Introduction to Ray Serve

Ray Serve allows for efficient multi-model serving by combining the benefits of microservices and monoliths.

Model Composition

* Efficient Hardware resource sharing between models.
* Independent scaling of models, allowing for flexible resource allocation.
* Simplification of application testing and monitoring.
* Example: Combining image pre-processing, classification, and detection models.

Multi-Application

* Supports multiple independent applications on a single cluster.
* Facilitates collaboration between teams and ensures independent upgrade cycles.
* Example: Managing autonomous driving algorithms for different scenarios and environmental conditions.

Multiplexing

* Addresses the challenge of serving a large number of models with limited Hardware resources.
* Dynamically manages model loading and routing traffic to specific replicas with cached models.
* Improves model cache hit rate, reducing latency and boosting throughput.
* Example: Support for numerous language models in an inference platform.

Case Studies

* **Samsara:** Reduced ML infrastructure costs by 50% using Ray Serve model composition.

* **Unscale Endpoints:** Boosted throughput by 30% using Ray Serve multiplexing.

* **Clary:** Enhanced model training speed by 80% and serving latency by 80% using Ray Serve.