ReplitLM: using Open-source from Training to Production for a Code Completion LLM

Michele Catasta

https://twitter.com/pirroh

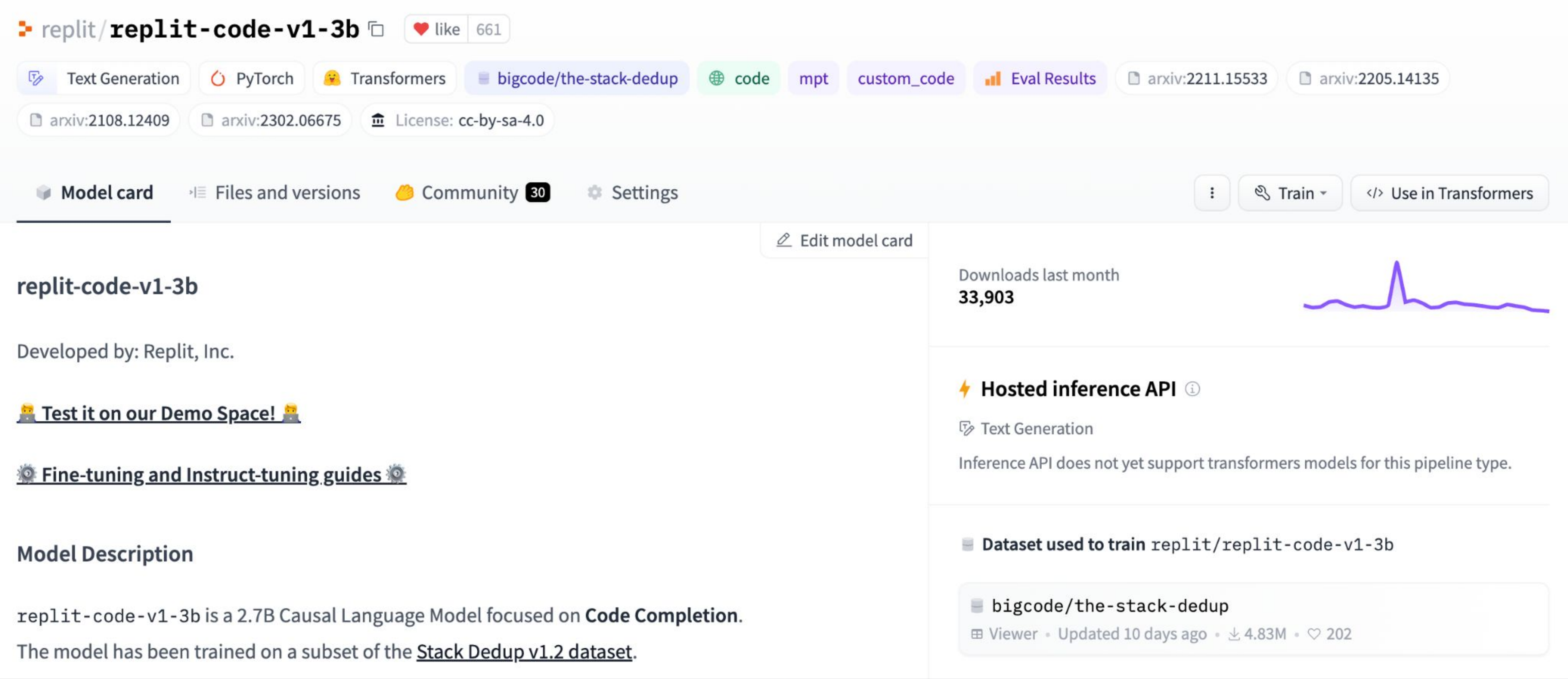
https://pirroh.fyi

Code Completion on Replit

style.css

```css
/* container with centered text and sans-serif font */
.

/* Style H1 with font size of 24 */

/* button add padding on top and box shadow */

/* .quotes add margin and padding */

/* .quote font size of 18 */

/* .author font size of 12 and bold text */


```

In early May 2023 we released replit-code-v1-3b, our bespoke Code Completion LLM serving a large number of Replit users



replit/replit-code-v1-3b
like 661

Text Generation | PyTorch | Transformers | bigcode/the-stack-dedup | code | mpt | custom_code | Eval Results | arxiv:2211.15533 | arxiv:2205.14135 | arxiv:2108.12409 | arxiv:2302.06675 | License: cc-by-sa-4.0

Model card | Files and versions | Community 30 | Settings | Train | Use in Transformers

Edit model card

replit-code-v1-3b

Developed by: Replit, Inc.

Test it on our Demo Space!

Fine-tuning and Instruct-tuning guides

Model Description

`replit-code-v1-3b` is a 2.7B Causal Language Model focused on **Code Completion**. The model has been trained on a subset of the Stack Dedup v1.2 dataset.

Downloads last month
33,903

Hosted inference API

Text Generation

Inference API does not yet support transformers models for this pipeline type.

Dataset used to train `replit/replit-code-v1-3b`

`bigcode/the-stack-dedup`
Viewer • Updated 10 days ago • 4.83M • 202

replit-code-v1-3b / **Data**

First Llama-style LLM for code

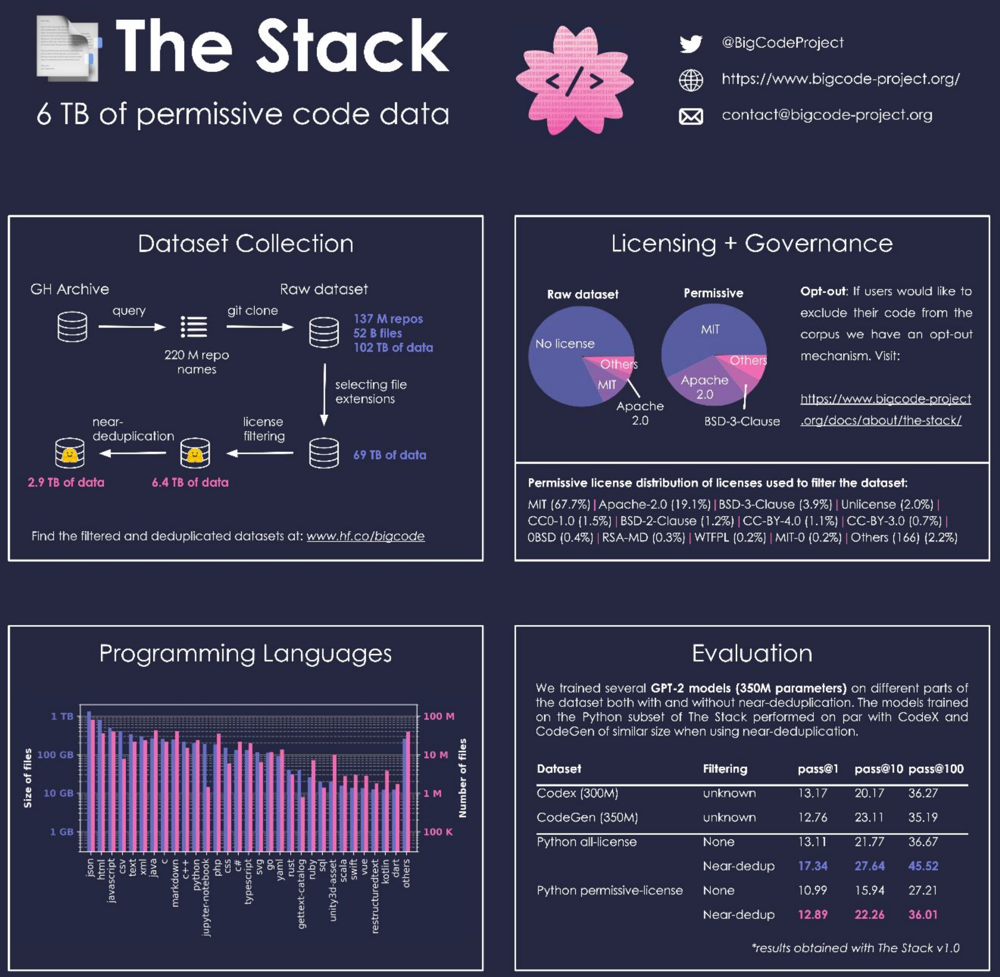
~195 tokens per parameter

Trained on 525B tokens of code

175B tokens over 3 epochs

20 languages

Markdown, Java, JavaScript, Python, TypeScript, PHP, SQL, JSX, reStructuredText, Rust, C, CSS, Go, C++, HTML, Vue, Ruby, Jupyter Notebook, R, Shell



- Pretraining data mixture based on The Stack v1.2 (released in March 2023)
- Selected the top 20 languages used on Replit
- Large number of code quality heuristics to filter the dataset (e.g., Codex paper, stripping long content from HTML/CSS files, etc.)
- Data processing on Spark, vocabulary training with Google SentencePiece

Scaling Data-Constrained Language Models

Niklas Muennighoff[1] Alexander M. Rush[1] Boaz Barak[2] Teven Le Scao[1]
Aleksandra Piktus[1] Nouamane Tazi[1] Sampo Pyysalo[3] Thomas Wolf[1] Colin Raffel[1]
[1] Hugging Face [2] Harvard University [3] University of Turku
n.muennighoff@gmail.com

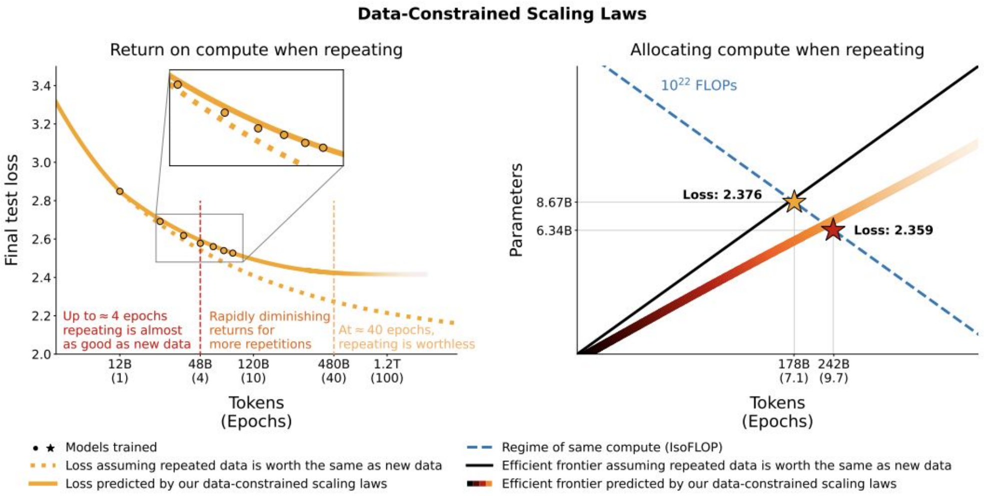


Figure 1: ***Return* and *Allocation* when repeating data.** *(Left):* Loss of LLMs (4.2B parameters) scaled on repeated data decays predictably (§6). *(Right):* To maximize performance when repeating, our data-constrained scaling laws and empirical data suggest training smaller models for more epochs in contrast to what assuming Chinchilla scaling laws [42] hold for repeated data would predict (§5).

- Published coincidentally just a few weeks after we released our LLM
- Highly recommended paper, confirming our ablation studies on repeated data
- This intuition allowed us to train to completion using only permissively-licensed code, hence we could release our model under CC BY-SA-4.0

replit-code-v1-3b / **Model Training**

2.7B parameters

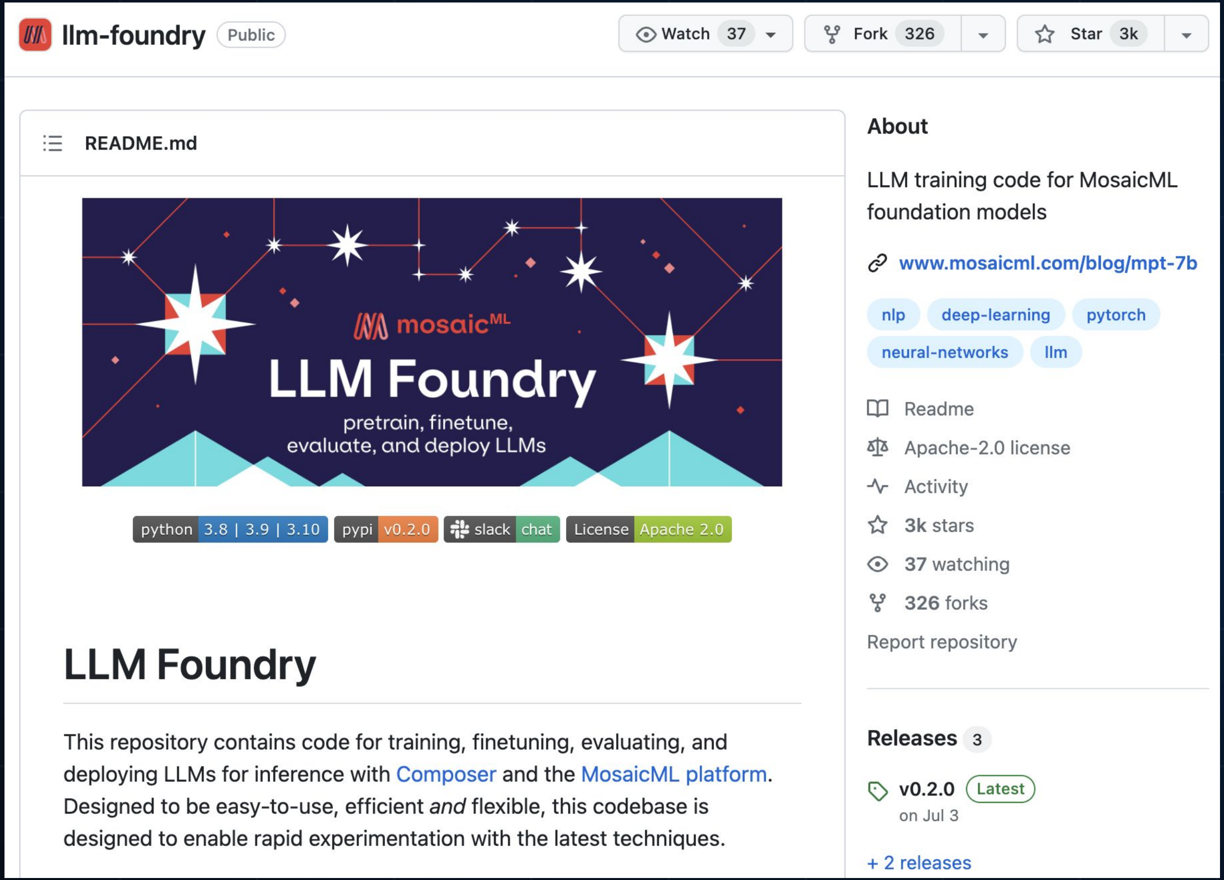
Custom 32k vocabulary focused on code

256 A100-40GB GPUs

For ~3 days on the MosaicML platform

LLM best practices

Flash Attention, AliBi positional embeddings, LionW optimizer, etc.



- All training runs based on an early release of [LLM Foundry] by MosaicML
- Same library used to train larger open-source models like MPT-7B and MPT-30B

replit-code-v1-3b / Evaluation

	Score pass@1
Python (OpenAI HumanEval)	22.56%
Python (MultiPL-E)	20.49%
Java (MultiPL-E)	20.25%
JavaScript (MultiPL-E)	19.25%
C++ (MultiPL-E)	18.63%
Rust (MultiPL-E)	16.02%
PHP (MultiPL-E)	13.04%

- To navigate the latest Code LLM releases, BigCode (🤗) created Multilingual Code Models Evaluation

- Based on MultiPL-E, an extension of the original OpenAI HumanEval benchmark to 18 languages

- replit-code-v1-3b was trained only on **10 languages out of the 18** supported by MultiPL-E

T	Models	Average score
	CodeLlama-34b-Instruct	35.09
	CodeLlama-34b	33.89
	CodeLlama-34b-Python	33.87
	WizardCoder-15B-V1.0	32.07
	CodeLlama-13b-Instruct	31.29
	CodeLlama-13b-Python	28.67
	CodeLlama-13b	28.35
	CodeLlama-7b-Instruct	26.45
	CodeLlama-7b	24.36
	OctoCoder-15B	24.01
	CodeLlama-7b-Python	23.5
	StarCoder-15B	22.74
	StarCoderBase-15B	22.4
	CodeGeex2-6B	21.23
	OctoGeeX-7B	20.79
	StarCoderBase-7B	20.17
	CodeGen25-7B-multi	20.04
	StarCoderBase-3B	15.29
	CodeGen25-7B-mono	12.1
	Replit-2.7B	11.62
	CodeGen-16B-Multi	9.89
	StarCoderBase-1.1B	9.81
	StableCode-3B	8.1
	DeciCoder-1B	5.86
	SantaCoder-1.1B	4.92

replit-repltuned-v1-3b / Data & Training

Further pretraining on 111B tokens of code

37B tokens over 3 epochs

Code authored by our users in public Repls

A lot of Python and Javascript

Same languages, same data filtering heuristics

The problem

Yao Fu @Francis_YAO_

Nowadays everybody finetune / continue train LLaMA. A practical problem is learning rate re-warm: the pretraining learning rate schedule stops at 3e-5, naively increasing the continue train lr to 3e-4 typically causes double descent. Is there a good way to mitigate this issue? 🤔

11:09 AM · Aug 15, 2023 · **46K** Views

Our experience

Yam Peleg @Yampeleg · Aug 15

I just schedule (& warmup) the gradient clipping along the lr and it works fine

Also: suboptimal training is usually not that suboptimal.. yolo just go for it, worse case the initial steps won't be the best and you end up with only 97% of the performance you could have..

The solution?

- Continual Pre-Training of Large Language Models: How to (re)warm your model?
- A pragmatic hack explained by Shital Shah in this thread, inspired by the LR schedule from "Scaling Vision Transformers"

replit-repltuned-v1-3b / **Evaluation**

	Score pass@1	Base model
Python (OpenAI HumanEval)	30.48%	22.56%
Python (MultiPL-E)	29.81%	20.49%
Java (MultiPL-E)	19.62%	20.25%
JavaScript (MultiPL-E)	27.95%	19.25%
C++ (MultiPL-E)	26.08%	18.63%
Rust (MultiPL-E)	15.38%	16.02%
PHP (MultiPL-E)	23.60%	13.04%

replit-*-v1-3b / **Inference**

~ 200 tokens / s on a single A100-40G
(no batching)

We made explicit architectural choices to support:
- https://github.com/NVIDIA/FasterTransformer
- https://github.com/triton-inference-server
for optimized inference on NVIDIA GPUs

Reliable inference evaluation across model architectures is still really **HARD**

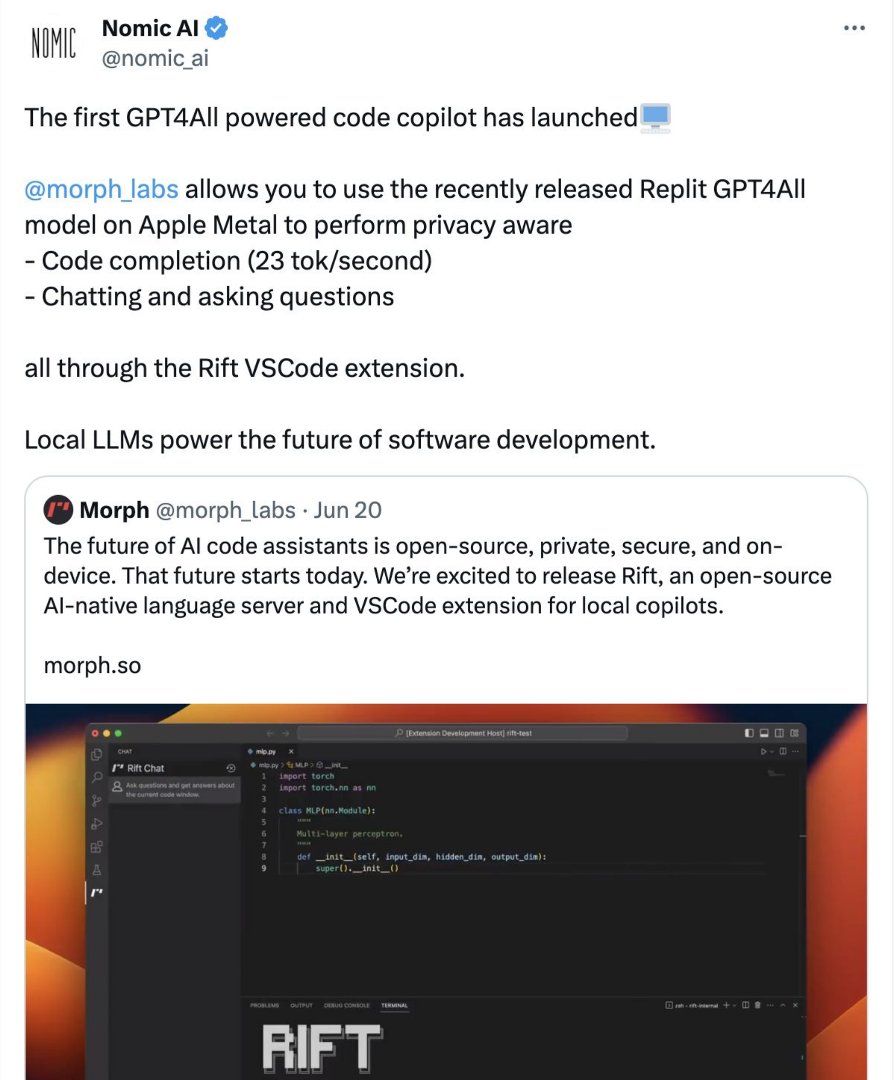
Models	Throughput (tokens/s)
CodeLlama-34b	15.1
CodeLlama-34b-Python	15.1
CodeLlama-13b	25.3
CodeLlama-13b-Python	25.3
CodeLlama-7b	33.1
StarCoder-15B	43.9
CodeLlama-7b-Python	33.1
StarCoderBase-15B	43.8
CodeGeex2-6B	32.7
StarCoderBase-7B	46.9
CodeGen25-7B-multi	32.6
StarCoderBase-3B	50
Replit-2.7B	42.2
StarCoderBase-1.1B	71.4
CodeGen25-7B-mono	34.1
CodeGen-16B-Multi	17.2
StableCode-3B	30.2
DeciCoder-1B	54.6
SantaCoder-1.1B	50.8

https://huggingface.co/spaces/bigcode/multilingual-code-evals

- Since the open-source release, a lot of interesting projects spun up from replit-code-v1-3b

- Instruct fine tuned on CodeAlpaca and GPTeacher Code-Instruct: https://huggingface.co/teknium/Replit-v2-CodeInstruct-3B

- Quantization + ggml support to boost local inference for VSCode plugins



Nomic AI
@nomic_ai

The first GPT4All powered code copilot has launched

@morph_labs allows you to use the recently released Replit GPT4All model on Apple Metal to perform privacy aware
- Code completion (23 tok/second)
- Chatting and asking questions

all through the Rift VSCode extension.

Local LLMs power the future of software development.

Morph @morph_labs · Jun 20
The future of AI code assistants is open-source, private, secure, and on-device. That future starts today. We're excited to release Rift, an open-source AI-native language server and VSCode extension for local copilots.

morph.so

Links

https://github.com/replit/ReplitLM
https://huggingface.co/replit/replit-code-v1-3b
https://blog.replit.com/llm-training

Acknowledgements

- Madhav Singhal, Juan Sigler Priego, Bradley Heilbrun, Samip Dahal, Giuseppe Burtini, Reza Shabani, Amjad Masad & the whole Replit team
- Jonathan Frankle, Hanling Tang, Abhinav Venigalla, Vitaliy Chiley, Alexander Trott, Daya Khudia, Scott Sovine, Barry Dauber, Naveen Rao & the whole MosaicML team