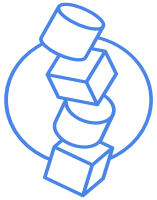
groq™

The All New

ONNX Model Zoo

Powered by MLAgility

Improved user experience



10x more models

Newer opsets

New Model Zoo

Outline

01 Thesis of MLAgility

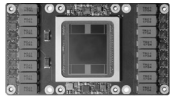
02 MLAgility Architecture

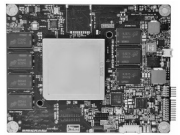
03 Collection of Models

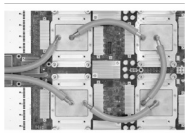
04 ONNX Benchmarking Tools

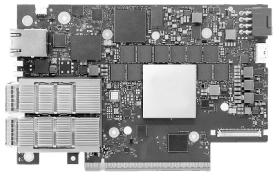
05 Benchmarking Results

05 Model Zoo Demo

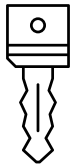




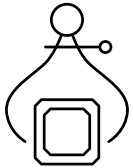




AMD	APPLE	INTEL	GROQ	NVIDIA	HABANA	AMAZON

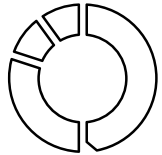


Turnkey Performance



Benchmark Results

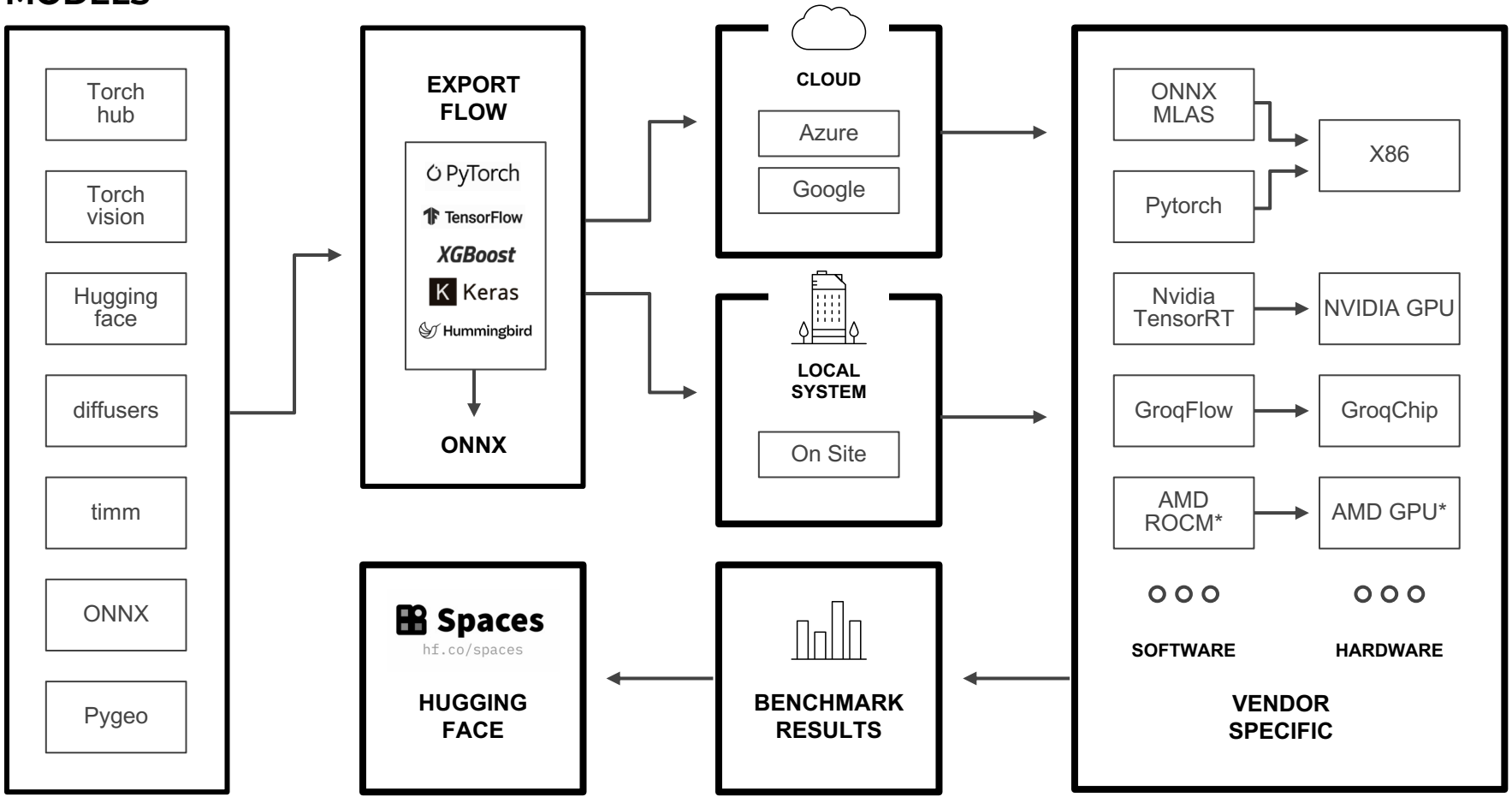
Vendor Agnostic



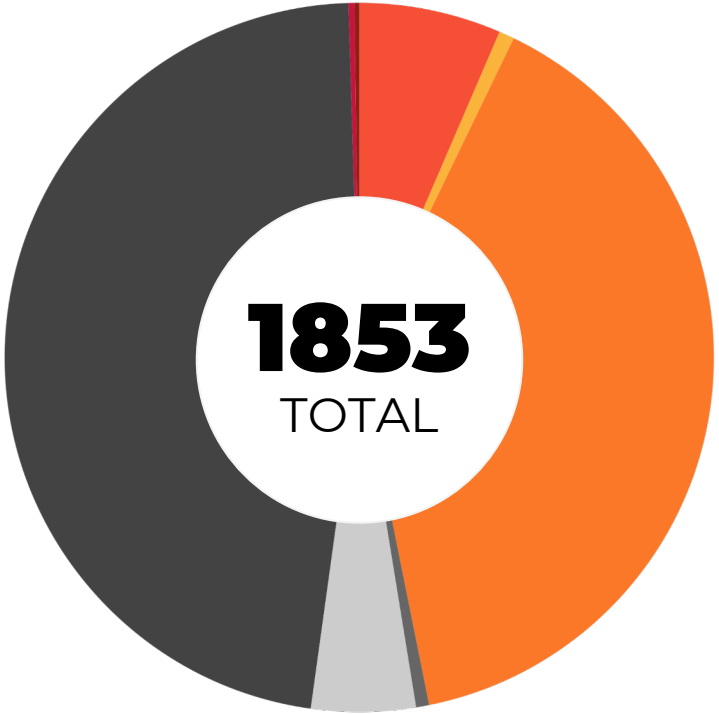
Diverse Ecosystem

The Potency of MLAgility

MODELS



The Model Zoo



	Model Repository	Count
●	Torch hub	120
●	Torch vision	13
●	timm	735
●	Graph convolutions (pygeo)	11
●	Transformers	88
●	Popular on huggingface	877
●	Diffusers	5
●	LLMs	4

ONNX Benchmarking Tools (API)

Python API

- Benchmark models by adding a single line of code to your Python script.

Input: PyTorch/Keras/Hummingbird model + Inputs

```
benchmark_model(model,inputs,device="x86")
```

```
Building "bert"
    ✓ Exporting PyTorch to ONNX
    ✓ Optimizing ONNX file
    ✓ Converting to FP16
    ✓ Finishing up

Woohoo! Saved to ~/.cache/mlagility/bert

Info: Benchmarking on local x86...

Info: Performance of build bert on Intel(R) Xeon(R) CPU @ 2.60GHz (ort v1.14.1) is:
        Mean Latency: 160.206 milliseconds (ms)
        Throughput: 6.2 inferences per second (IPS)
```

```
benchmark_model(model,inputs,device="nvidia")
```

```
Building "bert"
    ✓ Exporting PyTorch to ONNX
    ✓ Optimizing ONNX file
    ✓ Converting to FP16
    ✓ Finishing up

Woohoo! Saved to ~/.cache/mlagility/bert

Info: Benchmarking on local nvidia...

Info: Performance of build bert on NVIDIA A100-SXM4-40GB (trt v23.03-py3) is:
        Mean Latency: 0.792 milliseconds (ms)
        Throughput: 1228.1 inferences per second (IPS)
```

ONNX Benchmarking Tools (CLI)

- Benchmark models without even opening the Python file.
- Supports applications with multiple models
  (e.g. Stable Diffusion)

Input: One or more python files that instantiate and call a PyTorch model (think, a model card on huggingface)

`benchit bert.py --device x86`

```
Models discovered during profiling:

bert.py:
        model (executed 1x)
                Model Type:     Pytorch (torch.nn.Module)
                Class:          BertModel (<class 'transformers.models.bert.modeling_bert.BertModel'>)
                Location:       /home/rsivakumar/mlagility/models/transformers/bert.py, line 18
                Parameters:     109,482,240 (208.8 MB)
                Input Shape:    'attention_mask': (1, 128), 'input_ids': (1, 128)
                Hash:           95fb0413
                Build dir:      /home/rsivakumar/.cache/mlagility/bert_transformers_95fb0413
                Status:         Successfully benchmarked on Intel(R) Xeon(R) CPU @ 2.60GHz (ort v1.14.1)
                                Mean Latency:   155.729 milliseconds (ms)
                                Throughput:     6.4     inferences per second (IPS)


Woohoo! The 'benchmark' command is complete.
```

`benchit bert.py --device nvidia`

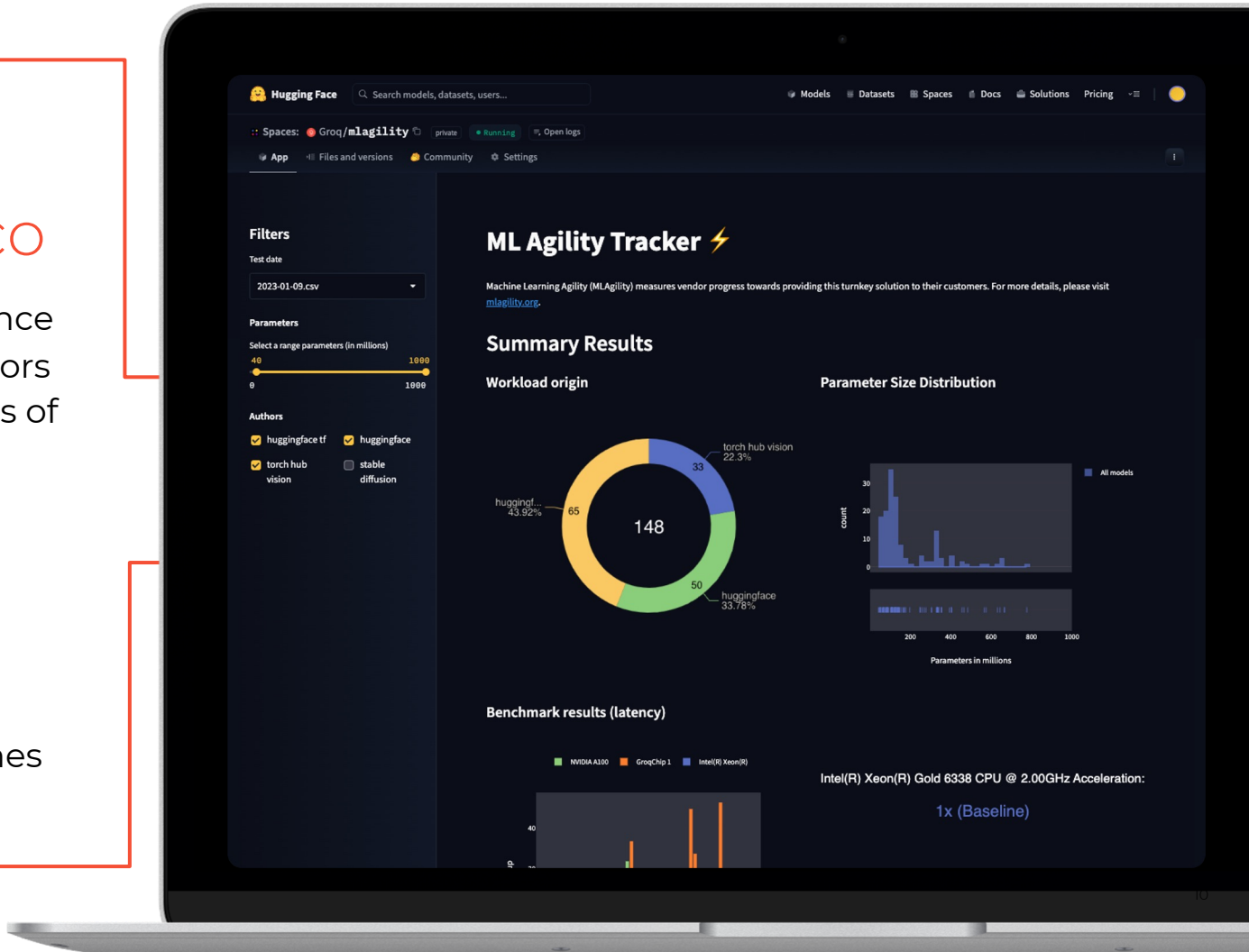
```
Models discovered during profiling:

bert.py:
        model (executed 1x)
                Model Type:     Pytorch (torch.nn.Module)
                Class:          BertModel (<class 'transformers.models.bert.modeling_bert.BertModel'>)
                Location:       /home/rsivakumar/mlagility/models/transformers/bert.py, line 18
                Parameters:     109,482,240 (208.8 MB)
                Input Shape:    'attention_mask': (1, 128), 'input_ids': (1, 128)
                Hash:           95fb0413
                Build dir:      /home/rsivakumar/.cache/mlagility/bert_transformers_95fb0413
                Status:         Successfully benchmarked on NVIDIA A100-SXM4-40GB (trt v23.03-py3)
                                Mean Latency:   0.780   milliseconds (ms)
                                Throughput:     1245.9  inferences per second (IPS)


Woohoo! The 'benchmark' command is complete.
```

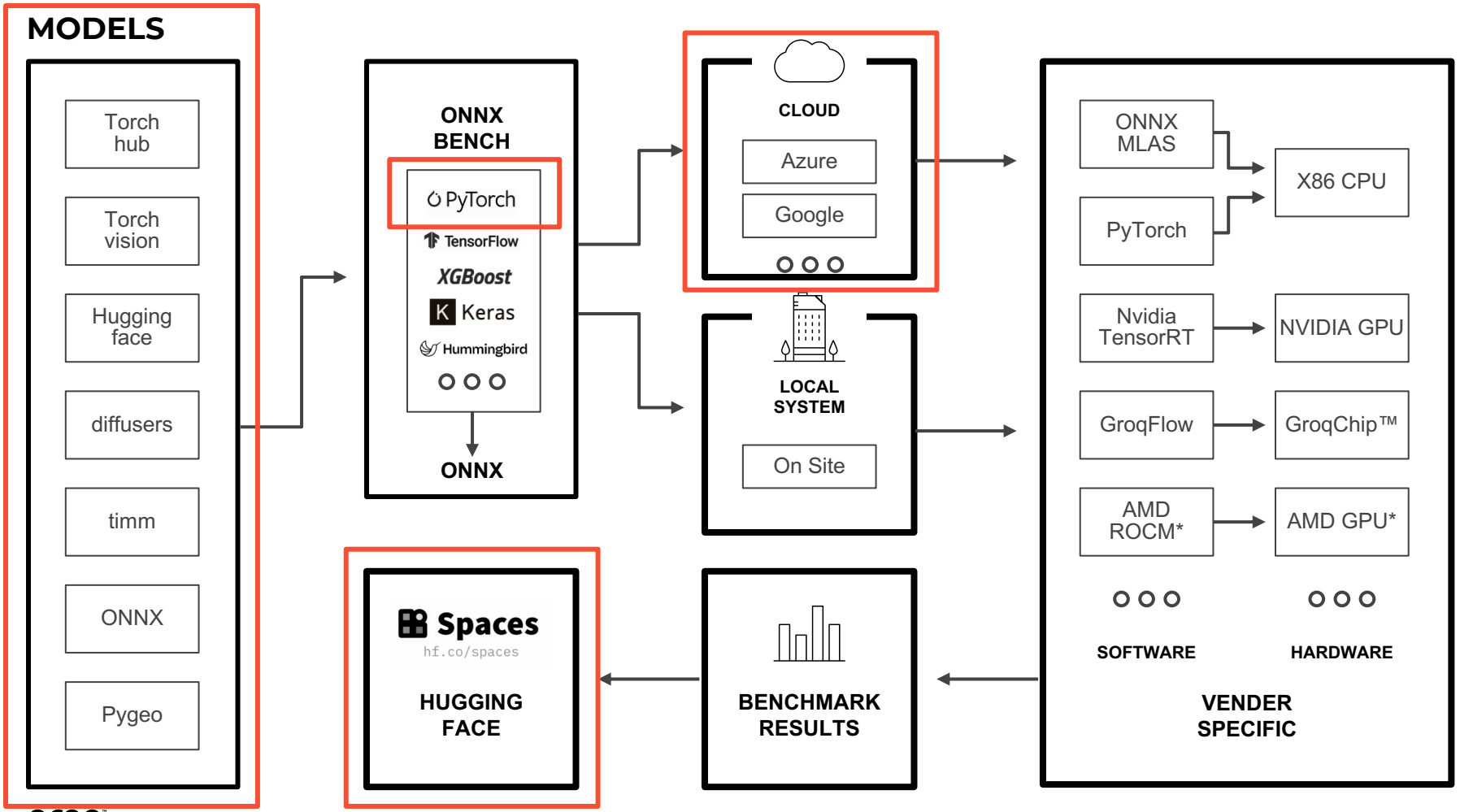
Benchmarking Results on

huggingface.co

The goal: Show performance stats of different accelerators on hundreds-to-thousands of PyTorch and TensorFlow models

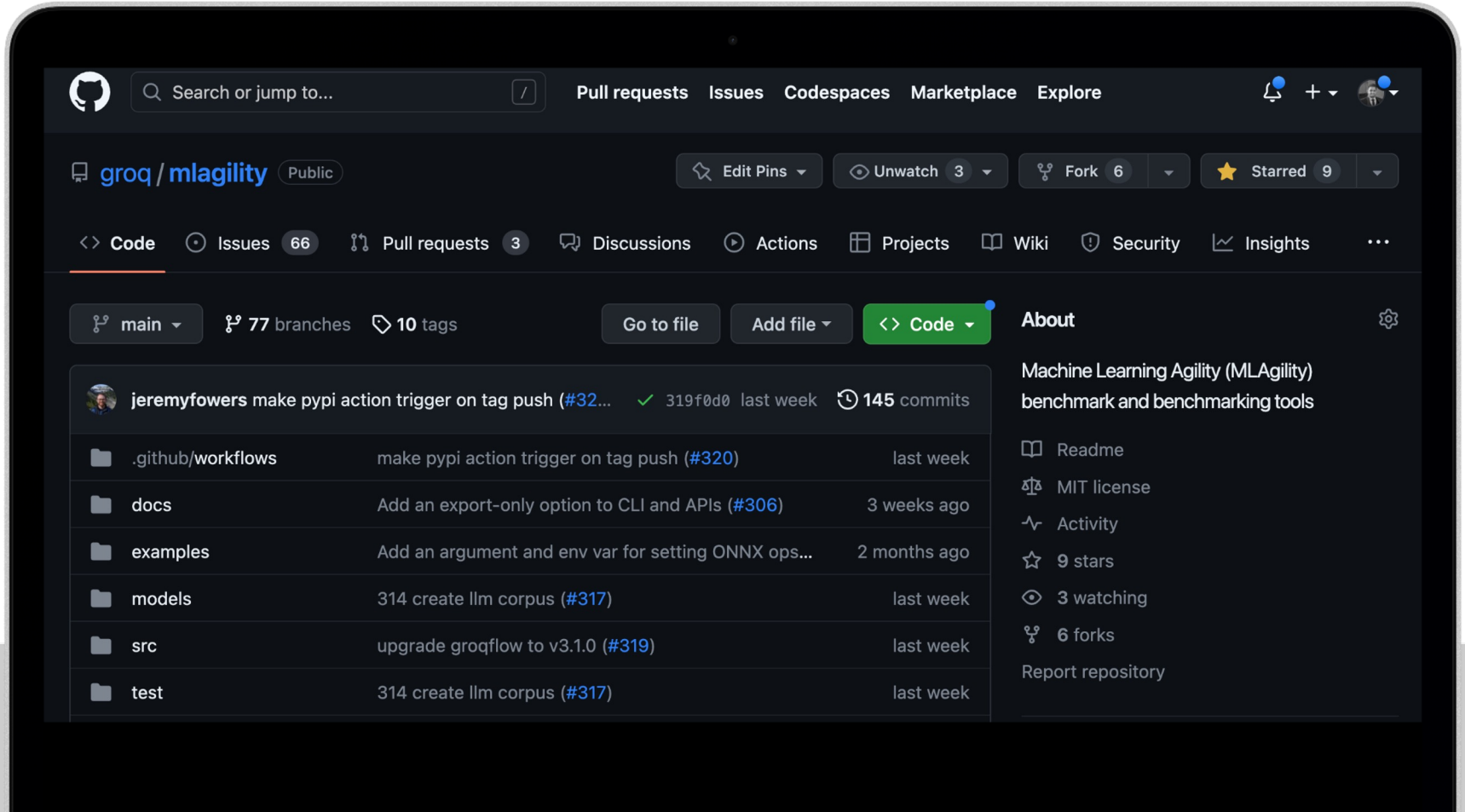
- Vendor-agnostic
- Open source
- Extensible - more accelerators, more models, more runtimes
- Reproducible



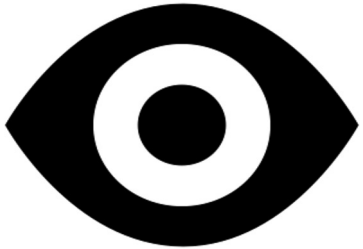


Demo

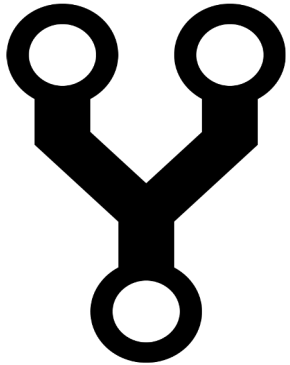
Model Zoo


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
groq / mlagility
Public
Edit Pins
Unwatch 3
Fork 6
Starred 9
Code
Issues 66
Pull requests 3
Discussions
Actions
Projects
Wiki
Security
Insights
main
77 branches
10 tags
Go to file
Add file
Code
jeremyfowers make pypi action trigger on tag push (#32...
319f0d0 last week
145 commits
.github/workflows
make pypi action trigger on tag push (#320)
last week
docs
Add an export-only option to CLI and APIs (#306)
3 weeks ago
examples
Add an argument and env var for setting ONNX ops...
2 months ago
models
314 create llm corpus (#317)
last week
src
upgrade groqflow to v3.1.0 (#319)
last week
test
314 create llm corpus (#317)
last week
About
Machine Learning Agility (MLAgility) benchmark and benchmarking tools
Readme
MIT license
Activity
9 stars
3 watching
6 forks
Report repository

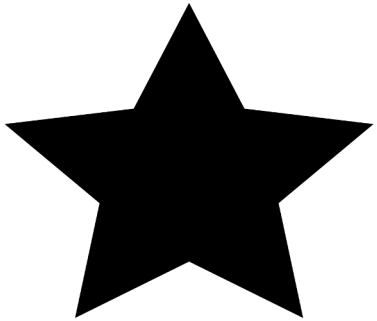
https://github.com/groq/mlagility



WATCH



FORK



STAR

groq™