.*
- *TensforFlow Dataset (TFDS) is a collection of datasets for use with TF, and other ML frameworks.*

Example 2.6

***##An example to demonstrate the use of TensorFlow
##Dataset (TFDS)***

TensorFlow Dataset (TFDS) can be installed with:

!pip install -q tfds-nightly tensorflow

Import the required libraries

import matplotlib.pyplot as plt

import tensorflow as tf

import tensorflow_datasets as tfds

To get the list of available dataset, use tfds.list_builders()

tfds.list_builders()

##Load the dataset with tfds.load

ds, info = tfds.load('mnist', split='train', shuffle_files=True, with_info=True)

##tfds.show_examples returns a matplotlib.figure to show selected images

fig = tfds.show_examples(ds, info)

- Part of the output of *tfds.list_builders()* is as follows:

<pre> ↳ ['abstract_reasoning', 'accentdb', 'aeslc', 'aflw2k3d', 'ag_news_subset', 'ai2_arc', 'ai2_arc_with_ir', 'amazon_us_reviews', 'anli', 'answer_equivalence', 'arc', 'asqa', 'asset', 'assin2', 'bair_robot_pushing_small', 'bccd', 'beans', 'bee_dataset', 'beir', 'big_patent', 'bigearthnet', 'billsum', 'binarized_mnist', </pre>	<pre> ↳ 'binary_alpha_digits', 'ble_wind_field', 'blimp', 'booksum', 'bool_q', 'bucc', 'c4', 'c4_wsrs', 'caltech101', 'caltech_birds2010', 'caltech_birds2011', 'cardiotox', 'cars196', 'cassava', 'cats_vs_dogs', 'celeb_a', 'celeb_a_hq', 'cfq', 'cherry_blossoms', 'chexpert', 'cifar10', 'cifar100', 'cifar100_n', </pre>	<pre> 'cifar10_1', 'cifar10_corrupted', 'cifar10_n', 'citrus_leaves', 'cityscapes', 'civil_comments', 'clevr', 'clic', 'clinc_oos', 'cmaterdb', 'cnn_dailymail', 'coco', 'coco_captions', 'coil100', 'colorectal_histology', 'colorectal_histology_large', 'common_voice', 'conll2002', 'conll2003', 'controlled_noisy_web_labels', 'coqa', 'cos_e', 'cosmos_qa', </pre> <hr/>
--	---	---

iii) PyTorch

- *It is a free and an open-source ML framework developed by **FAIR(Facebook's AI Research)** lab.*
- *It has support for Python, C++ and Java, but the the Python interface is more interactive.*
- *It can also be used for **NLP, game development** and **computer vision** tasks. It has support for GPU for fast execution of models.*

3.1 Coding Automation with AI for Catalyzing STEM Research

- Also known as AI-assisted coding, coding automation with AI technologies is revolutionizing the way researchers solve complex problems.

The pace of discovery is catalysed through coding automation tools, thereby enhancing the efficiency of scientific endeavors.

- *Selected Examples of Coding Automation Tools are:*

i) GitHub CoPilot

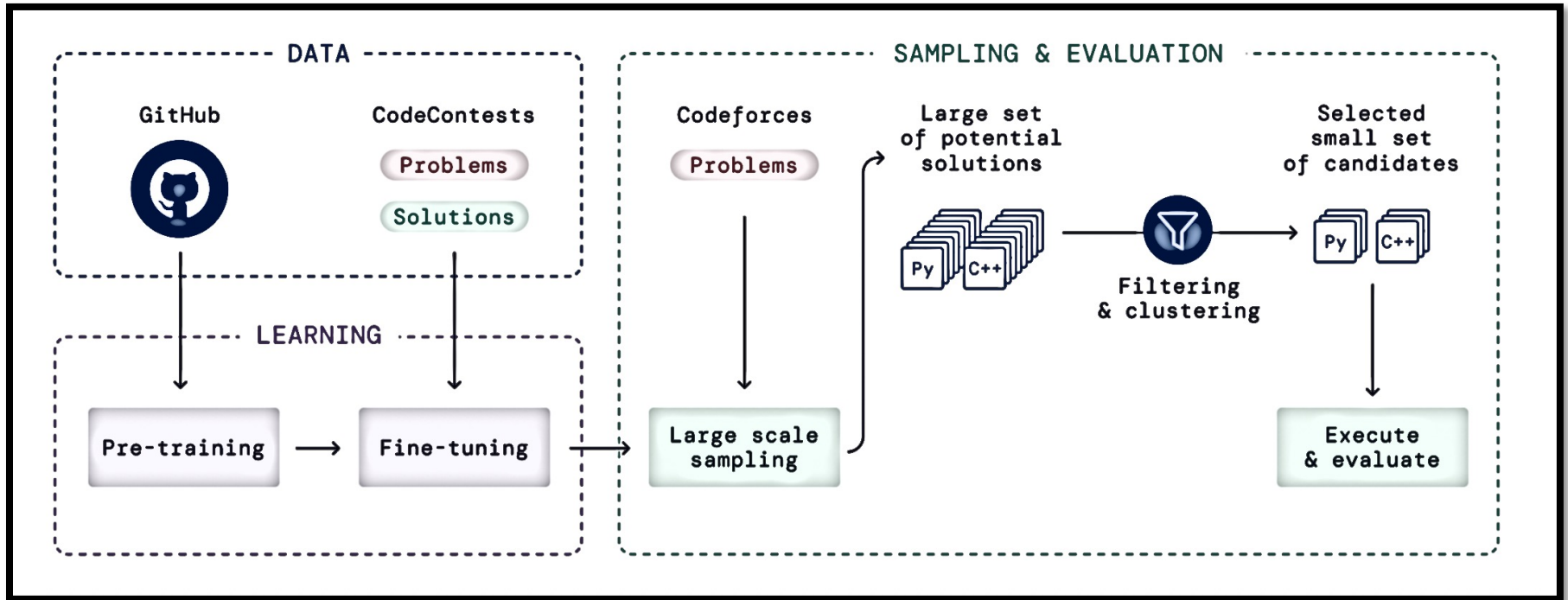
*A specific implementation of OpenAI Codex built by GitHub. It's primary focus is **code completion**, **code refactoring** and **automatic document generation**(<https://github.com/features/copilot/>).*

ii) TensorFlow Extended (TFX)

It is a powerful and versatile open source platform that was developed by Google to build, deploy, and manage production scale ML workflows. TFX seamlessly integrates with other TF libraries and tools. A comprehensive documentation on TFX is available at <https://github.com/tensorflow/tfx>

ii) AlphaCode

*It was developed by Google DeepMind for **coding automation** with design focus on competitive programming challenges. It has participated in several programming competitions achieving a rank within the top 54% of human coders. It is currently not publicly available but the publication available at <https://www.science.org/doi/10.1126/science.abq1158> and*



Development Architecture for AlphaCode

(<https://deepmind.google/discover/blog/competitive-programming-with-alphacode/>)

iii) General-Purpose Conversational AI Tools

These tools are intelligent assistants that leverage NLP and ML to generate code based on developer's instructions and specifications in addition to other general-purpose NLP tasks.

*They also handle other coding tasks such as **code completion**, **code debugging and error correction**, **code refactoring** and etc.*

The most common examples of these tools are:

- **Google Bard**

*It is build on Large Language Models (LLMs) such as **PaLM**(Pathway Language Model), **LaMDA**(Language Model for Dialogour Application), **Gemini**(a factual language model)*

- **OpenAI ChatGPT**

*The base LLMs for ChatGPT are **GPT-3**(Generative Pretrained Transformer 3), **GPT-4**. It also undergoes continuous finetuning for conversational purposes through **Training on dialogue data**, **Reinforcement Learning** (RH) as well as **Human Feedback** (HF).*

3.2 Prompt Engineering for Coding Automation

Prompt engineering involves crafting a clear and specific instruction for the LLM to generate the desired code in a desired programming language.


Example 3.1

Original Prompt:

"Write a Python program that includes a function named **calculate_simple_interest** to calculate simple interest. The function should take three parameters - **principal** (the principal amount), **rate** (the annual interest rate), and **time** (the time period in years). Additionally, implement a main section in the program where the user is prompted to input values for **principal**, **rate**, and **time**. Then, call the **calculate_simple_interest** function with the user-input values and display the result. Ensure appropriate user prompts and result formatting."

Generated Code:

python

 Copy code

```
def calculate_simple_interest(principal, rate, time):  
    """  
    Calculate simple interest.  
  
    Args:  
        principal (float): The principal amount.  
        rate (float): The annual interest rate.  
        time (float): The time period in years.  
  
    Returns:  
        float: The calculated simple interest.  
    """  
    simple_interest = (principal * rate * time) / 100  
    return simple_interest  
  
def main():  
    # Get user input  
    principal = float(input("Enter the principal amount: "))  
    rate = float(input("Enter the annual interest rate (%): "))  
    time = float(input("Enter the time period in years: "))  
  
    # Calculate simple interest using the function  
    result = calculate_simple_interest(principal, rate, time)  
  
    # Display the result  
    print(f"\nSimple Interest Calculation:")  
    print(f"Principal Amount: ${principal}")  
    print(f"Annual Interest Rate: {rate}%")  
    print(f"Time Period: {time} years")  
    print(f"Simple Interest: ${result:.2f}")  
  
# Call the main function if the script is run  
if __name__ == "__main__":  
    main()
```

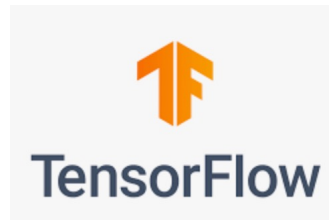
ChatGPT Generated Code Based on the Prompt

References

- Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville.
- Introduction to Machine Learning with Python: A Guide for Data Scientists by Andreas C. Muller & Sarah Guido
- <https://github.com/tensorflow/tfx>
- <https://github.com/awesomedata/awesome-public-datasets>

Thanks for Listening

Keep Coding!!!



AlphaCode



OpenAI Codex



jupyterb-gpu.fedgen.net