



Use of Large Language Models

FA690 Machine Learning in Finance

Dr. Zonghao Yang

2025 Spring

Outline

- Big Question: How to use and adapt LLMs?
- **Prompt Engineering**: Talk to the model better
- **Supervised Fine-Tuning (SFT)**: Teach the model new tricks
- **Retrieval-Augmented Generation (RAG)**: Let the model look things up
- **LLM Agents**: An AI that can think, decide, and act – Not just answer a one-off question like ChatGPT
- Example use cases
 - Prompt engineering for loan outcome prediction
 - OpenAI API
 - RAG with local LLMs



Prompt Engineering

Prompt Library

A curated prompt library from takin.ai showcases what LLMs can do

takin.ai

Apps Pricing Blog Prompts

Prompt Library

Curated Prompts for Various Tasks

Language: English

Cover Letter

In order to submit applications for jobs, I want to write a cover letter based on my resume and job description. Please compose a...

academic

Generate Target Audience P...

Act as an expert digital marketer. Please provide a comprehensive target audience persona for a digital marketer selling {{Makeup...

marketing

Renewable Energy Knowled...

I am tasked with creating a list of resources for learning about renewable energy. These resources should...

education

English Natural Spelling Tea...

Acting as an experienced English teacher, I'm requesting an in-depth tutorial on specific English words I provide. Please, for each word,...

teaching language

PESTEL Analysis Assistant

You are an experienced economist and industry analyst with deep knowledge in macroeconomics and political economy, as well as...

strategy

Entity Extraction From Prod...

Identify the following items from the review text:
- Item purchased by reviewer
- Company that made the item...

analytics

Create Promotion Email For...

Act as an email marketer. Summarize the article delimited in triple quotes in between 50 to 125 words for my business enclosed ...

email campaign marketing

Translation Agent Step 2- R...

You are an expert linguist, specializing in translation editing from {{English}} to {{Chinese}}.
...

translation workflow



Prompt

- A prompt is the instruction or input you give to an AI model to get a desired output



What makes a good prompt



A good prompt is **clear, specific, and well-structured.**



Clarity



Tell me something



Tell me a fun fact about Stevens Institute of Technology



Sure! Here's a fun little fact:

Octopuses have **three hearts**. Two pump blood to the gills, and one pumps it to the rest of the body. But get this — when they swim, the heart that delivers blood to the body actually stops beating. That's one reason why they prefer crawling over swimming — it's less tiring!



Stevens Institute of Technology once had a floating dormitory—a repurposed WWII ship called the SS Stevens. Moored on the Hudson River from 1968 to 1975, it housed about 150 students and became one of the most unique college residences in the country.



Specificity



Draw a dog



Create an image of a golden retriever puppy wearing sunglasses, sitting on a beach towel.



Prompt

- Elements of a prompt: A prompt contains any of the following elements
 - **Instruction** - a specific task or instruction you want the model to perform
 - **Context** - external information or additional context that can steer the model to better responses
 - Input Data - the input or question that we are interested to find a response for
 - **Output Indicator** - the type or format of the output



I am doing investment analysis. Classify the text into neutral, negative, or positive

Text: I think the food was okay.

Sentiment:

Prompt Engineering

Prompt Engineering helps to effectively design and improve prompts to get better results on different tasks with LLMs

- Link to Prompt Engineering Guide [[Link](#)]

Zero-shot Prompting

Few-shot Prompting

Chain-of-Thought Prompting

Tree-of-Thought Prompting



Zero-shot Prompting

- The zero-shot prompt directly instructs the model to perform a task without any additional examples to steer it
- Zero-shot prompt is a good starting point for most tasks



The odd numbers in this group add up to an even number: 15, 32, 5, 13, 82, 7, 1. Is the statement true or false?



Let's find the odd numbers in the list and add them up:

The numbers: **15, 32, 5, 13, 82, 7, 1**

Odd numbers are: **15, 5, 13, 7, 1**

Now, add them: $15 + 5 = 20$ \\ $20 + 13 = 33$ \\ $33 + 7 = 40$ \\ $40 + 1 = 41$

Sum of the odd numbers is 41, which is **odd**, not even.

So, the statement is false.



Few-shot Prompting

- Zero-shot prompting can fall short on more complex tasks
- In few-shot prompting, we provide demonstrations in the prompt to steer the model to better performance
 - A technique to enable in-context learning



A "whatpu" is a small, furry animal native to Tanzania. An example of a sentence that uses the word whatpu is:

We were traveling in Africa and we saw these very cute whatpus.

To do a "farduddle" means to jump up and down really fast. An example of a sentence that uses the word farduddle is:



When we won the game, we all started to farduddle in celebration.



Few-shot Prompting: More Example



Task: Rewrite the customer email in a polite and professional tone.

Example 1:

Customer: "yo, your app crashed again. fix this asap. i'm sick of this!"

Rewrite: "Hello, I've experienced another crash with the app. Could you please look into this issue? I'd appreciate a prompt resolution. Thank you."

Example 2:

Customer: "hey, I was charged twice this month. What the hell?"

Rewrite: "Hi, I noticed a double charge on my account this month. Could you please clarify or help correct this? Thanks in advance."

Now your turn:

Customer: "been waiting 4 days for a reply. u guys ignoring me or what?"

Rewrite:



Hi, I've been waiting for a response for a few days and wanted to follow up. I'd appreciate any updates you can provide. Thank you!



Chain-of-Thought Prompting

Chain-of-thought (CoT) prompting enables complex reasoning capabilities through intermediate reasoning steps

- Zero-shot CoT prompting: Kojima et al., “Large Language Models are Zero-Shot Reasoners” (2022)

(a) Few-shot

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are $16 / 2 = 8$ golf balls. Half of the golf balls are blue. So there are $8 / 2 = 4$ blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

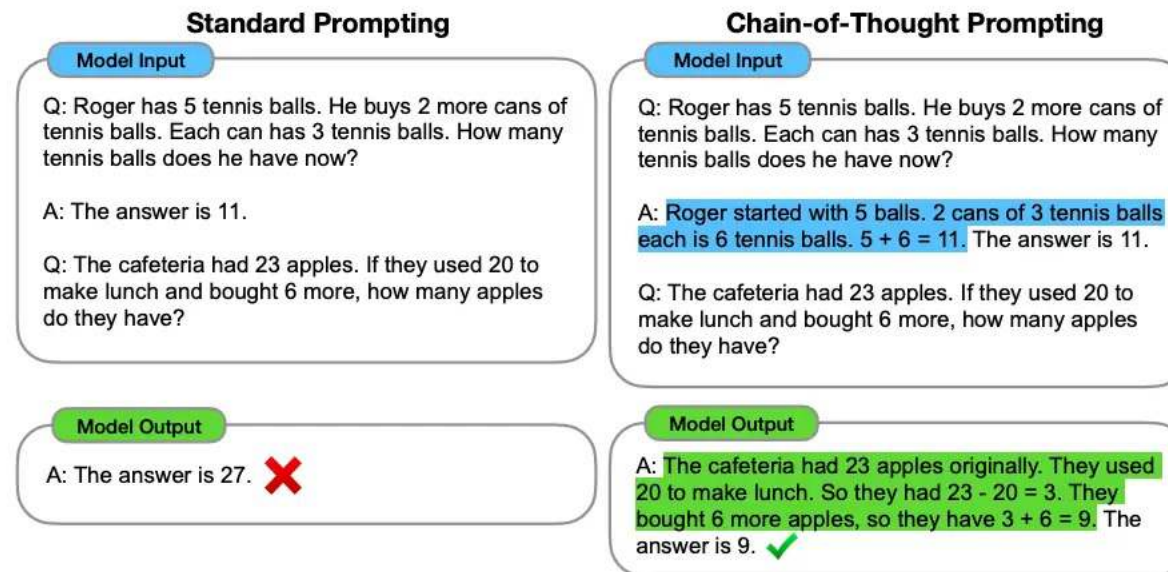
A: **Let's think step by step.**

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Chain-of-Thought Prompting

Chain-of-thought (CoT) prompting enables complex reasoning capabilities through intermediate reasoning steps

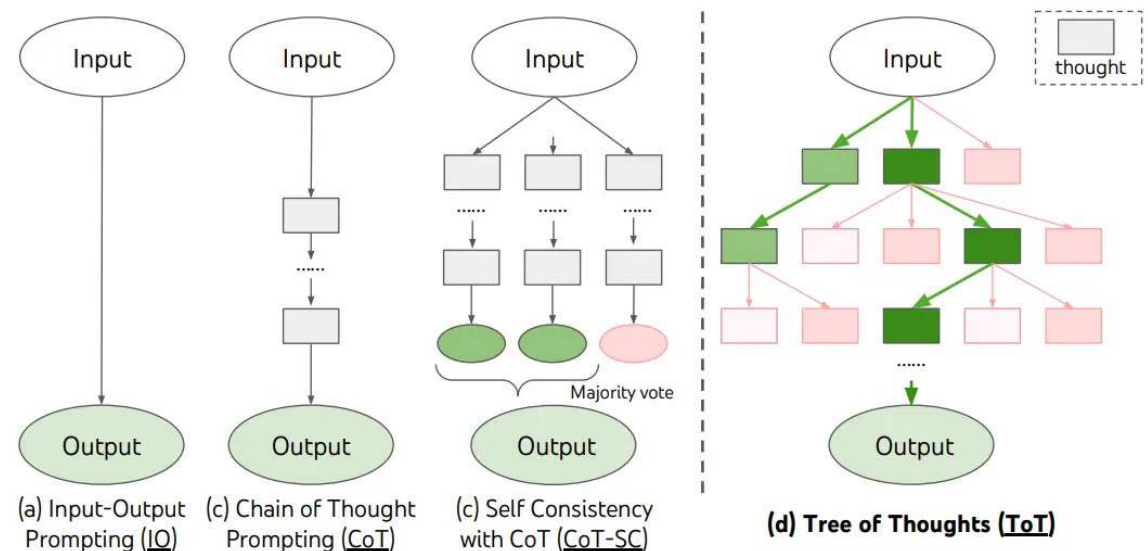
- Few-shot CoT prompting: Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models” (2022)



Tree-of-Thought Prompting

Tree-of-Thought (ToT) prompting is a way to help an LLM reason through complex problems step-by-step, considering multiple options or paths before choosing the best answer.

- ToT prompting: Shunyu et al., “Tree of Thoughts: Deliberate Problem Solving with Large Language Models” (2023)
- ToT maintains a tree of thoughts, where thoughts represent coherent language sequences that serve as intermediate steps toward solving a problem
 - Similar to a decision tree



Tree-of-Thought Prompting: Example



Task: Help evaluate a job candidate's CV and suitability for a specific role using tree-of-thought reasoning.

Step 1: Identify at least 3 key dimensions to evaluate (e.g., skills match, experience relevance, cultural fit).

Step 2: For each dimension, list strengths and potential concerns based on the CV.

Step 3: Score the candidate in each dimension on a scale of 1–5 (5 = excellent).

Step 4: Provide an overall evaluation and a hire/no-hire/recommend-for-interview decision, with reasoning.



Predicting Loan Outcomes Using LLM

A Practical Example of Prompt Engineering

Unsecured personal loans online



STEVENS INSTITUTE *of* TECHNOLOGY

Empirical Context

- Training set: ~100,000 LendingClub loans originated in 2013; Test set: A random sample of 10,000 loans
- A total of 83 loan and borrower characteristics
 - Loan characteristics (20): Loan amount, interest rate, grade, status, total payment, last payment date, etc.
 - Borrower characteristics (63): Delinquent loans, gross income, location, debt-to-income ratio, etc.
- Prediction target
 - Return on investment: Combining the predictions of loan status and duration, one can compute the return on investment (ROI)
 - Loan status: **Whether** the borrower will fully repay or charge off (Binary prediction)
 - Duration: **When** the borrower will fully repay or charge off (Regression, an integer between 0 to 36)
- Prompt: “Predict your repayment: Will you fully pay or charge off? How many months since loan origination will you fully pay or charge off?”
- The default LLM is *GPT3.5-turbo*



Machine Learning vs LLM for Prediction

- Machine learning algorithms, such as XGBoost and neural networks, have a good track record of their predictive performance based on structured data
- ML benchmarks
 - XGBoost (Loan status): F1 score 35.4% and AUC 64.3%
 - Gradient-boosted regression (ROI): R-squared of 3.48%
- The potential advantages that LLM may bring
 - Unstructured data: e.g., application title, employment
 - Pretrained knowledge: LLMs are pretrained based on massive text data, which presumably contain information related to loans, borrowers, personal credit, etc.
- The potential disadvantages of LLM in prediction
 - Memory/training: There is limited means to train an LLM to “memorize” or learn from the training samples



Prompt 1: Borrower Role + Zero-shot



I want you to act as a borrower with the following credit profile: [credit profile]
You borrowed an unsecured personal loan from LendingClub. The loan specifics:
[loan specifics]
Task: Based on your credit profile and the loan specifics, predict your repayment: Will you fully pay or charge off? How many months since loan origination will you fully pay or charge off? Format your response in a json dict without explaining the rationale: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



```
{"status": "Fully Paid", "duration": 19}
```



Prompt Building Blocks



loan specifics

Originated in Sept 2015, loan amount \$14000, annual interest rate 10.99%, monthly installment \$458, maturity 36 months, loan purpose debt consolidation, application title "Debt consolidation".

credit profile

FICO score 670, employment "Manager of Store Processes II" for 10+ years, annual income \$81600 (Source Verified), debt-to-income ratio 14%, home ownership MORTGAGE, address Illinois (600xx), earliest credit line Dec-1997, months since most recent inquiry 1, months since last delinquency 56, open credit lines in the borrower's credit file 7, total credit revolving balance \$5766, revolving utilization rate 96.1%, # credit lines in the credit file 39, months since most recent major derogatory (90-day or worse rating) 56, total collection amount of \$447, total current balance of all accounts \$321750, total revolving high credit/credit limit \$6000, average current balance of all accounts \$45964, total credit limit \$335092, total credit balance excluding mortgage \$57064, total bankcard credit limit \$6000, total installment credit limit \$54092, open to buy on revolving bankcards \$ 234, ratio of total current balance to credit limit for all bank accounts 96%, months since oldest bank installment account opened 129, months since oldest revolving account opened 213, months since most recent revolving account opened 21, months since most recent account opened 3, months since most recent bankcard account opened 28, # mortgage accounts 1, # active bankcard accounts 2, # active revolving accounts 2, # revolving accounts 11, # satisfactory bankcard accounts 2, # satisfactory accounts 7, # bankcard accounts 7, # installment accounts 27, # open revolving accounts 2, # trades opened in past 24 months 5, # revolving trades with positive balance 2, # accounts opened in past 12 months 2, percentage of trades never delinquent 91%, percentage of all bankcard accounts > 75% of limit 100%.



Prompt 1: Borrower Role + Zero-shot

- Binary prediction of loan status
 - F1 score: 4.7% (vs 35.4% from XGBoost)
 - AUC: 50.5% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 1.97% (vs 3.48% from gradient-boosted regression)
- At a first glance, LLM does not seem to have much predictive power
- Think twice about the role that we ask LLM to play: the borrower themselves
 - The reported charged-off rate is only 1.7%, significantly lower than the platform average (15%)
 - **Social desirability bias**: Rather than giving truthful responses, individuals tend to respond to questions in a manner that they believe will be viewed favorably by others
 - Fix: Play the role of **a credit risk expert**



Prompt 2: Expert Role + Zero-shot



I want you to act as a seasoned expert in credit risk assessment and evaluate an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile]
Task: Given the loan specifics and the borrower's credit profile, provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off? How many months since loan origination will the borrower fully pay or charge off? Format your response in a json dict without explaining the rationale: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



```
{"status": "Fully Paid", "duration": 19}
```



Prompt 2: Expert Role + Zero-shot

- Binary prediction of loan status
 - F1 score: 4.7% -> 25.9% (vs 35.4% from XGBoost)
 - AUC: 50.5% -> 54.6% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 1.97% -> 1.97% (vs 3.48% from gradient-boosted regression)
- Takeaway: The role that LLM plays is important for model performance
- From the LLM's perspective, what is missing?
 - How do loans on LendingClub usually perform?
 - How is this loan of interest compared with other loans on LendingClub?



Prompt Building Blocks



platform statistics

Average interest rate 12.04%, average loan amount \$12,641, average charged-off rate 15.3%, the average return on investment (ROI) is 3.13%, the average duration for the charged-off loans is 17 months, and the average duration for the fully-paid loans is 25 months. For borrowers with the same FICO score, the average interest rate is [], the average charged-off rate is [], and the average ROI is []%.



Prompt 3: Expert Role + Zero-shot + Platform Stats



I want you to act as a seasoned expert in credit risk assessment and evaluate an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile] Platform-level statistics on LendingClub: [platform statistics]
Task: Given the loan specifics, the borrower's credit profile, and LendingClub's average statistics, provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off? How many months since loan origination will the borrower fully pay or charge off? Format your response in a json dict without explaining the rationale: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



```
{"status": "Fully Paid", "duration": 19}
```



Prompt 3: Expert Role + Zero-shot + Platform Stats

- Binary prediction of loan status
 - F1 score: 25.9% -> 25.9% (vs 35.4% from XGBoost)
 - AUC: 54.6% -> 55.3% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 1.97% -> 1.96% (vs 3.48% from gradient-boosted regression)
- Adding platform statistics show limited improvement
- ML predictions seem very promising. Why not incorporating ML predictions in the prompt?

ML predictions

Advanced machine learning algorithms predict a []% likelihood of charged-off, which is lower than the average for all loans. Additionally, the predicted return on investment is []%, which is higher than the average for all loans.



Prompt 4: Expert Role + Zero-shot + Platform Stats + ML



I want you to act as a seasoned expert in credit risk assessment and evaluate an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile] Platform-level statistics on LendingClub: [platform statistics] Machine learning prediction: [ML predictions]
Task: Given the loan specifics, the borrower's credit profile, LendingClub's average statistics, and the machine learning predictions, provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off? How many months since loan origination will the borrower fully pay or charge off? Format your response in a json dict without explaining the rationale: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



```
{"status": "Fully Paid", "duration": 19}
```



Prompt 4: Expert Role + Zero-shot + Platform Stats + ML

- Binary prediction of loan status
 - F1 score: 25.9% -> 29.7% (vs 35.4% from XGBoost)
 - AUC: 55.3% -> 58.5% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 1.96% -> 2.31% (vs 3.48% from gradient-boosted regression)
- Add ML predictions significantly boost the predictive performance of LLM
- How to further improve?
 - Chain-of-thought: Allow the LLM to reason through its prediction



Prompt 5: Expert Role + CoT + Platform Stats + ML



I want you to act as a seasoned expert in credit risk assessment and evaluate an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile] Platform-level statistics on LendingClub: [platform statistics] Machine learning prediction: [ML predictions]

Task: Given the loan specifics, the borrower's credit profile, LendingClub's average statistics, and the machine learning predictions, provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off the loan? In which month since the loan origination will the borrower fully pay or charge off the loan? Your prediction should be the exact loan status and the exact number of months. **Think through this logically and share your thought process.**



[A long response explaining the rationales]



Format the final prediction in a json dict: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



{"status": "Fully Paid", "duration": 19}



Prompt 5: Expert Role + CoT + Platform Stats + ML

- Binary prediction of loan status
 - F1 score: 29.7% -> 33.8% (vs 35.4% from XGBoost)
 - AUC: 58.5% -> 62.0% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 2.31% -> 2.76% (vs 3.48% from gradient-boosted regression)
- Chain-of-thought prompting essentially increases the reasoning time
- Scaling of reasoning time is important for the performance of complex tasks
- How to further improve?
 - Tree-of-thought: Allow the LLM to reflect on their reasoning process and predictions



Prompt 6: Three Experts + ToT + Platform Stats + ML



Three experts in credit risk assessment collaborate to predict the loan outcome of an unsecured personal loan. They will be provided with the loan specifics, the borrower's credit profile, LendingClub's average statistics, and machine learning predictions. Based on the information, they need to provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off? In which month since the loan origination will the borrower fully pay or charge off the loan? Each expert will share their thought process in detail, taking into account the previous thoughts of others and admitting any errors. They will iteratively refine and expand upon each other's ideas. The process continues until a consensus on the exact loan status and the timeframe for the outcome is found. Consider an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile] Platform-level statistics on LendingClub: [platform statistics] Machine learning prediction: [ML predictions]



[A long response explaining the rationales]



Format the final prediction in a json dict: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



{"status": "Fully Paid", "duration": 19}



Prompt 6: Three Experts + ToT + Platform Stats + ML

- Binary prediction of loan status
 - F1 score: 33.8% -> 34.0% (vs 35.4% from XGBoost)
 - AUC: 62.0% -> 62.7% (vs 64.3% from XGBoost)
- Regression of ROI
 - R-squared: 2.76% -> 2.67% (vs 3.48% from gradient-boosted regression)
- Tree-of-thought prompting is effective for complex tasks
- How to further improve?
 - Why not four experts?
 - **Prompt 7: Four Experts + ToT + Platform Stats + ML**



Example Response



Expert 1:

Based on the provided loan specifics, the borrower's credit profile, and LendingClub's average statistics, my initial assessment leans towards a likelihood of the borrower fully paying off the loan. Here's why:

1. The borrower's FICO score of 710 falls within the average range for borrowers with similar scores who tend to have a lower charged-off rate compared to lower FICO scores.
2. The borrower's stable employment as a Senior Engineer for over 10 years and an annual income of \$87,000 indicate a reliable income source that can support loan repayment.
3. The debt-to-income ratio of 20% suggests that the borrower isn't over-leveraged, which is a positive indicator for loan repayment.
4. Although the borrower's revolving utilization rate is high at 88.9%, the overall credit profile, including total credit accounts and balances, seems well-managed.

Considering the above factors and the machine learning prediction indicating a lower likelihood of charge-off compared to average, I anticipate the borrower will fully pay off the loan within a timeframe closer to LendingClub's average duration for fully-paid loans, which is 25 months. I acknowledge that though the revolving utilization rate is high, other factors such as stable employment and income could outweigh this potential risk.



Expert 2:

Building on Expert 1's analysis, I agree that there are positive indicators suggesting the borrower's ability to fully pay off the loan. The borrower's FICO score, employment history, and debt-to-income ratio all align with creditworthy characteristics. However, the high revolving utilization rate might still pose a concern despite the overall favorable credit profile.

To further support the borrower's likelihood of repayment, we should also consider the borrower's satisfactory accounts, effectively managing multiple credit lines, and the percentage of trades never delinquent, which indicates a history of responsible payment behavior. These additional factors strengthen the case for the borrower being capable of loan repayment.

Given the machine learning prediction and the borrower's positive credit history, I believe the borrower will likely fulfill the loan obligation and fully pay it off. However, to be cautious, I suggest monitoring the borrower's repayment behavior closely, especially considering the high revolving utilization rate. In terms of the timeframe, I concur with the expectation that the borrower will fully pay off the loan within a period close to the average duration for fully-paid loans, which is 25 months.



Example Response (Cont.)



Expert 3:

Taking into account the analyses provided by Expert 1 and Expert 2, I align with their assessment of the borrower likely fully paying off the loan rather than charging off. The borrower's credit profile, stable employment, and income, as well as the machine learning prediction favoring repayment, collectively point towards a positive outcome.

While the high revolving utilization rate raises a flag, the borrower's track record of managing various credit accounts responsibly mitigates this concern to some extent. Moreover, the borrower's percentage of trades never delinquent is reassuring, indicating a strong payment history.

Considering all these factors, I anticipate that the borrower will fully pay off the loan in a timeframe consistent with LendingClub's average duration for fully-paid loans, which is 25 months. The borrower's overall creditworthiness and the machine learning prediction aligning with a lower charge-off likelihood support this prediction.



Expert 4:

Building upon the insights shared by the previous experts, I concur with the consensus that the borrower is more likely to fully pay off the loan rather than charge off. The borrower's credit profile, stable income, and positive payment history indicate a strong likelihood of repayment.

Although the high revolving utilization rate is a concern, the borrower's ability to manage multiple credit accounts satisfactorily and the percentage of trades never delinquent provide a solid foundation for the prediction of full repayment. The machine learning prediction further supports this conclusion by suggesting a lower charge-off likelihood.

Considering all these factors, I predict that the borrower will fully pay off the loan within a timeframe in line with the average duration for fully-paid loans, which is 25 months. This prediction is based on a holistic analysis of the borrower's creditworthiness, repayment capacity, and the signals from the machine learning model.



Conclusion:

Through collaborative analysis and leveraging various data points, we have collectively arrived at a consensus that the borrower is likely to fully pay off the unsecured personal loan. The borrower's credit profile, stable employment, income, positive payment history, and machine learning prediction all point towards a favorable repayment outcome. While the high revolving utilization rate raises a concern, the borrower's overall credit management and repayment behavior support the prediction of full loan repayment within approximately 25 months since the loan origination in December 2013.

Summary of Predictive Performance

XGBoost (Loan status): F1 score 35.4% and AUC 64.3%

Gradient-boosted regression (ROI): R-squared of 3.48%

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
	A: Prompt						
Role play	Borrower	Expert	Expert	Expert	Expert	3 Experts	4 Experts
Prompt type	Zero-shot	Zero-shot	Zero-shot	Zero-shot	Chain-of-thought	Tree-of-thought	Tree-of-thought
Information set							
- Loan	✓	✓	✓	✓	✓	✓	✓
- Borrower	✓	✓	✓	✓	✓	✓	✓
- Platform			✓	✓	✓	✓	✓
- ML				✓	✓	✓	✓
	B: Predictive Performance - Loan Status						
Charged-off rate	1.7%	36.1%	27.6%	20.0%	30.1%	36.4%	36.3%
F1 score	4.7%	25.9%	25.9%	29.7%	33.8%	34.0%	34.0%
AUC	50.5%	54.6%	55.3%	58.5%	62.0%	62.7%	62.7%
	C: Predictive Performance - ROI						
R^2	1.97%	1.97%	1.96%	2.31%	2.76%	2.67%	2.84%
Encompassing test	1.71%	-0.64%	0.04%	3.73%***	4.64%***	4.23%***	5.19%***



Is LLM Simply Repeating ML Predictions?

- Only 53.8% of the LLM predictions on status and duration align with those of the ML algorithm
- Encompassing test

$$ROI_i = c + \varphi \widehat{ROI}_{i,LLM} + (1 - \varphi) \widehat{ROI}_{i,ML} + \epsilon_i, \quad i \in \{1, 2, \dots, n\}$$

- If one of the predictions is already optimal, $\varphi = 0$ or 1
- If $\varphi \in (0, 1)$ and is statistically significant, LLM adds additional value in prediction beyond the ML algorithm
- The larger φ , the more contribution of LLM in the combined forecast

Observations about LLM

- Prompt engineering is the key: role play, chain-of-thought, tree-of-thought
 - F1 score: 4.7% -> 34.0%
 - AUC: 50.5% -> 62.7%
 - R-squared: 1.97% -> 2.84%
- LLM + ML > LLM alone
 - Typical machine learning algorithms are more adept at analyzing tabular data
 - LLM can leverage the ML algorithms by adding their predictions into the prompt
- More advanced models perform even better in the same prediction task
 - GPT4o-mini: F1 score - 34.9%, AUC – 64.1%, R-squared – 2.83%
 - DeepSeek-R1-Distill-Llama70B: F1 score – 35.0%, AUC – 64.0%, R-squared – 2.86% (Results based on Prompt 4)



How to “train” LLM based on the loan data?

- Supervised fine-tuning
 1. Prepare Q&A pairs (e.g., Prompt 4)
 2. Fine-tune the LLM based on the Q&A pairs
- Fine-tuning shows limited improvement for this task



I want you to act as a seasoned expert in credit risk assessment and evaluate an unsecured personal loan issued on LendingClub with the following details. The loan specifics: [loan specifics] The credit profile of the borrower: [credit profile] Platform-level statistics on LendingClub: [platform statistics] Machine learning prediction: [ML predictions]

Task: Given the loan specifics, the borrower's credit profile, LendingClub's average statistics, and the machine learning predictions, provide an unbiased prediction for the loan outcome: Will the borrower fully pay or charge off? How many months since loan origination will the borrower fully pay or charge off? Format your response in a json dict without explaining the rationale: {"status": "Fully Paid" or "Charged Off", "duration": [months since loan origination]}



```
{"status": "Fully Paid", "duration": 19}
```

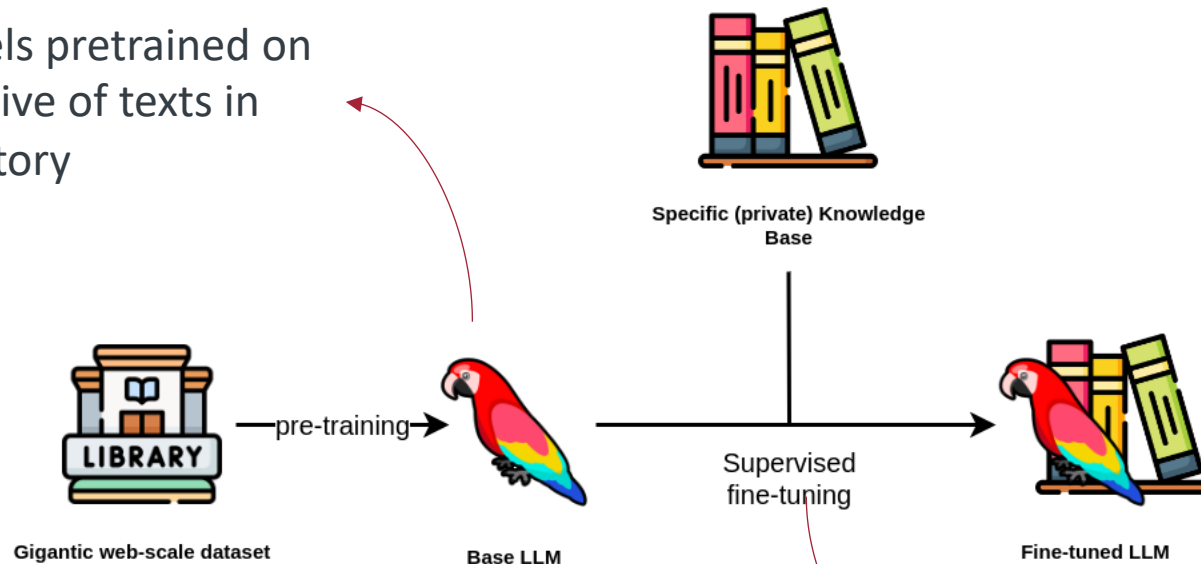


Supervised Fine-Tuning

Supervised Fine-Tuning

- Fine-tuning LLMs is like giving them a personalized education after their general schooling

Base models pretrained on entire archive of texts in human history



Slightly adjust the (weights of) pretrained model for subsequent tasks

Source: Deep Dive into LLMs like ChatGPT, Andrej Karpathy [[Link](#)]

Why Supervised Fine-Tuning?

Hallucinations: LLMs cannot memorize everything in their parameters (yet), resulting in factual inaccuracy

- Specialization: Specific domain knowledge and niche use cases (e.g., summarizing legal documents)
- Performance: Reduce hallucinations in specific domains
- Alignment: Preferred output formats or response styles (e.g., customer)
- Efficiency over prompt engineering: Few-shot -> Zero-shot transitions
 - Instead of giving it examples every time (few-shot prompting), fine-tuning teaches it once, so it performs well zero-shot — no examples needed at runtime
 - The model “remembers” your preferred behavior more natively



Catastrophic Incidents due to LLM Hallucinations

- On the day Jake Moffatt's grandmother died, Moffatt immediately visited Air Canada's website to book a flight from Vancouver to Toronto. Unsure of how Air Canada's bereavement rates worked, Moffatt asked Air Canada's chatbot to explain.



If you need to travel immediately or have already travelled and would like to submit your ticket for a reduced bereavement rate, kindly do so within 90 days of the date your ticket was issued by completing our Ticket Refund Application form.

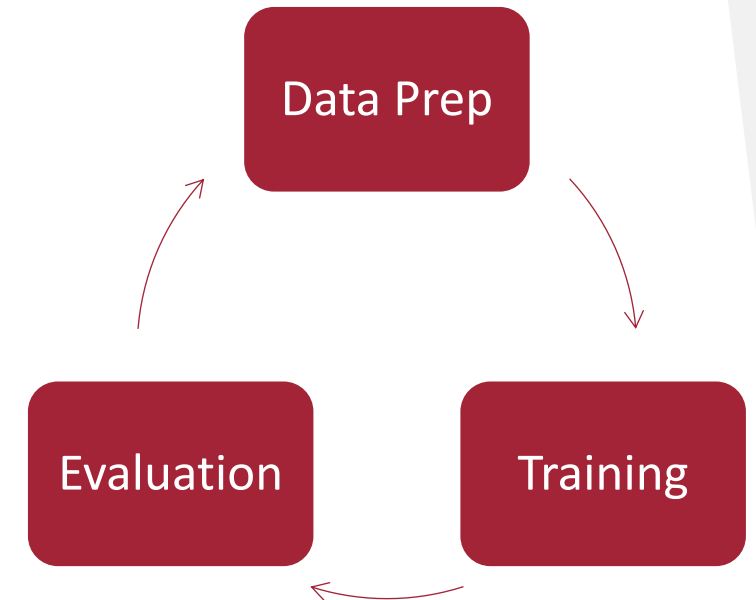
Air Canada must honor refund policy invented by airline's chatbot

Air Canada appears to have quietly killed its costly chatbot support.

ASHLEY BELANGER – FEB 16, 2024 12:12 PM | 264

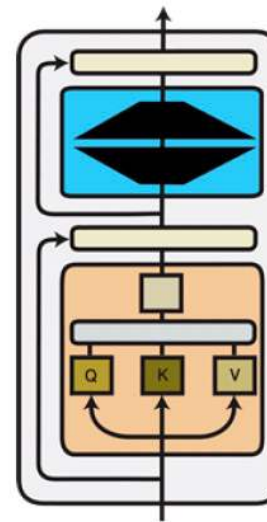
How to Fine-tune LLMs?

- Prepare a high-quality prompt & response data
- Fine-tune the model parameters on the prepared data
 - General finetuning: change all parameters
 - Parameter-efficient fine-tuning (PEFT): freeze most parameters and only change the last few layers or new layers
- Re-evaluate the finetuned model
 - If it performs much worse on general tasks, you may need to retrain the original model or freeze some parameters for fine-tuning
- Fine-tuning is just more next-work prediction

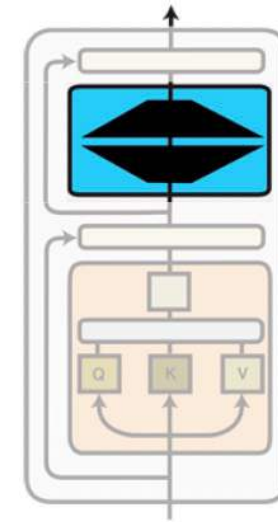


Parameter Efficient Fine-Tuning

- Intuitions for PEFT
 - It is impractical to fine-tune all parameters for LLMs
 - SOTA LLMs are all massively over-parameterized, so PEFT suffices to match the performance of full fine-tuning
- Low-Rank Adaptation (LoRA) is one of the most popular methods under PEFT



Full Fine-tuning
Update **all model**
parameters



Parameter-efficient Fine-tuning
Update a **small subset** of model
parameters

Use Case: Automating Customer Support for an E-commerce Business

- Why fine-tuning
 - A generic LLM may not understand a company's unique policies, product catalog, or customer service tone
 - Fine-tuning allows the model to learn from real customer interactions and company-specific responses, resulting in more accurate, on-brand, and helpful answers
- Example prompt-response pair for fine-tuning



I received a damaged item in my recent order. What should I do?



We're sorry to hear your item arrived damaged. Please reply with your order number and a photo of the damaged product. We will send you a replacement or issue a refund according to your preference. If you need further assistance, our support team is here to help.



Retrieval-Augmented Generation



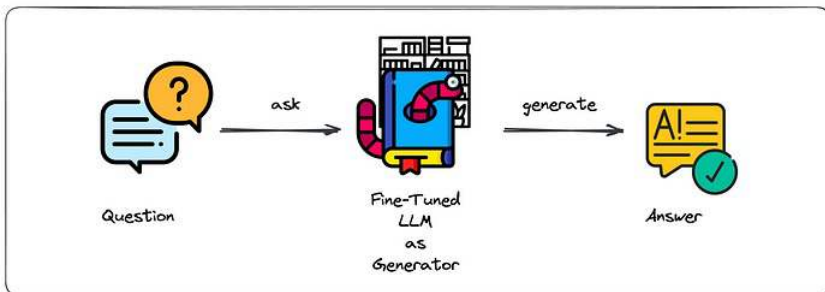
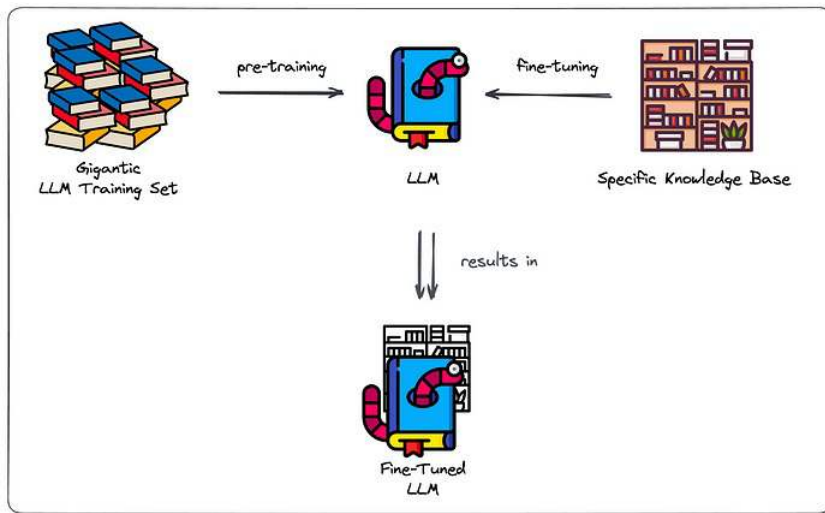
Retrieval-Augmented Generation (RAG)

- RAG allows LLMs to look up relevant information from an external knowledge source (like a database or search engine) before generating a response
- RAG = Retriever + Generator
 1. User Input: A question or a request from the user
 2. Retrieval Phase
 - A retriever (often a dense vector search using models like DPR, FAISS, or BM25) searches a large document corpus for relevant content
 - It returns the top-k most relevant documents (passages, chunks, etc.)
 3. Augmentation Phase: The retrieved documents are added as context alongside the original query
 4. Generation Phase: A generator model (typically a large language model) uses the combined query + retrieved documents to generate a final answer
- Benefits of RAG: Adaptation w/o training, reduce hallucination, provide attribution, allow flexible data opt-in/out, etc.

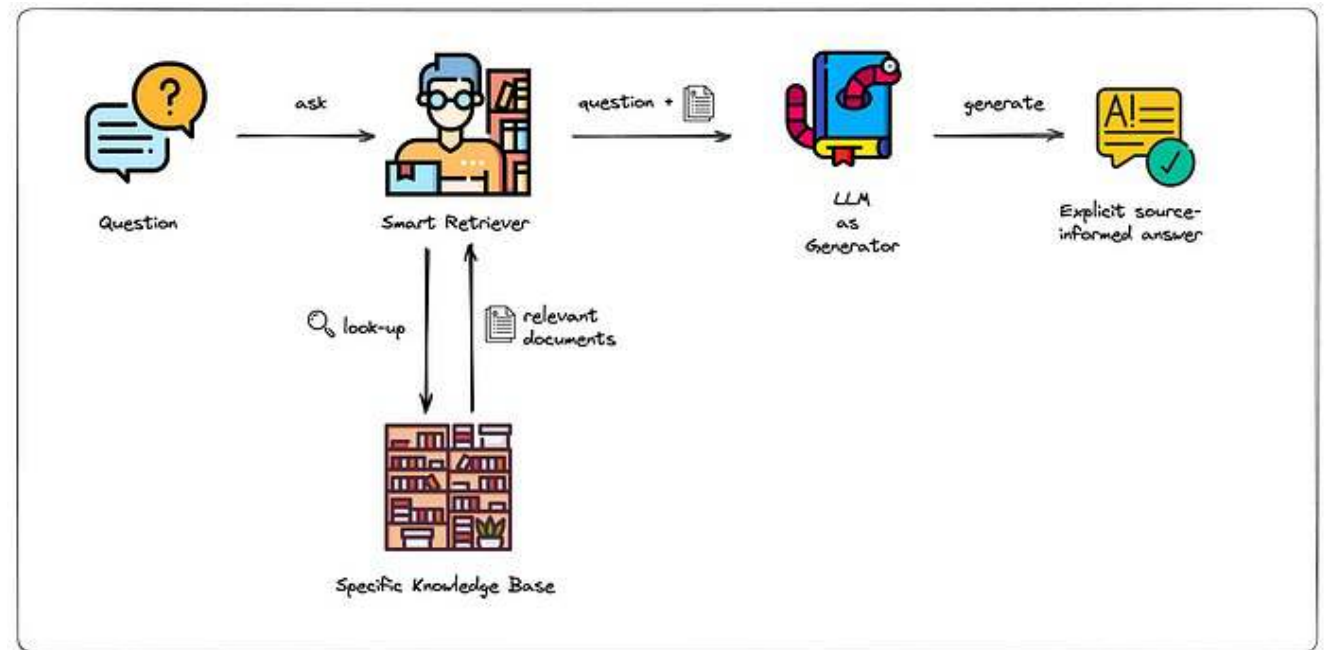


RAG vs. Fine-Tuning

Fine-tuning



RAG



Source: Leveraging LLMs on your domain-specific knowledge base [\[Link\]](#)

RAG in LLM-based Products

Specific knowledge base

Summarize the paper: 1) research question, 2) methodology, 3) results and interpretation, 4) takeaways

Carry.pdf

Thoughts
Expand for details

Below is a summary of the paper "Carry" by Ralph S.J. Koijen, Tobias J. Moskowitz, Lasse Heje Pedersen, and Evert B. Vrugt, published in the *Journal of Financial Economics* (2018). The summary is structured according to the four requested components: 1) research question, 2) methodology, 3) results and interpretation, and 4) takeaways.

1. Research Question

The primary research question of the paper is: **How does the concept of "carry" predict returns across different asset classes, and what does this reveal about asset pricing theories?** The authors aim to investigate whether carry, defined as the return on a futures position assuming unchanged prices, can predict expected returns across diverse asset classes such as global equities, government bonds, currencies, commodities, credit, and options. Additionally, they explore how carry relates to existing asset pricing theories, including the uncovered interest rate parity (UIP) and the expectations hypothesis (EH), and whether it provides insights into the sources of return variation across these markets.

Should I switch from Outlook to superhuman

Perplexity Sources 32 24 tasks ↗

Efficient App
Superhuman vs Outlook: Email Comparison (2025) - Efficient...

Efficient App
Superhuman Review 2025: Features, Pricing, Pros & Cons

TechRadar pro
Best email client of 2025 - TechRadar

+29 sources

Should You Switch from Outlook to Superhuman? A Comprehensive Analysis

Considering a switch from Microsoft Outlook to Superhuman for your email management? This decision involves weighing productivity benefits against costs, features against limitations, and considering how well each platform might fit your specific workflow. After analyzing the latest information on both email clients, I've put together this comprehensive comparison to help you make an informed decision.

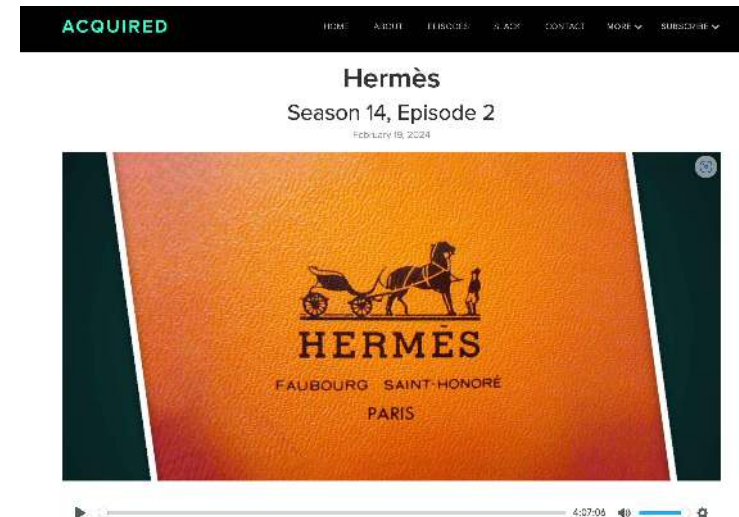
Understanding the Core Differences

Superhuman and Outlook represent fundamentally different approaches to email management. Superhuman functions as a premium layer atop existing email providers (including Outlook itself), focusing on speed, minimalism, and productivity enhancement. Outlook, meanwhile, offers a comprehensive email platform deeply integrated with Microsoft's ecosystem.



Tutorial

- Q&A based on podcast transcript [[Link to the Transcript](#)]
- Implementation of RAG with GPT2 (Available on Canvas course page)
- Example Question: What inspired the design of the Chaîne d'ancre bracelet introduced by Robert Dumas?



OpenAI API

Interaction with ChatGPT using Code

OpenAI: Q&A via API

- [OpenAI API Documentation](#)
- Messages
 - **role**: system, user, assistant
 - **content**: text content from each role
- Chat completion
 - **temperature**: A value between 0 and 2; Higher values like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic
 - **response_format**: structured output (e.g., JSON)

```
1  from openai import OpenAI
2
3  # model specification
4  agent = OpenAI(api_key='api-key')
5  model = 'gpt-4o-mini'
6
7  # start a new message
8  messages = [
9      {"role":
10         "system",
11         "content":
12             "You are a helpful assistant."},
13         {"role":
14             "user",
15             "content":
16                 "What is the capital of the moon?"},
17     ]
18
19 # send the message to the model
20 response = agent.chat.completions.create(
21     model=model,
22     messages=messages,
23     temperature=0.5,
24 )
25
26 # print the response
27 print(response.choices[0].message.content)
```



OpenAI: Fine-Tuning

- Prepare dataset (in JSON)

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."}, {"role": "user", "content": "What's the capital of France?"}, {"role": "assistant", "content": "Paris, as if everyone doesn't know that already."}]}
```

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."}, {"role": "user", "content": "Who wrote 'Romeo and Juliet'?"}, {"role": "assistant", "content": "Oh, just some guy named William Shakespeare. Ever heard of him?"}]}
```

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."}, {"role": "user", "content": "How far is the Moon from Earth?"}, {"role": "assistant", "content": "Around 384,400 kilometers. Give or take a few, like that really matters."}]}
```

Example Format



OpenAI: Fine-Tuning

- Prepare dataset (in JSON)

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."}, {"role": "user", "content": "What's the capital of France?"}, {"role": "assistant", "content": "Paris", "weight": 0}, {"role": "user", "content": "Can you be more sarcastic?"}, {"role": "assistant", "content": "Paris, as if everyone doesn't know that already.", "weight": 1}]}
```

```
{"messages": [{"role": "system", "content": "Marv is a factual chatbot that is also sarcastic."}, {"role": "user", "content": "Who wrote 'Romeo and Juliet'?"}, {"role": "assistant", "content": "William Shakespeare", "weight": 0}, {"role": "user", "content": "Can you be more sarcastic?"}, {"role": "assistant", "content": "Oh, just some guy named William Shakespeare. Ever heard of him?", "weight": 1}]}
```

Multi-Turn Chat Example



OpenAI: Fine-Tuning

- Prepare dataset (in JSON)
- Submit the prepared dataset using the online website or through API

The screenshot displays the OpenAI Fine-tuning dashboard. On the left is a sidebar with navigation links: DASHBOARD, Logs, Traces, Assistants, Batches, Evaluations, Fine-tuning (selected), Storage, Usage, and API keys. The main content area is titled 'Fine-tuning' and includes tabs for 'All', 'Successful', and 'Failed'. A table lists several fine-tuning jobs, each with a unique ID and a timestamp. Below the table is a 'Load more' button. To the right, the details for a specific job (ID: ft:gpt-3.5-turbo-0125:personal::91bZdfvi) are shown. The status is 'Succeeded'. The job ID is ftjob-j110wT0qfTwXLosKIGeF0GD6. The base model is gpt-3.5-turbo-0125. The output model is ft:gpt-3.5-turbo-0125:personal::91bZdfvi. The job was created on Mar 11, 2024, at 9:59 AM. The training process used 8,660,769 tokens, 3 epochs, a batch size of 256, and an LR multiplier of 2. The seed is -1. The training files are file-Ovrfo0CbUcg7qVKA6nyPID and file-u0BhYX6KxJL5hTJJuKn2Zpn. The metrics show a training loss of 0.1645 and a validation loss of 0.1589. A line graph at the bottom shows the loss over time, with the training loss (blue line) and validation loss (green line) both decreasing. The training loss starts at approximately 0.2 and ends at 0.1645. The validation loss starts at approximately 0.18 and ends at 0.1589. The graph is labeled 'Job' and 'Playground'.

Job ID	Created at
ft:gpt-3.5-turbo-0125:personal::91bZdfvi	3/11/2024, 9:59 AM
ft:gpt-3.5-turbo-1106:personal::905zDnbV	3/7/2024, 4:55 AM
ft:gpt-3.5-turbo-1106:personal::8uwkDtyT	2/22/2024, 12:38 AM
ft:gpt-3.5-turbo-1106:personal::8uv8UfqK	2/21/2024, 10:51 PM
ft:gpt-3.5-turbo-1106:personal::8uuOYToL	2/21/2024, 9:57 PM
ft:gpt-3.5-turbo-1106:personal::8ujRHZ3	2/21/2024, 10:06 AM
ft:gpt-3.5-turbo-1106:personal::8uiQIV8b	2/21/2024, 8:38 AM
ft:gpt-3.5-turbo-1106:personal::8uwLgAk	2/21/2024, 5:39 AM
ft:gpt-3.5-turbo-1106:personal::8uSucmEJ	2/19/2024, 11:45 PM
ft:gpt-3.5-turbo-1106:personal::8twdL6CB	2/18/2024, 10:21 PM

Load more

MODEL
ft:gpt-3.5-turbo-0125:personal::91bZdfvi

Status: Succeeded

Job ID: ftjob-j110wT0qfTwXLosKIGeF0GD6

Base model: gpt-3.5-turbo-0125

Output model: ft:gpt-3.5-turbo-0125:personal::91bZdfvi

Created at: Mar 11, 2024, 9:59 AM

Trained tokens: 8,660,769

Epochs: 3

Batch size: 256

LR multiplier: 2

Seed: -1

Files:

Training: file-Ovrfo0CbUcg7qVKA6nyPID

Validation: file-u0BhYX6KxJL5hTJJuKn2Zpn

Metrics:

Training loss: 0.1645

Validation loss: 0.1589

5.92

Job

Playground



OpenAI: Pricing

API calls are priced based on the number of tokens [[Link to OpenAI Pricing](#)]

OpenAI ⓘ

Log in

← Home

Platform Overview

Pricing

API Log in ↗

Documentation ↗

Developer Forum ↗

API Pricing

Contact sales

Our latest models

Reasoning models for complex, multi-step problems

OpenAI o3

Our most powerful reasoning model with leading performance on coding, math, science, and vision

Price

Input:
\$10.00 / 1M tokens

Cached input:
\$2.50 / 1M tokens

Output:
\$40.00 / 1M tokens

OpenAI o4-mini

Our faster, cost-efficient reasoning model delivering strong performance on math, coding and vision

Price

Input:
\$1.00 / 1M tokens

Cached input:
\$0.275 / 1M tokens

Output:
\$4.400 / 1M tokens



Summary

- How to use and adapt LLMs?
 - **Prompt Engineering**: Talk to the model better
 - **Supervised Fine-Tuning (SFT)**: Teach the model new tricks
 - **Retrieval-Augmented Generation (RAG)**: Let the model look things up
 - **LLM Agents**: An AI that can think, decide, and act – Not just answer a one-off question like ChatGPT
- Example use cases
 - Prompt engineering for loan outcome prediction
 - OpenAI API
 - RAG with local LLMs





THANK YOU

Stevens Institute of Technology
1 Castle Point Terrace, Hoboken, NJ 07030