1. What does a SavedModel contain? How do you inspect its content?

ANS:

A SavedModel contains a complete TensorFlow program, including trained parameters (i.e, tf. Variable s) and computation. It does not require the original model building code to run, which makes it useful for sharing or deploying with TFLite, TensorFlow. js, TensorFlow Serving, or TensorFlow Hub.

1. When should you use TF Serving? What are its main features? What are some tools you can use to deploy it?

ANS:

TensorFlow Serving also includes features such as model versioning, automatic batching of requests, and support for canary deployments, which make it easier to deploy and manage machine learning models in a production environment.

“TensorFlow Serving is a flexible, high-performance serving system for machine learning models, designed for production environments. TensorFlow Serving makes it easy to deploy new algorithms and experiments while keeping the same server architecture and APIs.

1. How do you deploy a model across multiple TF Serving instances?

ANS:

Create your model

1. Import the Fashion MNIST dataset.
2. Train and evaluate your model.
3. Add TensorFlow Serving distribution URI as a package source:
4. Install TensorFlow Serving.
5. Start running TensorFlow Serving.
6. Make REST requests.

How was it done?

1. Step 1: Create a new virtual environment using Pycharm IDE.
2. Step 2: Install necessary libraries.
3. Step 3: Build the best machine learning model and Save it.
4. Step 4: Test the loaded model.
5. Step 5: Create main.py file.
6. Step 6: Upload local project to Github.
7. When should you use the gRPC API rather than the REST API to query a model served by TF Serving?

ANS:

1. High-performance communication and low latency are crucial.
2. You need to support real-time streaming between client and server.
3. You prefer a more function-driven API design.
4. Your system relies heavily on microservices.

REST does not have built-in code generation. Instead, you must use a third-party service to produce the code for an API request. On the server side, API requests must be converted to the server's programming language. gRPC has built-in code generation and uses a compiler to convert programming languages

1. What are the different ways TFLite reduces a model’s size to make it run on a mobile or embedded device?

ANS:

Quantization can reduce the size of a model in all of these cases, potentially at the expense of some accuracy. Pruning and clustering can reduce the size of a model for download by making it more easily compressible.

One of the most widely used methods for compressing models, quantisation, involves decreasing the size of the weights to improve efficiency. The smaller representations of the model weights by reducing them into smaller sizes reduces the size of the model along with increasing the speed of its processing and inference

1. What is quantization-aware training, and why would you need it?

ANS:

Quantization aware training emulates inference-time quantization, creating a model that downstream tools will use to produce actually quantized models. The quantized models use lower-precision (e.g. 8-bit instead of 32-bit float), leading to benefits during deployment.

1. What are model parallelism and data parallelism? Why is the latter generally recommended?

ANS:

Data parallelism vs. model parallelism

|  |  |
| --- | --- |
| Data parallelism | Model parallelism |
| Same model is used for every thread but the data given to each of them is divided and shared. | Same data is used for every thread, and model is split among threads. |

In model parallelism, every model is partitioned into 'N' parts, just like data parallelism, where 'N' is the number of GPUs. Each model is then placed on an individual GPU. The batch of GPUs is then calculated sequentially in this manner, starting with GPU#0, GPU#1 and continuing until GPU#N.

1. When training a model across multiple servers, what distribution strategies can you use? How do you choose which one to use?

ANS:

distribute. Strategy is a TensorFlow API to distribute training across multiple GPUs, multiple machines, or TPUs. Using this API, you can distribute your existing models and training code with minimal code changes

There are two types of communication approaches. This applies to both the data-parallel and model parallel methods.