Evaluating Chatbots

Dan DeGenaro
Georgetown University
drd92@georgetown.edu

Abstract

In this paper, we perform a simplistic, qualitative evaluation of a small selection of chatbots.

1 Introduction

1.1 Why compare multiple models?

Large Language Models (LLMs), language models which are pre-trained on vast amounts of data (overwhelmingly natural language text), are now ubiquitous. Such models excel at various natural language tasks, including tasks on which they were not explicitly trained. With the advent of instruction tuning (Ouyang et al., 2022) and reinforcement learning from human feedback (RLHF) (Christiano et al., 2017), many LLMs have been converted into chatbots capable of holding natural language conversations. Such models vary greatly in performance on various tasks, generally a function of their parameter count and the amount and utility of their training data. In this paper, we select three open-weight models that can be run locally, as well as three to which we connect via the Cerebras inference API.[1] We compare the six models for inference latency, as well as accuracy on a few basic tasks with the goal of understanding what makes a chatbot good.

1.2 Overview of transformer architecture relevance

All LLM architectures in common use today are based on the transformer architecture introduced by (Vaswani et al., 2017). Transformers rely on the attention computation introduced by (Bahdanau et al., 2014), which produces pairwise scores for every pair of inputs in a sequence of length n in $O(n^2)$ time. Such scores allow any downstream loss function to receive signal from any pair of inputs. Downstream tasks thus train the weights of the attention mechanism to learn to prioritize useful pairwise relationships between any two inputs.

This mechanism, along with advances in hardware such as cheaper and faster graphics processing units (GPUs) and generally cheaper compute overall, are responsible for the success of LLMs. Prior to the widespread use of the attention mechanism, language models were implemented with recurrent neural networks (RNNs) such as in (Mikolov et al., 2010). Recurrent networks suffered from two major problems which attention aimed to solve: long-context forgetting and lack of parallelizability.

The RNN architecture suffered from various information bottlenecks, particularly in encoder-decoder style architectures. Namely, a single context vector, generally taken as the final hidden state output of the encoder network, was expected to encode the entire source-side input effectively, to provide sufficient information on which the decoder could condition its outputs. In practice, this was often extremely lossy, and RNNs tended to struggle with longer inputs.

Additionally, even encoder-only or decoder-only RNNs would be limited by the long distance in the RNN's computation graph that distant information would need to travel. Famously, the LSTM network (Hochreiter and Schmidhuber, 1997), a modified RNN designed to handle longer contexts, was developed and somewhat successfully addressed this issue.

However, in an attention computation, every input is compared with every other input *directly,* allowing a much closer-to-lossless transfer of information across input tokens. It is this "instant recall" that brought about the LLM hype as well as genuinely impressive intelligent systems, capable of high-level conversation, translation, summarization, and many other challenging natural-language tasks.

[1] https://github.com/Cerebras/cerebras-cloud-sdk-python

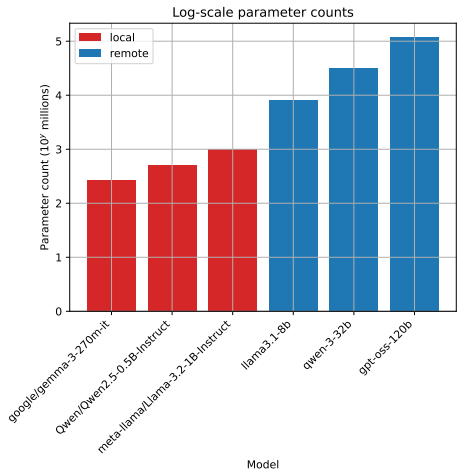

Figure 1: Log-scaled parameter count by model.

1.3 Research Questions

1. How does latency compare across locally-run and remote models accessed via API? How does it scale with parameter count? What about the number of tokens generated per unit time?

2. Which LLMs are able to handle basic tasks best? Which are most reliable, even on challenging tasks?

2 Methods

2.1 Hardware specifications and setup

All experiments with local models are conducted on a 16 GB Apple M4 chip, whose memory is shared by the CPU and MPS device, with all computation handled by the MPS system. Local models are loaded using the bfloat16 (Intel, 2018) specification as is common practice. All experiments with remote models use Cerebras as the inference provider.

2.2 Model selection rationale

We select six models produced by four different organizations, three of which are run locally (and the other three being accessed via the Cerebras inference API). We select a broad range of model sizes, conforming nearly to an exponential scaling by parameter count. This can be seen clearly in figure 1. This choice is motivated by work such as (Kaplan et al., 2020), which found that performance on a given task as measured by loss on a held out test set decreases roughly linearly as a function of the logarithm of the parameter count.

To this end, we select Google's Gemma-3-270M-IT (Gemma Team et al., 2024), Alibaba's Qwen2.5-0.5B-Instruct (Yang et al., 2024), and Meta's Llama-3.2-1B-Instruct (Grattafiori et al., 2024) to be run locally, and Alibaba's Qwen-3-32B (Yang et al., 2025), Meta's Llama3.1-8B (Grattafiori et al., 2024), and OpenAI's GPT-OSS-120B (OpenAI et al., 2025) to be accessed via API. A diverse selection of producing organizations is chosen to smooth out any organization-specific idiosyncrasies in model architectures, training techniques, and other modifications.

However, we still include two pairs of models in which each model in the pair is produced by the same organization, namely the two Qwen models and the two Llama models. We select an earlier iteration of Qwen distilled to few parameters (as compared with a newer and bigger version), and a later iteration of Llama distilled to few parameters (as compared with an older but bigger version). We aim to examine whether a distilled, new model outperforms a larger, older one in the case of Llama, while also examining the need for continuously updating these models in the first place by comparing an older, smaller Qwen with a supposedly superior, larger, newer version.

The other two models selected represent roughly the extremes of open-weight LLM sizes: the 270-million parameter version of Gemma-3 is among the smallest LLMs ever developed (smaller even than BERT or GPT-1, which didn't have the same capabilities) and the 120-billion parameter version of GPT-OSS, which is among the largest LLMs whose weights are publicly available.

2.3 Evaluation methodology and metrics

We evaluate all six models on an identical testbed of 12 prompts, without allowing the model's response to be conditioned on any prior history, yielding independent samples. We test each prompt five times for precision, choosing to forego a random seed and allowing a softmax temperature, top-p sampling, and all other generation hyperparameter settings enabled by default for all models.

We evaluate the model for a subjective accuracy (out of five trials) on every prompt across four tasks:

1. Basic chat functionality:
    a. Easy: `Tell me a joke.`
    b. Hard: `What's your favorite color?`

2. Question answering:
    a. Easy: `What is the capital of France?`
    b. Hard: `Explain photosynthesis.`
3. Machine translation (Chinese-English, both directions):
    a. Easy: `Translate the following into Chinese: "Good morning, how are you?"`
    b. Easy: `Translate the following into English: "早上好，你好吗？"`
    c. Hard: `Translate the following into Chinese: "The mitochondrion is the powerhouse of the cell."`
    d. Hard: `Translate the following into English: "线粒体是细胞的动力源。"`
4. Machine translation (Russian-English, both directions):
    a. Easy: `Translate the following into Russian: "Good morning, how are you?"`
    b. Easy: `Translate the following into English: "Доброе утро, как дела?"`
    c. Hard: `Translate the following into Russian: "The mitochondrion is the powerhouse of the cell."`
    d. Hard: `Translate the following into English: "Митохондрия — это электростанция клетки."`

These prompts were selected for minimality and broad coverage. Namely, assuming the model has some ability to do each of the four tasks presented to it, we judge the model's overall capability by testing whether it can complete a simple example of the task, as well as a more challenging example. We then rate the model's performance on each task using a 0-2 scale defined in Table 1.

We also document CPU, GPU, and disk usage for local models, as well as latency for both local and remote models (defined as the time needed to process a string end-to-end, including any preprocessing and string manipulations external to the computation by the model itself).

Score	Rationale
0	The model gives a reasonable/correct answer less than 2/5 times when presented with the simpler prompt.
1	The model gives a reasonable/correct answer at least 4/5 times when presented with the simpler prompt, but less than 2/5 times when presented with the harder prompt.
2	The model gives a reasonable/correct answer at least 4/5 times when presented with either prompt.

Table 1: How a model might achieve each score, 0-2.

3 Results

3.1 Performance on tasks

Key results are summarized in Tables 2 and 3. Overall, GPT-OSS and Qwen-3 were the top performers, while Llama-3.2 had the poorest scores as shown in Table 2. Gemma-3, Qwen-2.5, and Llama-3.1 were middling. Looking at the raw scores in Table 3, Gemma-3 actually performed best for simpler queries, while GPT-OSS was the best at handling more difficult queries. Gemma-3 notably struggles with difficult queries, while GPT-OSS is ranked 2nd for simple queries.

In general, all models performed better when translating either Chinese or Russian into English than in the other direction, likely because translation is fundamentally a generation task, and all the models used were trained primarily to produce English text. The Qwen models are clearly the best at translating Chinese, though in one case, Qwen-2.5 seemed to forget to finish a translation before beginning a rant about the topic of the text to be translated. Furthermore, Qwen-3 tended to tell jokes that did not make sense and anthropomorphize itself to excess, seemingly going for a "quirky" personality. There was no clear front-runner for Russian translation. Additionally, Llama-3.2 tended to mix up languages when translating, inserting closely related, or even correct words, in the wrong language (such as using the German or Spanish translation of the correct Russian word). Several models struggled with the word "mitochondrion," often defaulting to using English or trying to transliterate it into one language or the other.

3.2 Hardware considerations

Figures 2, 3, and 4 contain information about latency for each model. Notably, API models are unreasonably fast given their parameter counts as compared with the much smaller models run locally on an M4 chip. This is due to the API models being run on better hardware as well as engineering

Model	Chat	QA	CN-EN	RU-EN	Avg.
Gemma-3-270M	1	2	1	0	1
Qwen-2.5-0.5B	2	2	2	0	1.5
Llama-3.2-1B	0	2	0	0	0.5
Llama-3.1-8B	2	2	0	0	1
Qwen-3-32B	0	2	2	0	1
GPT-OSS-120B	2	2	2	0	1.5

Table 2: 0-2 rating as defined previously for each of the 4 tasks, along with a task-agnostic averaged rating.

Model	Chat	QA	CN-EN	RU-EN	Avg.
Gemma-3-270M	(5,2)	(5,5)	(4.5,0)	(3.5,0)	(4.5,1.75)
Qwen-2.5-0.5B	(5,5)	(5,5)	(4.5,4)	(1.5,0)	(4,3.5)
Llama-3.2-1B	(3,5)	(5,5)	(3,2.5)	(3,2.5)	(3.5,3.75)
Llama-3.1-8B	(5,5)	(5,5)	(2.5,4.5)	(2.5,3)	(3.75,4.375)
Qwen-3-32B	(1,0)	(5,5)	(5,5)	(3,3.5)	(3.5,3.375)
GPT-OSS-120B	(5,5)	(5,5)	(5,5)	(2.5,5)	(4.375,5)

Table 3: Precise scoring out of 5 trials for each prompt (translation tasks are averaged across the two directions) along with a task-agnostic average score. The first number in each tuple indicates the number of reasonable responses, out of 5, given to the easier prompt (or pair of prompts in translation tasks), while the second number indicates the same for the harder prompt (or pair of prompts). Notably, the easier prompt was not always addressed as well as the harder prompt for a given task.

optimizations suited to said hardware which are not available on an M4 chip's instruction set, such as forward pass compilation, KV-caching, and alternative implementations of the attention computation that are faster. Gemma-3 seems to run somewhat slow as compared to other models with double or quadruple its parameter count such as Qwen-2.5 and Llama-3.2. Indeed, Qwen-2.5 in particular is quite fast for its parameter count, generating the most tokens per second of any local model. That said, Llama-3.2 is about 5/6 the speed at twice as many parameters.

Figures 5, 6, and 7 indicate hardware utilization. CPU and disk usage appears pretty much identical for all the local models, while GPU utilization is comparably higher for Gemma-3 and Llama-3.2, but not for Qwen-2.5. This might indicate that in addition to being the fastest, Qwen-2.5 is also the most memory-efficient.

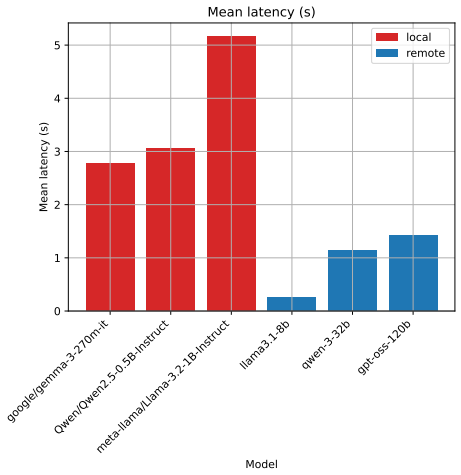

Figure 2: Latency of each model, averaged over all queries.

4 Discussion

Overall, larger models perform better, know more obscure things, respond more consistently, and are worth the expense in mission-critical scenarios where reliability is important. But for unimportant or casual use, smaller models can be just as useful as conversation partners, prone to serious errors just as humans are.

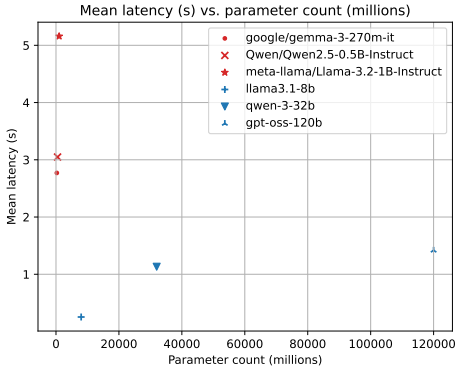

Figure 3: Latency of each model, averaged over all queries, versus parameter count.

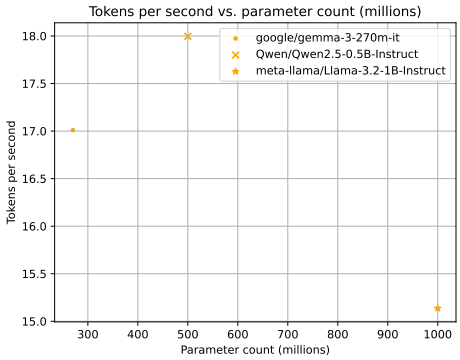

Figure 4: Tokens generated per second, averaged over all queries, versus parameter count.

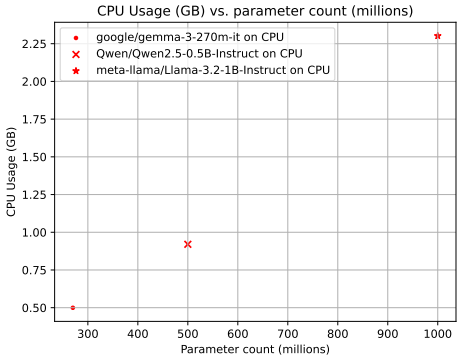

Figure 5: CPU utilization versus parameter count.

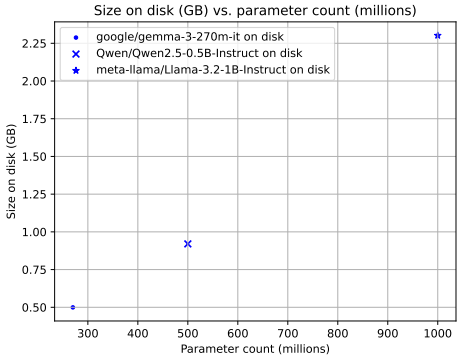

Figure 6: Disk utilization versus parameter count (nearly identical to Figure 5).

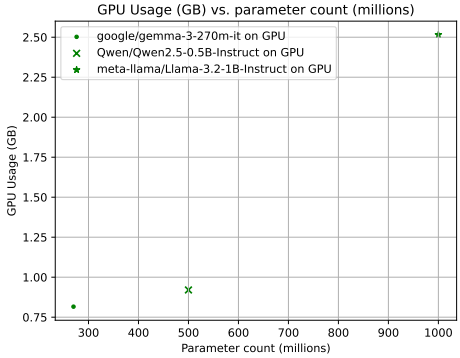

Figure 7: GPU utilization versus parameter count. Notably higher for Gemma-3 and Llama-3.2 as compared with their CPU/disk usage in Figures 5 and 6.

If speed is a must and only inference is needed, an API is the way to go. Inference API providers have invested greatly in fast hardware and compute-optimized code that allows responses to be returned extremely fast. Barring a slow internet connection, they are all but guaranteed to run faster, even when using very large models. Additionally, the need for a high parameter count often prevents the usage of truly high-performing LLMs on local hardware, given extremely limited memory. This is a further argument for using an API when performance is a must; the models that can be accessed via API are simply better than those that can fit on local hardware on most personal computers.

Interestingly, not all "easy" tasks were genuinely easier than their hard counterparts. As an example, some models struggled with translating "how are you?" appropriately, replacing it with phrases near-in-meaning but which were either overly colloquial, paraphrased, or metaphorically linked to the appropriate, accepted translations, while having a clearer idea about translating more advanced terminology, where there is less room for linguistic variation. This is arguably an artefact of the very subjective evaluation method, but is nonetheless an interesting problem. It shows that genuinely evaluating the performance of an LLM is very hard, particularly when there are many ways to say something. There is often "no best answer" to optimize for or to score against.

Other noteworthy findings include a general propensity to abstain from having opinions about things and to tell jokes about why scientists don't trust atoms (although both Llama models tend to repeat a joke about Pavlov and Schrödinger). All models, nearly 100% of the time, resoundingly re-

jected the contention that they should or could have a favorite color (Gemma sometimes frustratingly refusing to acknowledge the question and turning it back at the user), though a few begrudgingly acknowledged blue as a popular choice. This is clear evidence that all these models have seen similar RLHF data, and perhaps even the same data. Qwen-3 was particularly interesting in this regard, as it refused to name a favorite color, but freely acknowledged other aspects of its personality, claiming a genuine interest in how humans perceive and feel about color. It also referred to itself as a "byte-colored blob." Qwen-2.5 notably does not anthropomorphize itself this way. A final interesting note is about GPT-OSS-120B, which uses punctuation marks differently from the other models. It tends to prefer U+2011 rather than the hyphen (U+002D), as well as the narrow no-break space (U+202F), the n-dash (U+2013), the m-dash (U+2014), and stylized apostrophes and double quotes (U+2019, U+201C, U+201D) instead of the ordinary ones (U+0027 and U+0022) whereas the other models largely avoid these. Qwen-3 occasionally produces some of these, indicating it may have been trained with supervision from an OpenAI model.

5 Limitations

The limitations of this study are essentially, all. Very few models have been evaluated, and of those six, only half of them were run locally - it is unknowable what may be happening at the other end of an API call in principle, though Cerberas' Python SDK is open source, and it may be possible to find out for sure.

That said, the problem of few models is still there. Even worse, few prompts have been evaluated. A serious test bed ought to include thousands of prompts, each evaluated a few times over as I have done here. This would likely require automatic evaluation or a large team of annotators who can validate responses as correct or not, as well as carefully constructed guidelines for evaluation.

Additionally, none of the authors speak Chinese or Russian natively, and are thus poor judges of translation quality. We frequently relied upon Google Translate as a sanity check, as well as to produce faithful translations of our prompts to be tested bidirectionally.

Finally, no strict guidelines were used to evaluate any response. Logic was fairly consistently applied, though not very carefully, and various biases, such as the order in which models were evaluated, could have influenced the scoring. A careful study would test independent inter-annotator agreement and attempt to quantify the reliability of the evaluation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024. Gemma: Open Models Based on Gemini Research and Technology. *arXiv preprint*. ArXiv:2403.08295 [cs].

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 181 others. 2024. The Llama 3 Herd of Models. *arXiv preprint*. ADS Bibcode: 2024arXiv240721783G.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Intel. 2018. bfloat16 - Hardware Numerics Definition.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *arXiv preprint*. ArXiv:2001.08361 [cs].

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari. Issue: 3.

OpenAI, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, and 107 others. 2025. gpt-oss-120b & gpt-oss-20b Model Card. *arXiv preprint*. ArXiv:2508.10925 [cs].

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 Technical Report. *arXiv preprint*. ArXiv:2505.09388 [cs].

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 Technical Report. *arXiv preprint*. ArXiv:2407.10671 [cs].