TensorFlow on Google Cloud

Course Summary and
Key Takeaways

Learning Objectives

- Create TensorFlow 2.x and Keras machine learning models

- Describe Tensorflow 2.x key component

- Use the tf.data library to manipulate data and large datasets

- Use the Keras Sequential and Functional APIs for simple and advanced model creation

- Train, deploy, and productionalize ML models at scale with Vertex AI

Module Breakdown

- Module 0: Introduction to the Course

- Module 1: Introduction to the TensorFlow ecosystem

- Module 2: Design and Build an Input Data Pipeline

- Module 3: Building Neural Networks with the TensorFlow and Keras API

- Module 4: Training at Scale with Vertex AI

Summary

Enterprises that have and want to utilize big data need a machine learning solution.

TensorFlow is commonly used for deep Learning, classification and predictions, image recognition, and transfer learning. So, its portability to multiple platforms and devices and production readiness can solve complex business and academic research problems.

Used in production environments with Keras, TensorFlow gives you the ultimate machine learning solution.

Key takeaways

Module 1: Introduction to the TensorFlow ecosystem

TensorFlow is an open-source, high-performance library for numerical computation that uses directed graphs (DAGs).

DAGs are used in models to illustrate the flow of information through a system and is simply a graph that flows in one direction, with nodes.

The nodes represent mathematical operations. Connecting the nodes are the edges which are the input and output of the mathematical operations. These are arrows that point in one direction. These arrays of data, or **tensors**, flow towards the output.
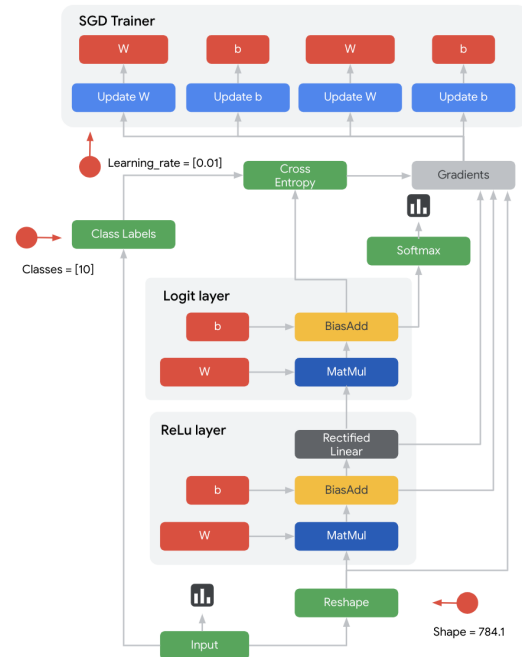
A tensor is simply an array of data, with its dimensioning determining its **rank**. So, your data in TensorFlow are tensors. They flow through the graph. Hence the name TensorFlow.

Rank 0 Tensor	Rank 1 Tensor	Rank 2 Tensor	Rank 3 Tensor	Rank 4 Tensor
scalar	vector	matrix		

TensorFlow uses DAGS to represent computation because of **portability**. TensorFlow applications can be run on almost any platform: local machines, cloud clusters, iOS and Android devices, CPUs, GPU, or TPUs.

TensorFlow contains multiple abstraction layers, **tf.estimator**, **tf.keras**, and **tf.data**. It's possible to run TF at scale with **Vertex AI Platform**.

tf.estimator, tf.keras, tf.data	High-level APIs for distributed training	
tf.losses, tf.metrics, tf.optimizers, etc.	Components useful when building custom NN models	Vertex AI
Core TensorFlow (Python)	Python API gives you full control	
Core TensorFlow (C++)	C++ API is quite low level	
CPU GPU TPU Android	TF runs on different hardware	

	Common name	Rank (Dimension)	Example	Shape of example
■	Scalar	0	`x = tf.constant(3)`	()
▮	Vector	1	`x = tf.constant([3, 5, 7])`	(3,)
▬	Matrix	2	`x = tf.constant([[3, 5, 7],` `              [4, 6, 8]])`	(2, 3)

◰	3D Tensor	3	`tf.constant([[[3, 5, 7],[4, 6, 8]],`
			`             [[1, 2, 3],[4, 5, 6]]`
			`            ])`
				(2, 2, 3)

▤	nD Tensor	n	`x1 = tf.constant([2, 3, 4])`
			`x2 = tf.stack([x1, x1])`
			`x3 = tf.stack([x2, x2, x2, x2])`
			`x4 = tf.stack([x3, x3])`
			`...`
				(3,)
				(2, 3)
				(4, 2, 3)
				(2, 4, 2, 3)

Module 2: Design and build a TensorFlow Input Data Pipeline

Models which are deployed in production require lots and lots of data. This is data that likely won't fit in memory and can possibly be spread across multiple files or may be coming from an input pipeline.

The **tf.data API** enables you to build complex input pipelines from simple, reusable pieces. The tf.data API makes it possible to handle large amounts of data, read from different data formats, and perform complex transformations.

The Dataset API will help you create input functions for your model that will load data *progressively*.

There are specialized Dataset classes that can read data from text files like CSVs, TensorFlow records, or fixed length record files.

Datasets can be created from different file formats:

- Use **TextLineDataset** to instantiate a Dataset object which comprises lines from one or more text files.
- **TFRecordDataset** comprises records from one or more TFRecord files.
- **FixedLengthRecordDataset** is a dataset object from fixed-length records from one or more binary files.
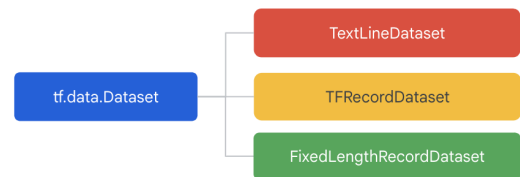
For anything else, you can use the generic Dataset class and add your own decoding code.

Module 3: Building Neural Networks with the TensorFlow and Keras API

As the number of categories of a feature grows large, it becomes infeasible to train a neural network using one-hot encodings.

We can use an **embedding column** to overcome this limitation. Instead of representing the data as a one-hot vector of many dimensions, an embedding column represents that data as a lower-dimensional, dense vector in which each cell can contain *any* number, not just 0 or 1.

Keras preprocessing layers allow you to build and export models that are truly end-to-end: models that accept raw images or raw

structured data as input and models that handle feature normalization or feature value indexing on their own.

Deep dive into model building using the **Keras Sequential**, **Functional**, and **Model subclassing APIs**:

Sequential API	Functional API	Functional API	Subclassing
+ Built-in layers	+ Built-in layers	+ Custom layers + Custom metrics + Custom losses	Write everything yourself from scratch
— 01 →	— 02 →	— 03 →	— 04 →
New users, simple models	Engineers with standard use cases	Engineers requiring increasing control	Researchers

Progressive disclosure of complexity

Our goal while training a model is to minimize the loss value.

Loss on Test Data

Loss on Training Data

Training iterations
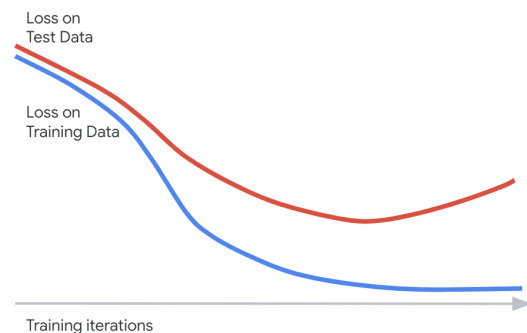
```
Early Stopping
Parameter Norm Penalties
        L1 regularization
        L2 regularization
        Max-norm regularization
Dataset Augmentation
Noise Robustness
Sparse Representations
. . .
```

L1 and L2 regularization

Regularization is one of the major fields of research within machine learning and refers to any technique that helps generalize a model. A generalized model performs well not just on training data, but also on never-seen test data.

Module 4: Training at Scale with Vertex AI

You can train TensorFlow models at scale using Vertex AI. You'll first need to use TensorFlow to create your Keras model, package your trainer application, and configure and start a Vertex AI training job.