# Accelerating Transformers with Hugging Face Optimum and Infinity



**MLOps World - June 2022**

https://github.com/huggingface/optimum

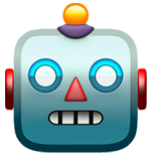
https://huggingface.co/infinity

# About us

👨‍💻 **Philipp Schmid**
💼 Technical Lead
🐦 @_philschmid
📎 /philipp-schmid-a6a2bb196

👨‍💻 **Lewis Tunstall**
💼 Machine Learning Engineer
🐦 @_lewtun
📎 /lewis-tunstall

# Plan of attack

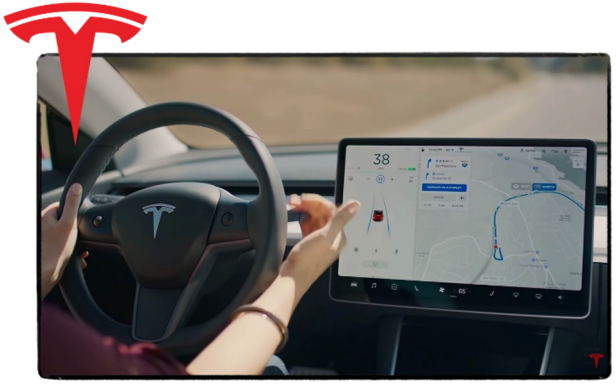


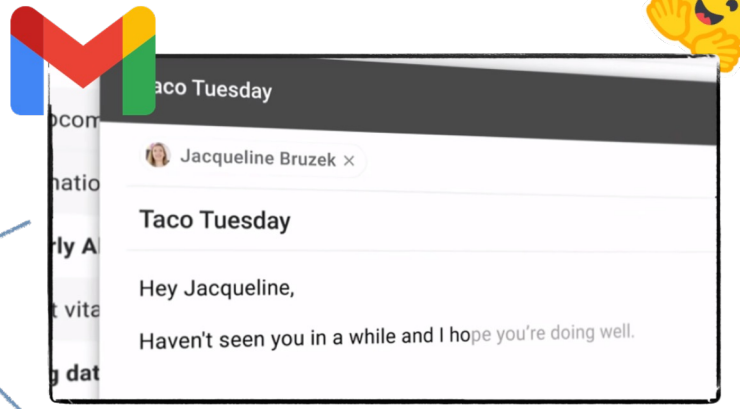
Transformers

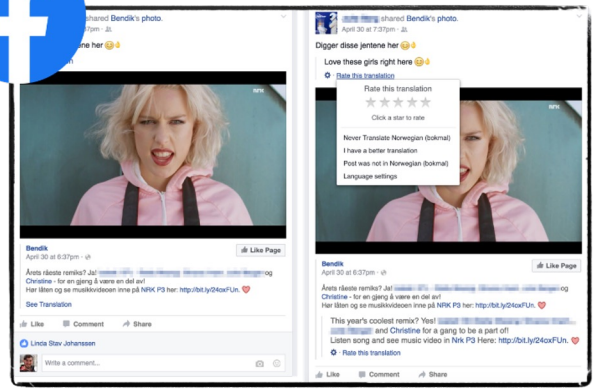
Hugging Face

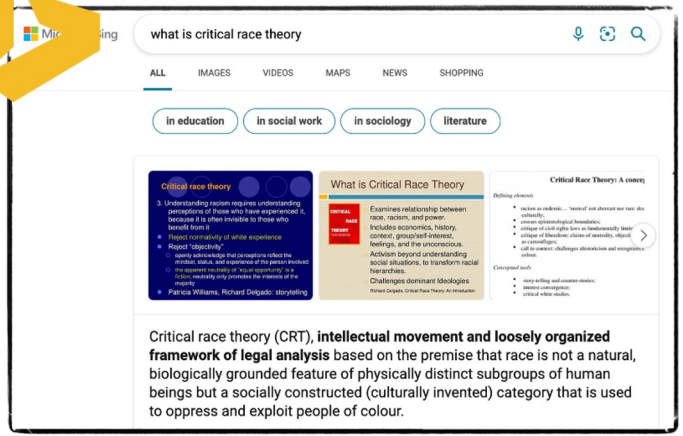
Optimum

Infinity






https://huggingface.co/

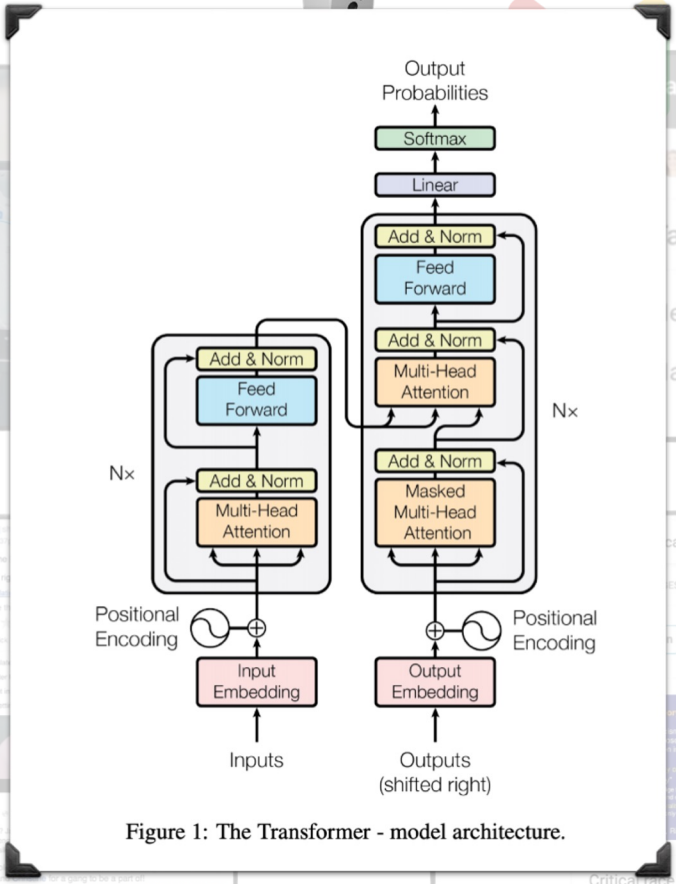
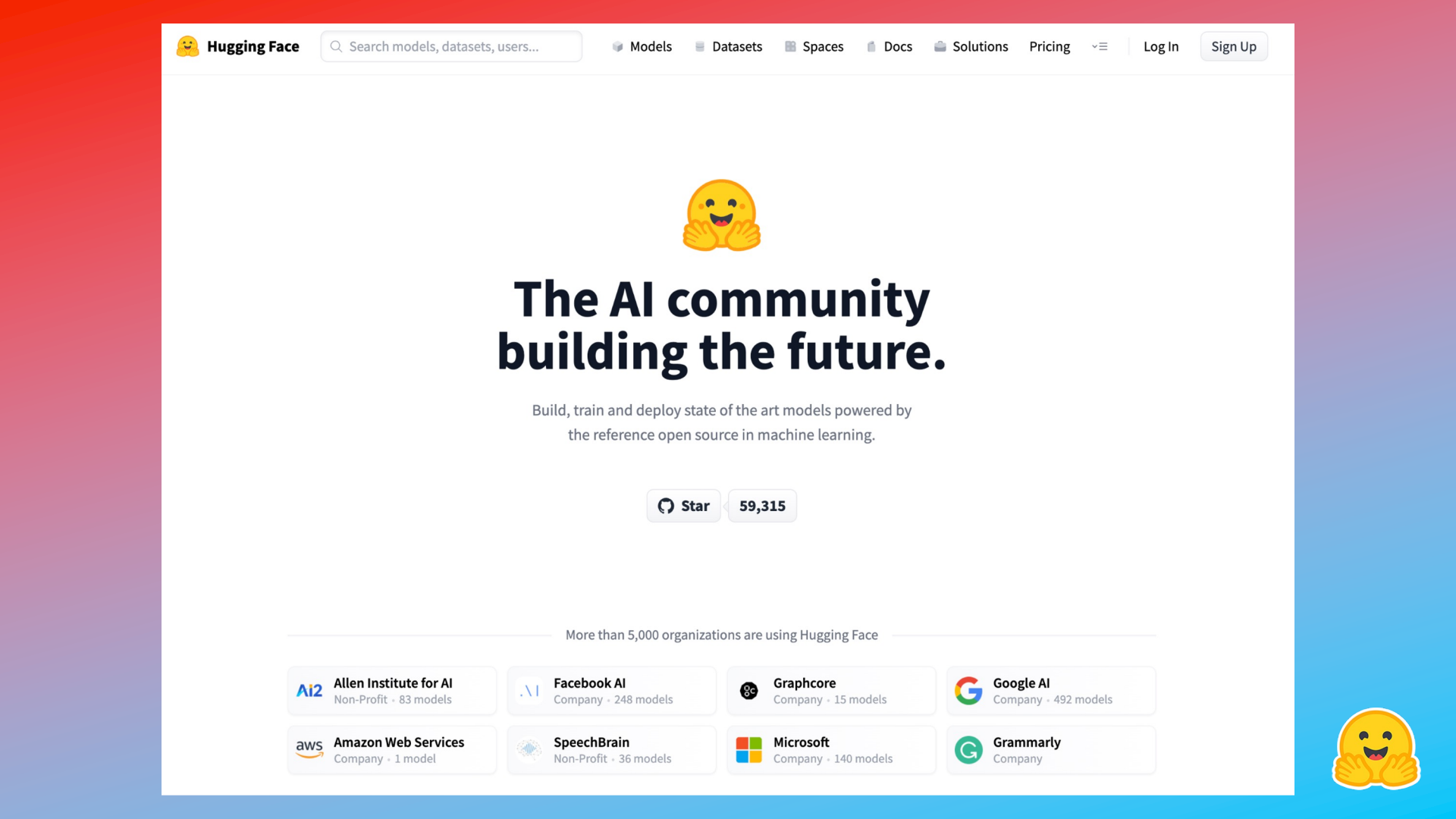



Figure 1: The Transformer - model architecture.

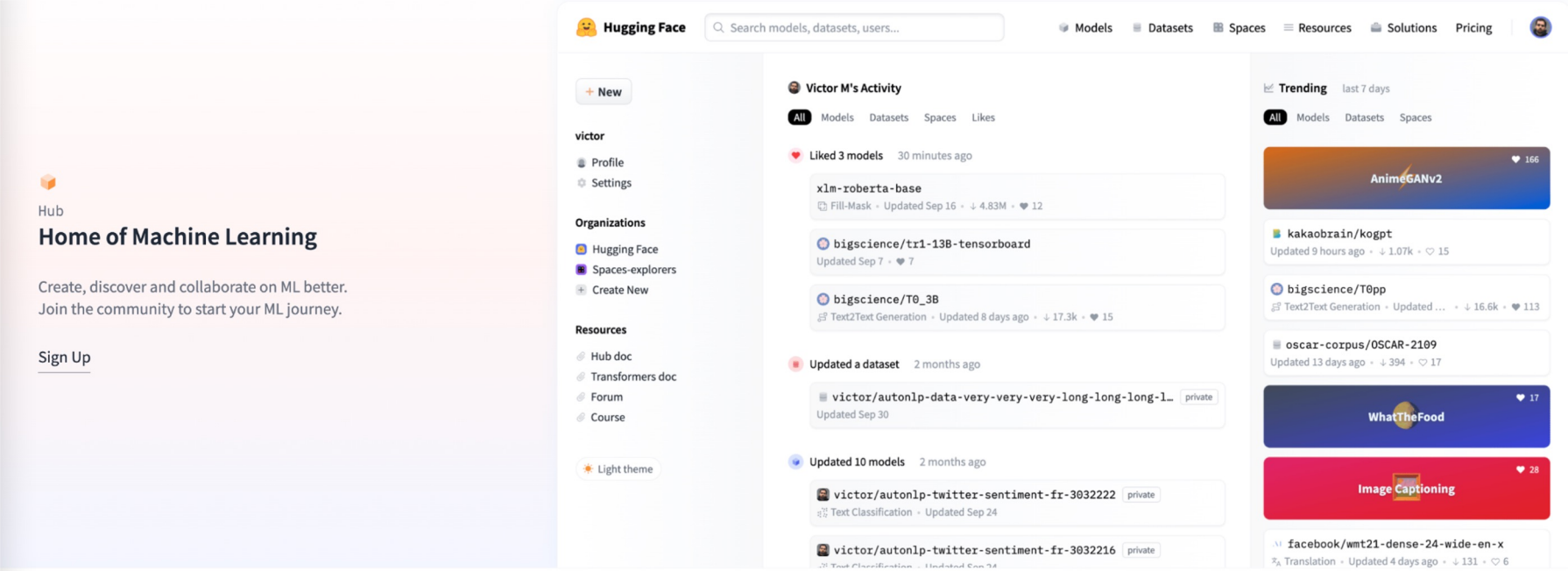
https://huggingface.co/

# Transformers for the rest of us?


Humble ML engineer


Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Docs
Solutions
Pricing
Log In
Sign Up
The AI community building the future.
Build, train and deploy state of the art models powered by the reference open source in machine learning.
Star
59,315
More than 5,000 organizations are using Hugging Face
Allen Institute for AI
Non-Profit • 83 models
Facebook AI
Company • 248 models
Graphcore
Company • 15 models
Google AI
Company • 492 models
Amazon Web Services
Company • 1 model
SpeechBrain
Non-Profit • 36 models
Microsoft
Company • 140 models
Grammarly
Company

# Hugging Face Hub - where the magic happens




https://huggingface.co/

# Solving all kinds of problems

NLP

Speech

Vision

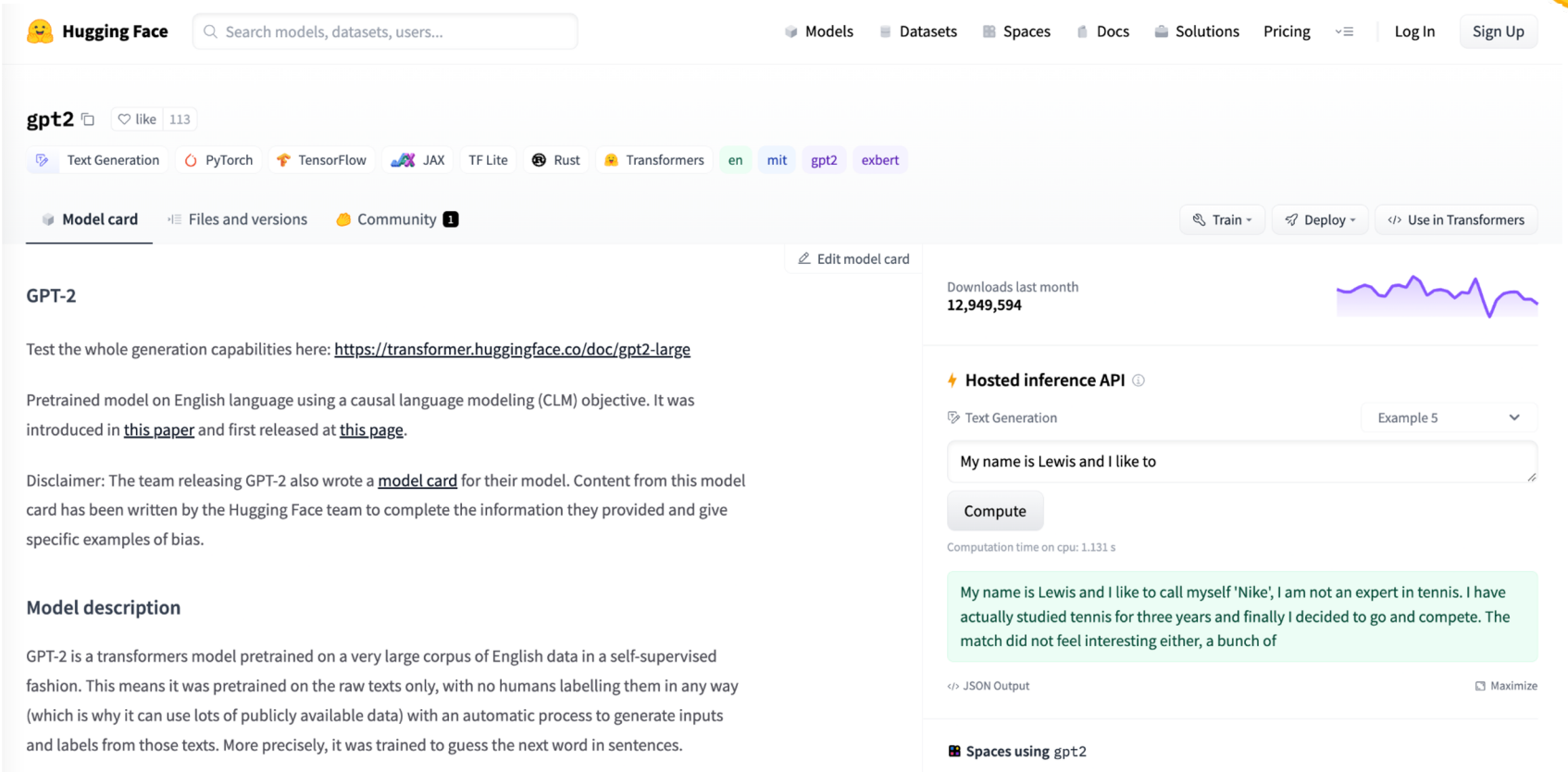

Domain X

Bio & CH

Time

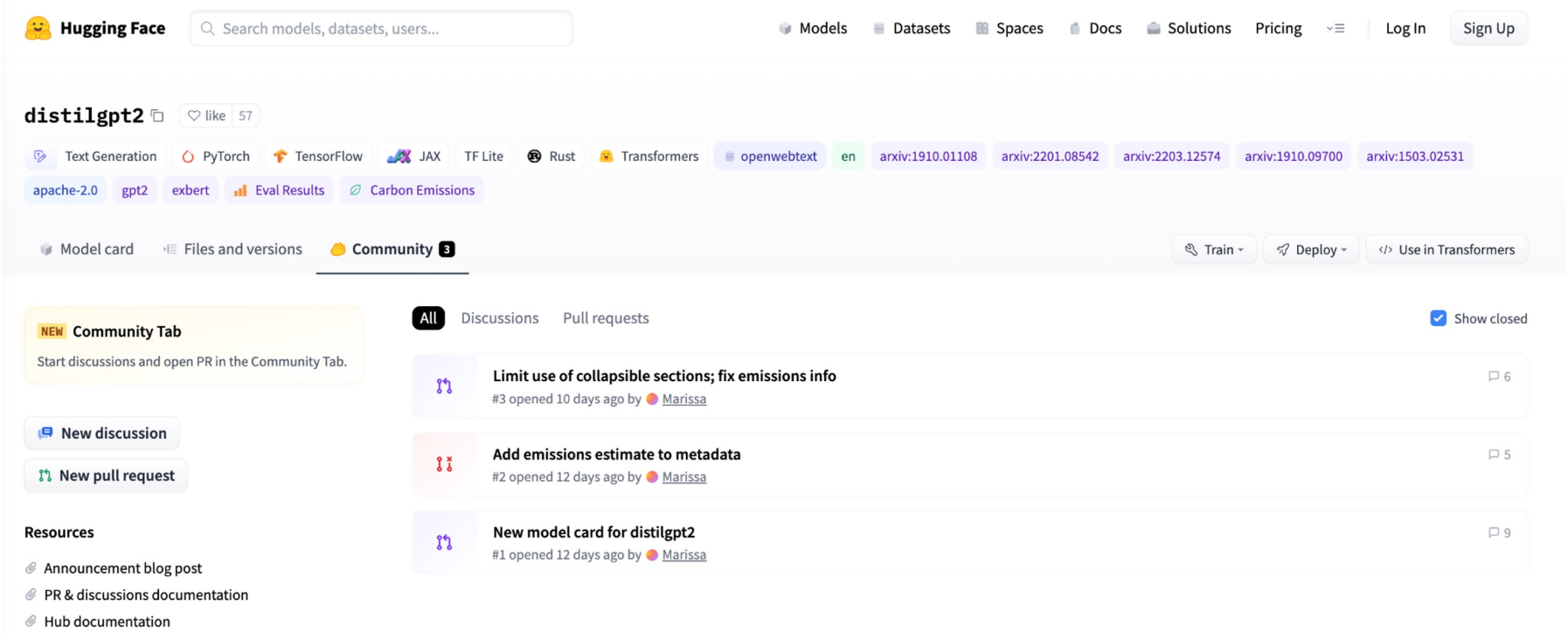
RL

# Model repositories - optimised for ML collaboration




https://huggingface.co/

# Pull requests and discussions




https://huggingface.co/

# Built on top of open-source

**transformers** Public

🤗 Transformers: State-of-the-art Machine Learning for Pytorch, TensorFlow, and JAX.

Python ☆ 64.5k ⑂ 15.1k

**datasets** Public

🤗 The largest hub of ready-to-use datasets for ML models with fast, easy-to-use and efficient data manipulation tools

Python ☆ 13.6k ⑂ 1.7k

**tokenizers** Public

💥 Fast State-of-the-Art Tokenizers optimized for Research and Production

Rust ☆ 5.7k ⑂ 478

**accelerate** Public

🚀 A simple way to train and use PyTorch models with multi-GPU, TPU, mixed-precision

Python ☆ 2.5k ⑂ 176

**optimum** Public

🏎️ Accelerate training and inference of 🤗 Transformers with easy to use hardware optimization tools

Python ☆ 546 ⑂ 43

**huggingface_hub** Public

All the open source things related to the Hugging Face Hub.

Python ☆ 439 ⑂ 107

https://huggingface.co/

# Hugging Face Optimum

https://github.com/huggingface/optimum

- An open-source library and extension of Hugging Face Transformers
- Provides a unified API of performance optimization tools to achieve maximum efficiency to train and run models on accelerated hardware
- Can be used for ***accelerated training***, ***quantization***, and ***graph optimization***, with inference support for transformers pipelines.

GRAPHCORE

Train Transformers faster with **IPUs**

habana

Accelerate Transformers Training on **Gaudi**

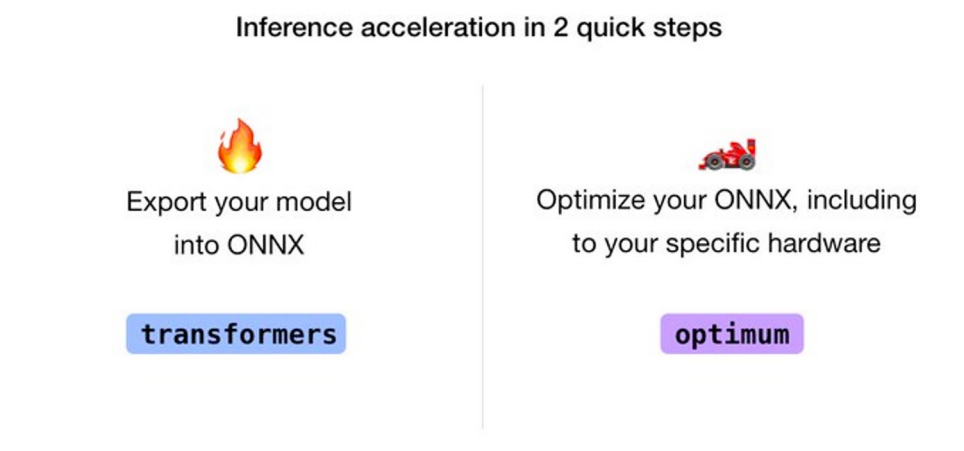
ONNX RUNTIME

intel

Scale with **Xeon**

# Hugging Face Optimum

https://github.com/huggingface/optimum




Today's workshop:

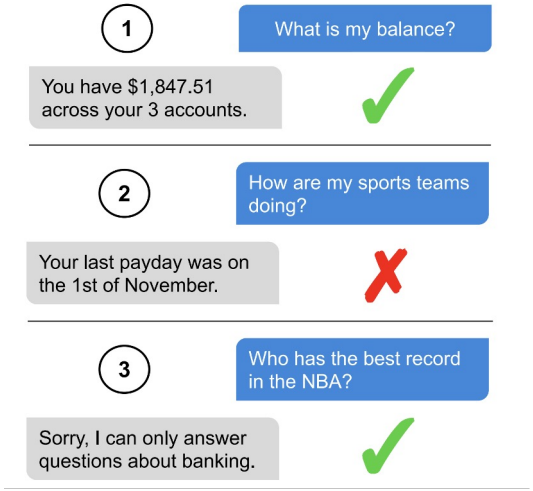
- Acceleration with ***graph optimization*** & ***quantization***
- Running inference with **ONNX Runtime** in Optimum

# Intent classification as a use case

- **Goal:** classify customer queries into ***intents*** such as Purchase, Unsubscribe, Lost Card etc
- A common component in digital assistants → low latencies (sub millisecond) are critical!
- Many challenges around identifying in-scope vs out-of-scope queries

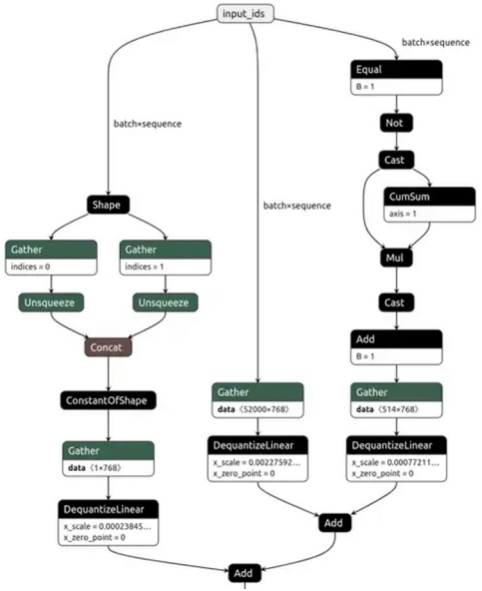
Today's workshop:

- Use the BANKING77 dataset (77 intents)
- Optimise a fine-tuned DistilBERT model



# Exporting Transformers to ONNX

- ONNX is an open standard that defines a ***common set of operators*** and file format for deep learning models
- Operators used to construct a computational graph or ***intermediate representation*** (IR) that represents flow of data through the network
- Shines when coupled with a dedicated accelerator like ONNX Runtime or TensorRT



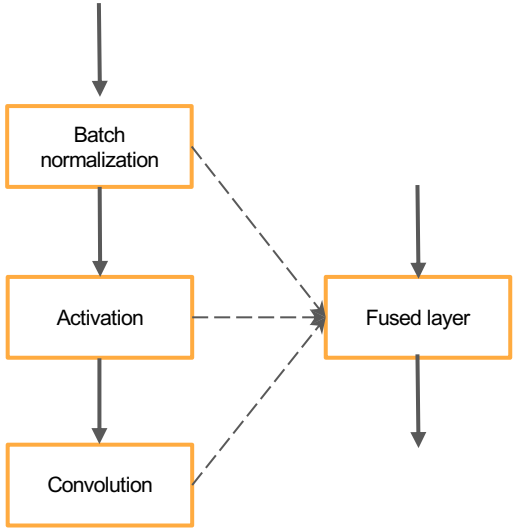

ONNX graph

# Making models faster with **graph optimization**

Basic idea:

- Apply ***transformations*** to the graph ***nodes*** and ***layout***

- **Constant folding:** evaluate constant expressions at compile time instead of runtime
- **Redundant node elimination:** remove redundant nodes without changing graph structure
- **Operator fusion:** merge one node (i.e. operator) into another so they can be executed together

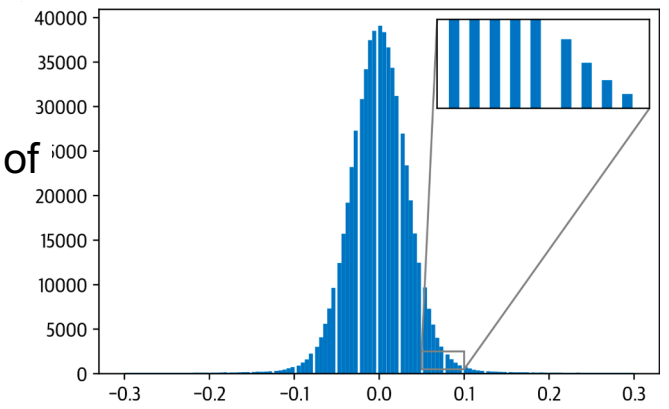



operator fusion

# Making models faster with **quantization**

Basic idea:

- Represent weights and activations with ***low-precision data types*** like 8-bit integer instead of 32-bit floating point.
- Less memory storage & faster matmuls!




In practice:

- Map range of floating-point values to ***smaller range***

$$real\_value = (int8\_value - zero\_point) \times scale$$

# Making models faster with **quantization**

Three main ways to quantize:

- **Dynamic quantization:** quantize weights & activations on-the-fly. Simplest to start with.

- **Static quantization:** precompute quantization scheme by observing activation patterns on sample of data. Generally gives better latency, but more complex to calibrate.

- **Quantization aware training:** simulate quantization during training with "fake" quantization of FP32 values.

# Hugging Face Infinity

### Plug and Predict

Infinity comes as a single-container and can be deployed in any production environment. It can easily be scaled to thousands of requests every second using orchestration services like kubernetes.

### Unmatched Performance

Infinity achieves unmatched performance for state-of-the-art transformer models. Infinity achieves 1ms latency for BERT-like models on GPU, and 4ms on CPU.

### Enterprise Ready

Infinity meets the highest security requirements and can be integrated any-where from public clouds to air gapped environments. You control your models, your data, and the traffic.

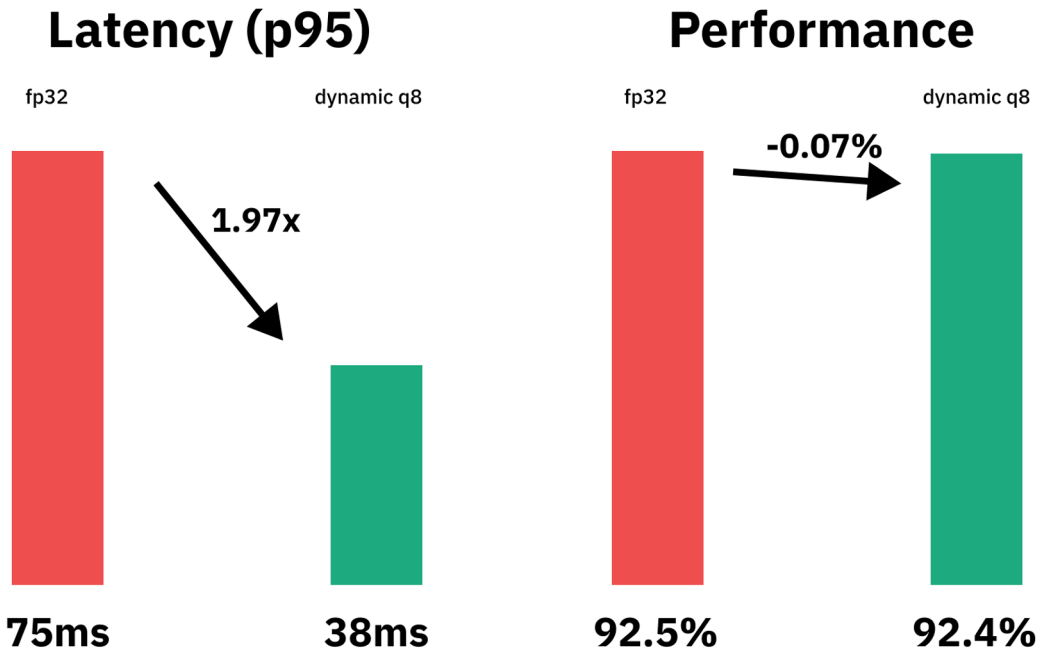
https://huggingface.co/



# Free Infinity on Hugging Face Inference Endpoint soon…

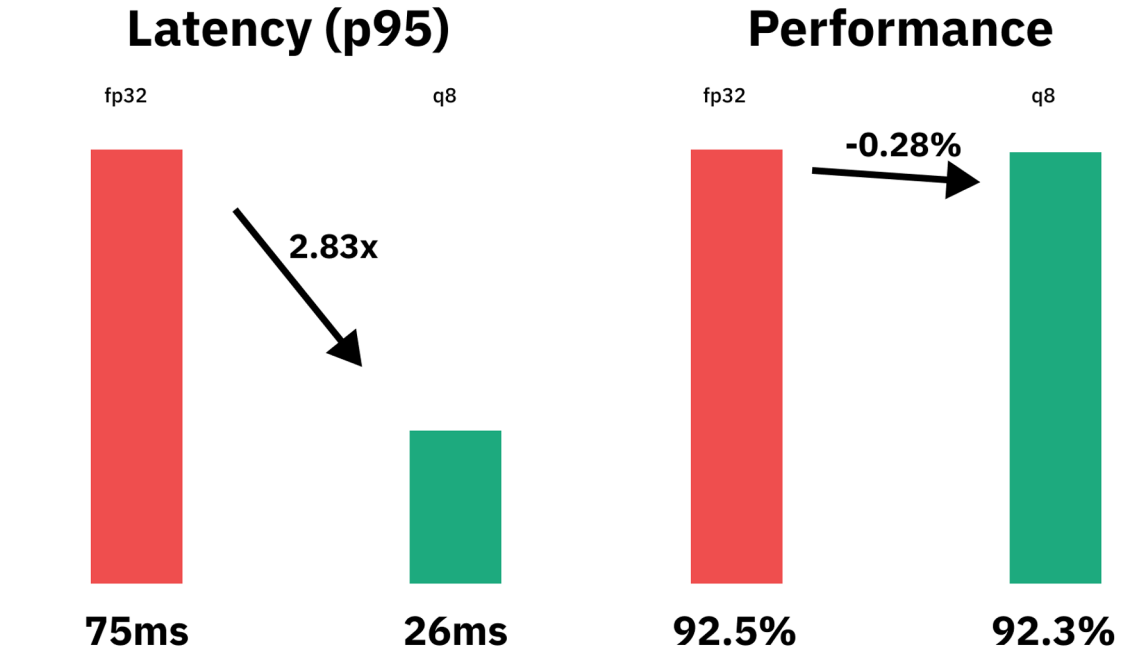
# Session 1: Dynamic Quantization & Optimization with Optimum

*run on c6i.xlarge with sequence length of 128



# Session 2: Post-Training Static Quantization with Optimum

*run on c6i.xlarge with sequence length of 128



# Where to get **help**

https://hf.co/join/discord

https://
discuss.huggingface.co/

https://github.com/
huggingface/optimum