meta-llama/Meta-Llama-3-8B-Instruct

Model Details

Meta developed and released the Meta Llama 3 family of large language models (LLMs), a collection of pretrained and instruction tuned generative text models in 8 and 70B sizes. The Llama 3 instruction tuned models are optimized for dialogue use cases and outperform many of the available open source chat models on common industry benchmarks. Further, in developing these models, we took great care to optimize helpfulness and safety.

Model developers Meta

Variations Llama 3 comes in two sizes — 8B and 70B parameters — in pre-trained and instruction tuned variants.

Input Models input text only.

Output Models generate text and code only.

Model Architecture Llama 3 is an auto-regressive language model that uses an optimized transformer architecture. The tuned versions use supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align with human preferences for helpfulness and safety.

	Training Data	Params	Context length	GQA	Token count	Knowledge cutoff
Llama 3	A new mix of publicly available online data.	8B	8k	Yes	15T+	March, 2023
		70B	8k	Yes		December, 2023

Llama 3 family of models. Token counts refer to pretraining data only. Both the 8 and 70B versions use Grouped-Query Attention (GQA) for improved inference scalability.

Model Release Date April 18, 2024.

Status This is a static model trained on an offline dataset. Future versions of the tuned models will be released as we improve model safety with community feedback.

License A custom commercial license is available at: https://llama.meta.com/llama3/license

Where to send questions or comments about the model Instructions on how to provide feedback or comments on the model can be found in the model README. For more technical information about generation parameters and recipes for how to use Llama 3 in applications, please go here.

Intended Use

Intended Use Cases Llama 3 is intended for commercial and research use in English. Instruction tuned models are intended for assistant-like chat, whereas pretrained models can be adapted for a variety of natural language generation tasks.

Out-of-scope Use in any manner that violates applicable laws or regulations (including trade compliance laws). Use in any other way that is prohibited by the Acceptable Use Policy and Llama 3 Community License. Use in languages other than English**.

**Note: Developers may fine-tune Llama 3 models for languages beyond English provided they comply with the Llama 3 Community License and the Acceptable Use Policy.

How to use

This repository contains two versions of Meta-Llama-3-8B-Instruct, for use with transformers and with the original `llama3` codebase.

Use with transformers

You can run conversational inference using the Transformers pipeline abstraction, or by leveraging the Auto classes with the `generate()` function. Let's see examples of both.

Transformers pipeline

```
import transformers
import torch

model_id = "meta-llama/Meta-Llama-3-8B-Instruct"

pipeline = transformers.pipeline(
    "text-generation",
    model=model_id,
    model_kwargs={"torch_dtype": torch.bfloat16},
    device_map="auto",
)

messages = [
    {"role": "system", "content": "You are a pirate chatbot who always responds in pirate speak!"},
    {"role": "user", "content": "Who are you?"},
]

terminators = [
    pipeline.tokenizer.eos_token_id,
    pipeline.tokenizer.convert_tokens_to_ids("<|eot_id|>")
]

outputs = pipeline(
    messages,
    max_new_tokens=256,
    eos_token_id=terminators,
    do_sample=True,
    temperature=0.6,
    top_p=0.9,
)
print(outputs[0]["generated_text"][-1])
```

Transformers AutoModelForCausalLM

```
from transformers import AutoTokenizer, AutoModelForCausalLM
import torch

model_id = "meta-llama/Meta-Llama-3-8B-Instruct"

tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(
    model_id,
    torch_dtype=torch.bfloat16,
    device_map="auto",
)

messages = [
    {"role": "system", "content": "You are a pirate chatbot who always responds in pirate speak!"},
    {"role": "user", "content": "Who are you?"},
]

input_ids = tokenizer.apply_chat_template(
    messages,
    add_generation_prompt=True,
    return_tensors="pt"
).to(model.device)

terminators = [
    tokenizer.eos_token_id,
    tokenizer.convert_tokens_to_ids("<|eot_id|>")
]

outputs = model.generate(
    input_ids,
    max_new_tokens=256,
    eos_token_id=terminators,
    do_sample=True,
    temperature=0.6,
    top_p=0.9,
)
response = outputs[0][input_ids.shape[-1]:]
print(tokenizer.decode(response, skip_special_tokens=True))
```

Use with `llama3`

Please, follow the instructions in the repository

To download Original checkpoints, see the example command below leveraging `huggingface-cli`:

```
huggingface-cli download meta-llama/Meta-Llama-3-8B-Instruct --include "original/*" --local-dir Meta-Llama-3-8B-Instruct
```

For Hugging Face support, we recommend using transformers or TGI, but a similar command works.

Hardware and Software

Training Factors We used custom training libraries, Meta's Research SuperCluster, and production clusters for pretraining. Fine-tuning, annotation, and evaluation were also performed on third-party cloud compute.

Carbon Footprint Pretraining utilized a cumulative 7.7M GPU hours of computation on hardware of type H100-80GB (TDP of 700W). Estimated total emissions were 2290 tCO2eq, 100% of which were offset by Meta's sustainability program.

	Time (GPU hours)	Power Consumption (W)	Carbon Emitted(tCO2eq)
Llama 3 8B	1.3M	700	390
Llama 3 70B	6.4M	700	1900
Total	7.7M		2290

CO2 emissions during pre-training. Time: total GPU time required for training each model. Power Consumption: peak power capacity per GPU device for the GPUs used adjusted for power usage efficiency. 100% of the emissions are directly offset by Meta's sustainability program, and because we are openly releasing these models, the pretraining costs do not need to be incurred by others.

Training Data

Overview Llama 3 was pretrained on over 15 trillion tokens of data from publicly available sources. The fine-tuning data includes publicly available instruction datasets, as well as over 10M human-annotated examples. Neither the pretraining nor the fine-tuning datasets include Meta user data.

Data Freshness The pretraining data has a cutoff of March 2023 for the 8B and December 2023 for the 70B models respectively.

Benchmarks

In this section, we report the results for Llama 3 models on standard automatic benchmarks. For all the evaluations, we use our internal evaluations library. For details on the methodology see here.

Base pretrained models

Category	Benchmark	Llama 3 8B	Llama2 7B	Llama2 13B	Llama 3 70B	Llama2 70B
General	MMLU (5-shot)	66.6	45.7	53.8	79.5	69.7
	AGIEval English (3-5 shot)	45.9	28.8	38.7	63.0	54.8
	CommonSenseQA (7-shot)	72.6	57.6	67.6	83.8	78.7
	Winogrande (5-shot)	76.1	73.3	75.4	83.1	81.8
	BIG-Bench Hard (3-shot, CoT)	61.1	38.1	47.0	81.3	65.7
	ARC-Challenge (25-shot)	78.6	53.7	67.6	93.0	85.3
Knowledge reasoning	TriviaQA-Wiki (5-shot)	78.5	72.1	79.6	89.7	87.5
Reading comprehension	SQuAD (1-shot)	76.4	72.2	72.1	85.6	82.6
	QuAC (1-shot, F1)	44.4	39.6	44.9	51.1	49.4
	BoolQ (0-shot)	75.7	65.5	66.9	79.0	73.1
	DROP (3-shot, F1)	58.4	37.9	49.8	79.7	70.2

Instruction tuned models

Benchmark	Llama 3 8B	Llama 2 7B	Llama 2 13B	Llama 3 70B	Llama 2 70B
MMLU (5-shot)	68.4	34.1	47.8	82.0	52.9
GPQA (0-shot)	34.2	21.7	22.3	39.5	21.0
HumanEval (0-shot)	62.2	7.9	14.0	81.7	25.6
GSM-8K (8-shot, CoT)	79.6	25.7	77.4	93.0	57.5
MATH (4-shot, CoT)	30.0	3.8	6.7	50.4	11.6

Responsibility & Safety

We believe that an open approach to AI leads to better, safer products, faster innovation, and a bigger overall market. We are committed to Responsible AI development and took a series of steps to limit misuse and harm and support the open source community.

Foundation models are widely capable technologies that are built to be used for a diverse range of applications. They are not designed to meet every developer preference on safety levels for all use cases, out-of-the-box, as those by their nature will differ across different applications.

Rather, responsible LLM-application deployment is achieved by implementing a series of safety best practices throughout the development of such applications, from the model pre-training, fine-tuning and the deployment of systems composed of safeguards to tailor the safety needs specifically to the use case and audience.

As part of the Llama 3 release, we updated our Responsible Use Guide to outline the steps and best practices for developers to implement model and system level safety for their application. We also provide a set of resources including Meta Llama Guard 2 and Code Shield safeguards. These tools have proven to drastically reduce residual risks of LLM Systems, while maintaining a high level of helpfulness. We encourage developers to tune and deploy these safeguards according to their needs and we provide a reference implementation to get you started.

Llama 3-Instruct

As outlined in the Responsible Use Guide, some trade-off between model helpfulness and model alignment is likely unavoidable. Developers should exercise discretion about how to weigh the benefits of alignment and helpfulness for their specific use case and audience. Developers should be mindful of residual risks when using Llama models and leverage additional safety tools as needed to reach the right safety bar for their use case.

Safety

For our instruction tuned model, we conducted extensive red teaming exercises, performed adversarial evaluations and implemented safety mitigations techniques to lower residual risks. As with any Large Language Model, residual risks will likely remain and we recommend that developers assess these risks in the context of their use case. In parallel, we are working with the community to make AI safety benchmark standards transparent, rigorous and interpretable.

Refusals

In addition to residual risks, we put a great emphasis on model refusals to benign prompts. Over-refusing not only can impact the user experience but could even be harmful in certain contexts as well. We've heard the feedback from the developer community and improved our fine tuning to ensure that Llama 3 is significantly less likely to falsely refuse to answer prompts than Llama 2.

We built internal benchmarks and developed mitigations to limit false refusals making Llama 3 our most helpful model to date.

Responsible release

In addition to responsible use considerations outlined above, we followed a rigorous process that requires us to take extra measures against misuse and critical risks before we make our release decision.

Misuse

If you access or use Llama 3, you agree to the Acceptable Use Policy. The most recent copy of this policy can be found at https://llama.meta.com/llama3/use-policy/.

Critical risks

CBRNE (Chemical, Biological, Radiological, Nuclear, and high yield Explosives)

We have conducted a two fold assessment of the safety of the model in this area:

- Iterative testing during model training to assess the safety of responses related to CBRNE threats and other adversarial risks.
- Involving external CBRNE experts to conduct an uplift test assessing the ability of the model to accurately provide expert knowledge and reduce barriers to potential CBRNE misuse, by reference to what can be achieved using web search (without the model).

Cyber Security

We have evaluated Llama 3 with CyberSecEval, Meta's cybersecurity safety eval suite, measuring Llama 3's propensity to suggest insecure code when used as a coding assistant, and Llama 3's propensity to comply with requests to help carry out cyber attacks, where attacks are defined by the industry standard MITRE ATT&CK cyber attack ontology. On our insecure coding and cyber attacker helpfulness tests, Llama 3 behaved in the same range or safer than models of equivalent coding capability.

Child Safety

Child Safety risk assessments were conducted using a team of experts, to assess the model's capability to produce outputs that could result in Child Safety risks and inform on any necessary and appropriate risk mitigations via fine tuning. We leveraged those expert red teaming sessions to expand the coverage of our evaluation benchmarks through Llama 3 model development. For Llama 3, we conducted new in-depth sessions using objective based methodologies to assess the model risks along multiple attack vectors. We also partnered with content specialists to perform red teaming exercises assessing potentially violating content while taking account of market specific nuances or experiences.

Community

Generative AI safety requires expertise and tooling, and we believe in the strength of the open community to accelerate its progress. We are active members of open consortiums, including the AI Alliance, Partnership in AI and MLCommons, actively contributing to safety standardization and transparency. We encourage the community to adopt taxonomies like the MLCommons Proof of Concept evaluation to facilitate collaboration and transparency on safety and content evaluations. Our Purple Llama tools are open sourced for the community to use and widely distributed across ecosystem partners including cloud service providers. We encourage community contributions to our Github repository.

Finally, we put in place a set of resources including an output reporting mechanism and bug bounty program to continuously improve the Llama technology with the help of the community.

Ethical Considerations and Limitations

The core values of Llama 3 are openness, inclusivity and helpfulness. It is meant to serve everyone, and to work for a wide range of use cases. It is thus designed to be accessible to people across many different backgrounds, experiences and perspectives. Llama 3 addresses users and their needs as they are, without insertion unnecessary judgment or normativity, while reflecting the understanding that even content that may appear problematic in some cases can serve valuable purposes in others. It respects the dignity and autonomy of all users, especially in terms of the values of free thought and expression that power innovation and progress.

But Llama 3 is a new technology, and like any new technology, there are risks associated with its use. Testing conducted to date has been in English, and has not covered, nor could it cover, all scenarios. For these reasons, as with all LLMs, Llama 3's potential outputs cannot be predicted in advance, and the model may in some instances produce inaccurate, biased or other objectionable responses to user prompts. Therefore, before deploying any applications of Llama 3 models, developers should perform safety testing and tuning tailored to their specific applications of the model. As outlined in the Responsible Use Guide, we recommend incorporating Purple Llama solutions into your workflows and specifically Llama Guard which provides a base model to filter input and output prompts to layer system-level safety on top of model-level safety.

Please see the Responsible Use Guide available at http://llama.meta.com/responsible-use-guide

Citation instructions

@article{llama3modelcard,

title={Llama 3 Model Card},

author={AI@Meta},

year={2024},

url = {https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md}

}

Contributors

Aaditya Singh; Aaron Grattafiori; Abhimanyu Dubey; Abhinav Jauhri; Abhinav Pandey; Abhishek Kadian; Adam Kelsey; Adi Gangidi; Ahmad Al-Dahle; Ahuva Goldstand; Aiesha Letman; Ajay Menon; Akhil Mathur; Alan Schelten; Alex Vaughan; Amy Yang; Andrei Lupu; Andres Alvarado; Andrew Gallagher; Andrew Gu; Andrew Ho; Andrew Poulton; Andrew Ryan; Angela Fan; Ankit Ramchandani; Anthony Hartshorn; Archi Mitra; Archie Sravankumar; Artem Korenev; Arun Rao; Ashley Gabriel; Ashwin Bharambe; Assaf Eisenman; Aston Zhang; Aurelien Rodriguez; Austen Gregerson; Ava Spataru; Baptiste Roziere; Ben Maurer; Benjamin Leonhardi; Bernie Huang; Bhargavi Paranjape; Bing Liu; Binh Tang; Bobbie Chern; Brani Stojkovic; Brian Fuller; Catalina Mejia Arenas; Chao Zhou; Charlotte Caucheteux; Chaya Nayak; Ching-Hsiang Chu; Chloe Bi; Chris Cai; Chris Cox; Chris Marra; Chris McConnell; Christian Keller; Christoph Feichtenhofer; Christophe Touret; Chunyang Wu; Corinne Wong; Cristian Canton Ferrer; Damien Allonsius; Daniel Kreymer; Daniel Haziza; Daniel Li; Danielle Pintz; Danny Livshits; Danny Wyatt; David Adkins; David Esiobu; David Xu; Davide Testuggine; Delia David; Devi Parikh; Dhruv Choudhary; Dhruv Mahajan; Diana Liskovich; Diego Garcia-Olano; Diego Perino; Dieuwke Hupkes; Dingkang Wang; Dustin Holland; Egor Lakomkin; Elina Lobanova; Xiaoqing Ellen Tan; Emily Dinan; Eric Smith; Erik Brinkman; Esteban Arcaute; Filip Radenovic; Firat Ozgenel; Francesco Caggioni; Frank Seide; Frank Zhang; Gabriel Synnaeve; Gabriella Schwarz; Gabrielle Lee; Gada Badeer; Georgia Anderson; Graeme Nail; Gregoire Mialon; Guan Pang; Guillem Cucurell; Hailey Nguyen; Hannah Korevaar; Hannah Wang; Haroun Habeeb; Harrison Rudolph; Henry Aspegren; Hu Xu; Hugo Touvron; Iga Kozlowska; Igor Molybog; Igor Tufanov; Iliyan Zarov; Imanol Arrieta Ibarra; Irina-Elena Veliche; Isabel Kloumann; Ishan Misra; Ivan Evtimov; Jacob Xu; Jade Copet; Jake Weissman; Jan Geffert; Jana Vranes; Japhet Asher; Jason Park; Jay Mahadeokar; Jean-Baptiste Gaya; Jeet Shah; Jelmer van der Linde; Jennifer Chan; Jenny Hong; Jenya Lee; Jeremy Fu; Jeremy Teboul; Jianfeng Chi; Jianyu Huang; Jie Wang; Jiecao Yu; Joanna Bitton; Joe Spisak; Joelle Pineau; Jon Carvill; Jongsoo Park; Joseph Rocca; Joshua Johnstun; Junteng Jia; Kalyan Vasuden Alwala; Kam Hou U; Kate Plawiak; Kartikeya Upasani; Kaushik Veeraraghavan; Ke Li; Kenneth Heafield; Kevin Stone; Khalid El-Arini; Krithika Iyer; Kshitiz Malik; Kuenley Chiu; Kunal Bhalla; Kyle Huang; Lakshya Garg; Lauren Rantala-Yeary; Laurens van der Maaten; Lawrence Chen; Leandro Silva; Lee Bell; Lei Zhang; Liang Tan; Louis Martin; Lovish Madaan; Luca Wehrstedt; Lukas Blecher; Luke de Oliveira; Madeline Muzzi; Madian Khabsa; Manav Avlani; Mannat Singh; Manohar Paluri; Mark Zuckerberg; Marcin Kardas; Martynas Mankus; Mathew Oldham; Mathieu Rita; Matthew Lennie; Maya Pavlova; Meghan Keneally; Melanie Kambadur; Mihir Patel; Mikayel Samvelyan; Mike Clark; Mike Lewis; Min Si; Mitesh Kumar Singh; Mo Metanat; Mona Hassan; Naman Goyal; Narjes Torabi; Nicolas Usunier; Nikolay Bashlykov; Nikolay Bogoychev; Niladri Chatterji; Ning Dong; Oliver Aobo Yang; Olivier Duchenne; Onur Celebi; Parth Parekh; Patrick Alrassy; Paul Saab; Pavan Balaji; Pedro Rittner; Pengchuan Zhang; Pengwei Li; Petar Vasic; Peter Weng; Polina Zvyagina; Prajjwal Bhargava; Pratik Dubal; Praveen Krishnan; Punit Singh Koura; Qing He; Rachel Rodriguez; Ragavan Srinivasan; Rahul Mitra; Ramon Calderer; Raymond Li; Robert Stojnic; Roberta Raileanu; Robin Battey; Rocky Wang; Rohit Girdhar; Rohit Patel; Romain Sauvestre; Ronnie Polidoro; Roshan Sumbaly; Ross Taylor; Ruan Silva; Rui Hou; Rui Wang; Russ Howes; Ruty Rinott; Saghar Hosseini; Sai Jayesh Bondu; Samyak Datta; Sanjay Singh; Sara Chugh; Sargun Dhillon; Satadru Pan; Sean Bell; Sergey Edunov; Shaoliang Nie; Sharan Narang; Sharath Raparthy; Shaun Lindsay; Sheng Feng; Sheng Shen; Shenghao Lin; Shiva Shankar; Shruti Bhosale; Shun Zhang; Simon Vandenhende; Sinong Wang; Seohyun Sonia Kim; Soumya Batra; Sten Sootla; Steve Kehoe; Suchin Gururangan; Sumit Gupta; Sunny Virk; Sydney Borodinsky; Tamar Glaser; Tamar Herman; Tamara Best; Tara Fowler; Thomas Georgiou; Thomas Scialom; Tianhe Li; Todor Mihaylov; Tong Xiao; Ujjwal Karn; Vedanuj Goswami; Vibhor Gupta; Vignesh Ramanathan; Viktor Kerkez; Vinay Satish Kumar; Vincent Gonguet; Vish Vogeti; Vlad Poenaru; Vlad Tiberiu Mihailescu; Vladan Petrovic; Vladimir Ivanov; Wei Li; Weiwei Chu; Wenhan Xiong; Wenyin Fu; Wes Bouaziz; Whitney Meers; Will Constable; Xavier Martinet; Xiaojian Wu; Xinbo Gao; Xinfeng Xie; Xuchao Jia; Yaelle Goldschlag; Yann LeCun; Yashesh Gaur; Yasmine Babaei; Ye Qi; Yenda Li; Yi Wen; Yiwen Song; Youngjin Nam; Yuchen Hao; Yuchen Zhang; Yun Wang; Yuning Mao; Yuzi He; Zacharie Delpierre Coudert; Zachary DeVito; Zahra Hankir; Zhaoduo Wen; Zheng Yan; Zhengxing Chen; Zhenyu Yang; Zoe Papakipos