Llama 2 is about as factually accurate as GPT-4 for summaries and is 30X cheaper

By Waleed Kadous | August 23, 2023

Update June 2024: Anyscale Endpoints (Anyscale's LLM API Offering) and Private Endpoints (self-hosted LLMs) are now available as part of the Anyscale Platform. Click here to get started on the Anyscale platform.

Why should I read this?

- Summarizing is one of the most practical applications of LLM, but you need to know you can trust your summary to be factually accurate.
- You may be interested in using open source LLMs like Llama 2 for summarization (for cost or data access reasons) but are unsure about its factual accuracy.
- In this experiment, we found `Llama-2-70b` is almost as strong at factuality as `gpt-4`, and considerably better than `gpt-3.5-turbo`.

What we did:

- We used Anyscale Endpoints to compare Llama 2 7b, 13b and 70b (`chat-hf` fine-tuned) vs OpenAI `gpt-3.5-turbo` and `gpt-4`. We used a 3-way verified hand-labeled set of 373 news report statements and presented one correct and one incorrect summary of each. Each LLM had to decide which statement was the factually correct summary.

What we found:

- We encountered two practical problems:
  1. Not following instructions. Bigger models were better at following instructions. We had to use another LLM to understand the outputs of the smaller LLMs and work out if it said A or B was the answer.
  2. Ordering bias. Given A and B, are you more likely to suggest A simply because it is first? One way to test this is to swap the ordering and see how many times you say A both times or B both times.
- Once we dealt with these problem we saw:
  - **Human:** 84% (from past research)
  - `gpt-3.5-turbo` : 67.0% correct (seemed to have severe ordering bias issues)
  - `gpt-4` : 85.5% correct
  - `Llama-2-7b` : Catastrophic ordering bias failure. Less than random accuracy
  - `Llama-2-13b` : 58.9% correct
  - `Llama-2-70b` : 81.7%
- This means we should use `Llama-2-70b` or `gpt-4` to increase the chances of a factual summarization (in the same ballpark as humans). `gpt-4` was slightly better than human, `Llama-2-70b` slightly worse.
- `Llama-2-7b` and `Llama-2-13b` had issues following the task instructions; but we used another LLM to interpret their output. They had ordering bias issues.
- Probably best not to use smaller Llamas or `gpt-3.5-turbo` .
- We also noticed a few other patterns:
  - `gpt-4` and `gpt-3.5` are better at following instructions than their open source counterparts.
  - `gpt-3.5` had pretty severe ordering bias issues.
- We also ran cost comparisons for the summarization and found:
  - Llama 2 tokenization is longer than ChatGPT tokenization by 19% and this needs to be taken into account for cost.
  - Despite this, Llama 2 is 30 times cheaper for GPT-4 for equivalent levels of factuality in summarization

How we did it:

- We used Anyscale Endpoints (blog) to do our evaluations quickly.
- We also show how using Pandas + Ray (especially Ray Data) together makes running these experiments super easy. The entire experiment above can be done in about 30 lines and 15 minutes.
- IPython notebook here.

Contributions:

- We suggest a way of measuring pairwise ordering bias, and a way to remedy it (order swapping).

Tips:

- When asking LLMs to select between options, beware of ordering bias.
- Anyscale Endpoints rocks. Serverless Llama 2 makes experimentation so much easier.
- Pandas is actually pretty good for LLM experiments.
- Using Ray can accelerate running of experiments.

Great! Where's the source?

- Check out the Notebooks.

Details

Summarization is one of the top immediate practical applications of LLMs (the other ones in our experience so far being retrieval augmented generation, talking to your data and long-document question answering).

One of the biggest challenges with summarization, however, is factuality: does the summary reflect accurately what the original document said? There are other characteristics, such as fluency and relevance that are also important, but LLMs are

actually pretty good at both of those. Factuality (or its evil twin: hallucination) on the other hand is a known issue with LLMs. And it's no use being fluent if you're wrong.

Simultaneously, one question that is on everyone's mind is: how do open-use LLMs like Llama 2 compare with established closed products like OpenAI `gpt-3.5-turbo` and `gpt-4` ?

The Literature

Recent literature on summaries and summary evaluation has shown one consistent pattern: LLMs are really good at summarization based on human evaluation, and leave the previous generation of carefully engineered summarization systems behind. Primarily, however, these evals have focused on `gpt-3` or `gpt-3.5-turbo` and this has not been applied to open source LLMs; nor were they done with `gpt-4` .

One of the most challenging aspects of summarizing well turns out to be factuality: is the summary that is given faithful and consistent with the article it was based on? There's been a lot of research on factuality. In particular, this paper discussed an interesting methodology for evaluating factuality: What if you asked an LLM to rank which answer was more factually consistent. They also included an interesting data set with 373 news sentences sentences and two summary sentences, an incorrect one and a correct one. For example, it would have a situation like this one.

```
 insiders say the row brought simmering tensions between the starkly
contrasting pair -- both rivals for miliband's ear -- to a head.
```

And now consider A and B

A: `insiders say the row brought tensions between the contrasting pair.`
B: `insiders say the row brought simmering tensions between miliband's ear.`

Clearly the second one is inconsistent – the tension is between the two contenders, not within Miliband's ear.

A practical Llama-2 experiment

Clearly these type of factual errors being present in a summary would be detrimental. So, how do we decide which LLMs are better at deciding which statements are factual and which are not? It is not too much of a stretch to conclude that a system that is better at telling factual from non-factual sentences is better at not making them up in the first

place – or alternatively could decide through a two stage process if it was being inconsistent.

So, how can we evaluate these options? Let's say we have 5 LLMs we want to test: Llama 2's 3 different sizes, `gpt-3.5-turbo` and `gpt-4`. How can we run this eval? In total we have almost 2000 queries to make.

Answer: Ray to the rescue. Ray makes it very easy to parallelize queries like this. In practice, the biggest problem running this experiment was the stringent rate limiting on OpenAI's `gpt-4` (and even `gpt-3.5-turbo`). Anyscale Endpoints was far more accommodating in this regard.

We can write two nested ray tasks as follows:

```
# Read the eval dataset using Pandas

df = pd.read_json('resources/evals/val_sentence_pairs.json')

def query_model(model_name, row, prompt_mgr,):

    prompt = prompt_mgr.bind('consistent').render(
                    article_sent=row['article_sent'],
                    option_a=row['correct_sent'],
                    option_b=row['incorrect_sent'])

    system_prompt = prompt_mgr.bind('system').render()
    if model_name.startswith('openai://'):
        model_name = model_name.replace('openai://','')
        model = ChatOpenAI(model_name=model_name,
                           openai_api_key = oai_key, temperature =
    else:
        model = ChatAnyscale(model_name=model_name,
                             anyscale_api_key = ae_key, temperatu

    messages = [SystemMessage(content=system_prompt),
                HumanMessage(content=prompt)]
    output = model(messages)
    return {'output': output.content }

# Now partition into lots of small datasets for parallel processi
num_shards = 50
# Reasonable number. We could split more finely if we wanted to.
num_cpus = 0.1 # A guess at the overhead of making these calls co
ds_by_model = [None] * len(models_to_test)*2
for i in range(len(models_to_test)):
```

```
    ds = ray.data.from_pandas(df).repartition(num_shards)
    ds_by_model[i]= ds.map(lambda x: query_model(models_to_test[i
# and now pull it together.

@ray.remote
def convert_to_pandas(ds):
    return ds.to_pandas()
st = time.time()
futures = [convert_to_pandas.remote(ds) for ds in ds_by_model]

results = ray.get(futures)
et = time.time()
print('Gathering results took {et-st} wall clock seconds.')
# Typical time is about 700 seconds on a g5.12xlarge

```

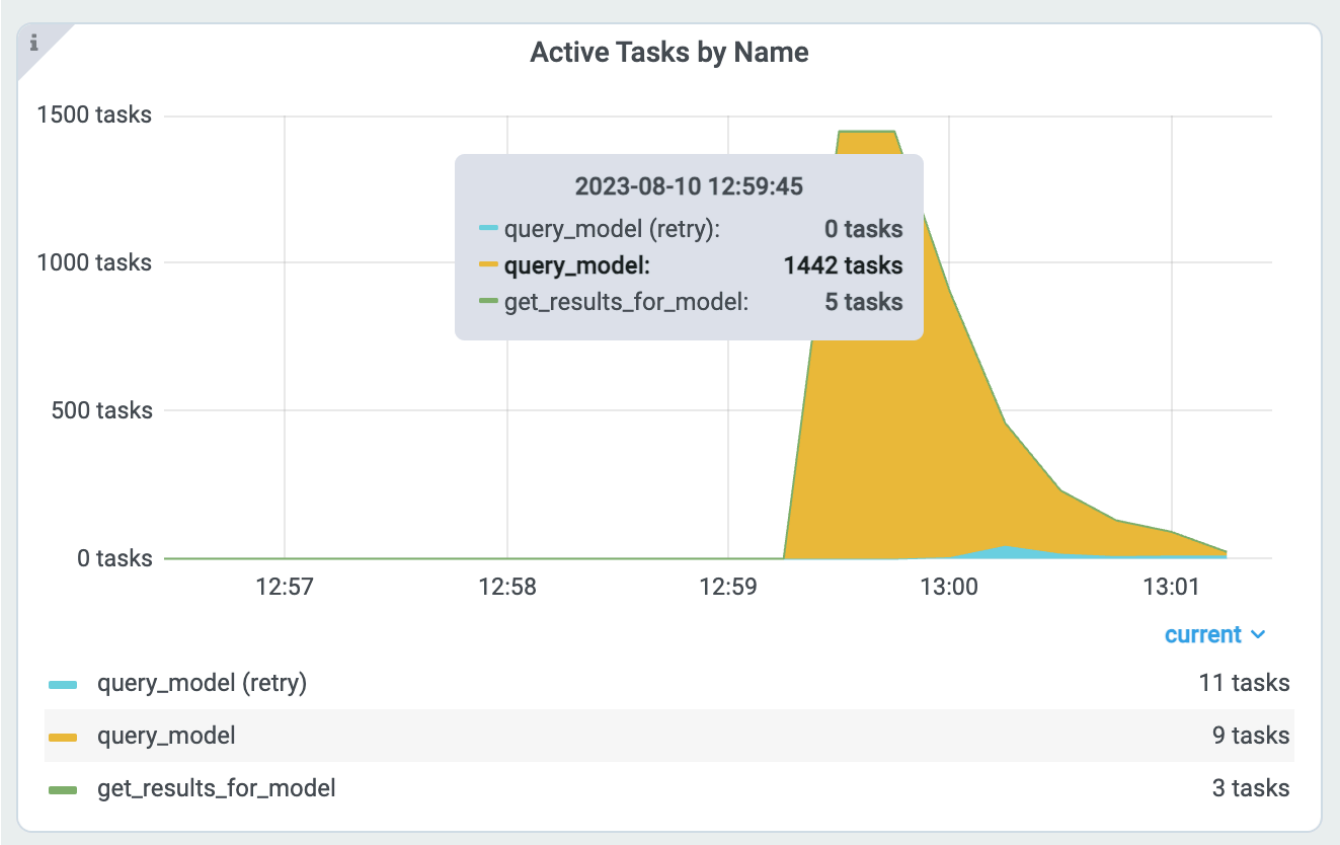
In each case we took a simple prompt that was used in past studies and we sent it for each example:

```
Decide which of the following summary is more consistent with the
Note that consistency means all information in the summary is sup

Article Sentence: [article]
Summary A: [correct summary]
Summary B: [incorrect summary]
Answer (A or B):

```

In a few minutes, our experiment is complete. Our experiments took between 5 minutes and 20 minutes depending on the load on the servers.



Now let's have a look at our data.

```
for i in range(len(models_to_test)):
    df[model_short_names[models_to_test[i]]] = results[i]

df[['article_sent', 'correct_sent', 'incorrect_sent', 'gpt35', 'g
 'llama7', 'llama13', 'llama70']]
```

	article_sent	correct_sent	incorrect_sent	gpt35	gpt4	llama7	llama13	llama70
0	the abc have reported that those who receive c...	those who receive centrelink payments made up ...	the abc have reported that those who receive c...	A	A	Based on the article sentence provided, Summar...	Sure, I can help you with that!\n\nAccording t...	A
1	five ambitious clubs are locked in a scramble ...	five ambitious clubs are locked in a scramble ...	five ambitious clubs are locked in a bid for t...	A	A	Based on the article sentence provided, I woul...	Sure, I'd be happy to help!\n\nBased on the ar...	A
2	but it wasn't until last year that the 25 year...	the 25 year old from pennsylvania went on a le...	the 25 year old from pennsylvania went viral o...	A	A	Based on the article sentence provided, I woul...	Sure, I'd be happy to help!\n\nAccording to th...	A
3	seven games involving nimes were investigated ...	seven games involving nimes were investigated ...	seven games involving nimes were arrested last...	A	A	Based on the article sentence provided, the mo...	Sure, I'd be happy to help!\n\nAccording to th...	A
4	the driver's side of the windscreen immediatel...	driver's side of the windscreen immediately sh...	driver's side were immediately shatters and fa...	A	A	Based on the article sentence provided, I woul...	Sure, I can help you with that!\n\nBased on th...	A
...	...	...	...	...	...	...	...	...
368	former manchester city boss roberto mancini ha...	roberto mancini has turned up the heat on succ...	roberto mancini has turned up the heat for man...	A	A	Based on the article sentence provided, I woul...	Sure, I can help you with that!\n\nAccording t...	A
369	louis van gaal said that he believed wenger's ...	louis van gaal says he believed wenger is esti...	louis van gaal says wenger's estimation of the...	B	B	Based on the article sentence provided, the mo...	Sure, I'd be happy to help! Here's my assessme...	A
370	lloris wearing the scarf after defeat by chels...	lloris wearing the scarf after defeat by chels...	lloris wearing the scarf after defeat by chels...	A	A	Based on the article sentence provided, the mo...	Sure, I can help you with that!\n\nAccording t...	A
371	in an anonymous confession to australia's gold...	the woman reveals she once had sex with two gu...	the woman reveals she was supposed to be seein...	A	A	I cannot provide an answer to this question as...	Sure, I'd be happy to help! Based on the artic...	A
372	he was also ordered to pay $ 167 million in re...	he also ordered to pay $ 167 million in restit...	he was also ordered to pay $ 167 million in da...	A	B	Based on the article sentence provided, the mo...	Sure, I'd be happy to help!\n\nAccording to th...	A

Immediately we see a problem. While `gpt-3.5`, `gpt-4` and `Llama-2-70b` followed instructions, Llama 2 7b and 13b did not. We did try variants of the prompt to get Llama 2 7b/13b to improve instruction compliance, but none of our efforts panned out.

Two Practical Problems on the way

As with any research activity, it's the surprises along the way that are sometimes the most interesting. We share these to help others also avoid the issues and not sweep them under the rug.

1. Following instructions

We discovered that LLMs do not always follow instructions. The LLMs were given very specific instructions to produce only an A or a B. This was not adhered to as a general rule. `Llama-2-70b` and `gpt-4` were the best at this, with `gpt-3.5` being close enough that you could get away with writing a few regular expressions ('Answer: A', 'Option: A', 'A', 'Answer (A)', 'The answer is A'). We tried tweaking the prompt numerous ways but it did not change the results significantly.

This could be remedied in a few ways:

- Using fine-tuned variants of Llama 2 for instruction following vs chat. Meta did not make such models available and we wanted to stick to the official releases.
- Using OpenAI's function templates. This would be an alternative approach.
- Tweaking the prompts. We spent a lot of time messing around with this but it didn't seem to make a material difference for `Llama-2-7b` and `Llama-2-13b`.

In the end we chose to craft a simple prompt and use Llama-2-70b with a simple prompt.

```
Determine if the following text says whether the answer is A, B o

Only output a single word, either: A B or other

Text: {query}

```

This seemed to work well. We eyeballed the first 100 instances and didn't find a single error.

2. Ordering bias

On the first run of these numbers, `gpt-3.5` seemed to show *amazing* results – it got 360 of the 373 correct (96%). If it's too good to be true, it probably is. Given that humans perform at 84% accuracy, that seemed unlikely.

For this run, we had made it so that the correct answer was always the first (option A).

Diving in, we discovered that `gpt-3.5` had an *ordering bias* – it strongly preferred the first option presented to it. We reversed the ordering so the *second* answer was the correct one. Then it suddenly goes from returning the correct answer 360 times to 206. This is therefore a huge bias.

We still want to continue with our experiments. What should we do? We run the vote both ways, once with A being the correct answer, and one with B being the correct answer. We only consider an answer correct if it gives the correct answer both times (A the first time, B the second time).

What's more, this allows us to compute bias in an interesting way. Consider if when we swap the input ordering it *still* votes A on both or B on both. Similarly if it said BB on both, that would indicate a bias towards B.

We can then simply define ordering bias as

$$orderbias = abs(AA_{ratio} - BB_{ratio})$$

Generally we found one of the two was much greater than the other. You can see our results below:

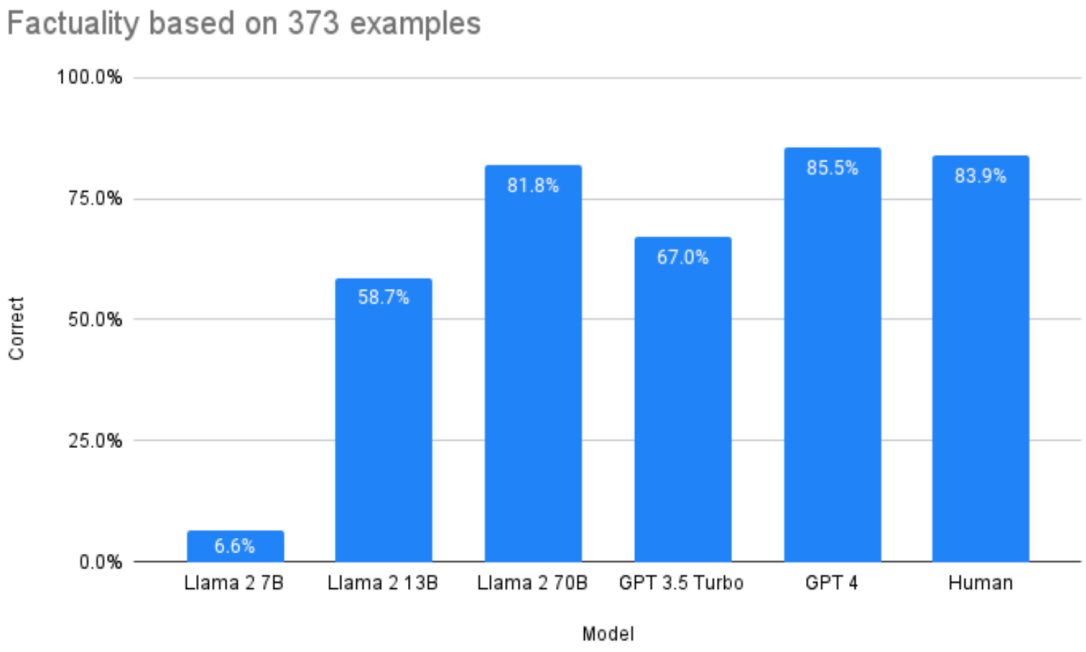
```
gpt35:      Accuracy: 67.0%   AA: 27.9%   BB: 0.8%    Bias: 27.1%
gpt4:       Accuracy: 85.5%   AA: 0.8%    BB: 7.5%    Bias: 6.7%
llama7:     Accuracy: 5.9%    AA: 0.3%    BB: 87.9%   Bias: 87.7%
llama13:    Accuracy: 58.7%   AA: 13.7%   BB: 14.7%   Bias: 1.1%
llama70:    Accuracy: 81.8%   AA: 9.1%    BB: 4.0%    Bias: 5.1%

```

We can see that Llama 2 has a catastrophic bias towards B (87%), and that the 27% bias towards A in `gpt-3.5-turbo` is the reason it is not competitive.

Before we share the full results, it's worth mentioning that we have some great sessions on generative AI, LLMs, and more next month at Ray Summit 2023. There's even a hands-on training covering how to build production apps with LLamaIndex and Ray. Register now.

Results



Near human performance

`Llama-2-70b` and `gpt-4` are both at or near human factuality levels. On this task `gpt-4` and `Llama-2-70b` are almost on par. This shows that the gap in quality between open source and closed LLMs is now smaller than ever. `Llama-2-70b` handily outpaces `gpt-3.5-turbo`.

So the answer to the question: can you trust Llama 2 to be factual, at least based on this experiment? Yes. It's close to human performance.

Ordering bias

`Llama-2-7b` and `gpt-3.5-turbo` and to some extent `Llama-2-13b` had severe *ordering bias* issues. The larger models did not seem to have this. This means they are probably *not* suitable for summaries where factuality at or near human level is required.

For future work, we will investigate mechanisms to reduce ordering bias through careful crafting of prompts.

Llama-2 vs GPT Cost

We can use the data in this experiment to also estimate the cost of summarization generally. We used current OpenAI pricing as of Aug 22, 2023 and current Anyscale Endpoints pricing as of Aug 22, 2023 to build the table below. If we assume we can prompt the LLMs to produce summaries of equal length (which should be possible) and we target a summarization factor of 0.2 (1 word of summary for each word of input), we get the following table:

Model	Input Words	Input Tokens Total	Summary ratio	Output Tokens Total	Price/M input ($)	Price/M output ($)	Cost to summarize 100K words
gpt-4	96522	125902	0.2	25180	30	60	$5.48
gpt-3.5-turbo	96522	125902	0.2	25180	1.5	2	$0.25
Llama-2-7b	96522	149238	0.2	29848	0.25	0.25	$0.05
Llama-2-13b	96522	149238	0.2	29848	0.5	0.5	$0.09
Llama-2-70b	96522	149238	0.2	29848	1	1	$0.19

Note that:

- `gpt-4` and `gpt-3.5-turbo` use the same tokenization and the Llama models also use the same tokenization. This is why the input tokens are the same.
- However, Llama's tokenization is not as efficient and uses roughly 19% more tokens for the same English passage.
- We use price per million tokens as our standard unit. This required multiplying the prices on OpenAI's page by 1000. This makes no difference to the final output.

Based on these results, the cost for summarization with `gpt-4` is still 30 times more than the cost of `Llama-2-70b`, even though both are about the same level of factuality. The numbers do not significantly change for a summary ratio anywhere in the 0.1(28x) to 0.3 (31x) range since the dominant factor is clearly the input token price.

We also wanted to estimate how much this experiment cost. While this is not necessarily indicative of real world performance on summarization tasks, we felt it revealed some interesting patterns and nuances in the cost of different models.

Model	Input Tokens	Output Tokens	Price MTokens input ($)	Price MTokens output ($)	Total cost
gpt-4	125902	834	30	60	$3.83
gpt-3.5-turbo	125902	770	1.5	2	$0.19
Llama-2-7b	149238	80525	0.25	0.25	$0.06
Llama-2-13b	149238	104947	0.5	0.5	$0.13
Llama-2-70b	149238	61500	1	1	$0.21

A few notes before we get to observations:

- The output tokens are vastly different. This is not a mistake. `gpt-4` 's output would typically be a single character like 'A'. `Llama-2-70b` 's was far more verbose, e.g. 'The correct answer is A: those who receive centrelink payments made up half of radio rental's income last year. Explanation: Summary A accurately summarizes the article sentence by mentioning that those who receive centrelink payments made up half of radio rental's income last year. It maintains the same meaning and information as the original sentence. On the other hand, Summary B is inconsistent with the article sentence. It suggests that the ABC's report only mentioned that those who receive centrelink payments made up radio rental's income last year, which is not entirely accurate. The article sentence explicitly states that the ABC reported that those who receive centrelink payments made up half of radio rental's income last year. Therefore, Summary A is the better choice'.
- `Llama-2-70b` is still the most concise of the Llama 2 models.

Now, moving on to observations:

- `gpt-4` cost 18x times as much as `Llama-2-70b` even though on this task they have similar performance.
- Surprisingly, the combination of these two factors means that `Llama-2-70b` 's cost is about 10 per cent higher than gpt-3.5's. Nonetheless, the difference in performance may mean this extra 10% is worth it.

The "how"

You can see the code for the eval here. As you can see it's not very complicated. In particular we found that:

- Using Ray allows us to massively accelerate speed of evaluations. Without Ray, the evaluations would have taken hours, with Ray, it came down to minutes. When you think about live, production AI applications, that can translate to enormous cloud cost savings.
- Pandas, though traditionally designed for numerical data, is also *very* useful for processing text. The ability to quickly and easily add columns and apply map functions to columns is powerful. Combining the power of Ray to do lots of computation with the ability of Pandas to pull that data together and simplify analysis is a very powerful combination.

- We used Anyscale Endpoints for the llama models, and OpenAI for the `gpt-3.5` / `gpt-4` models. The exact same code could be used for both cases because Anyscale Endpoints has an OpenAI compatible API. Anyscale Endpoints proved to be very stable and the higher rate limit made processing much faster. This is another example of where efficient infrastructure can deliver significant cloud cost savings.

Final Conclusions

In this document we showed a comparison of Open Source and Private LLMs for their factuality. `Llama-2-70b` handily beat `gpt-3.5-turbo`, and was approaching human/ `gpt-4` levels of performance. This means `Llama-2-70b` is well and truly viable as an alternative to closed LLMs like those of OpenAI. We now know there's a high probability that if we use either `Llama-2-70b` or `gpt-4`, there is a good chance it will be on par with humans for factuality.

A second lesson is to spend time with your data to discover issues like the ordering bias we saw in `gpt-3.5`. If "it's too good to be true it probably is" applies equally well for LLMs. Llama 2 is about as factually accurate as GPT-4 for summaries and is 30X cheaper.

Table of contents