TECHNICAL OVERVIEW

# SIMPLIFY AI MODEL DEPLOYMENT IN PRODUCTION WITH NVIDIA TRITON INFERENCE SERVER

The idea of a system that can learn from data, identify patterns, and make decisions with minimal human intervention is exciting. AI machine learning (ML) and deep learning (DL) are becoming effective tools for solving diverse computing problems, from object classification to conversational AI to recommender systems. However, deploying AI models in applications and services can be challenging for infrastructure and operations teams. Factors like diverse frameworks, underutilized infrastructure, and lack of standardized implementations can even cause AI projects to fail. In this overview, we'll explore how to navigate these challenges and deploy AI models in production in the data center, in the cloud, or at the edge.

In many organizations, multiple teams are usually involved in AI development and deployment to production: data scientists, ML engineers, application developers, and IT operations. And while they work for the same organization, each has their own specific goals.

**Data scientists** seek to create the most accurate AI models to solve business problems, which can involve significant experimentation. They're best served by a production platform that supports all major DL and ML frameworks and networks, so they can choose the best framework and network for each use case. They also need to continuously update and retrain their models based on new data to improve accuracy.

**Machine learning engineers** train the models on large, real-life datasets and optimize and package them for production. They seek to implement MLOps and continuous integration and continuous deployment (CI/CD) processes, and they rely on MLOps platforms and tools.

**Application developers** integrate the trained models into the business applications they're developing. Their goal is to rapidly deliver the applications to solve identified business problems. They're also looking to optimize development costs.

**IT and operations managers** seek to run production applications aligned to business needs. To do this, they need to integrate all applications and models with their standard production infrastructure. Their key performance indicators are high throughput, scalability, application service-level agreements (SLAs), and utilization of servers.

## CHALLENGES

These disparate goals present challenges to deploying AI models at scale in production for several reasons

1. Existence of multiple DL and ML frameworks with each framework having its own model execution backend
2. Different types of inferences—real time, batch, audio streaming, and ensemble—that require serving optimizations

3. Need for mixed infrastructure, including CPU and GPU in the cloud, in the data center, and at the edge. Disparate serving solutions for each can increase cost
4. Underutilized GPU and CPU systems that can result in increased cost
5. Scaled deployment that needs load balancers, cluster orchestration, monitoring, and visualization tools. Lack of integration with standard tools drives costly and non-scalable implementation

## NVIDIA TRITON INFERENCE SERVER

Organizations can address these challenges with the NVIDIA Triton™ Inference Server, an open-source software that simplifies and streamlines AI models at scale in production.in production. It brings the three teams together by accelerating model development and deployment with flexibility.

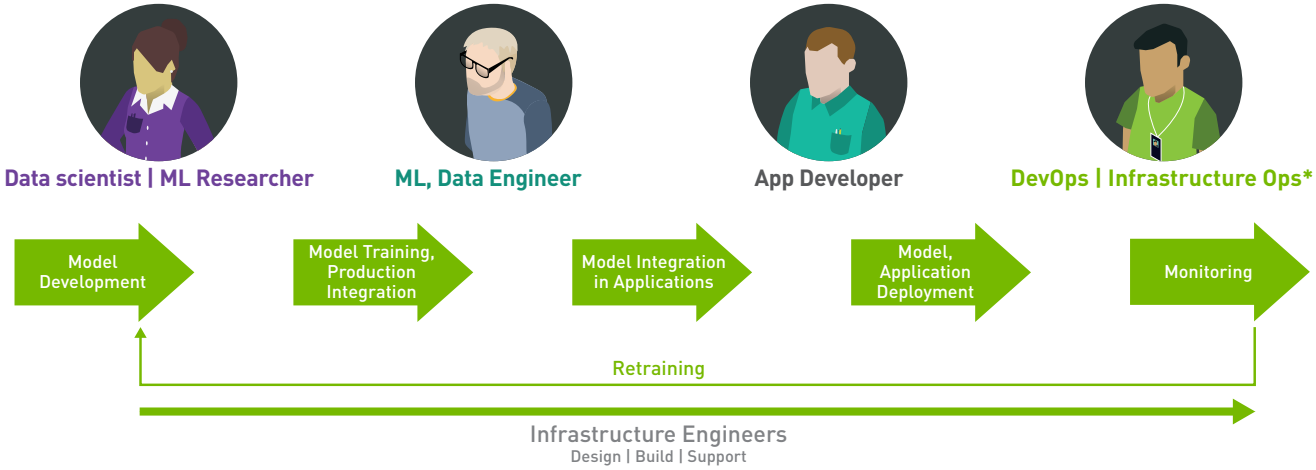


*Figure 1. Organizational AI workflow*

The Triton Inference Server lets teams deploy trained AI models from any framework (TensorFlow, NVIDIA® TensorRT™ Plan, PyTorch, ONNX Runtime, OpenVINO, RAPIDS™ Forest Inference Library (FIL), or a custom C++/Python framework) from local or public cloud storage on any GPU- or CPU-based infrastructure (cloud, data center, or edge).

Figure 2 shows how the Triton Inference Server fits in the overall data center AI inference architecture. This open-source software is available as a Docker container **image** or as code at **GitHub**.

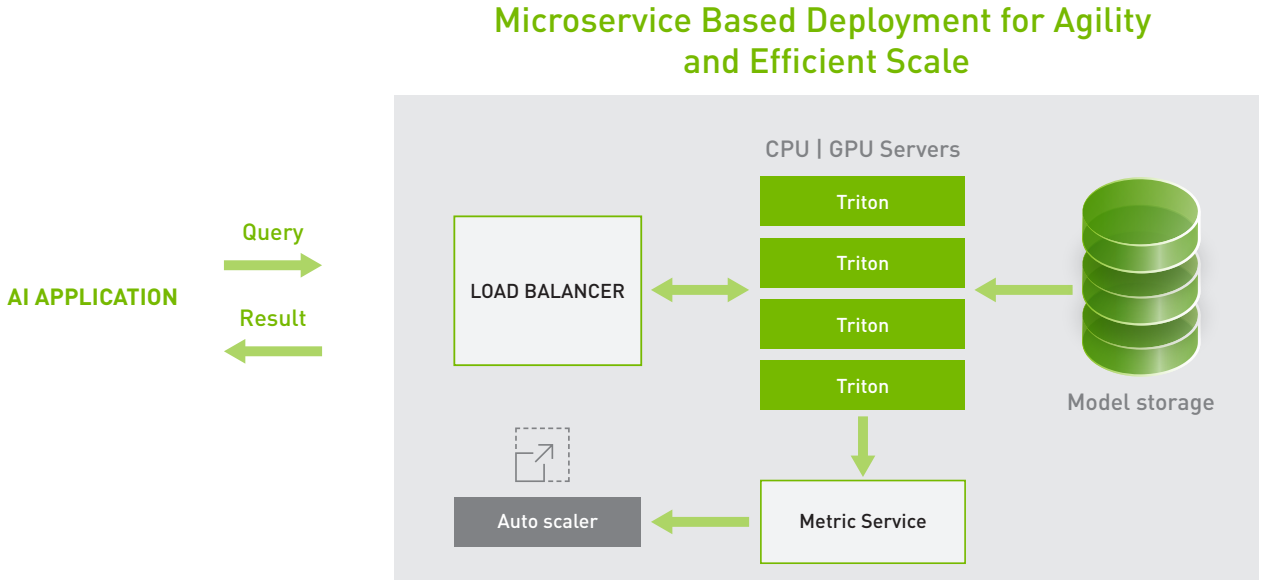


*Figure 2. High-level Triton Inference Server architecture*

Let's look at the benefits of NVIDIA Triton Inference Server.

1. **Designed to serve models from any framework:** NVIDIA Triton Inference Server supports all major frameworks, including TensorFlow, TensorRT, PyTorch, OpenVINO, RAPIDS FIL for XGBoost, LightGBM and ONNX Runtime (MXNet), and even custom C++/Python frameworks. Models from these frameworks can be deployed without any conversion on GPU or CPU servers. Data scientists have the freedom to choose their preferred framework and network, and both developers and IT operators can support these trained networks without incurring unneeded complexities. In addition, developers can easily add and update models live in production with Triton. As a result, developers can rapidly deliver their AI-integrated applications, and IT/DevOps can deploy retrained models without bringing down applications. and network, and both developers and IT operators can support these trained networks without incurring unneeded complexities.
2. **Designed for production IT/DevOps/MLOps infrastructure:** Available as a Docker container, Triton integrates with Kubernetes, the container management platform for orchestration, metrics, and autoscaling. It also integrates with Kubeflow, KFServing, and public cloud-managed Kubernetes for an end-to-end AI workflow. It exports Prometheus metrics for monitoring and supports the standard HTTP/gRPC interface to connect with other applications like load balancers. Triton Inference Server is also integrated in MLOps platforms like Azure ML, AWS SageMaker, Google Vertex AI, Seldon, and ClearML. All these integrations help IT deploy a streamlined inference-in-production platform with lower complexity, higher visibility into resource utilization, and scalability.

3. **Designed to scale and maximize GPU and CPU utilization:** Triton Inference Server has the ability to run multiple models concurrently and dynamically batch input requests for optimal GPU and CPU utilization. It can utilize all available GPUs on a server and it can easily scale to any number of servers to handle increasing inference loads, eliminating inefficient single-model-per-GPU deployments.
4. **Designed for different inference query types:** There are different types of inference queries—real time, batch, streaming, and ensemble. Real-time inference needs low latency, as end users are waiting for the results while the inference happens. Batch inference is generally done offline and can trade latency for high throughput. Audio streaming requires the state information to be maintained for the different inputs. Ensemble is a pipeline of pre-processing, model, and post-processing algorithms. The Triton Inference Server optimizes the model execution for all these types of queries. It supports batch inference to increase utilization and, at the same time, allows latency limits for real-time inference, handles stateful audio streaming, and allows an ensemble of models to be executed on CPU or GPU.
5. **Designed for any platform:** The Triton Inference Server can be used to deploy models on CPU-only servers or GPU servers in the public cloud, in the data center, at the edge, and in embedded devices (e.g., NVIDIA Jetson™). The above benefits of Triton Inference Server are available on CPUs too. Migration from CPU inferencing to GPU is easy with a simple change to the model configuration file. Supporting a heterogeneous cluster with both GPUs and CPUs helps standardize inference across platforms and helps dynamically scale out to any CPU or GPU to handle peak loads. Triton is also integrated with many other NVIDIA application frameworks, such as NVIDIA Clara™, DeepStream, Riva, Merlin, and Morpheus.
6. **Designed for high-performance inference:** Triton packs in many serving optimizations like concurrent execution, dynamic batching, multi-GPU support, ragged input batching for streaming inputs, and advanced scheduling that help deliver high-performance inference. As the figure 3 shows, Triton delivers nearly equal or identical performance as the highly optimized bare-metal execution in MLPerf 1.0 Inference benchmarks.

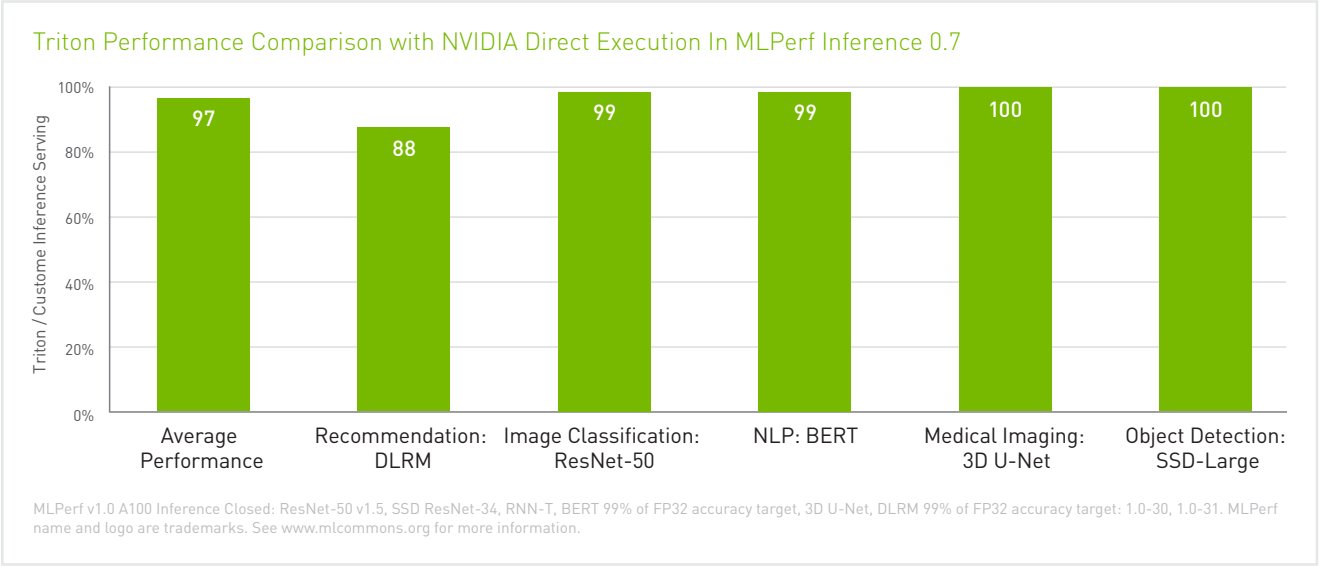


*Figure 3. Triton versus custom bare-metal execution performance in MLPerf Inference 1.0*

7. **Delivers automatic model conversion and analysis:** AA trained model is generally not optimized for deployment in production. ML engineers must go through a series of conversions, optimizations, and validations for their specific target environment. This process can take significant time and effort.

   Triton Model Navigator can be used to automate the process. Available as an alpha version, it can convert an input model from any framework (TensorFlow and PyTorch are currently supported) to TensorRT, validate the conversion for correctness, automatically find and create the most optimal model configuration, and generate the repository structure for the model deployment. What could take days for each type of model can now be done in hours.

8. **Optimizes model performance:** For efficient inference serving, optimal model configurations such as the batch size and number of concurrent models are important. Today, there's no easy way; ML engineers need to manually try different combinations to find the optimal configuration for throughput, latency, and memory utilization.

   Triton Model Analyzer is another optimization tool that can automatically find the best model configuration to get the highest performance. ML engineers can specify performance requirements, such as a latency constraint, throughput target, or memory footprint. The model analyzer searches through different model configurations and finds the one that provides the best performance under those constraints. To help visualize the performance of the top configurations, it then outputs a summary report, which includes charts (see figure 4).

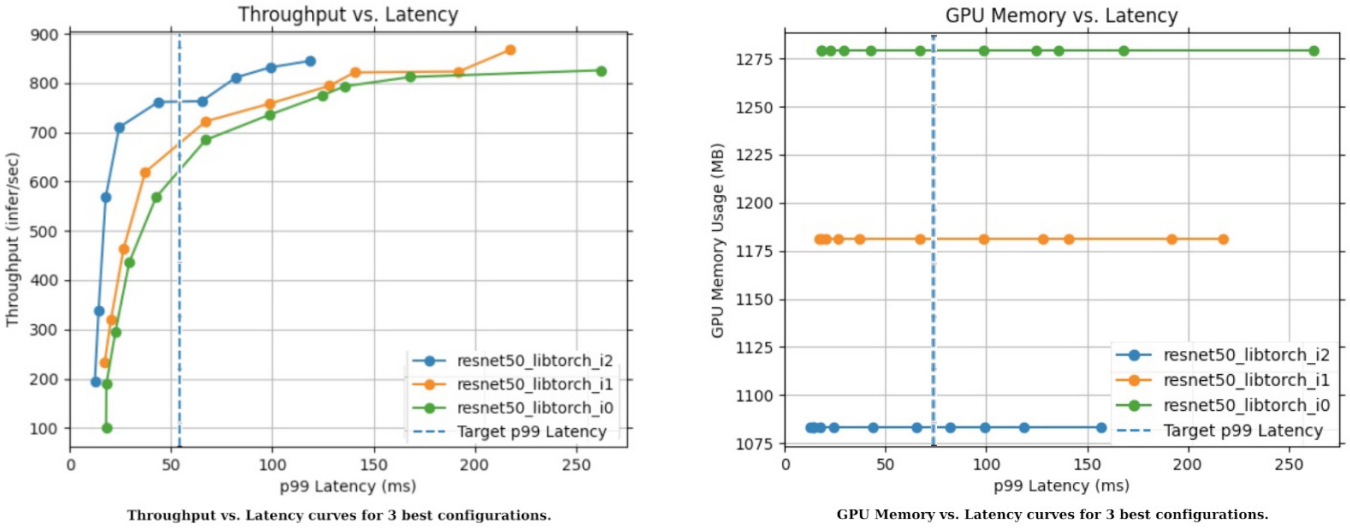


*Figure 4. Choosing an optimal configuration with Triton Model Analyzer*

There's no need to settle for a less optimized inference service because of the inherent complexity in getting to an optimized model. NVIDIA Triton easily delivers the most efficient inference deployment.

## GETTING STARTED WITH THE TRITON INFERENCE SERVER

Getting started is easy. Here are the steps IT operators and developers can take to deploy and use the Triton Inference Server.

**IT, DevOps, ML Engineer**

1. Pull the Triton Inference Server container from the **NVIDIA NGC™ catalog**. Source code is also available at **GitHub**.
2. Set up a model repository, which is a directory with information about the models to be deployed and the models themselves.
3. Run the Triton Inference Server container on GPU or CPU-only servers.
4. Metrics can be used for monitoring and autoscaling. Alternatively, Kubernetes can be used to deploy the inference server container.

**Developer, ML Engineer**

1. Integrate the Triton Inference Server client library into the business application.
2. Optionally, add or update the model configuration in the model repository.
3. Finally, deploy the application.

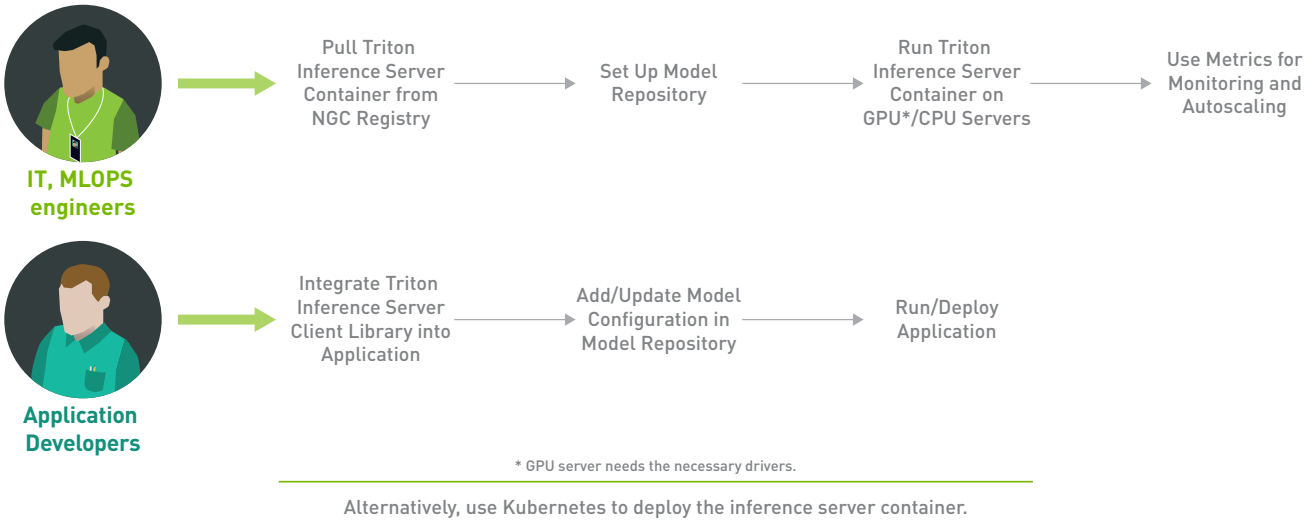


*Figure 6. Steps to deploy the Triton Inference Server*

## SIMPLIFY AI DEPLOYMENT

**The NVIDIA Triton Inference Server** simplifies deployment of AI models at scale in production. It supports all major frameworks, runs multiple models concurrently to increase throughput and utilization, and integrates with DevOps and MLOps tools for a streamlined production that's easy to set up. These capabilities combine to bring data scientists, ML engineers, developers, and IT/DevOps together to accelerate AI development and deployment to production.

Try it today, either on GPU or CPU. The Triton Inference Server can be downloaded **here**, and its source code can be found **here**.



NVIDIA