RedisAI

RedisAI is a Redis module for executing Deep Learning/Machine Learning models and managing their data. Its purpose is being a “workhorse” for model serving, by providing out-of-the-box support for popular DL/ML frameworks and unparalleled performance. RedisAI both simplifies the deployment and serving of graphs by leveraging on Redis’ production-proven infrastructure, as well as maximizes computation throughput by adhering to the principle of data locality.

This introduction is intended to present the core concepts it uses and the functionality it provides.

In broad strokes, RedisAI looks like this:

How RedisAI works

RedisAI bundles together best-of-breed technologies for delivering stable and performant computation graph serving. Every DL/ML framework ships with a runtime for executing the models developed with it, and the common practice for serving these is building a simple server around them.
RedisAI aims to be that server, saving you from the need of installing the backend you’re using and developing a server for it. By itself that does not justify RedisAI’s existence so there’s more to it. Because RedisAI is implemented as a Redis module it automatically benefits from the server’s capabilities: be it Redis’ native data types, its robust eco-system of clients, high-availability, persistence, clustering, and Enterprise support.

Because Redis is an in-memory data structure server RedisAI uses it for storing all of its data. The main data type supported by RedisAI is the Tensor that is the standard representation of data in the DL/ML domain. Because tensors are stored in the memory space of the Redis server, they are readily accessible to any of RedisAI’s backend libraries at minimal latency.

The locality of data, which is tensor data in adjacency to DL/ML models backends, allows RedisAI to provide optimal performance when serving models. It also makes it a perfect choice for deploying DL/ML models in production and allowing them to be used by any application.

Furthermore, RedisAI is also an optimal testbed for models as it allows the parallel execution of multiple computation graphs and, in future versions, assessing their respective performance in real-time.

**Data structures**

RedisAI provides the following data structures:

- **Tensor**: Represents an n-dimensional array of values
- **Model**: Represents a computation graph by one of the supported DL/ML framework backends
- **Script**: Represents a **TorchScript** program

**DL/ML backends**

RedisAI supports DL/ML identifiers and their respective backend libraries, including:

- **TF**: The TensorFlow backend
- **TFLITE**: The TensorFlow Lite backend
- **TORCH**: The PyTorch backend
- **ONNX**: ONNXRuntime backend

A complete list of supported backends is in the release notes for each version.

Backend libraries are dynamically loaded as needed, but can also be loaded during booting or at runtime. Refer to these pages for more information on loading backends:

- AI.CONFIG command
- Backend configuration

**More info**

- RedisAI commands
- RedisAI source

*Updated: April 26, 2022*