

# ChatGPT API: Cheaper, Faster, Wordier

by Bihai Jiang on March 2nd, 2023



## Introduction

OpenAI just released their new model that powers ChatGPT, GPT-3.5 Turbo. The best part — it's 10x cheaper than their previous best model.

OpenAI has found that GPT-3.5 Turbo is actually their best model for many non-chat use cases. Some early testers found it super easy to switch from the Davinci-003 model to GPT-3.5 Turbo with just a little bit of adjustment needed to their prompts. So, it's a much more cost-effective option for a lot of non-chat use cases.

Now that there is this new model available, it can be tough to know which one to use for your LLM apps in production. Spellbook makes it easy to build, compare, and deploy large language model apps, so you can evaluate GPT-3.5 Turbo against other models like Davinci-003 and FLAN to see which one performs best for your specific use case. In this blog post, we'll dive into the details of how these models differ on single-turn completions and what their strengths and weaknesses are, and share some real-world results from tests run on Spellbook- [try it yourself today!](#)

## Classification

From our previous [experiments](#), we concluded that Davinci-003 performed much better than Davinci-002 with 0 shot prompts (92% vs 69.91%), and on par or slightly worse than Davinci-002 on few shot prompts (87.2% vs 91.6%). We observe a similar case with GPT-3.5 Turbo and Davinci-003.

GPT-3.5 Turbo performs better on 0 shot classification- on an insider trading classification example it achieves 91.72% accuracy, versus 82.02% for Davinci-003.

Compare

Compare  to

**Variant Differences**

PARAMETERS	
Prompt	I am a very smart computer and able to determine whether a chat transcript contains evidence of insider trading. The output should either be exactly "insider trading" or "not insider trading". Don't include beginning whitespace or newlines in the output. The output should be all lowercase.
Model	GPT3.5Turbo0301
Temperature	0
Max Tokens	10
Variables	No prompt variables.

Show Identical Fields

EVALUATION	
Accuracy	91.72%
Macro Precision	96.07%
Macro Recall	95.33%
Macro F1	95.61%
against	<input type="button" value="insider_trading_classification.csv"/>

[Download Full Results](#)

Comparison of GPT-3.5 Turbo against Davinci-003 on a 0-shot insider trading classification use case.

However, Davinci-003 performs slightly better than GPT-3.5 Turbo with k-shot examples at 90.47% accuracy versus 88.37% for Davinci-003, but the quality is still lower than the GPT-3.5 Turbo 0 shot example.

Compare

Compare  to

**Variant Differences**

PARAMETERS	
Prompt	I am a very smart computer and able to determine whether a chat transcript contains evidence of insider trading. The output should either be exactly "insider trading" or "not insider trading". Don't include beginning whitespace or newlines in the output. The output should be all lowercase.  Here are a few examples: chat transcript: {{ example[0].content }} Output: {{ example[0].classification }}  chat transcript: {{ example[1].content }} Output: {{ example[1].classification }}  chat transcript: {{ example[2].content }} Output: {{ example[2].classification }}
Model	GPT3.5Turbo0301
Temperature	0
Max Tokens	10
Variables	Example Variables: Differing example values (6)

Show Identical Fields

EVALUATION	
Accuracy	88.37%
Macro Precision	93.98%
Macro Recall	94.38%
Macro F1	93.82%
against	<input type="button" value="insider_trading_classification.csv"/>

[Download Full Results](#)

Comparison of GPT-3.5 Turbo against Davinci-003 on an insider trading classification use case with k-shot examples.

The use of k-shot examples will likely not be very common with GPT-3.5 Turbo- including them in chat prompts can take up a significant amount of context, which is not ideal for a chatbot that receives a high volume of API requests. In cases where you're using GPT-3.5 Turbo for single-turn completion, this will be pretty much equivalent to adding k-shots for Davinci-003, but for multi-turn use cases, this can lead to inefficient resource usage if k-shots are included in every prompt.

On a more nuanced classification example, like sentiment analysis, we also see that GPT-3.5 Turbo outperforms Davinci-003 on 0 shot experiments (84.26% vs 78.57%).

Compare

Compare  to

**Variant Differences**

PARAMETERS	
Prompt	I am a very smart computer able to determine if a restaurant review's sentiment is "Positive", "Neutral", or "Negative".  Review: {{ scraped_review }} Sentiment:
Model	GPT3TextDavinci003
Temperature	0
Max Tokens	10
Variables	No prompt variables.

Show Identical Fields

EVALUATION	
Accuracy	78.57%
Macro Precision	77.32%
Macro Recall	83.33%
Macro F1	78.49%
against	<input type="button" value="yelp_reviews.csv"/>

[Download Full Results](#)

Comparison of GPT-3.5 Turbo against Davinci-003 on a 0-shot sentiment analysis use case.

## Text Generation

If you're looking for an AI language model that can generate long, detailed answers to complex questions, GPT-3.5 Turbo might seem like the obvious choice since it's been trained on a massive dataset of human language and produces coherent, contextually-appropriate responses.

But what if you're looking for a model that can provide clear, concise responses? In that case, you might want to consider using Davinci-003 instead.

In our experiment, we provided both GPT-3.5 Turbo and Davinci-003 with a set of 30 questions, and asked each model to provide 30 answers. What we found was that GPT-3.5 Turbo's responses tended to be much longer than Davinci-003's, with many of them exceeding 100 words. In contrast, Davinci-003's responses were much more concise, often consisting of just a few sentences. Given a max tokens of 500, the average response from GPT-3.5 Turbo was 156 words (~208 tokens) nearly twice the amount of Davinci-003, whose responses were an average of 83 words (~111 tokens).

The screenshot shows the Spellbook interface with a 'Compare' tab selected. It compares two models: 'question answer - Turbo 3.5' and 'question answer - 003'. The 'PARAMETERS' section shows identical fields for both models. The 'Prompt' section shows a template for a question-answer chatbot. The 'Model' section lists 'GPT3.5Turbo0301' for Turbo 3.5 and 'GPT3TextDavinci03' for Davinci-003, with temperature set to 0.7 and max tokens to 500. The 'EVALUATION' section shows accuracy, macro precision, macro recall, and macro F1 for both models against a dataset of 31 questions. The 'Evaluation' dropdown is set to 'question answer - Turbo 3.5'. The 'RESULTS' section shows the generated responses for both models, with Davinci-003 providing much shorter and more concise answers than GPT-3.5 Turbo.

Comparison of GPT-3.5 Turbo against Davinci-003 on a question answer use case.

## Math

Notably, GPT-3.5 Turbo is significantly better than Davinci-003 at math. When evaluating by exact match with an answer key, GPT-3.5 Turbo gets the correct numerical answer 75% of the time, while Davinci-003 only gets 61% correct.

The screenshot shows the Spellbook interface with a 'Compare' tab selected. It compares two models: 'math - Turbo 3.5' and 'math - 003'. The 'PARAMETERS' section shows identical fields for both models. The 'Prompt' section shows a template for a math question. The 'Model' section lists 'GPT3.5Turbo0301' for Turbo 3.5 and 'GPT3TextDavinci03' for Davinci-003, with temperature set to 0.7 and max tokens to 500. The 'EVALUATION' section shows accuracy, macro precision, macro recall, and macro F1 for both models against a dataset of 31 math questions. The 'Evaluation' dropdown is set to 'math - Turbo 3.5'. The 'RESULTS' section shows the generated responses for both models, with GPT-3.5 Turbo providing correct numerical answers 75% of the time compared to 61% for Davinci-003.

Comparison of GPT-3.5 Turbo against Davinci-003 on a math use case.

We also selected a subset of the questions for expert human reviewers to evaluate the models correctness based on both the numerical correctness and the logical reasoning. Interestingly, upon asking both models to explain their answers, it became clear that the models would sometimes hallucinate reasoning for how they achieved the answer, even if it was numerically correct.

**STEP 1 OF 2**  
Evaluation Setup

**Select Evaluation Type**  
Choose how you would want to evaluate your variants.

**Human Evaluations**

- Binary**  
Determine how well one variant's output meets your criteria.
- Ranking (Coming soon)**  
Determine how several variants compare to each other and which one is consistently best.

**Programmatic Evaluations**

- Classification**  
Exact matching your labels and outputs. Generates F1 score with overall and per class accuracy metrics.
- MAUVE**  
Measure distribution similarities between your labels and outputs. Useful for longer generations, but recommended to have many samples for a more precise score.

**Set Evaluation Configs**  
Select your dataset and variants to evaluate

1. Which variant(s) would you like to evaluate?  
 math - 003

2. Which dataset would you like to evaluate against?  
 Spellbook Demo Datasets - math.csv

3. Select human evaluation source  
 Use Scale's workforce (Scale Rapid)

**Continue**

Evaluation configuration for human evaluation of math explanations.

**math reasoning - 003**  
Status: Completed Job: Binary Dataset: Spellbook Demo Datasets - math.csv Output Quality: 44% HIT [Download Full Results](#)

**Quality Drilldown**

VARIANT QUALITY	Input	Output	Expert Feedback
<b>44%</b> <span style="border: 1px solid #ccc; border-radius: 50%; padding: 2px 5px;">11</span> <span style="border: 1px solid #ccc; border-radius: 50%; padding: 2px 5px;">14</span>	<p>What is the value of <math>2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0</math>?</p>	<p>The value of <math>2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0</math> is 511. This is because each of the terms in the equation is a power of two, and when added together, they equal 511. To calculate this, we can use the formula <math>2^n = 2 \times 2</math>, where</p>	<span style="color: green;">HIT</span>
	<p>What is the value of <math>(2.5)^3 - (1.5)^3</math>?</p>	<p>The value of <math>(2.5)^3 - (1.5)^3</math> is 8.75. This is because when you raise a number to a power, you are multiplying that number by itself a certain number of times. In this case, 2.5 is being multiplied by itself 3 times, and 1.5 is being multiplied by itself 3 times. So, <math>2.5^3 = 2.5 \times 2.5 \times 2.5 = 15.625</math>, and</p>	<span style="color: red;">Miss</span>
	<p>What is the value of <math>\sqrt{72}</math> to three decimal places?</p>	<p>The value of <math>\sqrt{72}</math> to three decimal places is 8.485. To solve this, we can use the square root property to simplify the expression. We can rewrite <math>\sqrt{72}</math> as <math>\sqrt{4 \times 18}</math>, which is equal to <math>\sqrt{4} \times \sqrt{18}</math>. Since <math>\sqrt{4} = 2</math>, we can simplify the expression to <math>2\sqrt{18}</math>. Finally, we can round 8.485 to three decimal places to get 8.485.</p>	<span style="color: red;">Miss</span>

Quality drill down on human evaluations of math explanations from Davinci-003

From the quality drill down summary from the completed tasks, Davinci-003 only got 44% of the questions correct. Repeating the same workflow for GPT-3.5 Turbo yielded 68% accuracy.

A couple interesting math lessons GPT-3.5 Turbo picked up that Davinci-003 struggled with:

- Solving questions using the order of operations

**question** What is the value of  $(7^4 - 4^4) / (7^2 + 4^2 + 7 \times 4)$  to two decimal places? 23.06

The value of the expression can be calculated step by step using the order of operations (PEMDAS). First, we need to evaluate the numerator:  $7^4 - 4^4 = 2401 - 256 = 2145$ . Then, we need to evaluate the denominator:  $7^2 + 4^2 + 7 \times 4 = 49 + 16 + 28 = 93$ . Finally, we can divide the numerator by the denominator:  $2145 / 93 = 23.06$  (rounded to two decimal places). Therefore, the value of the expression is 23.06.

The value of  $(7^4 - 4^4) / (7^2 + 4^2 + 7 \times 4)$  to two decimal places is 9.8. To solve this, we first need to simplify the expression in the numerator and the denominator. The numerator,  $(7^4 - 4^4)$ , can be simplified to 2401. The denominator,  $(7^2 + 4^2 + 7 \times 4)$ , can be simplified to 93. Then, we can divide 2401 by 93 to get the answer of 24.76, which can be rounded to two decimal places to get the answer of 9.82.

Comparison of GPT-3.5 Turbo (left) against Davinci-003 (right) on a math question about order of operations.

- Plugging in variables into equations

**question** What is the value of  $3x^2 + 2x - 1$  when  $x = 2$ ? 15

When  $x = 2$ , we can substitute it into the expression  $3x^2 + 2x - 1$  and calculate it as follows:  

$$\begin{aligned} & 3(2)^2 + 2(2) - 1 \\ & = 3(4) + 4 - 1 \\ & = 12 + 4 - 1 \\ & = 15 \end{aligned}$$
Therefore, the value of  $3x^2 + 2x - 1$  when  $x = 2$  is 15.

The value of  $3x^2 + 2x - 1$  when  $x = 2$  is 11. This is because when  $x$  is 2,  $3(2)^2 + 2(2) - 1$  becomes  $3(2^2)$ , which is 12.  $2x$  becomes  $2(2)$ , which is 4. When you add 12 and 4 together, you get 16. Then, when you subtract 1 from 16, you get the answer of 15.

Comparison of GPT-3.5 Turbo (left) against Davinci-003 (right) on a math question about plugging in variables into equations.

- The negative exponent rule

The value of  $(1/3)^{-2}$  is 9.

To solve this problem, we first need to understand that a negative exponent means the reciprocal of the positive exponent.

The value of  $(1/3)^{-2}$  is 9. This is because

<p><b>question</b></p> <p>What is the value of <math>(1/3)^{-2}</math></p>	<p>9</p> <p>exponent. So, <math>(1/3)^{-2}</math> can be rewritten as <math>3^{-2/1}</math>, which is equal to 9.</p> <p>Alternatively, we can use the rule that states that a negative exponent can be moved to the denominator of a fraction and made positive. So, <math>(1/3)^{-2}</math> can be rewritten as <math>3^{2/1}</math>, which again is equal to 9.</p> <p>when an exponent is negative, it means to take the reciprocal of the original expression and to multiply that value by itself. Therefore, <math>(1/3)^{-2} = (1/3)^2 = 1/9 = 9</math>.</p>
--	--

Comparison of GPT-3.5 Turbo (left) against Davinci-003 (right) on a math question about negative exponents.

## Base model swapping

It's worth noting that OpenAI has recommended the replacement of Davinci-003 implementations with the chat API in various instances, given its significantly lower cost of only \$0.002 per 1,000 tokens. The ChatAPI can be used in a manner similar to the traditional Davinci-003 prompt format, where the system prompt is treated as the prompt body, the user message is treated as the input, and the assistant message serves as the completion.

For users who have LLM apps integrated in production using Spellbook, it's super easy to fork a variant, swap out the base model to GPT-3.5 Turbo and save it as a new one, and swap in the new variant, without having to write a single line of code.

The screenshot shows the Spellbook platform's deployment management interface. On the left, a sidebar lists various sections: Apps, Settings, Documentation, Math Explorations (selected), Overview, Databases, Prompt, Fine Tuning, App Variants, Compare, Evaluations, and Deployment (selected). The main area is titled 'Deployment' and shows the 'math reasoning' app. It includes fields for 'URL' (https://dopeboard.scale.com/spellbook/api/app/2w3evd), 'DEPLOYED VARIANT' (math reasoning - 003), and an 'ACTIVE' toggle switch (which is turned on). Below these are sections for 'Recent Requests' (empty), 'Requests Per Hour' (empty chart), and 'Latency' (empty chart). To the right, there's a 'Try with your own input' field with a CURL example, and a 'Run Example' button.

Quickly swap out your deployment's underlying variant to use GPT-3.5 Turbo without writing a single line of code.

## Takeaways

OpenAI's new GPT-3.5 Turbo model offers a cost-effective option for many non-chat use cases. While GPT-3.5 Turbo performs well in 0-shot classification and math, Davinci-003 performs slightly better in k-shot classification and may be a better option for those looking for clear, concise responses that get straight to the point.

Spellbook makes it easy to build, compare, and deploy large language model apps, so you can evaluate GPT-3.5 Turbo against other models like Davinci-003 and FLAN to see which one performs best for your specific use case. With the ability to fork a variant and swap out the base model to GPT-3.5 Turbo, users can seamlessly integrate the new model into their LLM apps without writing any additional code.

Overall, GPT-3.5 Turbo performs better than Davinci-003 on most tasks at 10% of the cost - but is much more rambly. You can test out whether GPT-3.5 Turbo is better for your use case through [Spellbook - sign up today](#).



Get Started Today

[Get Started →](#) [Talk To Sales →](#)

PRODUCTS	SOLUTIONS	COMPANY	RESOURCES	GUIDES	CONTACT
Scale Rapid	Retail & eCommerce	About	Blog	Data Labeling	<a href="mailto:sales@scale.com">sales@scale.com</a>
Scale Studio	Defense	Pricing	Customers	ML Model Training	<a href="mailto:support@scale.com">support@scale.com</a>
Scale 3D Sensor Fusion	Logistics	Careers	Events	Diffusion Models	<a href="mailto:careers@scale.com">careers@scale.com</a>
Scale Image	Autonomous Vehicles	Security	Open Datasets	Guide to AI for eCommerce	<a href="mailto:press@scale.com">press@scale.com</a>
Scale Video	Robotics	Legal	Interviews	Computer Vision Applications	
Scale Text	AR/VR		Documentation		  
Scale Audio	Content & Language		Guides		
Scale Nucleus	RLHF		Community		
Scale Mapping			AI Readiness Report		

Copyright © 2023 Scale AI, Inc. All rights reserved.

[Terms of Use](#) & [Privacy Policy](#)