

Generative AI - Programming Assistant in the Classroom



Debasis Bhattacharya, JD, DBA

Associate Professor

Bryson Uehara, Student

Jen Russo, Student

University of Hawaii Maui College

debasisb@hawaii.edu

maui.hawaii.edu/cybersecurity



NCYTE
CENTER



Microsoft



Generative AI (GenAI), as championed by conversation chatbots like ChatGPT, has greatly impacted higher education for the past year or so. This presentation delves into the basics of Large Language Models (LLMs), prompt engineering and the impact of these technologies in the classroom.

Abstract

This presentation will provide a case study about how GenAI was used in an Intermediate Programming course at the the University of Hawaii Maui College in Spring 2024.

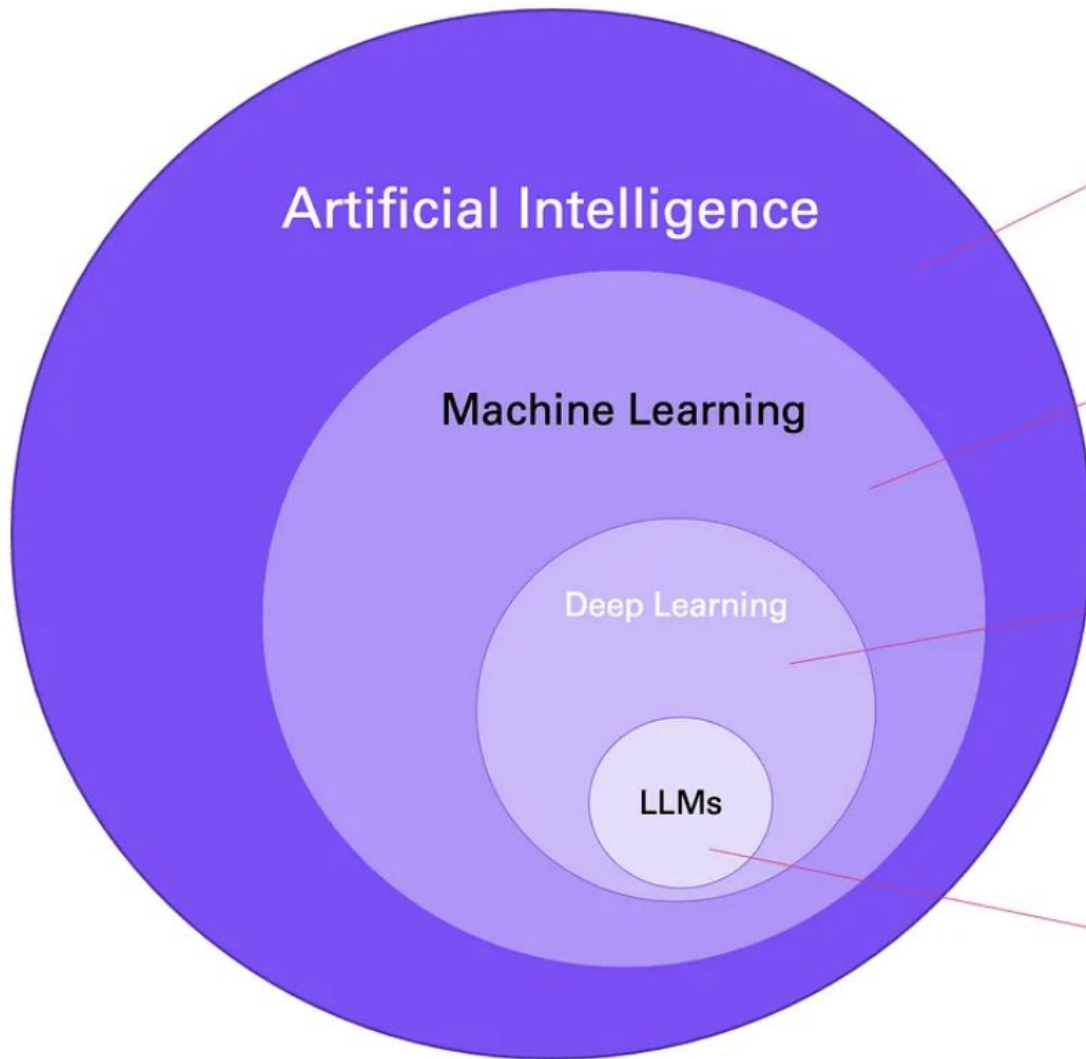
This presentation will provide the latest updates in the core features and usage of popular AI tools such as Replit, ChatGPT v4 from OpenAI etc.



Agenda

- Introduction
- Exploring Large Learning Models (LLMs)
- Impact of Generative AI in the Classroom
- Case Study - Using LLMs in a Programming Class
- Reflections
- Conclusions





Intelligent Machines

Broadly defined

Pattern Recognition

Learning general patterns from data

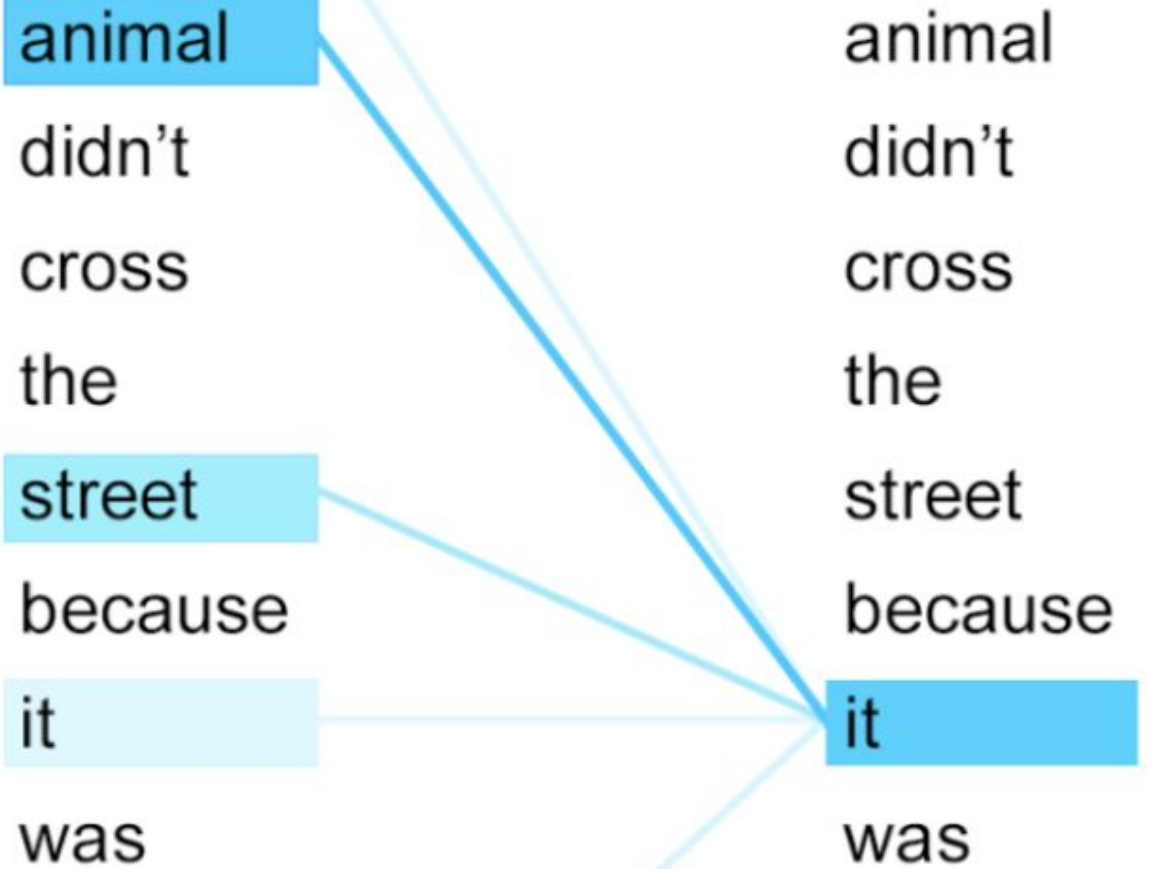
Neural Networks

Learning general patterns in **unstructured** data (i.e. images, text, audio, etc.)

Large Language Models

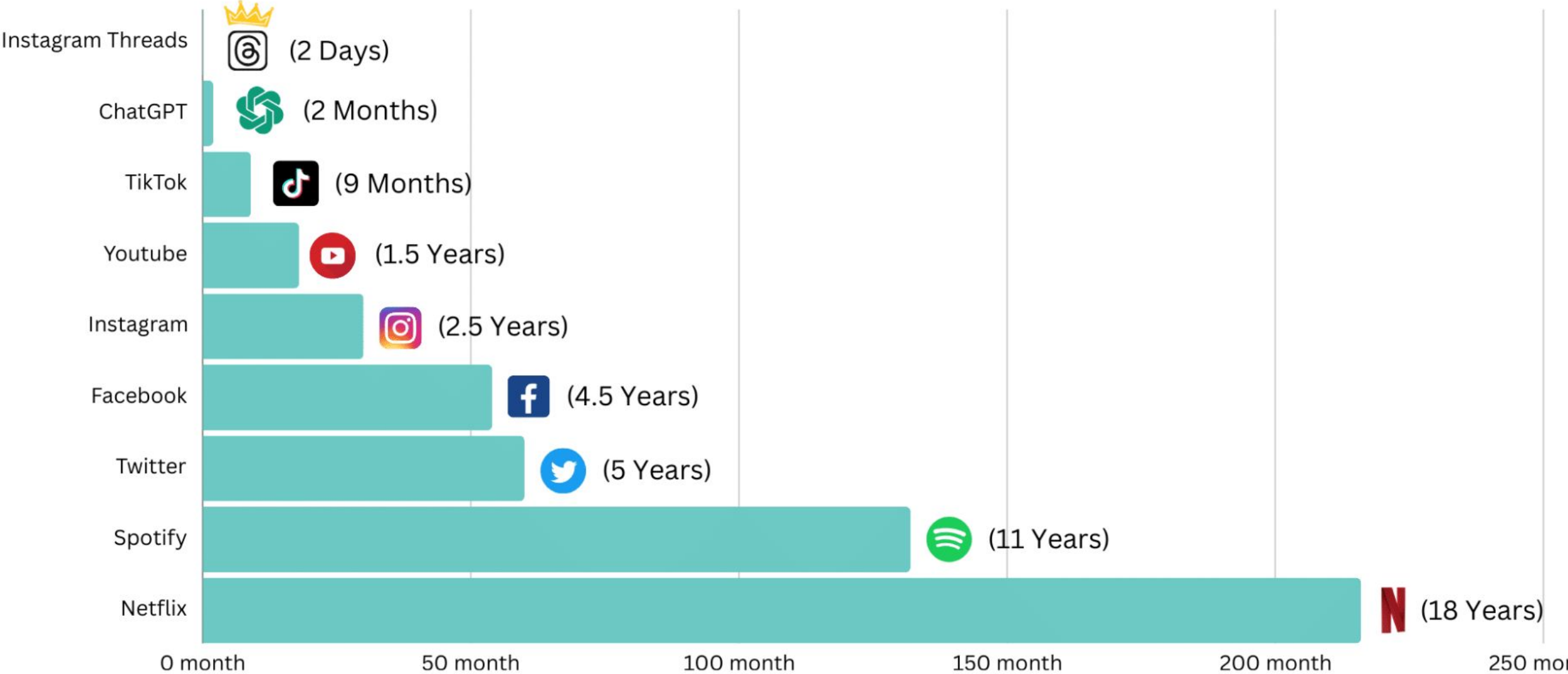
Learning to understand natural language (i.e. text)





Attention Is All You Need paper on Transformers, Vasvani et al. (2017)

Road To 100 Million Users For Various Platforms



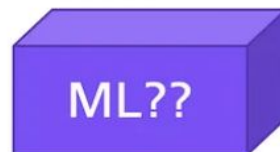
Exploring Large Language Models (LLMs)



What if the input is an image?

Classification

Is it a **tiger**, a **cat**, or a **fox**?



Class	Probability
Dog	0.03
Cat	0.96
Bird	0.01

=



=



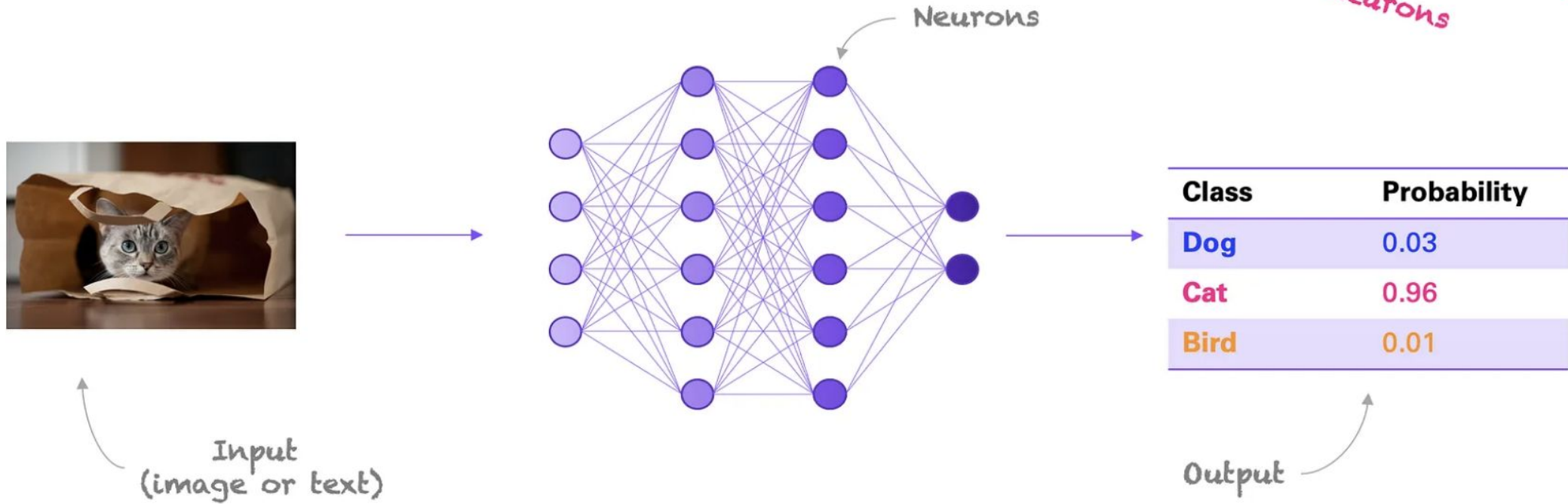
= 224x224x3 = **150,528 pixels (!!)**

Image classification example.



We need something way more powerful... **Neural Networks**

*GPT-3 has 175,000,000,000
(175B) Neurons
Brain: 100B Neurons*



Neural Networks are the most powerful Machine Learning models we have today.



Language modeling

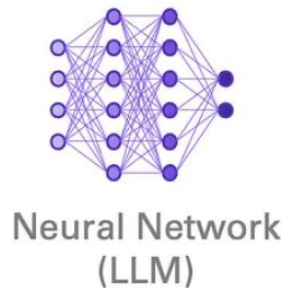
Imagine the following task: Predict the next word in a sequence

[The cat likes to sleep in the ___] → What **word** comes next?

Can we frame this as a ML problem? Yes, it's a classification task.

Now we have (say) ~50,000 classes (i.e. words)

[The cat likes to sleep in the]
Input



Word	Probability
ability	0.002
bag	0.071
box	0.085
...	...
zebra	0.001

Output

Language modeling is learning to predict the next word.



Massive training data

We can create **vast amounts of sequences** for training a language model

● Context ● Next Word ● Ignored

[The **cat** likes to sleep in the]
[The cat **likes** to sleep in the]
[The cat likes **to** sleep in the]
[The cat likes to **sleep** in the]
[The cat likes to sleep **in** the]

We do the same with much **longer sequences**. For example:

A language model is a probability distribution over sequences of words. [...] Given any sequence of words, the model predicts the **next** ...

Or also with **code**:

```
def square(number):  
    """Calculates the square of a number."""  
    return number ** 2
```

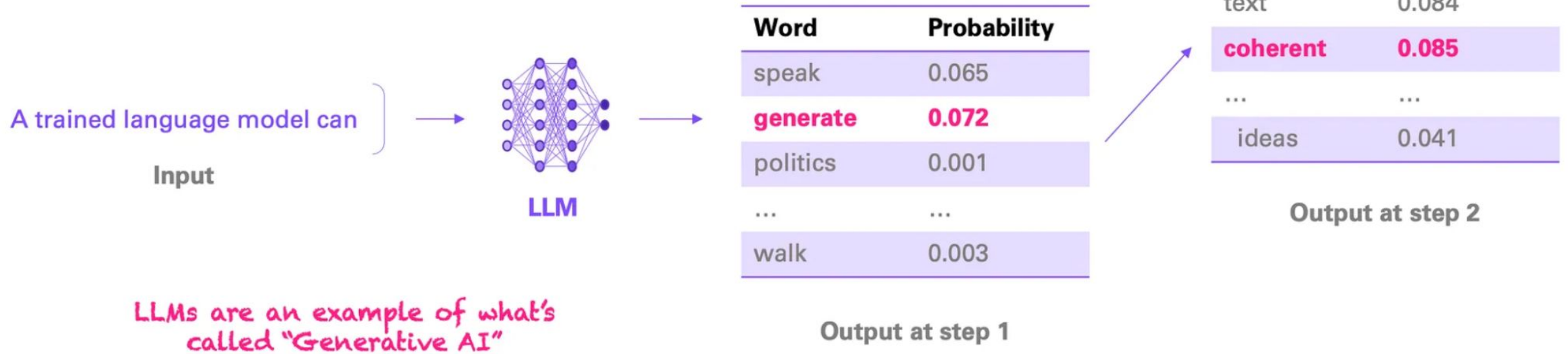
And as a result - the model becomes **incredibly good at predicting the next word** in any sequence.

Massive amounts of training data can be created relatively easily.



Natural language generation

After training: We can **generate text** by predicting **one word at a time**



What does **Generative Pre-trained Transformer (GPT)** mean

Generative

Means “next word prediction.”

As just described.

Pre-trained

The LLM is pretrained on massive amounts of text from the internet and other sources.

Transformer

The neural network architecture used (introduced in 2017).



Phases of training LLMs (GPT-3 & 4)



1. Pretraining

Massive amounts of data from the internet + books + etc.

Question: What is the problem with that?

Answer: We get a model that can babble on about anything, but it's probably not **aligned** with what we want it to do.

2. Instruction Fine-tuning

Teaching the model to respond to instructions.

Model learns to respond to instructions.

→ Helps **alignment**

"Alignment" is a hugely important research topic

3. Reinforcement Learning from Human Feedback

Similar purpose to instruction tuning.

Helps produce output that is closer to what humans want or like.



Zero-Shot Prompting

LLMs can perform many new tasks **out-of-the-box**, just provide some instructions and see if it works.



Few-Shot Learning

Providing **examples** helps the LLM understand and follow your task.

This is especially helpful to ensure a specific **output format**.



Chain-of-Thought Prompting

Ask the model to solve complex tasks **step by step**.

Why does this work?

It gives the model a **working memory**, similar to humans.

Who won the World Cup in the year before Lionel Messi was born? Think step by step.	User
Lionel Messi was born on June 24, 1987. The World Cup that took place before his birth was the 1986 World Cup. The winner of the 1986 FIFA World Cup was Argentina.	LLM



Impact of LLMs to Students



1. Personalized Learning Experiences

Generative AI enables more personalized and adaptive learning experiences for students. By analyzing individual learning patterns, preferences, and performance, AI-driven platforms can tailor content, suggest resources, and adjust difficulty levels in real time. This personalization helps meet students at their level, potentially improving understanding and retention.

2. Enhanced Accessibility of Information

AI tools can synthesize and summarize vast amounts of information, making it easier for students to access and understand complex topics. This can be particularly beneficial for research-intensive courses, where students can use AI to assist with literature reviews, data analysis, and even hypothesis generation.

3. Innovative Educational Tools

Generative AI powers a range of innovative educational tools, from AI tutors that provide instant feedback to immersive simulations that offer hands-on learning experiences in virtual environments. These tools can complement traditional learning methods and provide students with additional ways to engage with the material.

4. Improved Writing and Research Skills

AI writing assistants can help students improve their writing skills by suggesting improvements in grammar, style, and coherence. Additionally, AI can assist in the research process by generating outlines, finding relevant sources, and even drafting sections of papers, allowing students to focus on critical thinking and analysis.

Case Study : Use of Coding LLMs in a Programming Course



ICS 385 - Intro to Web Programmin g

1. Intermediate web programming course
2. Topics
 - HTML/CSS
 - JavaScript
 - NodeJS and Express
 - EJS Templates
 - GitHub
 - RESTful APIs
 - MongoDB and Mongoose
 - Authentication and Security
3. Two student TAs
 - Took course last spring without AI



Insert in Syllabus...

Use of Artificial Intelligence (AI)

- AI tools, such as coding generators from Replit, Github Copilot, and other similar tools are welcome to be used in all classroom and course assignments, unless when specifically prohibited from doing so. Students are expected to **cite and attribute their use of GenAI tools**, just like any other reference tool or source, when creating their homework or exam assignments.





.config

public

views

date.js

index.js

Packager files

package-lock.json

package.json

Config files

.replit

replit.nix

index.js

```
1 // This Express.js application creates two separate to-do lists, a fun to-do list and a weekend
  to-do list, each with pre-set arrays containing 2-3 items. The current date and a default to-do
  list with general items are shown on the main route ("/"). Users have the option of adding
  items to the work list (which is forwarded to the "/work" route) or the default list. Also,
  routes "/fun" and "/weekend" show listings designed for weekend chores and fun things to do.
2
3 // Preston Respicio
4 // 2/10/2024
5
6 const express = require("express");
7 const bodyParser = require("body-parser");
8
9 // create a date object that requires the date.js file
10 const date = require(__dirname + "/date.js");
11
12 const app = express();
13
14 // set an array for the default items in the list
15 let items = ["Wake up", "Brush teeth", "Get ready", "Start and warm up car", "Buy foods and
  drinks"];
16 // set an empty array for new work items
17 let workItems = ["Clock-in", "Curbside check-in service", "Wheelchair assistance service"];
18 // set an array for Fun and another for Weekend
19 let funItems = ["Watch anime episodes", "Play video games", "Walk the dog"];
```




Replit AI

Use Replit AI to set up new projects, answer questions about your code, and assist your thinking.

Generate

Create a personal assistant chatbot

Debug

Find bugs in index.js

Explain

How does index.js work?

Ask replit.com

How do I deploy a JavaScript project?

Type message...

Ask

README.md

Advanced

AI

Select a mode

Ask AI

Detect actions based on the prompt

Generate

Write starter code/tests

Modify

Refactor existing features

Explain

Ask a question about your code

Ask replit.com

Ask about how to do things on Replit

```
9 //displays index.html of root path
10 app.get("/", function(req, res) {
11     res.sendFile(__dirname + "/index.html"
12 });
13
14 //invoked after hitting go in the html f
15 app.post("/", function(req, res) {
```

Search actions

AI

Explain

Modify



AI just now Explain Advanced

This piece of code in the `index.js` file defines how your application responds when someone visits the root URL (`/`) of your website. Let's break it down into simpler parts:

- `app.get("/", function(req, res) {...});`: This tells your web server (created with Express, a web framework for Node.js) to listen for GET requests on the root URL (`/`). A GET request is what browsers typically send when a user tries to visit a webpage.
- `function(req, res) {...}`: This is a callback function that gets executed when a user hits the root URL. It takes two arguments:
 - `req` (short for request): This object contains all the information about the request made to the server, like any data the user sends.
 - `res` (short for response): This object is used to send back a response to the user. You fill it with what you want the user to receive.
- `res.sendFile(__dirname + "/index.html")`: Inside the callback function, `res.sendFile` is used to send a file back to the user's browser. `__dirname` is a built-in variable in Node.js that gives you the absolute path of the directory containing the current executing script. So, `__dirname + "/index.html"` creates a path to the `index.html` file located in the same directory as your `index.js` script.

In summary, this piece of code serves the `index.html` page to users when they visit the root (`/`) of your website. It's like telling your web server, "Whenever someone comes to the entrance (`/`), show them the `index.html` page."

```
// Generate code to compute the volume of a sphere
```

Generate code to compute the volume of a sphere



 Generate 

Reject

Accept

```
// Function to compute the volume of a sphere given its radius
function computeSphereVolume(radius) {
  return (4/3) * Math.PI * Math.pow(radius, 3);
}
```

Write test scripts to test above code to compute the volume of a sphere

 Generate 

Reject

Accept

```
// Test script for computeSphereVolume function
function testComputeSphereVolume() {
  const radius = 5; // Example radius
  const expectedVolume = (4/3) * Math.PI * Math.pow(radius, 3);
  const calculatedVolume = computeSphereVolume(radius);

  if (calculatedVolume === expectedVolume) {
    console.log(`Test Passed: Volume for radius ${radius} is correctly
  calculated as ${calculatedVolume}.`);
  } else {
    console.error(`Test Failed: Expected volume ${expectedVolume}, but
  got ${calculatedVolume}.`);
  }
}

testComputeSphereVolume();
```



Reflections!



Students

Pros

- Design, write pseudocode
- Explain code snippets
- Generate Code, sometimes unfamiliar!
- Debug code, explain bugs
- Create Auto Test Scripts
- Helps in documentation
- Document AI section!

Cons

- Generate Code, sometimes unfamiliar!
- Code works but students don't how...
- Trial and Error Approach
- Difficulty in Explaining Code
- Students who have less programming experience have more difficulty with AI



Instructor and TAs

Pros

Provides another point of view for students to learn

Students can generate working code easier

Helps students who are unfamiliar with setup and installation over Zoom

Students can turn to help for LLMs before asking/emails

Students can focus on design and tests more than before!

Cons

Over reliance on LLMs vs. learning by writing code

Beginners take shortcuts

Cheating

Homework Assignments

Exams

Harder to give partial credit

Unsure what students did vs. done by code generation!

Open book exams now have *images embedded in PDFs*





Here is a cartoon version of the image you provided.



Questions?
Comments...Discussions!
debasisb@hawaii.edu
maui.hawaii.edu/cybersecurity



Workshop Evaluation

NCyTE to add instructions / QR code

Be sure to fill out sign-in sheets before you leave!



NCyTE
CENTER



Microsoft

