FEval-TTC: Fair Evaluation Protocol for Test-Time Compute

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

The performance of Large Language Models (LLMs) and the associated dollar costs of API calls can fluctuate over time, potentially invalidating conclusions drawn in prior research. To address this, we propose a ***F****air* ***Eval****uation protocol for* ***T****est-****T****ime* ***C****ompute* (FEval-TTC), designed to ensure consistent assessment of test-time compute (TTC) methods, regardless of such fluctuations. FEval-TTC focuses on evaluation of TTC methods that utilize underlying Chains-of-Thought (CoT). It supports evaluations across multiple LLMs on a diverse set of mathematical and commonsense reasoning datasets. The few-shot prompting and answer extraction processes are standardized across datasets, reducing both time and monetary overhead for researchers. Furthermore, we provide a cost modeling procedure that estimates both the token and dollar cost per query, facilitating equitable comparisons of prevalent TTC methods. We open-source FEval-TTC for public use at anonymized code link.

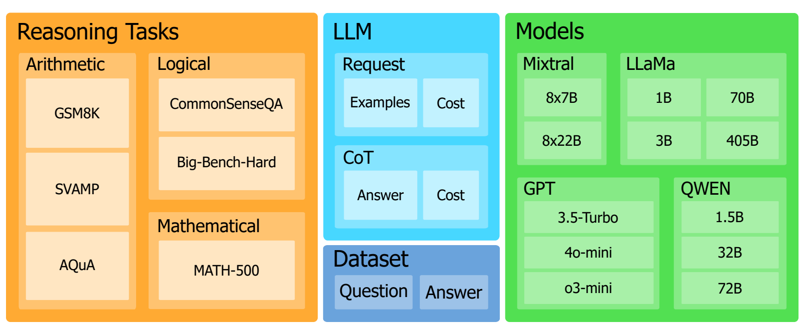

Figure 1: FEval-TTC comprises of three distinct groups of datasets, each consisting of question–answer pairs (see Section 2.1). Each dataset is queried by multiple LLMs from different families with standardized query format. We provide 40 sampled Chains-of-Thoughts (CoTs) with extracted answers, number of tokens, and the corresponding dollar cost of inference per question (see Section 2.2).

1 Introduction

The emergence of System-2 thinking in Large Language Models (LLMs) (Ji et al., 2025) has introduced a new paradigm that leverages inference-time computation to enhance reasoning capabilities (Snell et al., 2025). This paradigm involves allocating additional computational resources during inference, such as extended token generation, to improve performance on complex reasoning tasks (Yang et al., 2025). The additional computation may originate from a single LLM (Snell et al., 2025) or from coordination among multiple LLMs (Qi et al., 2025). However, this increase in

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

generation leads to substantial time and financial costs, posing practical challenges that hinder rapid experimentation and broader adoption.

In case a researcher is using API, the monetary cost is primarily determined by commercial API usage fees, that are typically charged by API providers based on token usage. For self-hosted LLMs, monetary costs arise from the electricity consumed to operate GPUs. The time costs in both cases primarily arise from the latency between initiating a request and receiving the response, as the LLM generates tokens sequentially. Concurrently, the LLM landscape evolves rapidly, with frequent model updates, new releases, and revisions to API pricing. Such volatility can undermine the validity of prior research or create unfair advantages for newer methods if experimental setups do not carefully control for differences in model performance and cost. Reusing results from published work without accounting for these changes can further exacerbate these issues, leading to inaccurate comparisons.

This **F**air **Eval**uation protocol for **T**est-**T**ime **C**ompute (**FEval-TTC**) addresses these challenges by enabling researchers to substantially reduce both computational and time costs, while preserving fair and reproducible comparisons with prior work. FEval-TTC includes a comprehensive set of pre-recorded model queries and responses, along with extracted answers and associated metadata. For instance, applying self-consistency with 20 samples on the GSM8K dataset (Cobbe et al., 2021a) using Mixtral 8×22B can take up to seven hours due to inference latency. In contrast, FEval-TTC allows this evaluation to be completed in seconds by eliminating the need for live LLM calls. We provide standardized requests and responses for sixteen datasets covering both commonsense and mathematical reasoning tasks. Additionally, we introduce a unified cost model to ensure consistent and fair estimation of both query and response costs across different methods and models.

The uniqueness of FEval-TTC lies in the following features:

- It supports several groups of reasoning tasks and multiple LLM model families.
- It is trivially extensible to incorporate additional models, datasets, and prompting techniques.
- It ensures a fair comparison of test-time algorithms by using a standardized set of LLM responses and a unified monetary/token cost model.
- FEval-TTC significantly reduces the evaluation time and cost of common test-time inference methods by leveraging pre-recorded LLM responses instead of issuing live queries.

2 FEval-TTC package overview

```
from feval_ttc import load, DatasetType, LLMType

dataset, [llm1,llm2] = load(DatasetType.SVAMP, \
                   [LLMType.LLaMA3B32, LLMType.Qwen72B25])

for question_id, dataentry in dataset:
    print("Question: ", dataentry.question)
    print("True answer: ", dataentry.answer)
    llm1_response = llm1(question_id, N=20)
    print("1st CoT answer: ",  llm1_response.cots[0].answer)
    print("Token cost: ", llm1_response.cots[0].tokens)
    print("USD Cost: ", llm1_response.cots[0].dollar_cost)
```

Listing 1: Example of an interaction with the *FEval-TTC* package

This section provides the architectural overview of the FEval-TTC. FEval-TTC is composed of two main parts: *Dataset* module and *LLM* module. The Dataset module holds the list of questions and answers and an interface to iterate over them (see Section 2.1). The LLM module stores multiple Chain-of-Thoughts (CoTs) responses along with their extracted answers. The design of both modules employs key-value dictionaries to facilitate seamless access to cached data. The main package features an interface to load a Dataset module instance and a set of corresponding LLM module instances. A usage example for research purposes is provided in Listing 1.

2.1 Dataset module

Dataset module instance contains a list of *Dataentries* (see Listing 2). Each Dataentry includes a question and its ground-truth answer, collected from the corresponding datasets. We did not change questions and answers, but the answer format was standardized across the package. For each dataset, we provide a system prompt that was used to obtain LLM responses.

FEval-TTC features datasets from three different reasoning categories: commonsense reasoning, arithmetic reasoning, and mathematical reasoning. The **commonsense reasoning** group includes tasks designed to assess inference capabilities using commonsense knowledge, such as *CommonSenseQA* (Talmor et al., 2019), and 11 *BIG-Bench-Hard* (Suzgun et al., 2023) tasks The **arithmetic reasoning** group contains datasets like *GSM8K* (Cobbe et al., 2021a), *SVAMP* (Patel et al., 2021), and *AQuA* (Ling et al., 2017), which require basic calculation skills. The **mathematical reasoning** category targets advanced problem-solving ability, represented by the *MATH-500* (Hendrycks et al., 2021), which consists of competition-style mathematical questions requiring rigorous algebraic and geometric manipulation.

```
class DatasetEntry(BaseModel):
    answer: str
    question: str

class Dataset(BaseModel):
    data: List[DatasetEntry]
    datatype: DatasetType
    system_prompt: str
```

Listing 2: Dataset module in FEval-TTC

2.2 LLM module

LLM instance represents a real-world API, such as OpenAI[1]. For each question, the instance returns an *LLMResponse* object. It provides access to a few-shot *LLMRequest* prompt, which includes the official few-shot examples for a corresponding dataset, and to a set of *CoT*. Each CoT object consists of the raw API response and extracted answer. Note that not all CoTs contain answers that could be extracted, therefore the answer field is set to *None*, when such failure occurs. The answers extracted from CoTs are standardized across datasets. The evaluation protocol is designed to be non-restrictive, allowing seamless integration with a researcher's existing methodology. In practice, evaluating a test-time compute algorithm using FEval-TTC simply involves replacing live LLM API calls with provided responses.

We feature five common LLM families. Each model is queried 40 times using a few-shot CoT prompt with standard few-shot examples. The reasoning model o3-mini is queried 3 times using zero-shot instructions to save cost. Specifically, FEval-TTC includes CoTs from the following LLMs.

LLaMA: Llama 3.2-1B-Instruct, Llama 3.2-3B-Instruct, Llama 3.3-70B-Instruct, and Llama-3.1-405B-Instruct.
QWEN: Qwen2.5-1.5B-Instruct, Qwen2.5-32B-Instruct, and Qwen2.5-72B-Instruct.
Deepseek: Deepseek-V3.
Mistral: Mixtral-8x7B, and Mixtral-8x22B.
GPT: GPT 3.5 Turbo, GPT-4o-mini, and o3-mini (reasoning).

```
class CoTMetadata(BaseModel):
    dollar_cost: float
    tokens: int

class CoT(BaseModel):
    raw_text: str
    answer: Optional[str]
    metadata: CoTMetadata

class LLMRequest(BaseModel):
    raw_text: str
    dollar_cost: float
    tokens: int

class LLMResponse(BaseModel):
    cots: List[CoT]
    request: LLMRequest
    answers: List[str]

class LLMConfig(BaseModel):
    name: LLMType
    temperature: float
    max_tokens: int

class LLM(BaseModel):
    config: LLMConfig
    responses: List[LLMResponse]
```

Listing 3: LLM modules in FEval-TTC

[1] https://platform.openai.com/docs/overview

2.3 Dollar cost modelling

FEval-TTC uses a unified monetary cost model to compute the dollar cost of a LLM response:

$$DollarCost(\text{INP}, \text{OUT}) = 10^{-6} \left(C_i \; Token(\text{INP}) + C_o \; Token(\text{OUT}) \right) , \tag{1}$$

where C_i is the input processing cost of the model in USD per million tokens, C_o is model's output cost in USD for generation of a million tokens, $Token(\text{INP})$ is the number of tokens in input prompt (from *LLMRequest*) and $Token(\text{OUT})$ is the number of tokens generated by the LLM (from *CoT*). In our cost model, we assume that an LLM can be prompted once to sample multiple outputs, therefore, OUT may include multiple CoTs for a single input INP.

We adopt this simplified cost model to enable fair comparisons of LLM responses, independent of external factors such as the query date or caching strategies used. We provide additional details in Appendix A.

3 Evaluation examples

In order to demonstrate the use of our protocol, we present some examples of common Test-Time Compute methods evaluated on FEval-TTC. Table 1 and Figure 2 show the results of Self-Consistency (Wang et al., 2023) and Best-of-N (Cobbe et al., 2021b) algorithms. FEval-TTC also supports the evaluation of many existing training-free, adaptive self-consistency methods (Aggarwal et al., 2023; Zhu et al., 2024; Wang et al., 2024) for reducing the sampling cost of Self-Consistency. Table 2 and Figure 3 demonstrate the evaluation of multi-LLM (cascade) methods such as Mixture of Thoughts (Yue et al., 2024) and ModelSwitch (Chen et al., 2025). Other cascade approaches, such as FrugalGPT (Aggarwal et al., 2024) and TREACLE (Zhang et al., 2024) can also be evaluated using FEval-TTC.

Table 1: Accuracies of Self-Consistency (SC) and Best-of-N (BoN) with 20 CoTs with AQuA dataset.

Method	Mixtral 8x22B	Qwen 32B
SC-20	0.787 ($1.76)	0.870 ($0.24)
BoN-20	0.606 ($1.76)	0.870 ($0.24)

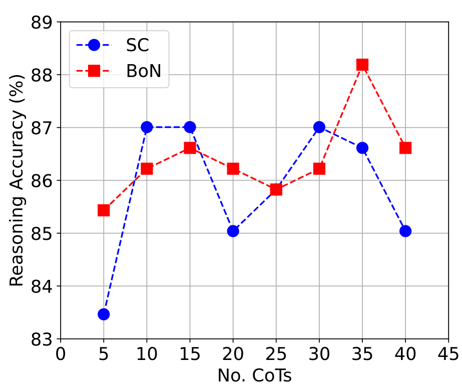

Figure 2: Evaluation of CoT+SC and Best-of-N algorithms on AQuA dataset using Qwen 32B for varying number of CoTs.

Table 2: Accuracies of Mixture of Thoughts (MoT) and ModelSwitch (MS) with LLaMA-70B and GPT-4o-mini.

Method	Ruin names	GSM8k
MoT	0.924 ($0.44)	0.960 ($2.21)
MS	0.916 ($0.13)	0.961 ($0.64)

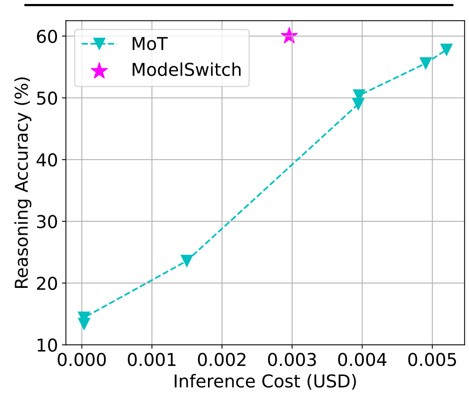

Figure 3: Evaluation of MoT and ModelSwitch algorithms on MATH-500 dataset using Llama models for varying computational costs.

4 Conclusion

We introduce FEval-TTC, an open-source framework for fast, fair, and low-cost evaluation of common test-time compute (TTC) methodologies. By replacing LLM API calls with FEval-TTC API calls, researchers can reduce evaluation time from hours to seconds at negligible cost. Our unified cost model enables fair comparisons across methods, independent of API pricing fluctuations. FEval-TTC facilitates the integration of new datasets and models through the application of standard prompting techniques.

References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proc. Conf. Empirical Methods in Natural Language Proces. (ACL)*, pp. 12375–12396, Singapore, Dec. 2023.

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam . Automix: Automatically mixing language models. In *Proc. Conf. Neural Info. Proces. Syst. (NeurIPS)*, pp. 131000–131034, Vancouver, Canada, Dec. 2024.

Jianhao Chen, Zishuo Xun, Bocheng Zhou, Han Qi, Hangfan Zhang, Qiaosheng Zhang, Yang Chen, Wei Hu, Yuzhong Qu, Wanli Ouyang, et al. Do we truly need so many samples? multi-llm repeated sampling efficiently scales test-time compute. *arXiv preprint arXiv:2504.00762*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021a.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021b.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Adv. Neural Inf. Process. Syst.*, 2021.

Yixin Ji, Juntao Li, Hai Ye, Kaixin Wu, Jia Xu, Linjian Mo, and Min Zhang. Test-time computing: from system-1 thinking to system-2 thinking. *arXiv preprint arXiv:2501.02497*, 2025.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proc. Conf. Empirical Methods in Natural Lang. Process.*, 2017.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proc. Conf. North Amer. Chapter Associ. Comput. Linguistics*, pp. 2080–2094, 2021.

Zhenting Qi, Mingyuan MA, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller LLMs stronger problem-solver. In *Proc. Int. Conf. Learn. Representations*, 2025.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *Proc. Int. Conf. Learn. Representations*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Proc. Assoc. Comput. Linguistics*, pp. 13003–13051, 2023.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, pp. 4149–4158, 2019.

Xinglin Wang, Shaoxiong Feng, Yiwei Li, Peiwen Yuan, Yueqi Zhang, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. Make every penny count: Difficulty-adaptive self-consistency for cost-efficient reasoning. *arXiv preprint arXiv:2408.13457*, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proc. Int. Conf. Learn. Representations*, 2023.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. Towards thinking-optimal scaling of test-time compute for LLM reasoning. *arXiv preprint arXiv:2502.18080*, 2025.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *Proc. Int. Conf. Learn. Representations*, 2024.

Xuechen Zhang, Zijian Huang, Ege Onur Taga, Carlee Joe-Wong, Samet Oymak, and Jiasi Chen. Efficient contextual LLM cascades through budget-constrained policy learning. In *Proc. Conf. Neural Info. Proces. Syst. (NeurIPS)*, pp. 91691–91722, Vancouver, Canada, Dec. 2024.

Jiace Zhu, Yingtao Shen, Jie Zhao, and An Zou. Path-Consistency: Prefix Enhancement for Efficient Inference in LLM. *arXiv e-prints*, art. arXiv:2409.01281, August 2024. doi: 10.48550/arXiv.2409.01281.

A Unified cost model details

Table 3: USD cost per million tokens for LLMs used in FEval-TTC. The costs are valid as of 02/06/2025.

LLM	**Input Cost C_i ($/M tokens)**	**Output Cost C_o ($/M tokens)**
LLaMA 3.2 1B-Instruct	0.005	0.01
LLaMA 3.2 3B-Instruct	0.01	0.02
LLaMA 3.3 70B-Instruct	0.13	0.40
LLaMA 3.1 405B-Instruct	1.00	3.00
Qwen 2.5 1B-Instruct	0.02	0.06
Qwen 2.5 32B-Instruct	0.06	0.20
Qwen 2.5 72B-Instruct	0.13	0.40
GPT 3.5-Turbo	0.50	1.50
GPT 4o-mini	0.15	0.60
OpenAI o3-mini	1.10	4.40
Mixtral-8x7B-Instruct	0.08	0.24
Mixtral-8x22B-Instruct	0.40	1.20
DeepSeek-V3	0.50	1.50

FEval-TTC includes CoTs from various LLaMA, QWEN, Deepseek, Mistral, and GPT models. We collected responses from the GPT model family using the OpenAI API[2]. OpenAI API prices are publicly available at `https://platform.openai.com/docs/pricing`. We used a commercial API service[3] to query other model families. The price information for LLaMA, QWEN, Deepseek, and Mistral model families can be found at `https://nebius.com/prices-ai-studio`. The detailed USD costs per million tokens for different LLMs are available in Tables 3. All costs were recorded as of June 2, 2025.

In out package we provide access to both token cost and dollar cost. Our unified model of dollar cost (1) is proportional to the number of tokens. Commercial API pricing is subject to change over time, typically at least once per year. By fixing the dollar cost model in our protocol, we ensure that cost comparisons remain consistent and are unaffected by such pricing changes. This design guarantees that comparisons between methods yield stable and fair conclusions, independent of future modifications to commercial API pricing policies.

[2]`https://openai.com/api`

[3]`https://docs.nebius.com/studio/inference/api`

B Licensing

The terms of use for OpenAI API [4] and Nebius API [5] services grant the users full ownership for the LLM inputs and outputs provided by the API. We distribute the collection of LLM inputs and CoT outputs under the Open Database License. We grant rights of distribution, utilization, modification, and extension of the collection under the condition of a copyright notice.

Our python package includes questions and ground truth answers for six datasets (including *causal judgement, date understanding, disambiguationQA, formal fallacies, geometric shapes, movie recommendation, penguins, ruin names, snarks, sports, and temporal sequences* tasks of Big-Bench-Hard). These datasets are provided for convenience of the users. We do not claim any ownership rights over the datasets included in the FEval-TTC package. These datasets are independent assets distributed under the following licenses:

- *CommonSenseQA* (Talmor et al., 2019) under an MIT license
- *Big-Bench-Hard* (Suzgun et al., 2023) under the MIT license
- *GSM8K* (Cobbe et al., 2021a) under the MIT license
- *SVAMP* (Patel et al., 2021) under the MIT license
- *AQuA* (Ling et al., 2017) under an Apache License, Version 2.0
- *MATH-500* (Hendrycks et al., 2021) under the MIT license

[4] `https://docs.studio.nebius.com/legal/terms-of-service#10-intellectual-property`
"You hold exclusive ownership of all rights, titles, and interests (including intellectual property rights) to Your Inputs"

[5] `https://openai.com/policies/row-terms-of-use/#content`
"As between you and OpenAI, and to the extent permitted by applicable law, you (a) retain your ownership rights in Input and (b) own the Output. We hereby assign to you all our right, title, and interest, if any, in and to Output."