AWS Whitepaper

Scale production AI with Hugging Face Optimum Neuron and AWS Trainium and Inferentia

Authored by

Sanhita Sarkar, PhD
Global Partner Solutions,
AI/ML and Generative AI,
Amazon Web Services

Armin Agha-Ebrahim
GTM Specialist, AWS AI Chips,
Annapurna ML,
Amazon Web Services

Jeff Boudier
Head of Products,
Hugging Face

Simon Pagezy
Partner Success Manager,
Hugging Face

Florent Gbelidji
ML Engineer,
Hugging Face

Reviewed by

Kamran Khan
Head of BD/GTM,
Annapurna ML,
Amazon Web Services

Table of Contents

Introduction

The rapid evolution of artificial intelligence has created an unprecedented demand for efficient, scalable, and cost-effective solutions for training and deploying AI models for inference. As organizations across industries seek to harness the power of generative AI and large language models (LLMs), the need for optimized hardware and software portfolios has become paramount.

This whitepaper explores the powerful synergy between Hugging Face, the leading platform for machine-learning (ML) models and datasets, and Amazon Web Services (AWS). Hugging Face uses both the AWS Neuron software development kit and AWS purpose-built AI chips to optimize model performance. Hugging Face Optimum Neuron serves as a specialized interface between Hugging Face Transformers and Diffusers libraries and AWS AI chips, making models more accessible to users, while providing hardware-specific optimizations. The technologies complement each other seamlessly to address the key challenges organizations face when implementing AI solutions: model selection, ease of use, fine-tuning, integration, and achieving optimal performance-to-cost ratios.

Using tools, libraries, and solutions that Hugging Face has built, it's easier than ever for ML developers to use AWS AI chips and optimize their applications. This represents a paradigm shift in how organizations approach AI development and deployment, offering unprecedented choice, simplicity, and performance optimization for AI workloads across diverse industry applications.

Hugging Face in the Generative AI Ecosystem

The foundation of modern AI development

Hugging Face has established itself as the central hub for hosting, sharing, and accessing pre-trained models and datasets in the AI community. The platform provides open-source libraries that enable developers to train and deploy AI models with just a few lines of code, democratizing access to state-of-the-art AI capabilities.

Key differentiators

The integration of Hugging Face Optimum Neuron with AWS AI chips delivers several advantages (Figure 1).

- **Price-performance excellence:** AWS Trainium and Inferentia provide performance-to-cost advantages compared to traditional GPU-based solutions.
- **Advanced optimizations:** Hugging Face has many optimized libraries, like Transformers, Diffusers, Transformer Reinforcement Learning (TRL), and others, that enable distributed training and inference at scale.
- **Accessibility and transparency:** Simple API calls provide users with transparent access to models and deployment capabilities.

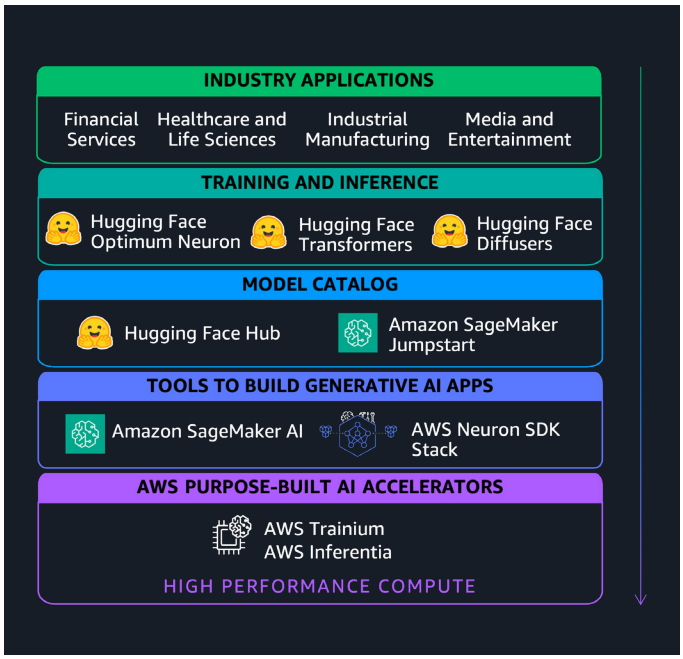


Figure 1. Hugging Face in the generative AI ecosystem

Industry applications

Hugging Face Optimum Neuron for AWS AI chips serves diverse industry verticals ranging from Financial Services and Healthcare and Life Sciences to Industrial and Manufacturing, Retail, and Media and Entertainment. Numerous applications, including multimodal AI systems, agentic AI workflows, and text summarization, are relevant across multiple industries (Figure 2).

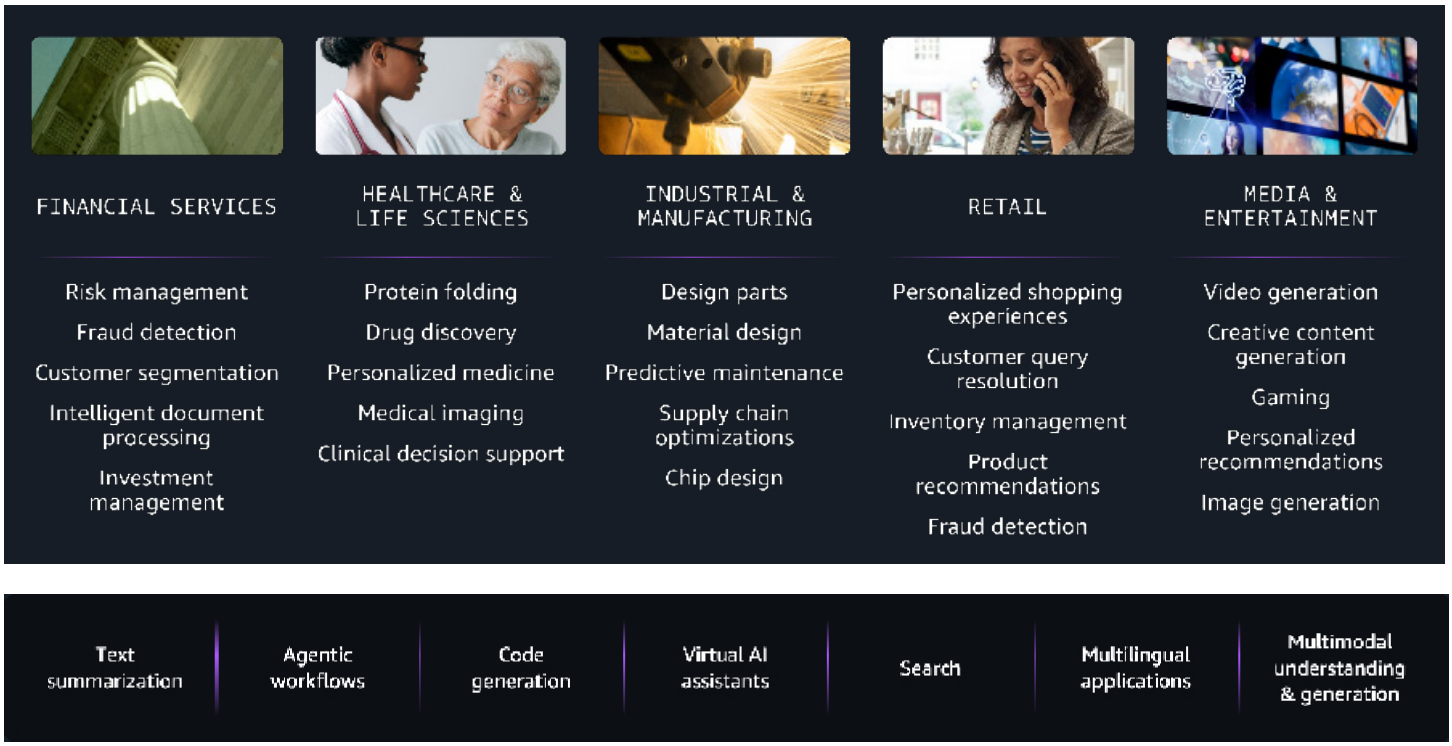


Figure 2. Customer use cases by industry

Customer Challenges and Solutions

The four fundamental questions

Organizations embarking on AI initiatives consistently face **four** critical questions:

1. **Which model should I use?** With thousands of available models and a rapidly evolving model space, it is difficult to keep up
2. **How can I move quickly?** Time-to-market pressures demand rapid deployment capabilities
3. **How can I achieve the best performance?** Optimization for specific use cases requires specialized expertise
4. **How can I lower my cost?** Budget constraints necessitate cost-effective solutions

The Hugging Face Optimum Neuron answer

The integrated platform addresses these challenges through **five** key pillars:

1. **Choice:** Access to the largest AI model catalog, combined with robust model selection tools and leaderboards, empowers more informed and confident decision-making
2. **Ease of use:** Streamlined development experience with simple API calls reduces complexity and accelerates time to value
3. **Price-performance:** Optimized hardware and software portfolios enhance price-performance and cost-effectiveness
4. **Integration:** Integration with pre-built containers enables seamless and efficient model deployment
5. **Fine-tuning:** Model fine-tuning provides the flexibility to address industry-specific needs

Model selection and deployment simplicity

The Hugging Face platform offers a comprehensive model portfolio and hosts an extensive collection of models spanning various architectures and use cases. Users can easily navigate this portfolio through:

- **Model leaderboards:** Performance comparisons across different tasks and datasets
- **Community ratings:** Peer reviews and usage statistics
- **Documentation:** Model cards and comprehensive guides and examples for each model

The deployment process is streamlined and simplified to **four** key steps:

1. **Model selection:** Choose from the world's library of pre-trained models for any workload
2. **API integration:** Utilize easy-to-use Hugging Face APIs like Estimator API, Transformers API, Pipelines API, Tokenizers API etc
3. **Deployment code:** Access example deployment code directly from the Hugging Face Model Hub
4. **Integration:** Seamlessly integrate models into existing workflows

This approach eliminates the traditional barriers to AI adoption, enabling organizations to move from concept to production rapidly.

Price-performance advantages

Modern AI workloads face significant challenges with distributed training and inference in production environments. Data and model parallelism challenges include communication bottlenecks and memory constraints when accommodating large models, along with network latency, load-balancing issues and synchronization overhead across multiple compute devices across distributed infrastructure.

Hugging Face Optimum Neuron for AWS AI chips addresses these challenges through several key optimizations.

Hugging Face optimized libraries

Hugging Face Optimum Neuron is the optimized vehicle to access other Hugging Face libraries, which include:

- **Hugging Face Transformers:** The foundational library providing access to thousands of pre-trained models with seamless integration with AWS Trainium and Inferentia.
- **Hugging Face Text Generation Inference (TGI):** High-performance text generation capabilities offering simple APIs and compatibility with various models from the Hugging Face Hub, optimized for AWS Trainium and Inferentia.

- **Hugging Face Accelerate:** Simplified scaling for training and inference, enabling the same PyTorch code to run efficiently across different hardware configurations.
- **Hugging Face Transformer Reinforcement Learning (TRL):** Designed for post-training foundation models using advanced techniques like Supervised Fine-Tuning (SFT), Proximal Policy Optimization (PPO), and Direct Preference Optimization (DPO).
- **Hugging Face Parameter Efficient Fine Tuning (PEFT):** Advanced fine-tuning techniques that reduce computational requirements while maintaining model performance.
- **vLLM:** Optimum Neuron is integrated as a plugin to vLLM to deliver out-of-the-box performance when deploying open LLMs using vLLM when Optimum Neuron is present.

Among many other built-in optimization features, Hugging Face Optimum Neuron enhances distributed training and inference workloads with:

- **Optimized communication algorithms** for data parallelism to improve communication speed and reduce latency
- **Smart model partitioning** with techniques like tensor parallelism and pipeline parallelism
- **Optimized frameworks** like PyTorch with better support for data and model parallelism
- **New parallelism techniques** to address limitations of traditional data and model parallelism
- **Continuous batching** for aggregating multiple incoming requests into single batches for increased throughput and reduced overall latency
- **Model quantization** for reducing precision of model weights and activations to improve inference speed and reduce memory footprint without significant accuracy loss

AWS AI chips for generative AI

AI model performance requirements vary based on workload type. For example:

Compute-bound workloads: Compute-bound workloads such as LLM training and LLM context encoding demand high TFLOPS performance. Similarly, vision model training has high computational demands. A key characteristic of such workloads is significant arithmetic intensity (ops/byte).

Memory-bound workloads: Memory-bound workloads such as LLM context decoding and LLM output generation require lower arithmetic intensity but demand high memory bandwidth.

Supporting the spectrum of workloads calls for advancements throughout the generative AI stack, including at the silicon layer. The AWS portfolio of AI chips, such as AWS Trainium and AWS Inferentia, is designed to support a wide range of workloads from LLM training and inference to multimodal and Mixture of Experts (MoE) models.

AWS Trainium2 and Inferentia2 architecture and performance

AWS Trainium2 is optimized for distributed training and inference workloads, featuring eight NeuronCore-v3 cores. Each core includes specialized Compute Engines (Scalar, Vector, Tensor, and GPSIMD—General Purpose Single Instruction Multiple data), on-chip SRAM (Static Random Access Memory) optimized for data locality and prefetch operations, dense compute capabilities, and advanced NeuronLink-v3 chip-to-chip interconnect technology. For more information, please refer to the architecture section.

An Amazon EC2 Instance trn2.48xlarge instance is powered by 16 Trainium2 chips interconnected in a 4x4, 2D torus topology. This configuration enables each chip to connect directly to its nearest neighbors, reducing latency for operations like all-reduce in distributed training. The NeuronLink-v3 chip-to-chip interconnect facilitates collective communication between Trainium2 chips during distributed training and inference while enabling memory pooling across all 16 chips.

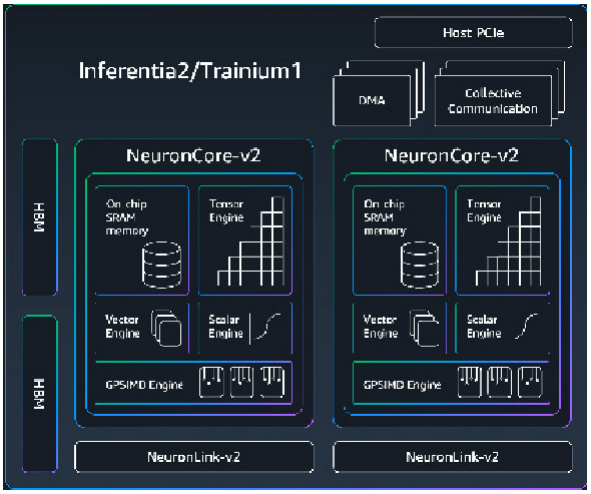
AWS Trainium2 delivers significant performance improvements, providing up to 1.5x faster training performance and 1.3x lower cost compared to similar EC2 instances. These performance gains

Figure 3. AWS Trainium2 Training performance

translate into substantial cost savings for organizations training large language models and other compute-intensive AI workloads (Figure 3).

Similarly, the **AWS Inferentia2/Trainium1** architecture enables low-latency, high-bandwidth communication (Figure 4a).

The Amazon EC2 Inf2.48xlarge instance supports up to 12 Inferentia2 chips, connected via low-latency, high-bandwidth NeuronLink-v2 chip-to-chip interconnect. This enables efficient collective communication operations like AllReduce and AllGather, optimizing model parallelism and minimizing communication overhead while sharding large models across AWS Inferentia2 devices for high-volume inference workloads (Figure 4b).



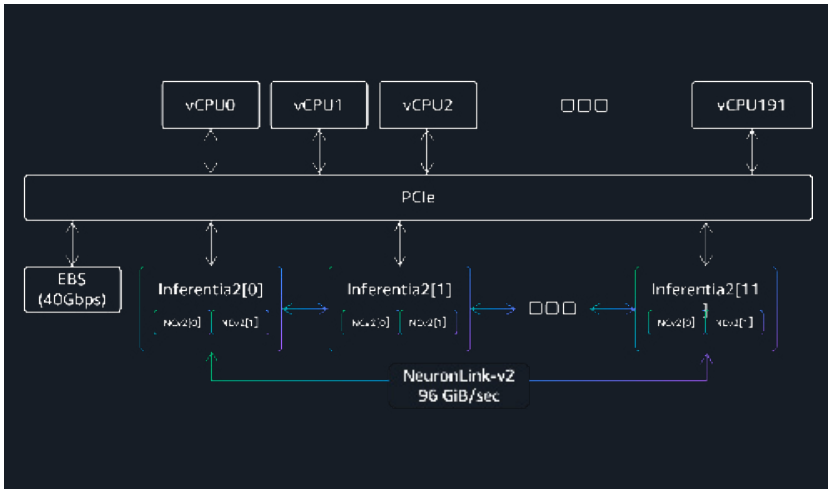


Figure 4a (left) AWS Inferentia2/Trainium1 Architecture, Figure 4b (right) Amazon EC2 inf2.48xlarge instance powered by Inferentia2/Trainium1 chips

Amazon EC2 Inf2 instances can deliver **up to 80 percent higher throughput-per-dollar** and **up to 74 percent lower latency** compared to similar EC2 instances. A performance case study with Llama-3.1-8B and Llama-3.1-70B on AWS Inferentia2.48xlarge instance showed that:

- Both 8B and 70B models demonstrate **predictable End-to-End (E2E) latency** across all input lengths (with 4096 sequence length). E2E latency represents the total time encompassing all steps in the LLM's operation: input processing, model inference, and output delivery, including the encoding and generation time (Figure 5a).

Figure 5a. AI Inference Serving: End-to-End Latency

- Larger batch sizes enable higher tensor parallelism, leading to increased throughput. The throughput and hence throughput-per-dollar (measured as **tokens-per-dollar**) remain predictable across all input lengths for both 8B and 70B models as shown in the tokens-per-dollar chart (Figure 5b).

Figure 5b. AI Inference Serving: Tokens-per-Dollar

AWS Neuron SDK optimization framework

The AWS Neuron SDK provides a comprehensive software stack for optimizing AI workloads on Trainium and Inferentia chips, integrating seamlessly with popular AWS services for distributed computing environments (Figure 6). Hugging Face's Optimum Neuron library offers a streamlined developer experience for using these chips with minimal code changes.

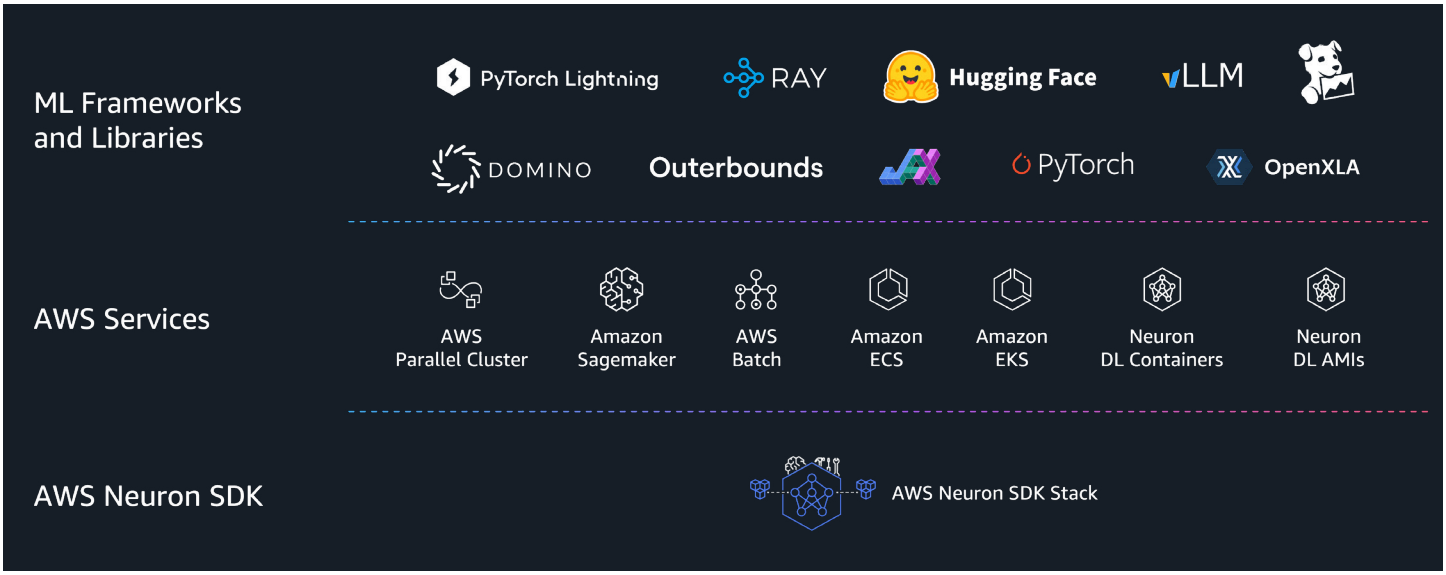


Figure 6. Enabling customers with AWS Neuron SDK

Hugging Face Optimum Neuron enables developers to easily deploy Hugging Face models on AWS Inferentia and Trainium with just a few lines of code. **For example, to run inference with a pre-trained model on Inferentia2:**

```python
from transformers import AutoTokenizer

from optimum.neuron import NeuronModelForCausalLM

model_id = “meta-llama/Meta-Llama-3-8B-Instruct” # Compile

model
tokenizer = AutoTokenizer.from_pretrained(model_id) compiler_args =
{“num_cores”: 8, “auto_cast_type”: ‘bf16’} input_shapes = {“batch_size”: 1,
“sequence_length”: 4096} model = NeuronModelForCausalLM.from_pretrained(
        model_id, export=True,
        **compiler_args,
        **input_shapes,
)

# Run inference 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”
)
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,
)
print(tokenizer.decode(outputs[0][input_ids.shape[-1]:]))
```

For distributed training on a trn1.32xlarge instance, Hugging Face Optimum Neuron integrates seamlessly with its Trainer API:

```python
from transformers import AutoModelForCausalLM, AutoTokenizer from peft
import LoraConfig
from optimum.neuron import NeuronSFTConfig, NeuronSFTTrainer

# Define the tensor_parallel_size tensor_parallel_size = 2

model_id = “meta-llama/Meta-Llama-3-8B”

tokenizer = AutoTokenizer.from_pretrained(model_id) tokenizer.pad_token =
tokenizer.eos_token

model = AutoModelForCausalLM.from_pretrained(model_id)

config = LoraConfig( r=16,
        lora_alpha=16,
        lora_dropout=0.05,
        target_modules=[
                “q_proj”,
                “gate_proj”,
                “v_proj”,
                “o_proj”,
                “k_proj”, “up_proj”,
                “down_proj”
        ],
        bias=”none”,
        task_type=”CAUSAL_LM”,
)

# training_args is an instance of NeuronTrainingArguments args =
training_args.to_dict()
sft_config = NeuronSFTConfig(
        max_seq_length=1024,
        packing=False,
        **args,
)

trainer = NeuronSFTTrainer(
        args=sft_config, model=model,
        peft_config=config,
        tokenizer=tokenizer,
        train_dataset=dataset,
        formatting_func=format_dataset,
)
```

To get started quickly with Optimum Neuron, developers can use the Hugging Face Neuron Deep Learning AMI, which comes pre-configured with all necessary dependencies. The AMI is available in AWS Marketplace.

Easy integration with Hugging Face Deep Learning Containers on AWS

Hugging Face Deep Learning Containers (DLCs) are Docker images pre-configured with deep learning frameworks like PyTorch and TensorFlow. They are optimized for AWS AI chips and include essential libraries like Transformers, Datasets, and Tokenizers; optimization tools including Hugging Face Optimum Neuron and AWS Neuron SDK components; and development tools for debugging and profiling.

Training DLCs are containers optimized for model training workflows on AWS Trainium, featuring distributed training capabilities and memory optimization. Inference DLCs are specialized containers for model deployment, including TGI containers optimized for Text Generation Inference on AWS Inferentia.

Amazon SageMaker AI integration

Hugging Face DLCs in Amazon SageMaker AI provide:

- **Pre-installed libraries:** Optimum Neuron and other ready-to-use optimization libraries
- **Integrated tools:** Complete development and deployment toolkit
- **Easy deployment:** Simple model deployment from Hugging Face Hub to SageMaker AI using simple API calls
- **Scalable infrastructure:** Automatic scaling based on demand

This integration enables users to easily train and deploy Transformers and Diffusers models in Amazon SageMaker AI while using AWS Trainium and Inferentia chips for optimal performance (Figure 7).

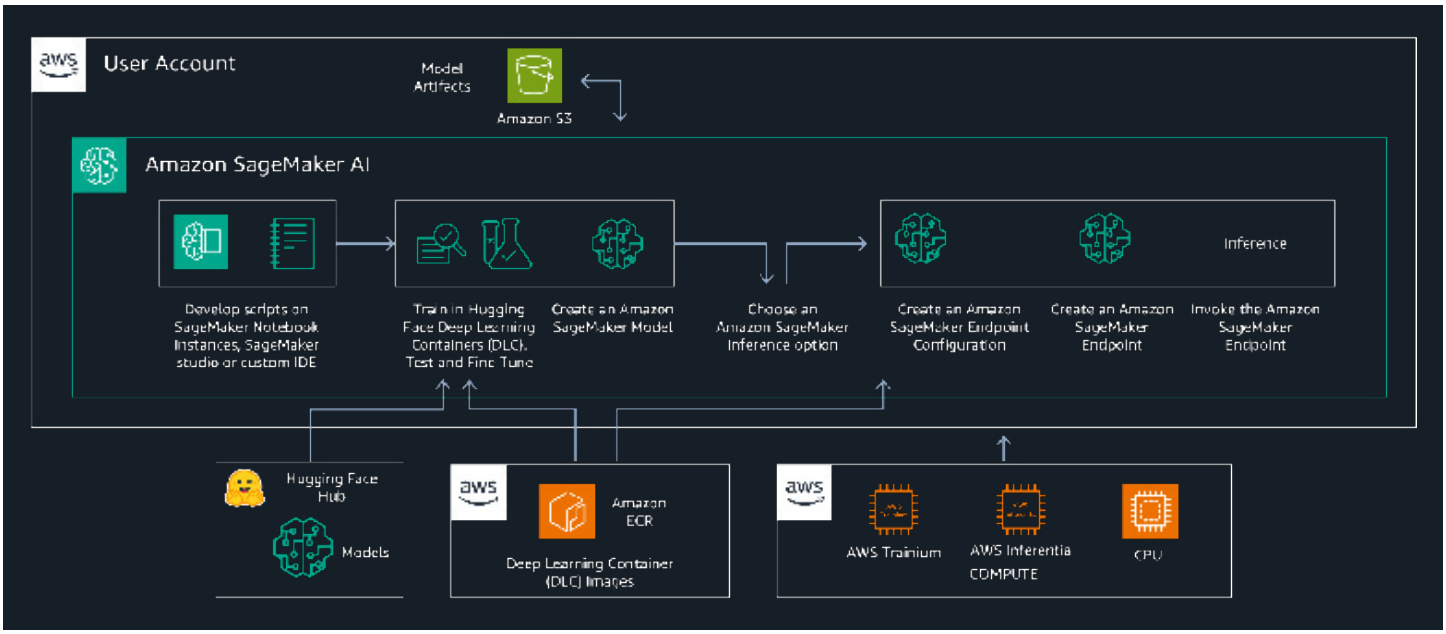


Figure 7. Hugging Face Model Deployment in Amazon SageMaker AI: A fully managed experience

Hugging Face Hub integration and compiled model caching

Hugging Face Optimum Neuron provides seamless integration with the Hugging Face Hub, enabling developers to efficiently share, discover, and deploy AWS Neuron-optimized models. Popular Hugging Face models are pre-compiled and available as cached compiled versions, reducing time to first token by 10x, when starting new deployments. This integration significantly enhances the model deployment workflow through compiled model caching and is supported by both Text Generation Inference (TGI) and vLLM with Optimum Neuron.

Hugging Face Hub integration

Developers can push and pull Neuron-compiled models directly to and from the Hugging Face Hub with minimal code changes:

```
```python

from optimum.neuron import NeuronModelForCausalLM # Load
and compile a model for Neuron
model = NeuronModelForCausalLM.from_pretrained( “meta-
llama/Meta-Llama-3-8B-Instruct”, export=True,
neuron_config={ “compiler_args”:{
            “batch_size”: 1,
            “sequence_length”: 4096,
            “num_cores”: 2,
            “auto_cast_type”: “bf16”
      }
}
)
# Push the compiled model to the Hub
model.push_to_hub(“my-username/Meta-Llama-3-8B-Instruct-Neuron”) # Later, pull
the pre-compiled model
compiled_model = NeuronModelForCausalLM.from_pretrained( “my-
username/Meta-Llama-3-8B-Instruct-Neuron”
)

```
```

This integration enables teams to collaborate effectively by sharing optimized models across their organization or with the broader community.

Compiled model caching benefits

One of the most significant advantages of the Hugging Face Hub integration is compiled model caching, which offers several key benefits:

1. **Elimination of compilation overhead:** Compiled model caching leads to a one-time compilation step, after which the compiled artifacts can be reused across deployments.
2. **Faster deployment times:** Pre-compiled models can be deployed in seconds, significantly reducing time-to-production and enabling more agile ML workflows.

```python
# Without caching – compilation is required each time model =
NeuronModelForCausalLM.from_pretrained(
“meta-llama/Meta-Llama-3-8B-Instruct”, export=True #
Triggers compilation
)

# With caching - instant loading of pre-compiled model model =
NeuronModelForCausalLM.from_pretrained(
“my-username/Meta-Llama-3-8B-Instruct-Neuron”
)
```

3. **Consistent performance:** Cached compiled models ensure consistent inference performance across different environments and deployments, eliminating variability that might arise from different compilation settings.
4. **Version control for optimized models:** The Hub enables proper versioning of compiled models, allowing teams to track changes, roll back to previous optimized versions if needed, and maintain a clear history of model optimizations.
5. **Reduced operational complexity:** By separating the model compilation and deployment phases, teams can optimize their workflows: data scientists can focus on model development while MLOps engineers handle deployment without the need to recompile.

For information about public cache system on Hugging Face Hub, refer to this guide.

Step 1: Check if a model is cached and if yes, which configurations.

```
HF_MODEL_ID = “mistralai/Mixtral-8x7B-Instruct-v0.1”

!optimum-cli neuron cache lookup $HF_MODEL_ID
```

Step 2: Pull the first configuration of the compiled model from the public cache.

The combination of Hugging Face Hub integration and compiled model caching significantly streamlines the deployment of optimized models on AWS Neuron hardware, reducing friction in the ML deployment pipeline and accelerating time-to-value for machine learning projects.

```
# without export=True
model = NeuronModelForCausalLM.from_pretrained( “meta-
llama/Meta-Llama-3-8B-Instruct”
)
```

Industry Use Cases

Financial services use case: Multimodal AI for insurance

Traditional AI pipelines in financial services face significant constraints: **text-only processing** with limited textual information and missing visual and audio data; **uni-modal analysis** with inability to process multiple information types simultaneously; and **processing bottlenecks** due to sequential processing of different data types.

A Multimodal Solution Architecture for **fast and intelligent document processing**, a predominant use case in Financial Services, utilizes Hugging Face multimodal models in Amazon SageMaker AI, powered by AWS AI chips. The solution enables:

- **Comprehensive data processing:** Simultaneous analysis of text, images, and structured data
- **Accelerated claims processing:** Rapid analysis of insurance documents, photos, and supporting materials
- **Enhanced accuracy:** Cross-modal validation and verification

The implementation (Figure 8) includes content processing, vector embeddings, Indexing and storing in an Opensearch Serverless vector store, followed by similarity search and contextual response to financial queries. The architecture utilizes Hugging Face Multimodal Foundation Models (MFMs) deployable from Hugging Face Hub to Amazon SageMaker AI, optimized for AWS AI chips. It also deploys multimodal embedding models from Hugging Face Hub to process information sources including images, audio and video, in addition to text. The solution requires AWS services like Textract, Transcribe, and Rekognition to handle multimodal content.

A future extension of the architecture utilizing AI Agents can deliver an autonomous, dynamic, and an adaptive AI pipeline for:

- **Intelligent document analysis:** Automated document classification and extraction
- **Fraud detection:** Cross-modal fraud pattern recognition
- **Autonomous decision making:** Automated claims approval for standard cases
- **Exception handling:** Intelligent escalation of complex cases

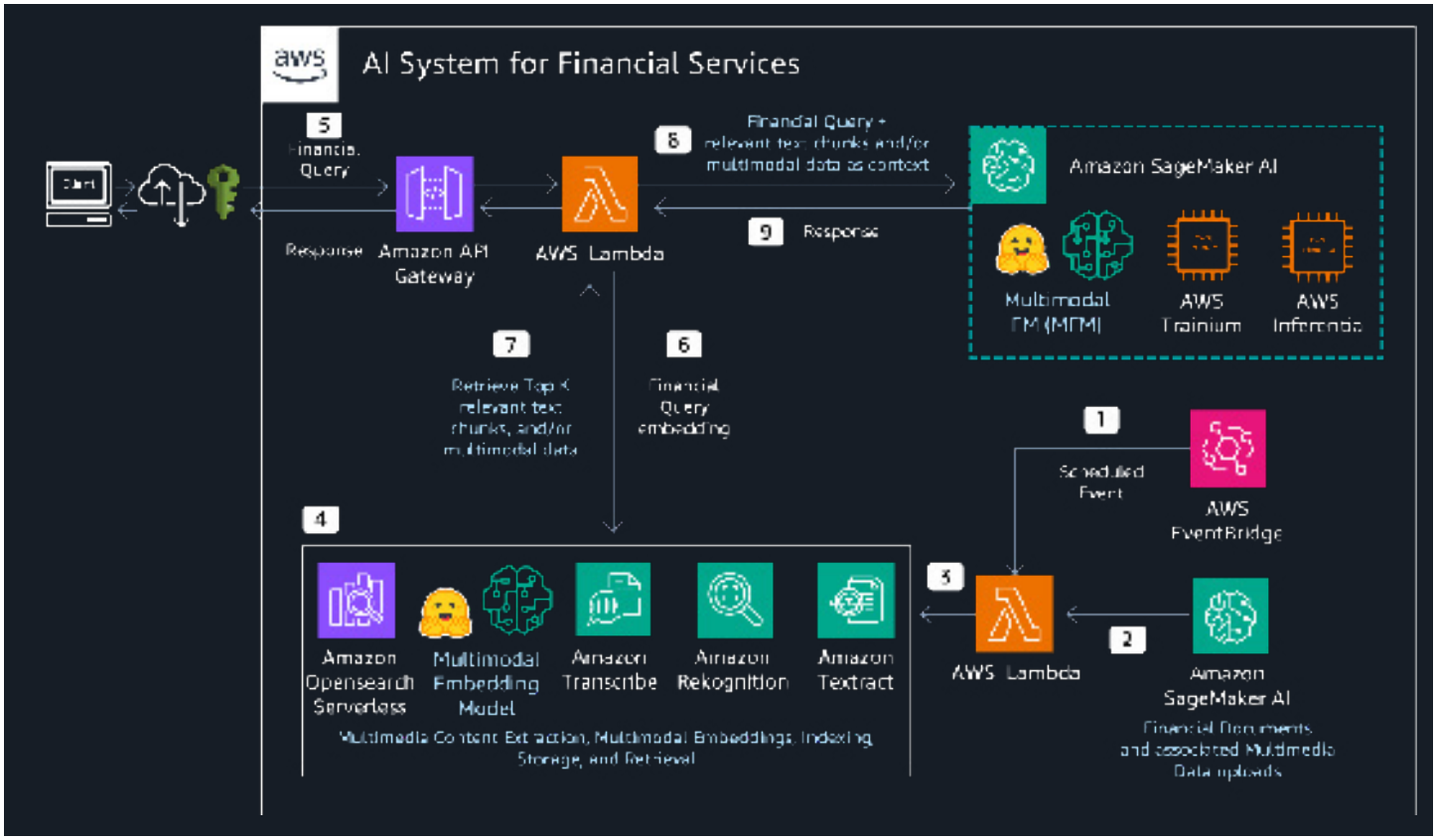


Figure 8. Multimodal Hugging Face models for intelligent document processing and media analysis for insurance claims

Natural Language Processing use case: Sentiment analysis

A Natural Language Processing (NLP) pipeline involves multi-class text classification and sentiment analysis using LLMs, prominent use cases for NLP, across multiple industries. Figure 9 shows how pre-trained Hugging Face models (like DistilBERT-base-uncased) can be customized and fine-tuned on SageMaker AI using emotion datasets and deployed for scalable, accurate sentiment analysis in production. This approach is valuable for understanding customer reactions to newly launched products across any industry.

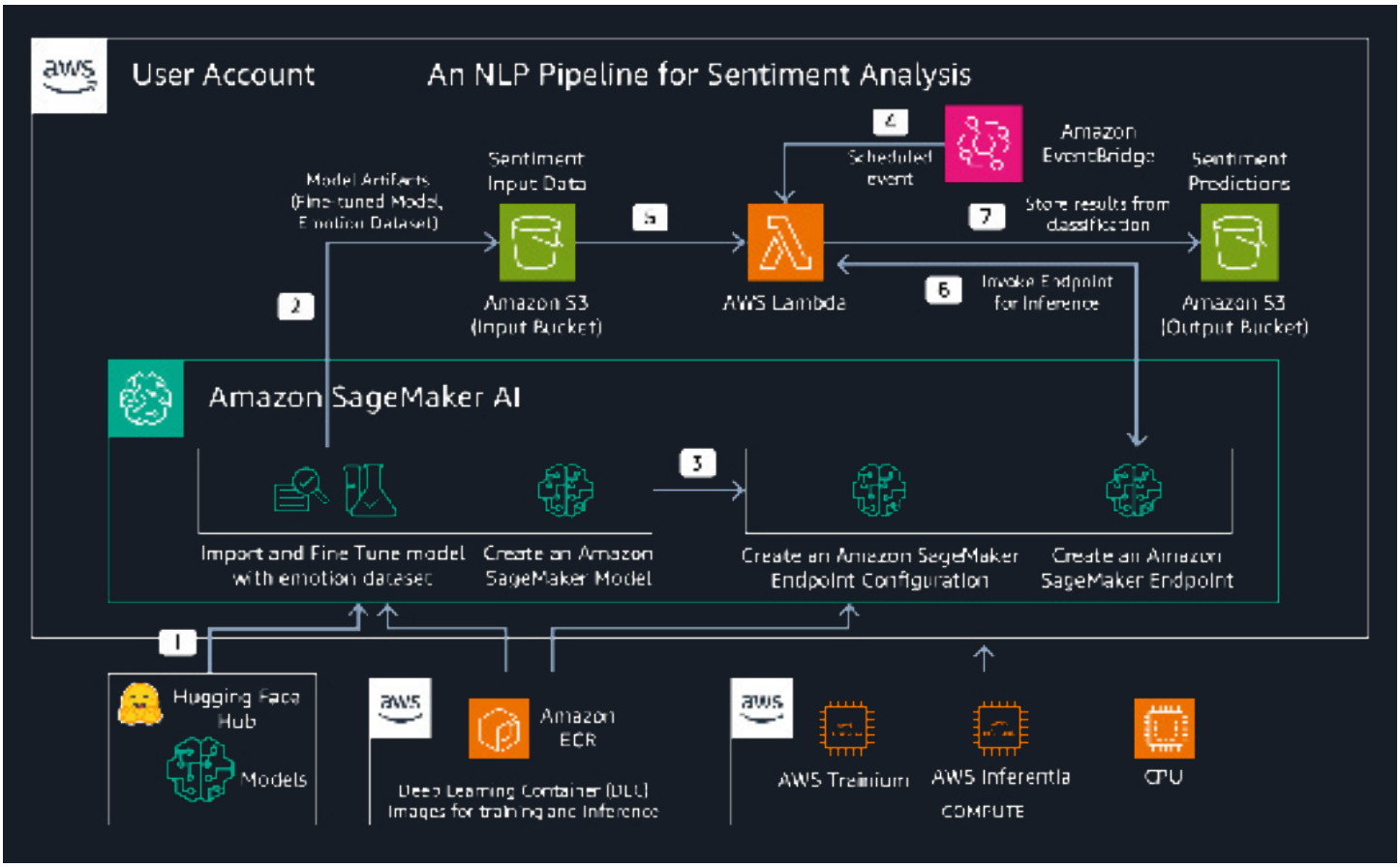


Figure 9. Fine-tuning Hugging Face Model for sentiment analysis

Summary

The Hugging Face-AWS Neuron integration represents a transformative approach to AI development and deployment, addressing fundamental challenges organizations face when implementing AI solutions at scale. Through the integration of Hugging Face's comprehensive model catalog and development tools with AWS purpose-built AI chips and optimized software stack, organizations can achieve unprecedented levels of choice, simplicity, customization, integration capabilities, and price-performance optimization.

Key Achievements include:

Technical excellence

The integration delivers robust technical capabilities through purpose-built hardware optimized for AI workloads, advanced distributed computing frameworks, and comprehensive software optimization. The integration of Hugging Face Optimum Neuron with AWS Trainium and Inferentia provides up to 1.5x performance improvements while reducing costs by 1.3x compared to traditional solutions.

Operational simplicity

Organizations can move from AI concept to production deployment through simplified workflows, pre-built containers, and intuitive API interfaces. The integration eliminates traditional barriers to AI adoption, enabling rapid prototyping and seamless scaling to production environments.

Economic value

The combination of optimized hardware and software delivers significant cost savings while maintaining high performance. Organizations achieve better price-performance ratios across training and inference workloads, with measurable improvements in total cost of ownership.

Industry impact

The integration serves diverse industry verticals, from financial services and healthcare to manufacturing and media, providing specialized solutions for complex use cases including multimodal AI, agentic workflows, and large-scale distributed computing.

As AI continues to evolve toward more sophisticated applications including multimodal understanding, agentic systems, and frontier model capabilities, the Hugging Face-AWS Neuron integration is well-positioned to support these advancing requirements. Continuous innovation in both hardware architecture and software optimization ensures that organizations can adapt to emerging AI paradigms while maintaining their competitive advantages.

The platform's comprehensive approach to AI development and deployment—combining choice, customization, integration, and price-performance optimization—establishes a foundation for sustained AI innovation and business value creation across industries and use cases.

Organizations adopting this platform position themselves at the forefront of AI innovation, with the tools, infrastructure, and support necessary to transform their operations and create new value propositions in an increasingly AI-driven economy.

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers, or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.