

# Hugging Face and Intel

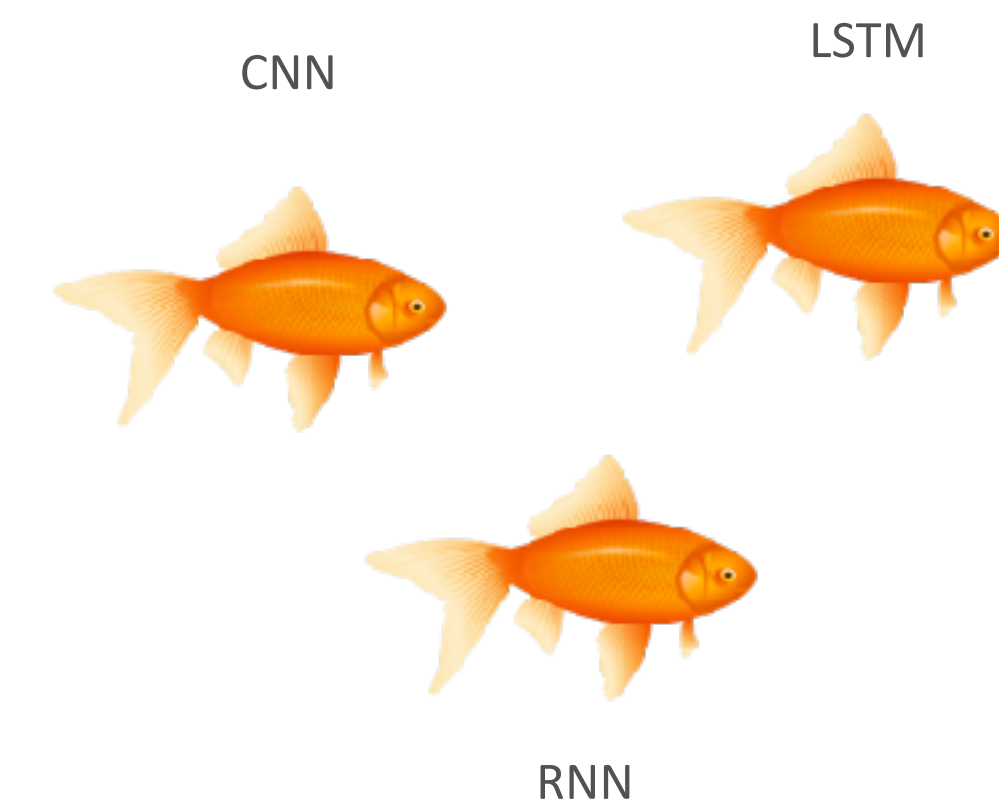
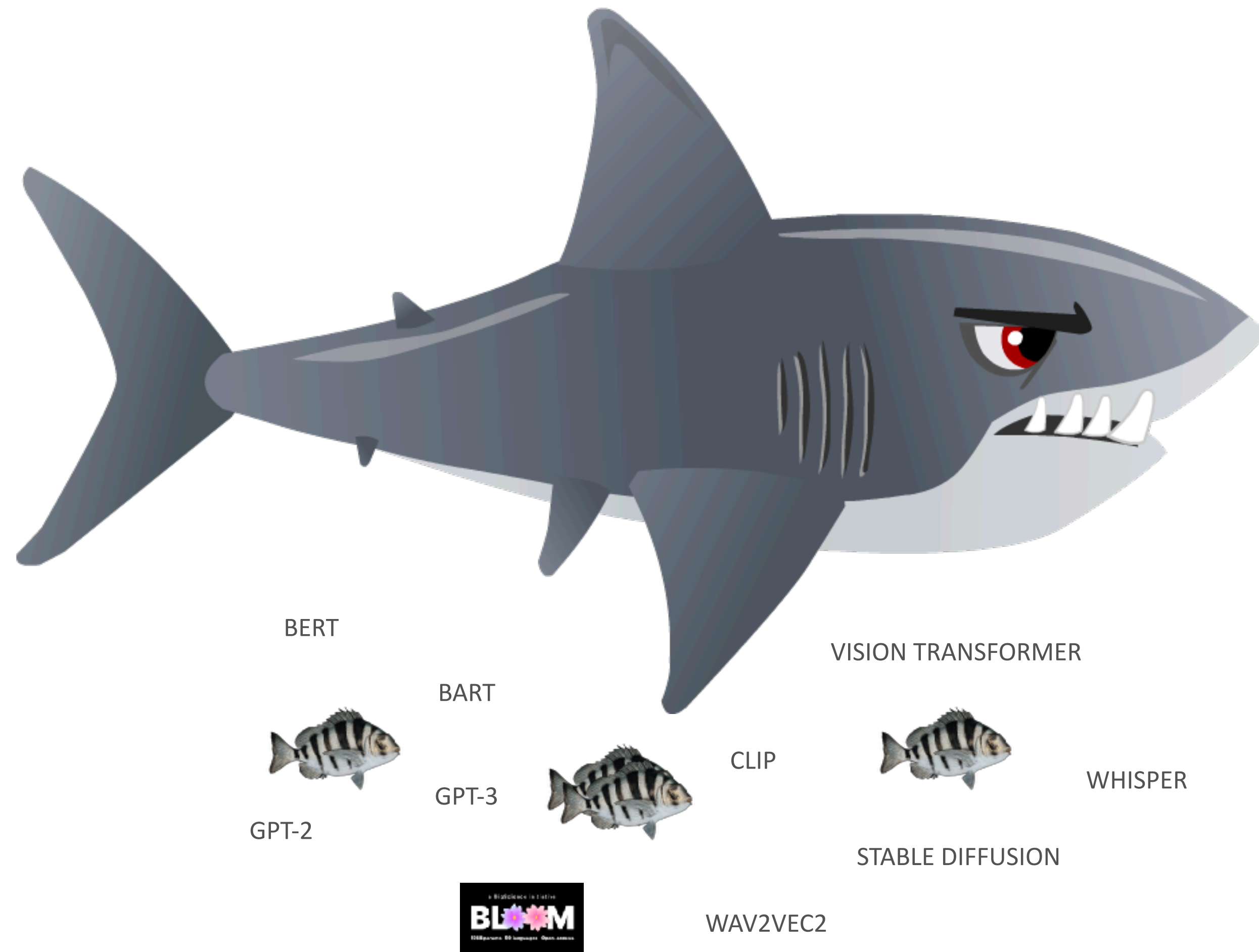
partnering to democratize ML hardware acceleration

Julien Simon, Chief Evangelist, Hugging Face



intel®

# 2022: Transformers are eating Deep Learning



*"Transformers are emerging as  
a general-purpose architecture for ML"*  
<https://www.stateof.ai> (2021)

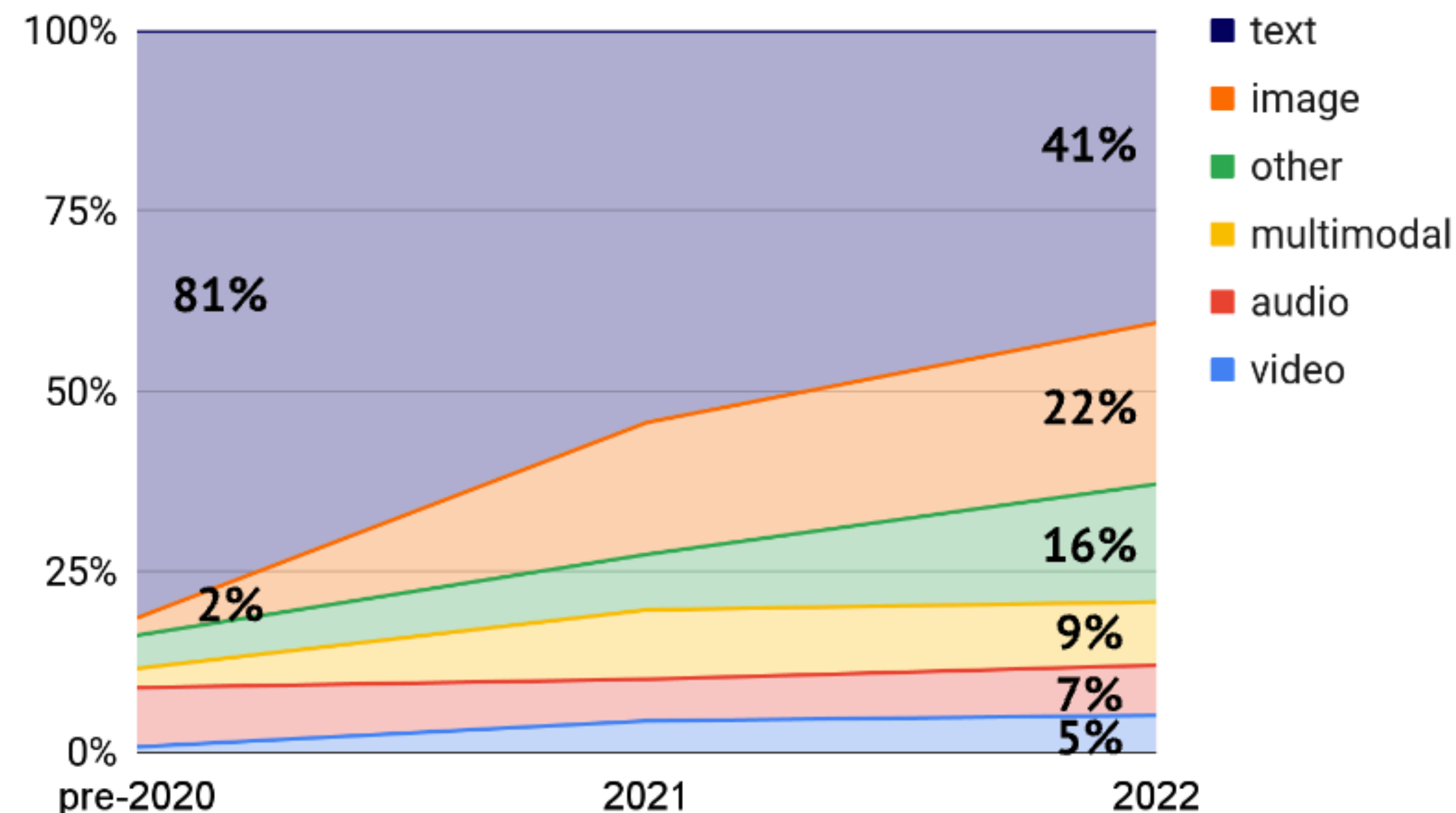
RNN and CNN usage down, Transformers usage up!  
<https://www.kaggle.com/kaggle-survey-2021>



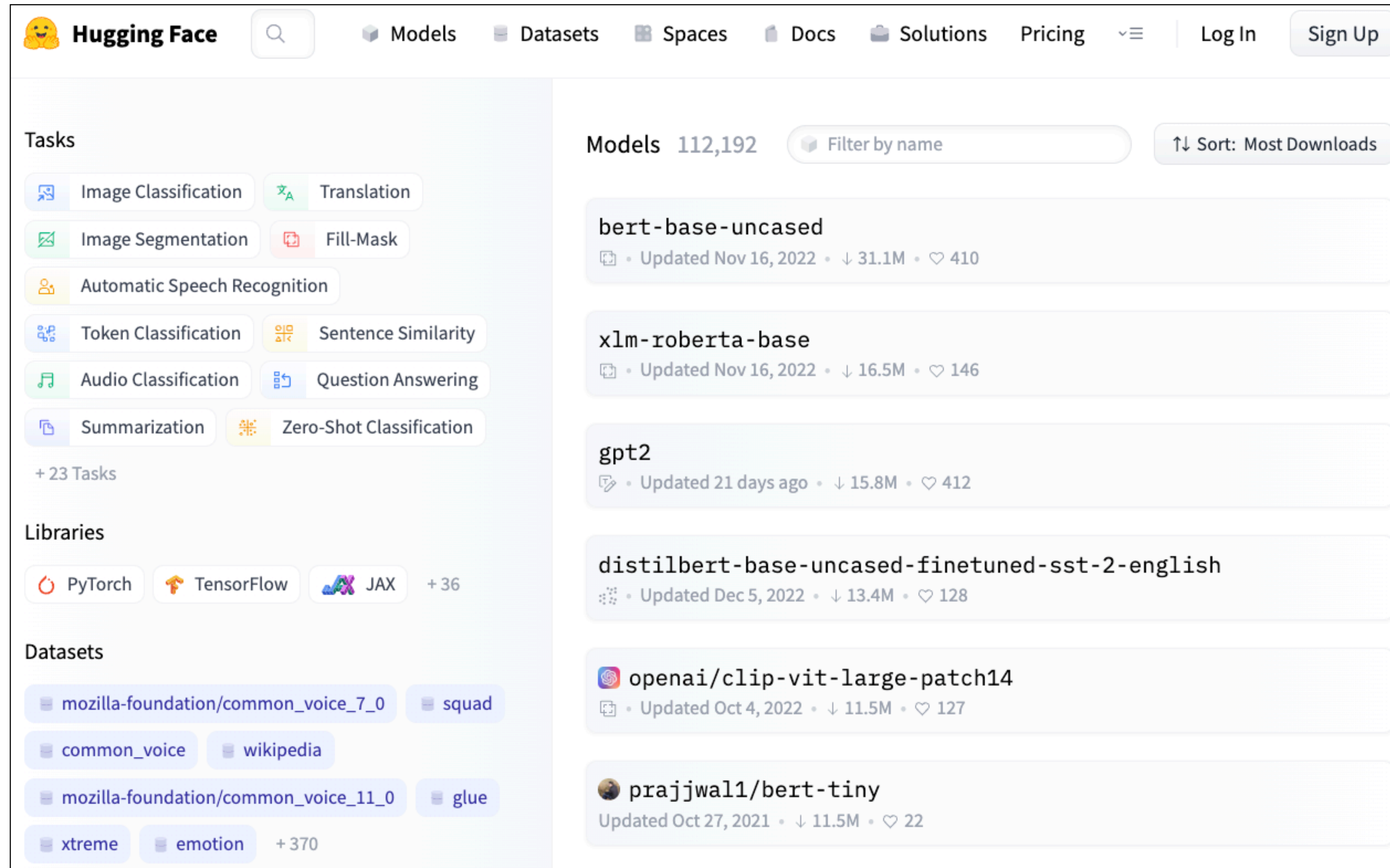
# All modalities, and multi-modal too

## Transformers are becoming truly cross-modality

- In the 2020 State of AI Report we predicted that transformers would expand beyond NLP to achieve state of the art in computer vision. It is now clear that transformers are a candidate general purpose architecture. Analysing transformer-related papers in 2022 shows just how ubiquitous this model architecture has become.



# The Hugging Face Hub: The Github of Machine Learning



The screenshot shows the Hugging Face Hub interface. At the top is a navigation bar with the Hugging Face logo, a search bar, and links to Models, Datasets, Spaces, Docs, Solutions, Pricing, Log In, and Sign Up. The left sidebar contains sections for Tasks (Image Classification, Translation, Image Segmentation, Fill-Mask, Automatic Speech Recognition, Token Classification, Sentence Similarity, Audio Classification, Question Answering, Summarization, Zero-Shot Classification, and +23 Tasks), Libraries (PyTorch, TensorFlow, JAX, and +36), and Datasets (mozilla-foundation/common\_voice\_7\_0, squad, common\_voice, wikipedia, mozilla-foundation/common\_voice\_11\_0, glue, xtreme, emotion, and +370). The main content area displays a list of models under the heading 'Models 112,192'. The list includes 'bert-base-uncased', 'xlm-roberta-base', 'gpt2', 'distilbert-base-uncased-finetuned-sst-2-english', 'openai/clip-vit-large-patch14', and 'prajjwal1/bert-tiny', each with its update date, download count, and heart count.

<https://huggingface.co>

150K models

24K datasets

25+ ML libraries: Keras, spaCY, Scikit-Learn, fastai, etc.

10K organizations

100K+ users daily

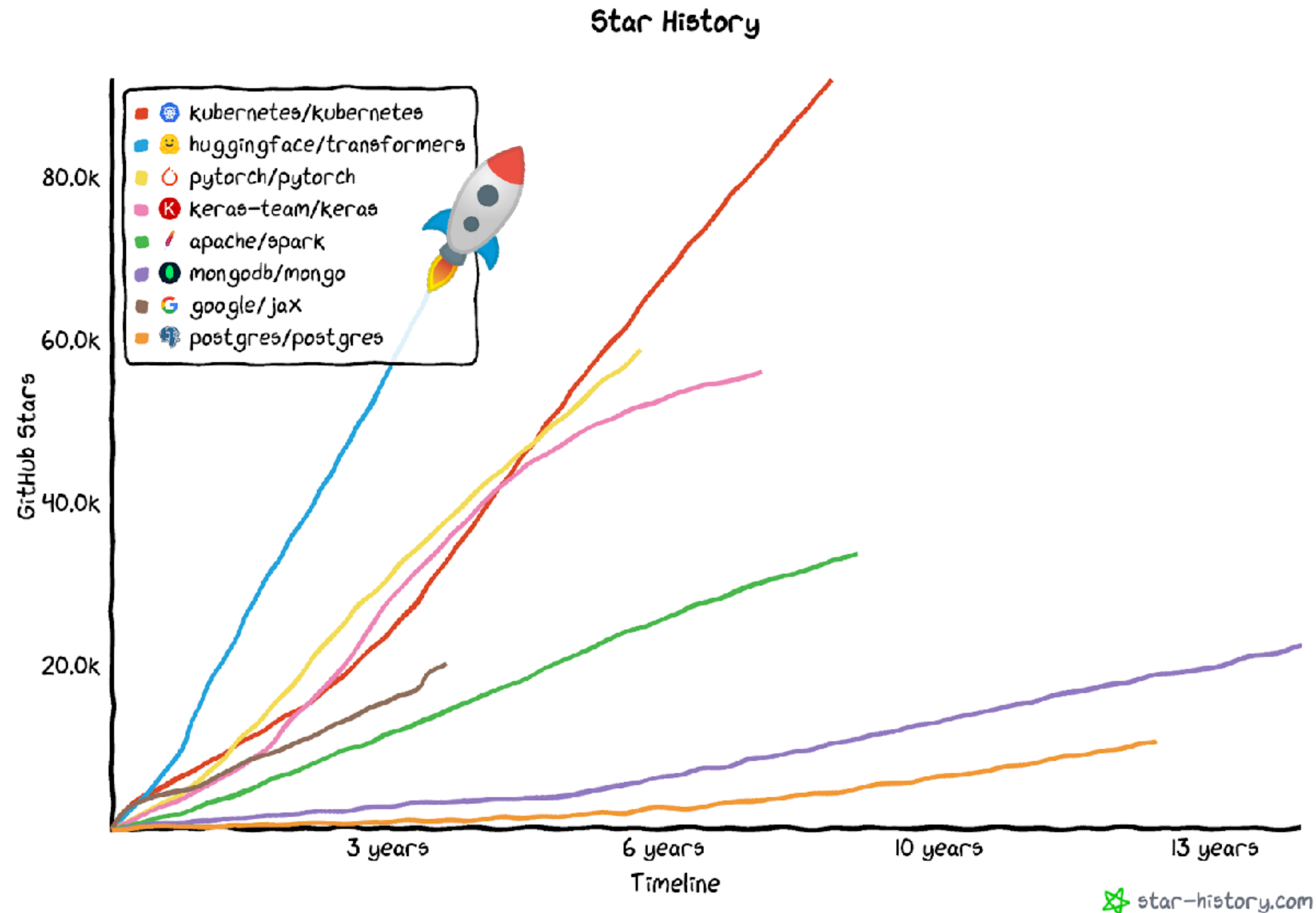
1M+ downloads daily



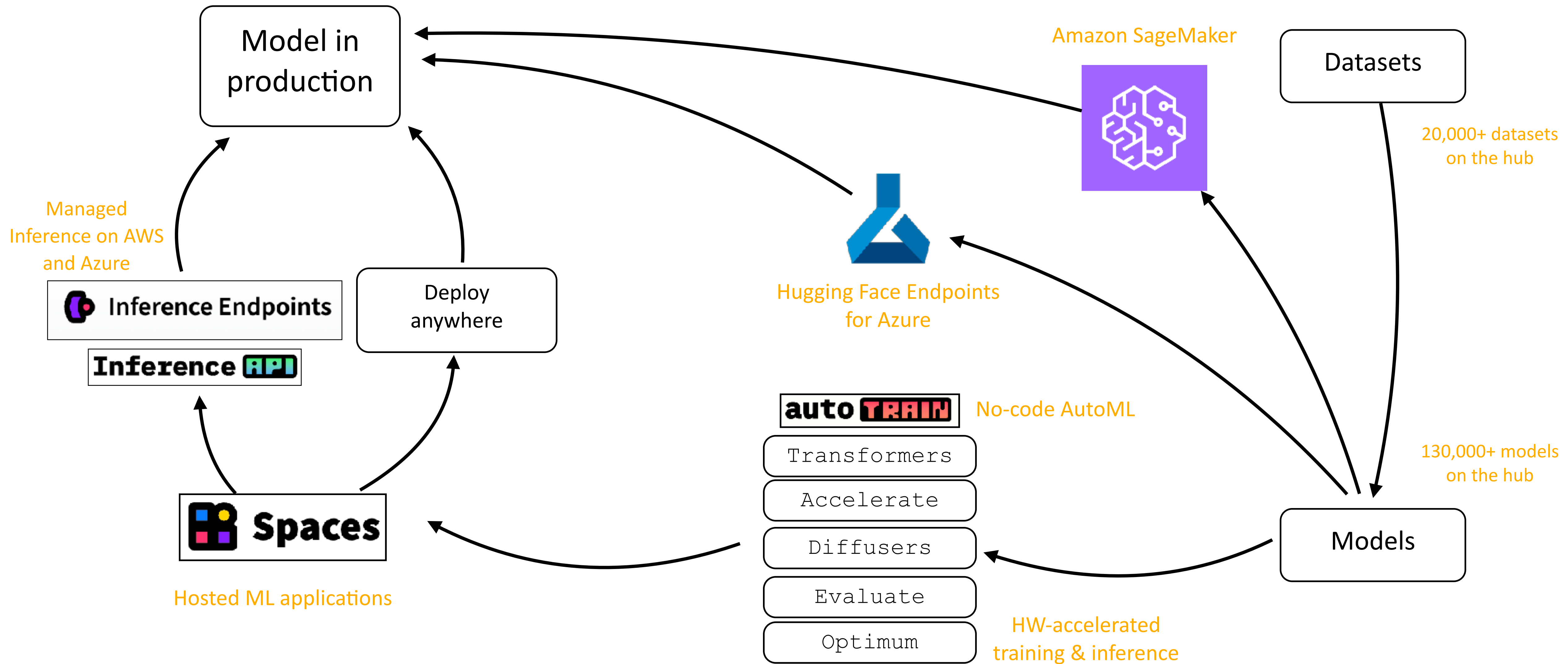


# Hugging Face: one of the fastest-growing open source projects

<https://github.com/huggingface/transformers/>



# Hugging Face at a glance



# Partnering to Democratize ML Hardware Acceleration



×

intel<sup>®</sup> ai

<https://huggingface.co/blog/intel>



# Hugging Face and Intel partnership

- Accelerate Transformer fine-tuning and inference on Intel platforms
- Deliver best-in-class cost-performance and scalability
- Build the simplest developer experience
- Share results with the Open Source community





# Accelerate Hugging Face transformers with Optimum Intel

<https://github.com/huggingface/optimum-intel>

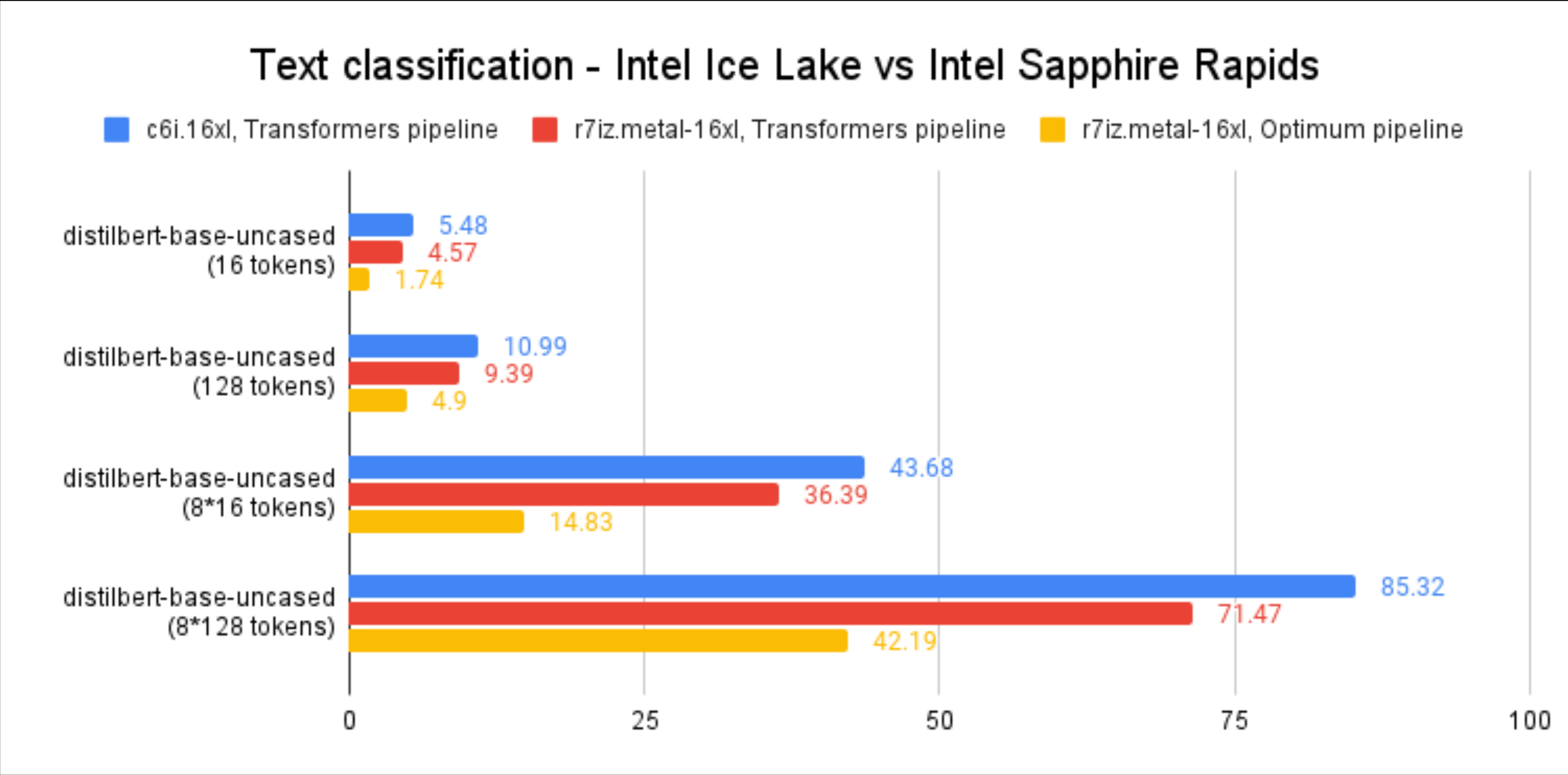
- Open source library, bringing Intel acceleration to the Hugging Face Transformers library
  - Hardware: **AVX**, **AMX**
  - Software: **Intel Extension for PyTorch** (IPEX), **Intel Neural Compressor**, **Intel OpenVINO**
- Acceleration, quantization and pruning in just a few lines of Python!

```
from transformers import AutoModelForQuestionAnswering
from neural_compressor.config import PostTrainingQuantConfig
from optimum.intel.neural_compressor import INCQuantizer

model_name = "distilbert-base-cased-distilled-squad"
model = AutoModelForQuestionAnswering.from_pretrained(model_name)
# The directory where the quantized model will be saved
save_dir = "quantized_model"
# Load the quantization configuration detailing the quantization we wish to apply
quantization_config = PostTrainingQuantConfig(approach="dynamic")
quantizer = INCQuantizer.from_pretrained(model)
# Apply dynamic quantization and save the resulting model
quantizer.quantize(quantization_config=quantization_config, save_directory=save_dir)
```



# Inference on Intel Sapphire Rapids with Optimum Intel and IPEX



# Text-to-image generation with Stable Diffusion on Sapphire Rapids

## Stable Diffusion Inference Demo Comparison

This demo shows the accelerated inference performance of a Stable Diffusion model on **Intel Xeon Gold 64xx (4th Gen Intel Xeon Scalable Processors codenamed Sapphire Rapids)** vs. **Intel Xeon Platinum 8375C (3rd Gen Intel Xeon Scalable Processors codenamed Ice Lake)**. Try it and see nearly 6x performance speedup on 4th Gen Intel Xeon!

Text-to-Image

Image-to-Image text-guided generation

Prompt

a photo of an astronaut riding a horse on mars

Inference Steps - increase the steps for better quality (e.g., avoiding black image)

20

Seed

1990633736

Guidance Scale - how much the prompt will influence the results

7,5

Generate Image

4th Gen Intel Xeon Scalable Processors (SPR)



16.9

<https://huggingface.co/spaces/Intel/Stable-Diffusion-Side-by-Side>





# Few-shot learning on Intel Sapphire Rapids with IPEx

```
Dataset({
  features: ['text', 'label'],
  num_rows: 16
})
Dataset({
  features: ['text', 'label'],
  num_rows: 38000
})
model_head.pkl not found on HuggingFace Hub, initialising classification head with random weights. You should TRAIN this model on a downstream task to use it for predictions and inference.
***** Running training *****
Num examples = 512
Num epochs = 1
Total optimization steps = 32
Total train batch size = 16
Iteration: 100%|██████████████████████████████████████| 32/32 [02:23<00:00, 4.49s/it]
Epoch: 100%|██████████████████████████████████████| 1/1 [02:23<00:00, 143.77s/it]
***** Running evaluation *****
{'accuracy': 0.9459736842105263}
```





# Distributed training on Intel Sapphire Rapids with IPEx and oneCCL

```
ubuntu@ip-172-31-41-227: ~/transformers/examples/pytorch/question-answering
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
[W reducer.cpp:1298] Warning: find_unused_parameters=True was specified in DDP constructor, but did not find any unused parameters in the forward pass. This flag results in an extra traversal of the autograd graph every iteration, which can adversely affect
performance. If your model indeed never has any unused parameters in the forward pass, consider turning this flag off. Note that this warning may be a false positive if your model has flow control causing later iterations to have unused parameters. (function
operator())
6% | 20/346 [00:27<07:20, 1.35s/it]
```

```
ubuntu@ip-172-31-19-119: ~ (ssh)
top - 18:59:30 up 8:50, 1 user, load average: 26.25, 18.91, 13.19
Tasks: 676 total, 3 running, 673 sleeping, 0 stopped, 0 zombie
%Cpu(s): 88.7 us, 2.0 sy, 0.0 ni, 8.9 id, 0.0 wa, 0.0 hi, 0.4 si, 0.0 st
MiB Mem : 515489.2 total, 499589.5 free, 13733.5 used, 2166.1 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 498289.8 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 274823 ubuntu    20   0 11.0g  6.5g 302824 R 2936   1.3 12:32.65 python3
 274824 ubuntu    20   0 11.0g  6.4g 304044 R 2875   1.3 12:24.01 python3
   183 root       rt    0     0     0  0 S  0.3  0.0  0:02.04 migration/28
   516 root      20   0     0     0  0 I  0.3  0.0  0:00.13 kworker/63:1
   621 root      20   0     0     0  0 I  0.3  0.0  0:00.10 kworker/11:1
  86107 root      20   0     0     0  0 I  0.3  0.0  0:00.07 kworker/14:1
  88023 ubuntu    20   0 11544  4492  3188 R  0.3  0.0  0:56.01 top
     1 root      20   0 168548 11164  8148 S  0.0  0.0  0:03.09 systemd
     2 root      20   0     0     0  0 S  0.0  0.0  0:00.00 kthreadd
     3 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_gp
     4 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_par_gp
     5 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 netns
     7 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:0+
     9 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:1+
    10 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 mm_percpu_wq

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 273842 ubuntu    20   0 11.0g  6.4g 306316 R 2775   1.3 12:50.70 python3
 273843 ubuntu    20   0 11.0g  6.5g 304636 R 2745   1.3 12:48.91 python3
   473 root      20   0     0     0  0 I  0.3  0.0  0:00.08 kworker/19:1-
   614 root      20   0     0     0  0 I  0.3  0.0  0:00.08 kworker/4:2--
   633 root      20   0     0     0  0 I  0.3  0.0  0:00.09 kworker/23:2-
   657 root      20   0     0     0  0 I  0.3  0.0  0:00.08 kworker/47:2-
   671 root      20   0     0     0  0 I  0.3  0.0  0:00.09 kworker/61:2-
   738 root       0 -20     0     0  0 I  0.3  0.0  0:00.10 kworker/44:1-
 72691 ubuntu    20   0 11672  4592  3268 R  0.3  0.0  2:55.81 top
 84753 root      20   0     0     0  0 I  0.3  0.0  0:00.05 kworker/33:0-
     1 root      20   0 168116 11384  8376 S  0.0  0.0  0:03.02 systemd
     2 root      20   0     0     0  0 S  0.0  0.0  0:00.00 kthreadd
     3 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_gp
     4 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_par_gp
     5 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 netns
     7 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:0H+
     9 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:1H+
    10 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 mm_percpu_wq
    11 root      20   0     0     0  0 S  0.0  0.0  0:00.00 rcu_tasks_rud+

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 273882 ubuntu    20   0 11.0g  6.5g 304656 R 2969   1.3 13:17.48 python3
 273883 ubuntu    20   0 11.0g  6.4g 306088 R 2894   1.3 13:18.45 python3
   14 root      20   0     0     0  0 I  0.3  0.0  0:02.11 rcu_sched
   184 root      20   0     0     0  0 S  0.3  0.0  0:00.16 ksoftirqd/28
   642 root      20   0     0     0  0 I  0.3  0.0  0:00.22 kworker/32:2+
   656 root      20   0     0     0  0 I  0.3  0.0  0:00.08 kworker/46:2+
     1 root      20   0 168412 11332  8328 S  0.0  0.0  0:03.14 systemd
     2 root      20   0     0     0  0 S  0.0  0.0  0:00.00 kthreadd
     3 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_gp
     4 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 rcu_par_gp
     5 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 netns
     7 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:0H+
     9 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 kworker/0:1H+
    10 root       0 -20     0     0  0 I  0.0  0.0  0:00.00 mm_percpu_wq
    11 root      20   0     0     0  0 S  0.0  0.0  0:00.00 rcu_tasks_rud+
```

<https://huggingface.co/blog/intel-sapphire-rapids>



# Getting started

Stay in touch!

@julsimon

[julsimon.medium.com](https://julsimon.medium.com)

[youtube.com/c/juliensimonfr](https://youtube.com/c/juliensimonfr)

<https://huggingface.co/intel>

<https://huggingface.co/hardware/intel>

<https://huggingface.co/docs/optimum/intel/index>

<https://github.com/huggingface/optimum-intel>

<https://www.intel.com/content/www/us/en/developer/partner/hugging-face.html>





# Hugging Face and Intel

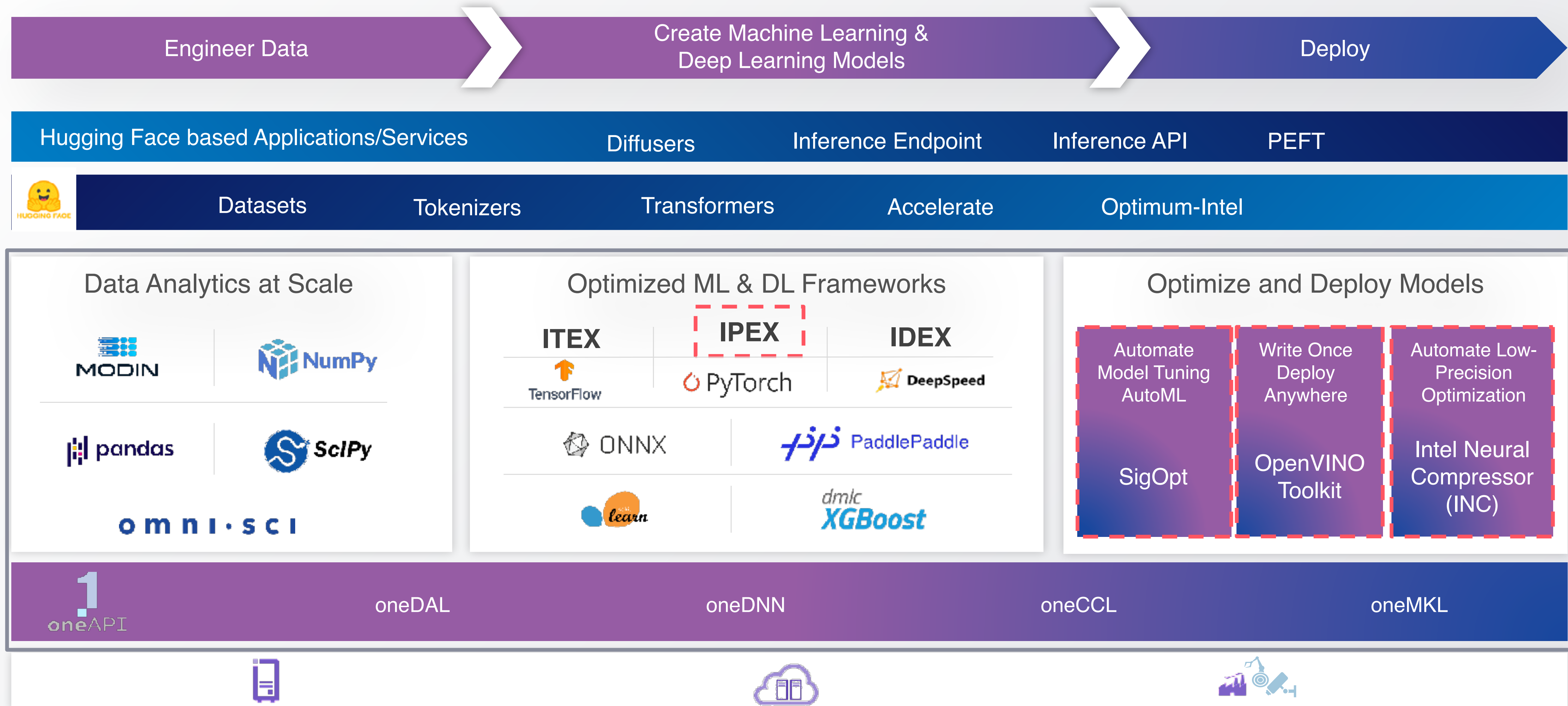
partnering to democratize ML hardware acceleration

Matrix Yao, Lead AI Software Architect, Intel Corporation

The Intel logo, consisting of the word "intel" in a lowercase, sans-serif font, followed by a registered trademark symbol (®).

intel®

# Intel Software Ecosystem deeply integrated w/ Hugging Face

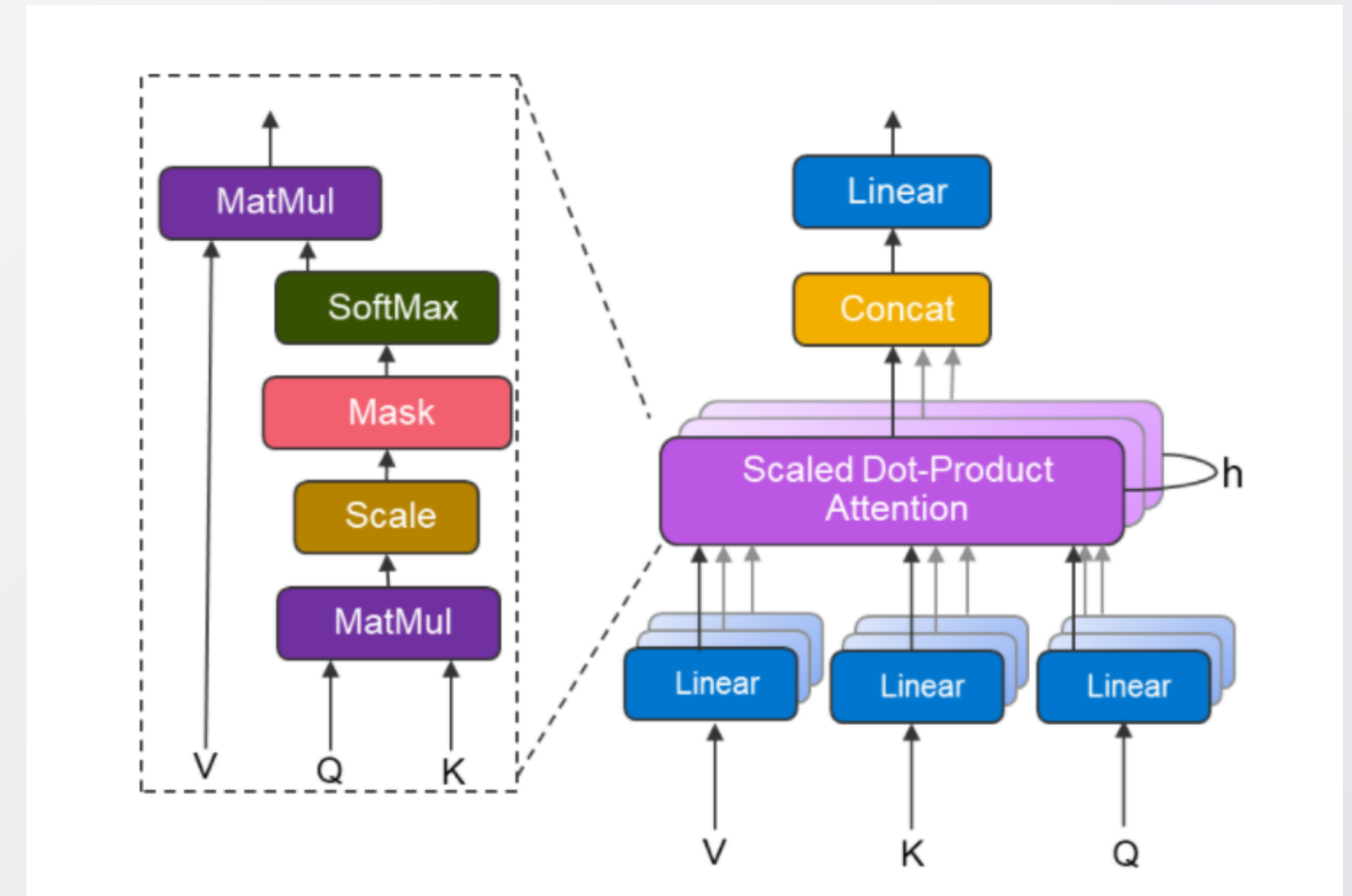
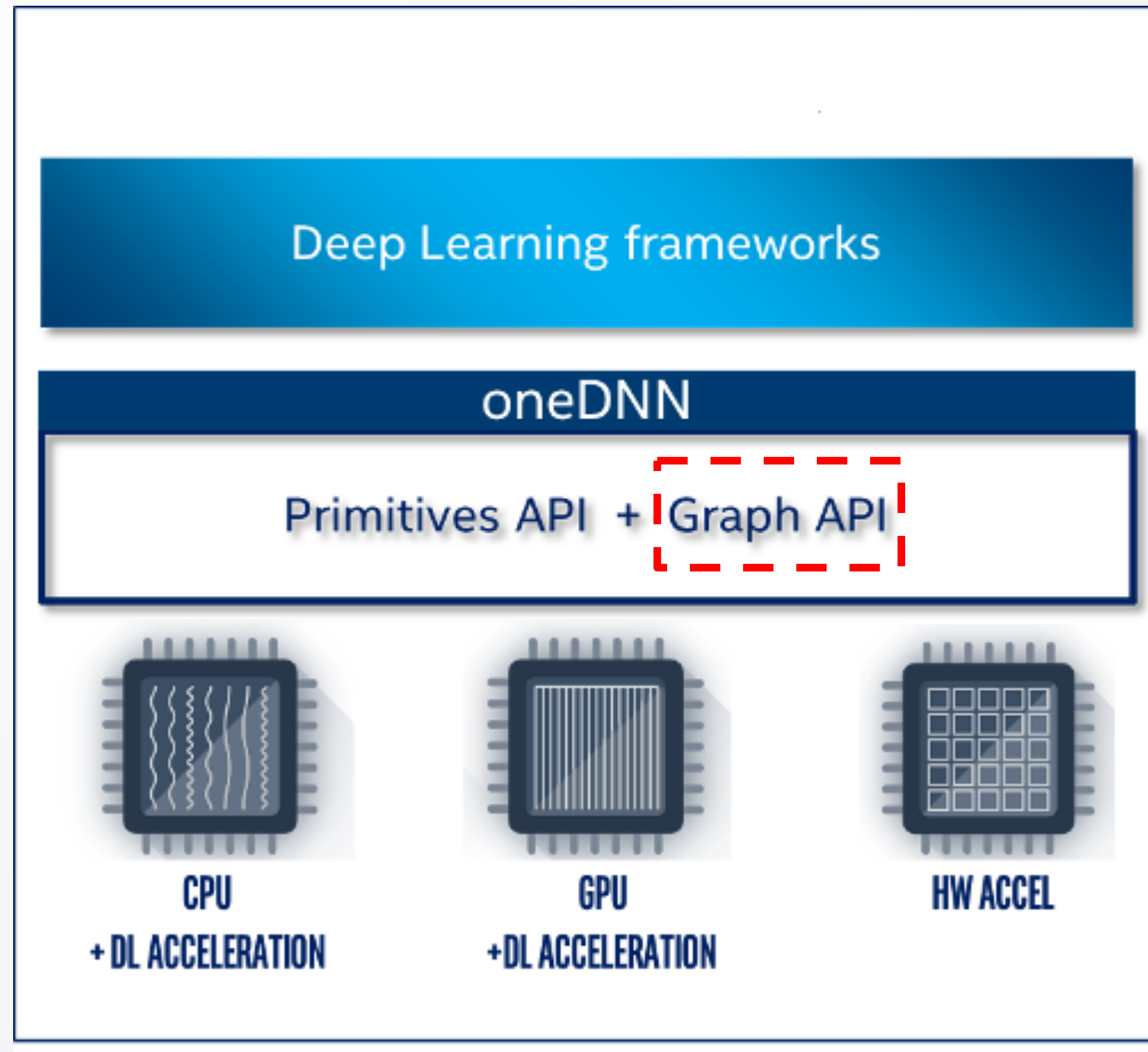


Build OOB is the best performance Hugging Face experience on Intel platforms



# Enjoy Intel SW Optimization Benefit with Zero Code Change

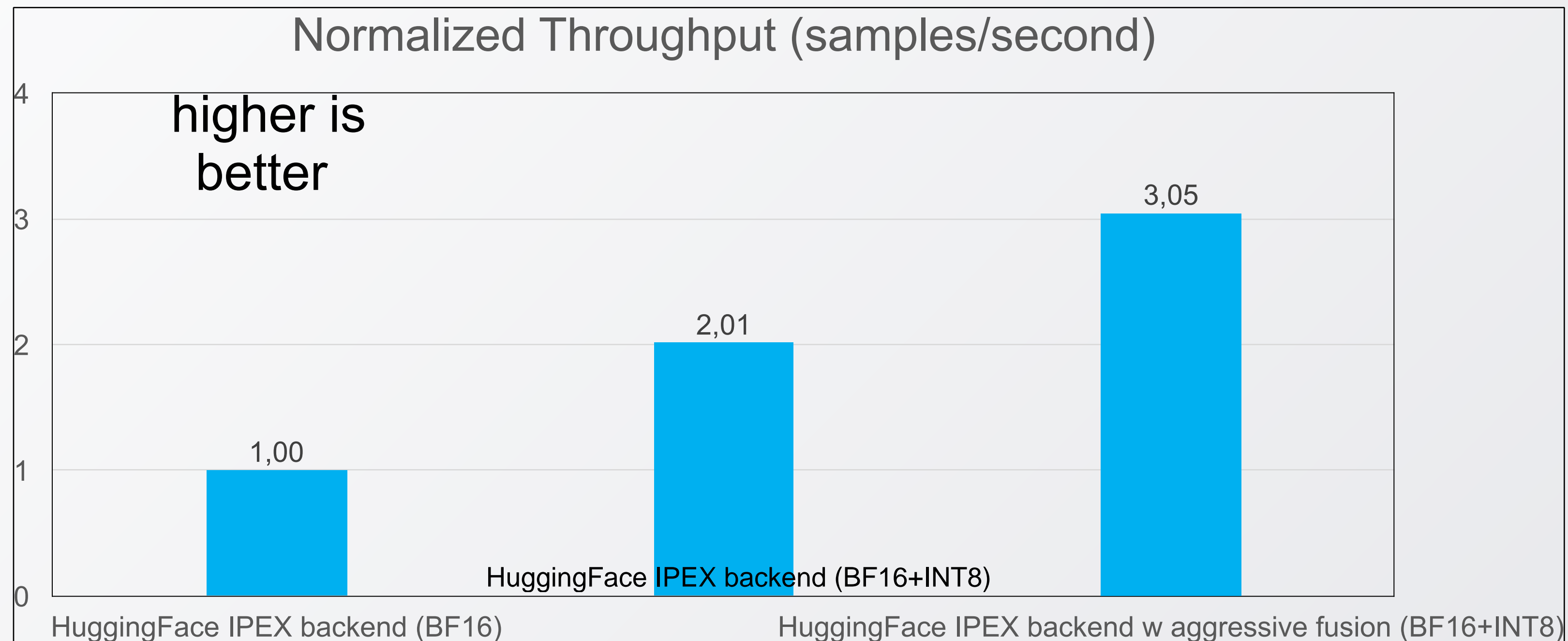
oneDNN-Graph brings INT8 MHA aggressive fusion capability and is integrated to IPEX.



*reshape + permute + transpose + bmm + mul + add + softmax + quantize + (bmm + transpose + reshape + quantize)*

fuse to single op in runtime

# Enjoy Intel SW Optimization Benefit with Zero Code Change



\* BERT-Large-uncased; SQUAD-v1.1; seq len=384, batch size per core = 1, single socket

\* Intel(R) Xeon(R) Platinum 8480+ 56 cores on Dennard Pass platform and software with 512GB memory (16x32GB DDR5 4800 MT/s [4800 MT/s]), microcode 0x90000c0, HT on, Turbo on, Rocky Linux 8.7, 4.18.0-372.32.1.el8\_6.crt2.x86\_64, 931.5G SSD. Multiple nodes connected with 200Gbps Mellanox EDR. PyTorch 1.13.1, IPEX 1.13.100, Transformers 4.26.1, oneDNN 2.6.0

Just pip install the new IPEX package, **zero code change**, HF users can enjoy oneDNN-  
Graph performance boost.