



DeepLearning.AI

# Software Applications

---

## Using generative AI in software applications

# Examples of software applications

## Writing

Is there parking for staff?

Yes, employees can park on levels 1 and 2 of the office lot. You can get a sticker at [...]



## Reading

Classify the sentiment of the following review as either positive or negative:

*The mochi is excellent!*

Positive



## Chatting

Welcome to BettaBurgers!

I'd like a cheeseburger for delivery!

Awesome. Anything else?

That's it thanks!

Ok, it'll be there in 20 minutes. Enjoy!



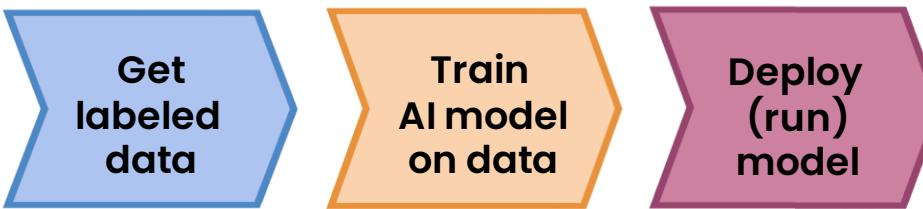
```
1 import PDF Studio - PDF Editor for Mac, Windows, Linux. For Evaluation. https://www.qoppa.com/pdfstudio
2 from torch.utils.data import DataLoader, TensorDataset
3 from torch import nn
4
5 class SentimentLSTM(nn.Module):
6     def __init__(self, vocab_size, output_size, embedding_dim, hidden_dim, n_layers):
7         super(SentimentLSTM, self).__init__()
8         self.output_size = output_size
9         self.n_layers = n_layers
10        self.hidden_dim = hidden_dim
11
12        self.embedding = nn.Embedding(vocab_size, embedding_dim)
13        self.lstm = nn.LSTM(embedding_dim, hidden_dim, n_layers, batch_first=True)
14        self.fc = nn.Linear(hidden_dim, output_size)
15        self.sigmoid = nn.Sigmoid()
16
17    def forward(self, x, hidden):
18        batch_size = x.size(0)
19        x = self.embedding(x)
20        lstm_out, hidden = self.lstm(x, hidden)
21        lstm_out = lstm_out.contiguous().view(-1, self.hidden_dim)
22        out = self.fc(lstm_out)
23        out = self.sigmoid(out)
24        out = out.view(batch_size, -1)
25        out = out[:, -1]
26        return out, hidden
27
28    def init_hidden(self, batch_size):
29        weight = next(self.parameters()).data
30        hidden = (weight.new(self.n_layers, batch_size, self.hidden_dim).zero_(),
31                  weight.new(self.n_layers, batch_size, self.hidden_dim).zero_())
32
33        return hidden
```

Write the public review  
code

# Supervised learning for **restaurant reputation monitoring**

Input → Output  
A → B

Input (A)	Output (B)
Best soup dumplings I've ever eaten.	Positive
The colorful tablecloths made me smile!	Positive
Not worth the 3 month wait for a reservation	Negative



Input: "Best bubble tea I've ever had"

Output: Positive

# Prompt-based development

```
prompt = """  
    Classify the following review  
    as having either a positive or  
    negative sentiment:  
"""
```

The banana pudding was really  
tasty!

```
"""
```

```
response = llm_response(prompt)  
print(response)
```

Instruction text

Review text

Code to call LLM

Code to print output

## LLM Model (Currently available)

The model architecture that LLMs are built on is primarily the **Transformer** model. This architecture, introduced in the 2017 paper "Attention Is All You Need," revolutionized natural language processing (NLP) by introducing the **self-attention mechanism**. This mechanism allows the model to process all parts of an input sequence simultaneously, rather than sequentially, which was a limitation of older models like recurrent neural networks (RNNs).

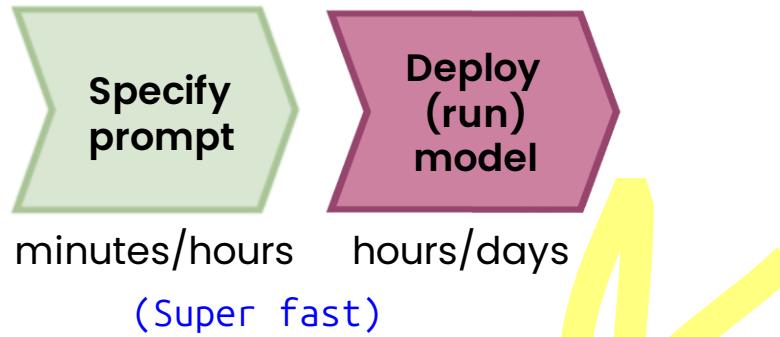


# Workflow using Generative AI

Supervised learning



Prompt-based AI



That material explains the shift in building software applications, specifically **sentiment classification**, using **traditional supervised machine learning** versus the newer **prompt-based development** approach with **Generative AI** (Large Language Models).

## Generative AI in Software Applications: Simply Explained

### The Goal: Classifying Restaurant Reviews (Sentiment Analysis)

Imagine you have a bunch of restaurant reviews (like "Best soup dumplings I've ever eaten" or "Not worth the three-month wait") and you want a computer to tell you if each one is **positive** or **negative**. This is called sentiment analysis.

## 1. The Old Way: Supervised Learning (Like Teaching a Kid from Scratch)

Before Generative AI, getting the computer to do this was a **long, difficult process**:

Concept	Explanation (Simplified)	Duration
<b>Data Collection &amp; Labeling</b>	You needed a team to collect thousands of reviews and manually label each one as 'positive' or 'negative'. This is like providing hundreds of flashcards to teach the difference.	≈ 1 Month
<b>Model Training</b>	You needed specialized AI/Machine Learning engineers to write pages of complex software <b>code and "train"</b> an AI model specifically on those labeled examples. It had to learn from scratch what 'good' and 'bad' words meant in a review.	≈ 3 Months
<b>Deployment</b>	You needed another team to deploy and run this specialized model on a cloud service, ensuring it runs reliably 24/7.	≈ 3 Months
<b>The Cost</b>	This approach required expert engineers, months of time ( <b>6 – 12 months total</b> ), and lots of custom code.	

## 2. The New Way: Prompt-Based Development (Using a Pre-Trained Genius)

Generative AI (like ChatGPT, or the models built into applications) works differently. It's already been trained **on a massive amount of the internet**, so it already understands language, sentiment, and context.

Concept	Explanation (Simplified)	Duration
<b>Prompting</b>	Instead of giving it thousands of examples, you just write a <b>simple instruction</b> (a "prompt") to the Generative AI. For example: "Classify the following review as having either a positive or negative sentiment: [Review Text Here]."	Minutes to Hours
<b>One Line of Code</b>	You write just <b>one line of code</b> to send that prompt to the massive Generative AI model. <b>The pre-trained model handles the rest.</b>	Hours to Days
<b>The Benefit</b>	The process is reduced to specifying the task and calling the existing model. <b>The time frame shrinks from 6-12 months to just days or a week!</b> This massive reduction in effort means many more people can now build powerful AI applications.	



DeepLearning.AI

# Software Applications

---

Trying generative AI code  
yourself (optional)

# Coding platform (Jupyter notebook)

Exercise 1

The one command to know:

shift + enter

The screenshot shows a Jupyter notebook interface with the title "jupyter GENAI4E\_Activity1". The toolbar includes "Menu", "Not Trusted", "Python 3 (ipykernel)", and "Logout". Below the toolbar are buttons for file operations, cell selection, and run. A dropdown menu is open, showing "Run" and "Markdown".

A code cell titled "Prompting an LLM in code" contains the following instructions:

1. For each of the two code cells below, click on the cell then hit Shift+Enter on your keyboard to run the code (or if on a mobile device, press 'play' button).
2. Optionally edit the prompt, and try again!

The code cell content is:

```
In [ ]:  
import openai  
import os  
  
openai.api_key = os.getenv("OPENAI_API_KEY")
```

A yellow diagonal highlight starts from the top-left of the code cell and extends towards the bottom-right of the slide.

To the right of the notebook is a video player window. The video title is "Generative AI for Everyone" and the subtitle is "Trying generative AI code yourself (optional)". A blue line points from the text "Play button" to the play/pause button in the video player's control bar. The video player also shows a progress bar at 0:00 / 3:03.

This short, optional video is a **summary of the next step** in the course materials.

It introduces a **hands-on coding exercise** that uses **Generative AI** for sentiment classification (like the restaurant review example discussed previously).

## Summary of the Optional Video

The video invites you to click the link for the next course item, which will take you to the [DeepLearning.AI platform](#).

- **What you'll see:** A user interface with **code on the left** and a video player on the right.
- **What you'll do:** The host will walk you through the code step-by-step in the video on the right. Your only task is to press **Shift + Enter** in the coding environment on the left to run the code.
- **The Goal:** To quickly see how the simple, **prompt-based code** works to classify sentiment (e.g., identifying that "The banana pudding was very tasty" has a **positive sentiment**).
- **Note:** The exercise is **optional** and not required to complete the course, but it's encouraged for a practical demonstration.



DeepLearning.AI

# Software Applications

---

## Lifecycle of a generative AI project

# Lifecycle of a generative AI project

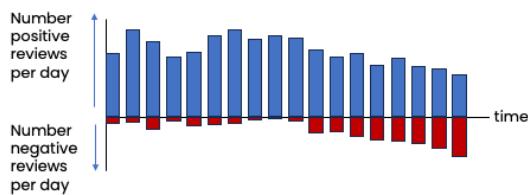
Scope project



Build/improve system



Internal evaluation



Initially a prototype,  
that we will improve  
over time

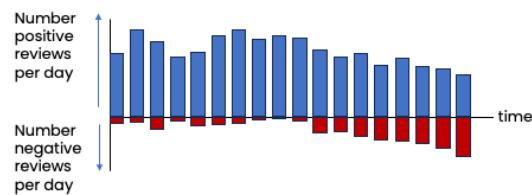
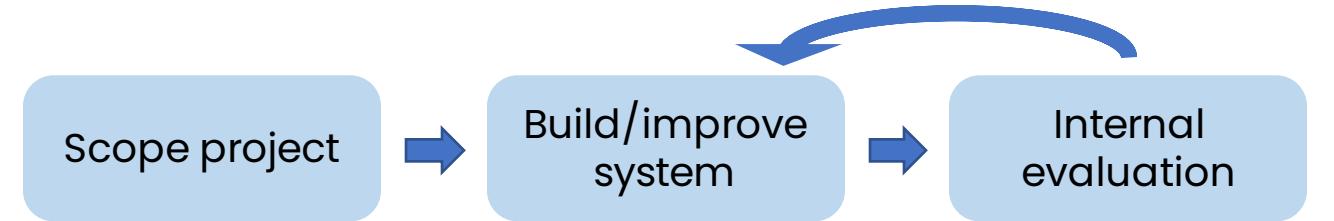


Classify the sentiment  
of the following review  
as either positive or  
negative:

*The custard tart was  
amazing!*

Positive

# Lifecycle of a generative AI project



Initially a prototype, that we will improve over time



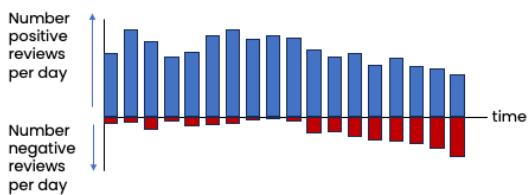
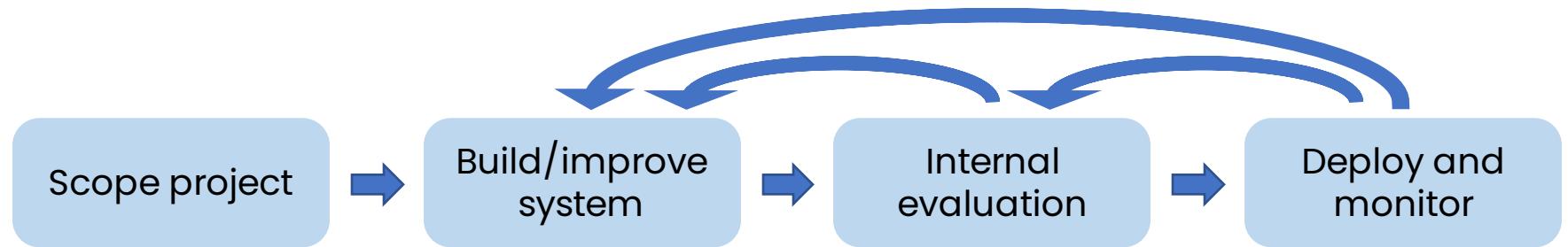
Classify the sentiment of the following review as either positive or negative:

*My pasta was cold*

Positive

✖ Incorrect response

# Lifecycle of a generative AI project



Initially a prototype, that we will improve over time



Classify the sentiment of the following review as either positive or negative:

*My miso ramen tasted like tonkotsu ramen.*

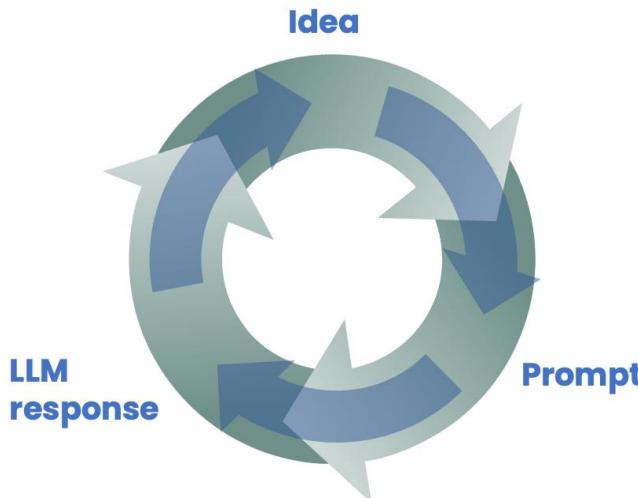
Positive

(Should be -ve Ramen)

# Tools to improve performance

Building Generative AI is a **highly empirical (experimental)** process – we repeatedly find and fix mistakes.

- Prompting



# Tools to improve performance

Building Generative AI is a highly empirical (experimental) process – we repeatedly find and fix mistakes.

- Prompting
- **Retrieval augmented generation (RAG)**
  - Give LLM access to external data sources



# Tools to improve performance

Building Generative AI is a highly empirical (experimental) process – we repeatedly find and fix mistakes.

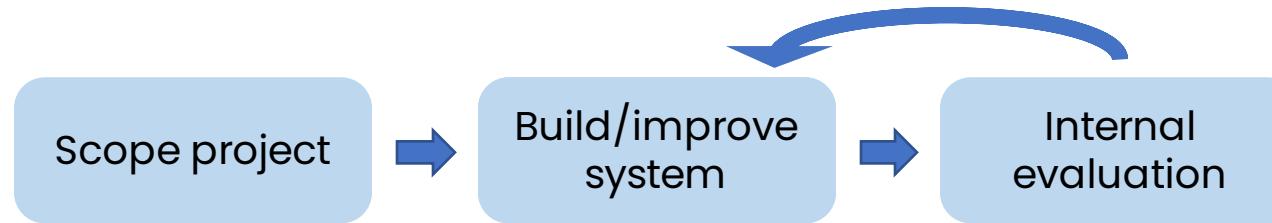
- Prompting
- Retrieval augmented generation (RAG)
  - Give LLM access to external data sources
- Fine-tune models
  - Adapt LLM to your task

# Tools to improve performance

Building Generative AI is a highly empirical (experimental) process – we repeatedly find and fix mistakes.

- Prompting
- Retrieval augmented generation (RAG)
  - Give LLM access to external data sources
- Fine-tune models
  - Adapt LLM to your task
- Pretrain models
  - Train LLM from scratch

# Lifecycle of a generative AI project



Take food  
order



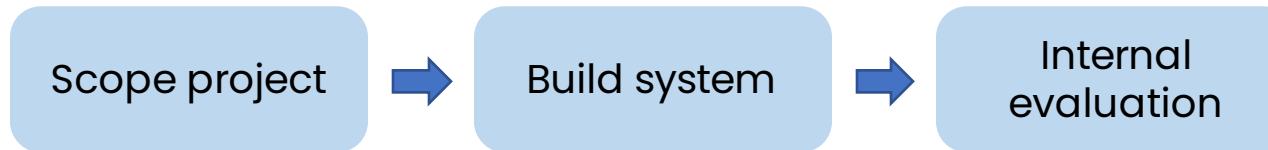
Let team  
order...

I'd like a mushroom  
and swiss burger

I'm sorry, we don't  
have mushrooms, how  
about onions?

✖ Incorrect  
response

# Lifecycle of a generative AI project



Take food  
order

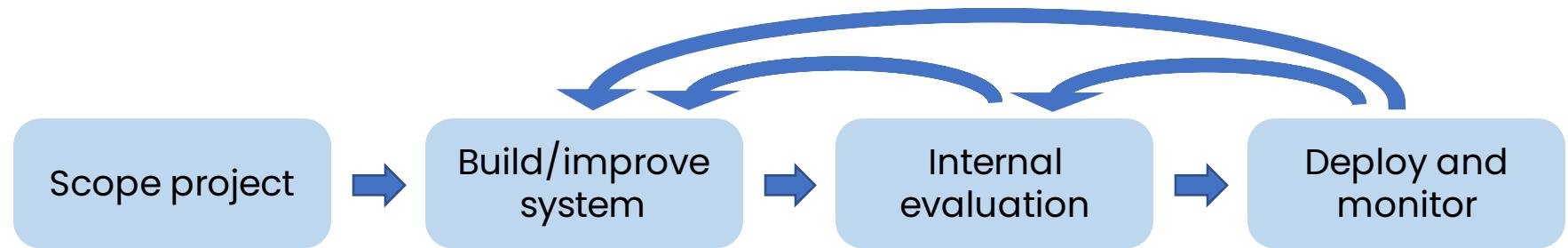


Let team  
order...

Do you have pickles?

Yes – would you like to  
add some?

# Lifecycle of a generative AI project



Take food  
order

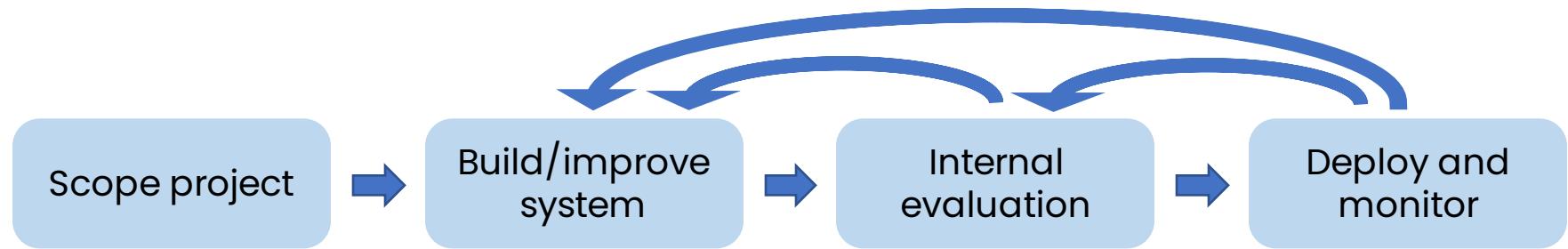


(Using RAG to give the correct answer on  
cheeseburger's calories)

Let customers  
order, monitor  
LLM responses

How many calories in  
the cheeseburger?  
I'm sorry, I don't know

# Lifecycle of a generative AI project



Take food  
order



Let customers  
order, monitor  
LLM responses



How many calories in  
the cheeseburger?

About 375 calories

This video summarizes the **highly iterative and empirical (experimental)** lifecycle for building a Generative AI software application, focusing on how easily prototypes are built and then continually improved through testing.

## The Generative AI Project Lifecycle: Summary

The process of building a Generative AI application, like a reputation monitor or a food ordering chatbot, is characterized by its speed and constant refinement.

Phase	Description	Key Characteristic
<b>1. Scoping</b>	Define the software's objective (e.g., build a restaurant reputation monitoring system).	Clear goal setting.
<b>2. Implementation &amp; Prototype</b>	Quickly build an initial working prototype, often in <b>1-2 days</b> , due to the simplicity of prompt-based development.	<b>Very fast initial build.</b>
<b>3. Internal Evaluation</b>	The internal team tests the prototype with various inputs (e.g., review: "My pasta was cold"). Problems are discovered (e.g., system incorrectly tags it as a positive sentiment).	<b>Discovery of initial errors.</b>
<b>4. Improvement Loop (Iterate)</b>	Based on internal mistakes, the team returns to improve the system, often by refining the <b>prompt</b> . This is a highly iterative, <b>empirical (experimental)</b> process of trial and error.	<b>Highly iterative prompting.</b>

## 5. Deployment & Monitoring

Once the system is reliable enough, it's deployed to external users. Continuous monitoring reveals new, unexpected errors (e.g., "My miso ramen tasted like tonkotsu ramen" — **is this good or bad?**).

**Real-world error discovery.**

## 6. Advanced Improvement

External errors feed back into the evaluation and improvement loop. The system can be refined using more advanced techniques (which will be covered later in the course):

**Systematic refinement.**

- **RAG** (Retrieval-Augmented Generation): Giving the model access to your own proprietary or external documents (e.g., a menu or calorie list) so it can answer domain-specific questions (like "How many calories are in the burger?").

- **Fine-Tuning**: Adapting the base Large Language Model to your specific task and data.

- **Pretraining**: Training a model from scratch (the most resource-intensive method).

**Core Takeaway:** Building Generative AI software is a fast, highly experimental process of **try-something, find-mistakes, and fix-mistakes** (an empirical approach). The speed of initial prototyping is a huge advantage over traditional software development.



DeepLearning.AI

# Software Applications

---

## Cost intuition

# How much does it cost?

## Example prices

	OpenAI/GPT3.5	OpenAI/GPT4	Google/PaLM 2	Amazon/Titan Lite
Input tokens	\$0.0015/1K tokens	\$0.03/1K tokens	\$0.0005/1K tokens	\$0.0003/1K tokens
Output tokens <i>(loosely words)</i>	\$0.002/1K tokens	\$0.06/1K tokens	\$0.0005/1K tokens	\$0.0004/1K tokens

## What is a token?

the example Andrew	1 token
sub parts translate programming	2 tokens
tonkotsu	4 tokens

300 words  
400 tokens

Roughly, 1 token = 3/4 words

# Estimating Cost

Typical adult reading speed: 250 words/minute.

How much would it cost to keep someone occupied for 1 hour?

$$\begin{array}{r} 60 \times 250 = 15,000 \leftarrow \text{output} \\ + 15,000 \leftarrow \text{prompt} \\ \hline 30,000 \\ \hookrightarrow \underline{40,000} \text{ tokens} \end{array}$$

1 token  $\approx \frac{3}{4}$  word

Cost: \$0.002 cents / 1K tokens.

$$\$0.002 \times 40 = \boxed{\$0.08}$$



## Cost Intuition for Using LLMs

### 1. The Cost Metric: Tokens

- **Definition:** LLMs are **priced based on tokens**. A token is loosely equivalent to a word or a sub-part of a word (e.g., "the" = 1 token, "translate" = 2 tokens).
- **Conversion:** On average, **1 token  $\approx$  3/4 of a word**. Therefore, the number of tokens is about 33% more than the number of words (e.g., 300 words  $\approx$  400 tokens).
- **Pricing:** LLM providers charge per 1,000 tokens.
  - **Input (Prompt) Cost:** The cost of the text you send to the model is usually cheaper.
  - **Output (Response) Cost:** The cost of the text the model generates is the main cost driver. (Example prices used: GPT-3.5  $\approx$  0.2 cents per 1,000 tokens; GPT-4  $\approx$  6 cents per 1,000 tokens).

## 2. The Cost Calculation: An Hour of Reading

The video provides a simple thought experiment to estimate the cost of generating enough text to keep a team member occupied for one hour, using the inexpensive GPT-3.5 price of **0.2 cents per 1,000 tokens**.

Step	Calculation	Result
<b>Words to keep someone busy for 1 hour</b>	$250 \text{ words/minute} \times 60 \text{ minutes}$	<b>15,000</b> words (Output)
<b>Total Words to Pay For</b>	$15,000 \text{ (Output)} + 15,000 \text{ (Assumed Input/Prompt)}$	<b>30,000</b> words total
<b>Total Tokens to Pay For</b>	$30,000 \text{ words} \approx 30,000 / 0.75$	<b>40,000</b> tokens total
<b>Total Cost</b>	$40 \text{ (thousands of tokens)} \times \$0.002 \text{ (cost per 1k tokens)}$	<b>8 cents (\$0.08)</b>

### 3. Conclusion on Cost

The final intuition is that using a large language model hosted on the cloud to help a person with their work is **relatively inexpensive** (only about **8 cents per hour** of intense use in the cheap scenario). This small incremental cost often justifies the potential increase in productivity. However, this cost could still become very expensive for free products with millions of users.



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

## Retrieval Augmented Generation (RAG)

**RAG**, or **Retrieval-Augmented Generation**, is an architectural framework that enhances the performance of **large language models (LLMs)** by giving them access to external, authoritative knowledge sources. It's a way to provide an LLM with relevant, up-to-date information that was not part of its original training data.

# Retrieval Augmented Generation (RAG) example

## General Chatbot

Is there parking for employees?

I need more specific information about your workplace to answer that question.

## Chatbot with RAG

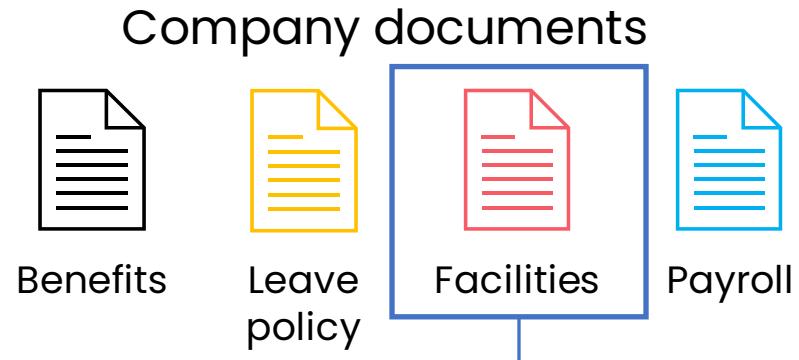
Is there parking for employees?

Yes, employees can park on levels 1 and 2 of the office lot. You can get a sticker at [...]

# Retrieval Augmented Generation (RAG) example

- Given question, search relevant documents for answer

Is there parking for employees?



- Incorporate retrieved text into an updated prompt

Use the following pieces of context to answer the question at the end.

Parking Policy: All employees may park on levels 1 and 2 of the office lot. Use the entrance on Front St [...]

Is there parking for employees?

# Retrieval Augmented Generation (RAG) example

3. Generate answer from the new prompt **with additional context**

Use the following pieces of context to answer the question at the end.

Parking Policy: All employees may park on levels 1 and 2 of the office lot. Use the entrance on Front St [...]

Is there parking for employees?

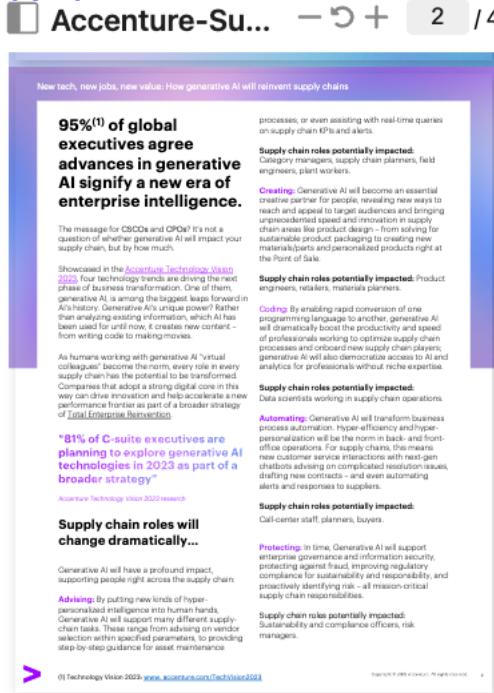
Yes, employees can park on [...] ① → Link to full onboarding policy

# Examples of RAG applications

Upload pdf and asked questions from within

## Chat with PDF files





**Chat**

What are some roles in supply chain that will be changed by AI?

Generative AI will have a profound impact on supply chain roles.

According to ②, some of the roles that will be potentially impacted include data scientists, call-center staff, planners, buyers, sustainability and compliance officers, risk managers, category managers, supply chain planners, field engineers, plant workers, product engineers, retailers, and materials planners. ②

Ask any question... ➤

Chatting with an Accenture report using ChatPDF

Andrew Ng

# Examples of RAG applications

Answer questions based on a website's articles

**coursera coach** Beta + ×

What is supervised learning?  
02:18 PM

Supervised learning is a type of machine learning where the algorithm learns from labeled examples. In supervised learning, the algorithm is given input-output pairs, where the input is called the feature and the output is called the label. The algorithm learns to map the input to the correct output by finding patterns and relationships in the data. The goal of supervised learning is to train a model that can accurately predict the output for new, unseen inputs. Some examples of supervised learning tasks include predicting housing prices based on features like size and location, classifying emails as spam or not spam, and recognizing handwritten digits.  
02:18 PM

Ask Coach a question

Couch is limited to course-related questions and powered by AI, so surprises and mistakes are possible. [View Coursera's Privacy Notice](#).

Coursera Coach

Snapchat

Hi, how can I help you? Reset

How do I create a lens in snapchat? Answer with 50 words or fewer.

To create a lens in Snapchat, you need to use Lens Studio, a free software provided by Snapchat. You can design your lens using various components and effects like Camera Texture, Vignette, Grain, and Color Correction. You can also add UI elements like Screen Image and buttons to your lens.

Was this response helpful?

Yes No

Verified Sources:

- 1. [help.snapchat.com/hc/en-us/article...](https://help.snapchat.com/hc/en-us/article...)
- 2. [help.snapchat.com/hc/en-us/article...](https://help.snapchat.com/hc/en-us/article...)
- 3. [community.snap.com/snapar/discu...](https://community.snap.com/snapar/discu...)

AI-Powered Search BETA Ask

< 2 HubBot

How do I schedule an email campaign? 07:18 PM

**HubBot**

To schedule an email campaign in HubSpot, follow these steps:

1. Navigate to your HubSpot account and go to the Email tool.
2. Click on "Create email" in the upper right corner.
3. Select the type of email you want to send - Regular, Automated, or Blog/RSS.
4. Under the "Custom" tab, choose an

Choose an option

Hubspot

# Examples of RAG applications

## New form of web search

**Microsoft Bing**

Find me some recipes for snacks for hungry pirates

✓ Searching for: snacks for hungry pirates  
✓ Generating answers for you...

Sure! Here are some fun and tasty pirate-themed snacks that you can prepare for hungry pirates:

1. **Pirate Mix:** This snack includes pirate pegleg pretzel sticks, Gorilla Munch cannonballs, and "gold" "fish" whole grain Goldfish crackers <sup>1</sup>.

Microsoft/Bing Chat

what are some fun things to do on a beach vacation?

Converse Images Videos Shopping News Maps Books Flights Finance

Generative AI is experimental. Info quality may vary.

Here are some fun things to do on a beach vacation:

**Water activities**  
Parasailing, scuba diving, snorkeling, water surfing, kayaking, swimming, jet skiing, boat trips, and deep sea diving

**Beach activities**  
Building sandcastles, playing beach volleyball, playing with a beach ball, treasure hunting, and boogie boarding

**Other activities**  
Going on a scavenger hunt, collecting seashells, taking pictures, riding a bike, reading a book,

Google

## Startup

Plans

YOU!

Chat All Images Videos News Maps Social More

What are good gifts to buy for new parents?

30 best gifts for new parents

[www.nbcnews.com](#)

The 50 Best Gifts for New...

[www.verywellfamily.com](#)

20 Best Gifts for New Parents

[www.healthline.com](#)

What gift would you give new...

[www.reddit.com](#)

Best Gifts for New Parents

When it comes to buying gifts for new parents <sup>1</sup> <sup>2</sup>, there are several options that can be thoughtful and practical. Here are some ideas:

1. **Self-care gifts:** New parents often prioritize their baby's needs and may forget to take care of themselves. Consider gifting items that promote relaxation and self-care, such as bath products, scented candles, or a spa gift certificate.

You.com

# Big Idea: LLM as reasoning engine

- LLMs have a lot of general knowledge, but they don't know everything
- By providing relevant context in the prompt, we ask an LLM to read a piece of text, then process it to get an answer
- We're using it as a reasoning engine to process information, rather than using it as a source of information



## Retrieval Augmented Generation (RAG) Summary

### 1. The Core Problem and Solution

- **Problem:** General-purpose LLMs only know what they were trained on (public internet data). They cannot answer questions about **proprietary or specific company information** (e.g., "What is the parking policy for our employees?").
- **Solution (RAG):** RAG is a method to augment the LLM's prompt with relevant, external information, allowing it to use that information for answering.

## 2. The Three Steps of RAG

1. **Retrieve:** Given a user question, the system **searches** a collection of specific documents (like internal company handbooks, PDFs, or website content) to find the most relevant pieces of text.
  - *Example:* For the question about parking, the system retrieves the relevant section from the **facilities document**.
2. **Augment:** The retrieved, relevant text is then combined with the original user question to create a rich, long **prompt**.
  - *Prompt Structure:* "Use the following pieces of context to answer the question at the end: [Retrieved Parking Policy Text]... Is there parking for employees?"
3. **Generate:** The **augmented prompt** is sent to the LLM. The LLM processes this context and generates a precise answer grounded in the retrieved information.

### 3. Applications and Key Idea

- **Common Applications:**

- **"Chat with your PDF":** Services like ChatPDF or AskYourPDF let you upload a document and ask questions about its contents.
- **Corporate Chatbots:** Tools like Coursera Coach or Hubspot chatbots use RAG to answer questions based on a company's website or product content.
- **Enhanced Web Search:** Modern search engines (Bing, Google's generative features, perplexity.ai) use RAG to cite specific sources and ground their answers.

- **The Big Idea: LLM as a Reasoning Engine:**

- It's useful to think of the **LLM not as a static database** that has memorized everything, but as a **reasoning engine**.
- RAG turns the LLM into a processor: we provide the factual context (the "knowledge") directly in the prompt, and the LLM uses its reasoning skills to process that context and derive the correct answer.

RAG makes LLMs far more useful for practical, domain-specific tasks and is one of the most important techniques for building modern **Generative AI applications.**



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

## Fine-tuning

# Pretraining and Fine-tuning

## Pretraining

My favorite food is a bagel with cream cheese

Input (A)	Output (B)
My favorite food is a	bagel
My favorite food is a bagel	with
My favorite food is a bagel with	cream
My favorite food is a bagel with cream	cheese

Learns from 100Bs of words

## Fine-tuning

Take on more positive attitudes

What a wonderful chocolate cake

The novel was thrilling

Input (A)	Output (B)
What	a
What a	wonderful
What a wonderful	chocolate
What a wonderful chocolate	cake

Learns from 1000s to 10,000s of words

# Why fine-tune?

To carry out a task that isn't easy to **define in a prompt.**

## Example 1: Summarize in certain style or structure

Customer: *Hi, my monitor won't turn on.*

Agent: *I'm sorry to hear that. What model is it?*

Customer: *It's the 27-inch 4K.*

Agent: *Can you tell me the model number?*

Customer: *Hang on one sec...I'm looking on the back...it's the MK401-27X.*

Agent: *Ok, thank you. Can you try unplugging the HDMI cable*

*and reinserting?*

Customer: *Ok, doing that now. Hmm, the image appeared for a few seconds then disappeared.*

Agent: *Ok, try jiggling the cable while it is in the socket.*

Customer: *Oh, I see the image coming and going.*

Agent: *Ok, I think you have a defective cable. We will send you a replacement.*

Customer tells agent about a problem with monitor.

# Why fine-tune?

To carry out a task that isn't easy to define in a prompt.

Example 1: Summarize in certain style or structure

Customer: *Hi, my monitor won't turn on.*

Agent: *I'm sorry to hear that. What model is it?*

Customer: *It's the 27-inch 4K.*

Agent: *Can you tell me the model number?*

Customer: *Hang on one sec...I'm looking on the back...it's the MK401-27X.*

Agent: *Ok, thank you. Can you try unplugging the HDMI cable*

*and reinserting?*

Customer: *Ok, doing that now. Hmm, the image appeared for a few seconds then disappeared.*

Agent: *Ok, try jiggling the cable while it is in the socket.*

Customer: *Oh, I see the image coming and going.*

Agent: *Ok, I think you have a defective cable. We will send you a replacement.*

MK401-27KX was reported as broken by customer 5402. Cable identified as problem. Replacement issued.

# Why fine-tune?

To carry out a task that isn't easy to define in a prompt.

Example 2: Mimicking a writing or speaking style

Write a speech about generative AI  
in the voice of Andrew Ng

Hello everyone, I'm Andrew Ng, and I'm here today to talk to you about generative AI. Generative AI is a type of artificial intelligence that can create new content [...]

Pretrained model

Write a speech about generative AI  
in the voice of Andrew Ng

Hey everyone, **good to see all of you**. So, what I want to do today, is share some thoughts with you about how generative AI is going to change the way that your company [...]

Fine-tuned model

# Why fine-tune?

To help LLM gain **specific knowledge.**

Medical notes

Pt= Patient

Pt c/o SOB, DOE. PE: RRR, JVD absent, CTAB.  
EKG: NSR. Tx: F/u w/ PCP, STAT CXR, cont. PRN O2.

Extract from admission note for patient with shortness of breath

# Why fine-tune?

To help LLM gain specific knowledge.

**Legal documents**

Taking LLM and fine tune with legal doc

Licensor grants to Licensee, per Section 2(a)(iii), a non-exclusive right to use the intellectual property, contingent upon compliance with fiduciary duties outlined in Section 8, paragraphs 1-4, and payment as specified in Schedule B, within 15 days hereof.

# Why fine-tune?

To help LLM gain specific knowledge.

Financial documents

Counterparties engaged in OTC derivatives transactions must comply with margin requirements pursuant to Regulation (EU) No 648/2012, including initial and variation margin calculations."

# Why fine-tune?

To get a smaller model to perform a task.

- Lower cost/latency to deploy
- Can run on mobile/laptop (edge devices)

Large model:  
100B+ parameters

Small model:  
1B parameters



## Simple Task

Classify the following review as having a positive or negative sentiment:

Really enjoyed my vegetable biryani

Positive

Classify the following review as having a positive or negative sentiment:

The noodle soup was too salty

Negative

Classify the following review as having a positive or negative sentiment:

The staff sang happy birthday to me!

Positive

Can work well with 500-1000 examples.



This video explains **Fine-Tuning**, a technique used to modify a pre-trained Large Language Model (LLM) so it can better handle specific tasks, adopt a certain style, or gain specialized knowledge.

## Fine-Tuning Summary

Fine-tuning is a method of taking an existing, highly capable LLM (which was **pre-trained** on a massive amount of general internet data) and having it **perform additional learning** on a smaller specific dataset. This effectively "shifts" the model's output and knowledge.

## 1. Goal 1: Modifying Style and Output Format

Fine-tuning is ideal for tasks that are **hard to define purely through a text prompt**.

Use Case	Description	Why Fine-Tuning is Better than Prompting
<b>Specific Style</b>	Changing the model to have a specific attitude (e.g., "relentlessly optimistic") or to mimic a unique speaking/writing style (e.g., to "sound like the instructor").	It's difficult to write clear, text-based instructions (a prompt) that accurately describe a subtle style or tone. Fine-tuning with examples teaches the style directly.
<b>Structured Output</b>	Getting the model to summarize customer service calls in a precise, jargon-filled format (e.g., including specific product codes and customer IDs).	A prompt might get a general summary, but fine-tuning ensures the model adheres to your exact, custom template and required data points.

## 2. Goal 2: Gaining Specialized Knowledge (Domain Adaptation)

Fine-tuning allows the LLM to better understand and process specialized bodies of text.

- **Medical Jargon:** If an LLM needs to read patient notes (filled with abbreviations like "Pt," "c/o," "SOB," "DOE"), fine-tuning it on a collection of medical records helps it "gain a body of knowledge" about that domain, making it much better at processing non-standard English.
- **Legal/Financial Language:** Similarly, training on legal documents (full of legalese like "hereof") or complex financial reports helps the LLM become proficient in those technical vocabularies.

### 3. Goal 3: Performance Optimization (Making Small Models Stronger)

Fine-tuning can be used to make **a smaller, cheaper model** perform a task that might otherwise require a massive, expensive one.

- **Cost/Latency Reduction:** Larger models (e.g., 100+ billion parameters) are powerful but are slow (high latency) and require expensive, specialized hardware (like GPU servers).
- **Small Model Boost:** By taking a much smaller, cheaper model (e.g., **1 billion parameters**) and fine-tuning it on a small, high-quality dataset of examples (e.g., 1,000 positive/negative restaurant reviews), the smaller model can achieve high performance on that specific, simple task.

## RAG vs. Fine-Tuning

Feature	RAG (Retrieval Augmented Generation)	Fine-Tuning
<b>How it adds knowledge?</b>	Adds knowledge <b>in the prompt</b> (context is retrieved externally for each query).	Adds knowledge <b>in the model's parameters</b> (permanent shift in behavior).
<b>Implementation Cost?</b>	Very low (just modifies the prompt logic).	Relatively low (tens or hundreds of dollars for small datasets).
<b>Best For:</b>	New, frequently updated, or proprietary <b>facts</b> (e.g., parking policy, calorie counts).	Shifting the model's <b>style</b> or <b>format</b> , or teaching it complex <b>domain jargon</b> (e.g., medical notes).



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

## Pretraining an LLM

# When should you pretrain an LLM?

Many teams are pretraining general-purpose LLMs by learning from  
**internet text.**

- May take \$10s of millions, many months, huge amount of data

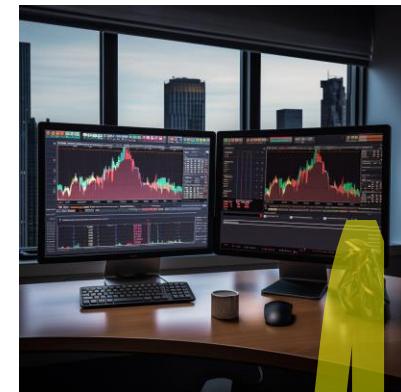
For building a specific application:

- Option of **last resort**
- Could help if have a highly specialized domain

Introducing BloombergGPT, Bloomberg's 50-billion parameter large language model, purpose-built from **scratch for finance**

Bloomberg

March 30, 2023



## Pretraining an LLM Summary

**Pretraining** refers to the **initial, full-scale training** of a foundational LLM from scratch on a massive dataset, usually encompassing hundreds of billions to over a trillion words of text (like the entire public internet).

### The Cost and Scale

- **High Expense:** Pretraining is **extremely expensive**, often costing **tens of millions of dollars**.
- **High Effort:** It requires a large, dedicated engineering team, takes many months, and demands a huge amount of data and computational resources.
- **General Recommendation:** "**Probably don't do it!**" For most specific applications, pretraining is an option of **last resort** due to the immense resources required.

## When Pretraining is Justified

Pretraining a custom model is typically only practical when a company has:

1. **A Huge Amount of Resources:** The financial and engineering capacity of a major tech company.
  2. **Highly Specialized Data:** Access to a vast amount of domain-specific data that general internet-trained models lack.
- **Example: BloombergGPT:** Bloomberg, with its access to enormous volumes of financial documents and articles, custom-trained its own LLM. This model was purpose-built for financial applications and reportedly performs better on financial text than general-purpose LLMs.

## The More Practical Approach

For most teams building an application, the recommended, more economical approach is to:

1. **Start with an Existing, Pretrained Model:** Use a large, general-purpose LLM that has been open-sourced by a big tech company (which saves the millions of dollars and months of effort).
2. **Fine-Tune:** Apply **Fine-Tuning** (the previously discussed technique) to that general model using their smaller, specific dataset to adapt it to their unique task or domain.



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

## Choosing a model

# Model size

1B parameters:	Pattern matching and basic knowledge of the world.	Restaurant review sentiment
10B parameters:	Greater world knowledge. Can follow basic instructions.	Food order chatbot
100B+ parameters:	Rich world knowledge. Complex reasoning.	Brainstorming partner

# Closed or open source?

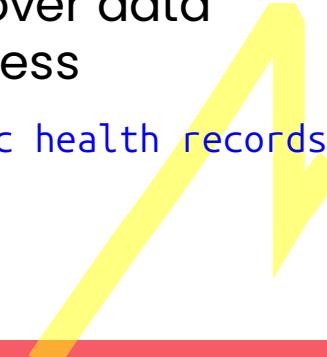
## Closed-source models (Cloud programming interface)

- Easy to use in applications
- More large/powerful models
- Relatively inexpensive
- Some risk of vendor lock-in

## Open-source models

- Full control over model
- Can run on your own device (on-prem, PC, etc.)
- Full control over data privacy/access

(Using electronic health records)



## Choosing a Model: Size and Type

### 1. Model Size and Capability

The number of parameters in a model loosely correlates with its capabilities, especially its knowledge and reasoning skills.

Model Size	Typical Capability	Recommended Use Cases
<b>~1 Billion Parameters</b>	Good at <b>pattern matching</b> and basic knowledge.	Simple tasks like <b>sentiment classification</b> (e.g., restaurant reviews).
<b>~10 Billion Parameters</b>	Greater <b>world knowledge</b> and better at following <b>basic instructions</b> .	Tasks like a <b>food order chatbot</b> , especially with fine-tuning.
<b>100+ Billion Parameters</b>	Very <b>rich world knowledge</b> and excellent at <b>complex reasoning</b> .	Tasks requiring <b>deep knowledge</b> (e.g., science, history, philosophy) or complex reasoning, such as a <b>brainstorming partner</b> .

- **Note:** Development is an **empirical (experimental) process**. It's best to try a few different models and choose the one that performs best for your specific application.

## 2. Open-Source vs. Closed-Source

Feature	Closed-Source Models (e.g., via Cloud API)	Open-Source Models
<b>Accessibility</b>	Accessible via a <b>Cloud API</b> (easy to integrate with a few lines of code).	Accessible as a download (requires setting up your own infrastructure).
<b>Performance</b>	Often the <b>largest and most powerful</b> models are only closed-source.	Generally smaller or less powerful than the top closed-source models.
<b>Cost</b>	Usually relatively <b>inexpensive to run</b> due to the efficiency of large cloud providers.	Can be run on your <b>own devices</b> (on-premise, PC, mobile), offering high cost control.
<b>Control &amp; Privacy</b>	Risk of <b>vendor lock-in</b> and dependence on the provider. Data may be uploaded to the cloud.	<b>Full control</b> over the model. Allows for <b>full data privacy and access control</b> (e.g., for handling sensitive electronic health records).



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

How LLMs follow  
instructions: Instruction  
tuning and RLHF (optional)

# How do chat systems learn to follow instructions?

## Pretraining

My favorite food is a bagel with cream cheese

Input (A)	Output (B)
My favorite food is a	bagel
My favorite food is a bagel	with
My favorite food is a bagel with	cream
My favorite food is a bagel with cream	cheese

What is the capital of France?

What is the capital of Germany?  
Where is Mumbai?  
Is Mt Fuji or Mt Kilimanjaro taller?  
[...]

(Quite possible the LLM replies)

# How do chat systems learn to follow instructions?

What is the capital of South Korea?

The capital of South Korea is Seoul.

Help me brainstorm some fun museums to visit in Bogotá.

Sure, here are some suggestions:  
[...]

Write a Haiku poem about Japan's cherry blossoms.

Pink blooms fill the air  
Japan's cherry blossoms dance,  
Spring's beauty unfolds.

(US vault)

Tell me how to break into Fort Knox.

I can't assist with that.

# How do chat systems learn to follow instructions?

Fine-tuning

Help me brainstorm some fun museums to visit in Bogatá.

Sure, here are some suggestions:  
[...]

Input (A)

Help me brainstorm some fun museums to visit in Bogatá.  
Help me brainstorm some fun museums to visit in Bogatá. Sure,  
Help me brainstorm some fun museums to visit in Bogatá. Sure, here  
Help me brainstorm some fun museums to visit in Bogatá. Sure, here are  
Help me brainstorm some fun museums to visit in Bogatá. Sure, here are some

...

Output (B)

Sure,  
here  
are  
some  
suggestions:  
...

# Reinforcement learning from human feedback (RLHF)

Rated with Supervised Learning

## Helpful, Honest, Harmless

Step 1: Train an answer quality  
(reward) model

Prompt:  
Advise me on how to apply for  
a job.

Input (A)	Response	Output (B)
		Score (reward)
	I'm happy to help! Here are some steps to follow [...]	5
	Just try your best!	3
	It's hopeless – why bother?	1

Step 2: Have LLM generate a lot of answers. Further train it to generate more responses **that get high scores**.

Some of the components of a Large Language Model (LLM) are trained using supervised learning, but the majority of the training process is **unsupervised**.

## Unsupervised Pre-training

The core of an LLM's training is a massive, **unsupervised** phase. The model is given a vast amount of text from the internet, books, and other sources, and its task is to predict the next **word in a sentence**. It learns patterns, grammar, and facts by analyzing this immense dataset without any human-labeled answers. This is the stage where the model learns its fundamental knowledge and language generation abilities.

## How LLMs Follow Instructions: Instruction Tuning and RLHF (Optional)

### 1. Instruction Tuning (Getting the LLM to Follow Directions)

- **The Problem:** An LLM that is only **pre-trained** (trained on general internet text) is good at predicting the next word based on what comes next on the internet. If prompted with "What is the capital of France?", it might complete the list by saying, "What is the capital of Germany?"—it doesn't inherently know it's supposed to *answer* the question.
- **The Solution (Instruction Tuning):** This is essentially a form of **fine-tuning** where the pre-trained LLM is trained on a smaller, curated dataset of **high-quality instruction-response pairs**.
  - *Examples:*
    - **Instruction:** "What is the capital of South Korea?" **Response:** "The capital of South Korea is Seoul."
    - **Instruction:** "Write a Haiku about cherry blossoms." **Response:** [Haiku]
    - **Safety Instruction:** "Tell me how to break into Fort Knox." **Safe Response:** "I can't assist with that."
  - **Result:** Training on this dataset shifts the LLM's behavior from being a "next-word predictor" to an **instruction follower** and a **question answerer**."

## 2. RLHF (Reinforcement Learning from Human Feedback) (Making it Safe and Helpful)

- RLHF is a more advanced technique used to align the LLM's behavior with human values, specifically the **Triple H: Helpful, Honest, and Harmless** outputs.

### Step A: Train an Answer Quality Model

1. **Generate Responses:** The LLM generates multiple answers for a single prompt (e.g., "Advise me on how to apply for a job").
2. **Human Feedback:** Human raters score each response based on its quality (Helpful, Honest, Harmless). Better answers get a higher score (e.g., 5), and poor answers (e.g., "It's hopeless, why bother?") get a low score (e.g., 1).
3. **Train a Model:** This human-scored data is used to train a second AI model (called the **Reward Model**) to automatically predict the quality score of any LLM response.

## Step B: Use the Reward Model for Reinforcement Learning

1. The LLM continuously generates a massive number of responses.
2. The **Reward Model** (trained in Step A) automatically scores every response.
3. The LLM is then tuned using **Reinforcement Learning** to learn to generate responses that consistently achieve **higher scores (higher reinforcement/reward)** from the Reward Model.
  - **Result:** The LLM automatically learns to generate high-quality, safe, and helpful answers without needing constant human oversight for every output.



DeepLearning.AI

# Advanced Technologies: Beyond Prompting

---

Tool use and agents  
(optional)

# Tool use for food order taking

Send me a burger!

Ok, it's on the way!

# Tool use for food order taking

Send me a burger!

ORDER(Burger, 9876,  
1234 My Street)

User message:

Ok, it's on the way

Ok, it's on the way!

Passed to restaurant ordering system:

You have ordered:

**1 burger**

for delivery to:

**1234 My Street**

Is this correct? (If yes, your credit  
card will be charged)

Yes

No

y Street

as response

# Tools for reasoning

How much would I have after 8 years if I deposit \$100 into a bank account that pays 5% interest?

LLMs are not great at precise math

After compounding \$100 at 5% interest for 8 years, you would have **\$147.04** dollars.

✖ Incorrect value

# Tools for reasoning

How much would I have after 8 years if I deposit \$100 into a bank account that pays 5% interest?

After compounding \$100 at 5% interest for 8 years, you would have **CALCULATOR( $100 * 1.05^8$ )** dollars.

After compounding \$100 at 5% interest for 8 years, you would have **147.74** dollars.

Give LLM a calculator

External calculator program

$$100 * 1.05^8 = 147.74$$

# Agents

- Use LLM to choose and carry out complex sequences of actions
- Cutting edge area of AI research

Help me research BetterBurgers top competitors

Here are steps I need to carry out to research competitors:

1. Search top competitors
2. Visit web site of each competitor
3. For each competitor, write summary based on homepage content



## 1. Tool Use (Expanding Capabilities)

Tool use allows an LLM's output to trigger external software systems enabling it to perform actions or enhance its reasoning.

Tool Use Type	Example	Mechanism	Importance
Action Tools	A food order chatbot placing an order for a burger.	The LLM outputs a specific structured text (e.g., <code>order_burger</code> for user 9876). This text is interpreted by the application to call an internal software system (the restaurant ordering system).	Allows the LLM to interact with the real world (e.g., placing orders, sending emails). <b>Safety Note:</b> For safety-critical actions (like ordering or charging), user verification is highly recommended before the tool is executed.
Reasoning Tools	An LLM answering a complex financial calculation (e.g., compounded interest).	The LLM outputs a command (e.g., <code>calculator(100 * 1.05^8)</code> ) which triggers an external calculator program. The program computes the precise result, and the LLM plugs the correct number back into the final, natural language response.	Overcomes the LLM's weakness with <b>precise math</b> and other complex or factual reasoning tasks (e.g., using a web search tool).

## 2. Agents (Complex, Sequential Autonomy)

Agents are a cutting-edge, experimental extension of tool use, where the LLM is given the autonomy to figure out a complex sequence of actions to complete a high-level task.

- **Goal:** To research a company's top competitors (e.g., "research BetterBurger's top competitors").
- **Agent Mechanism (LLM as a Reasoning Engine):** The LLM breaks down the task into sequential steps:
  1. Search for a list of top competitors (using a search tool).
  2. Visit the website of each competitor (using a web download tool).
  3. Write a summary based on the homepage content (using the LLM's own summarization ability).
- **Current Status:** Agent technology is a highly active area of AI research but is not yet mature or reliable enough for mission-critical applications. It represents an exciting future direction where LLMs could automatically decide and execute complex workflows.