BDA 6-2

1

Hello everyone, I am Haiying Che, from Institute of Data Science and knowledge Engineering

School of Computer Science, in Beijing Institute of Technology, in this session we will talk about the powerful Ml tool TensorFlow.

2

In this chapter, we introduce some useful platform， Spark MLlib and TensorFlow.

In this session we will discuss about the concepts and mechanism of TensorFlow.

3

TensorFlow is Created by the Google Brain team, it is an open source library for numerical computation and large-scale machine learning.

TensorFlow bundles together a lot of machine learning and deep learning (aka neural networking) models and algorithms and makes them useful by way of a common metaphor.

TensorFlow has grown to become one of the most loved and widely adopted ML platforms in the world

A tensor is the core data unit of TensorFlow, which is essentially an array of arbitrary dimensions.

Available tensor types include constants, variables, tensor placeholders, and sparse tensors.

**How TensorFlow works**

TensorFlow allows developers to create dataflow graphs—structures that describe how data moves through a graph, or a series of processing nodes.

Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.

TensorFlow provides all of this for the programmer by way of the Python language.

Python is easy to learn and work with, and provides convenient ways to express how high-level abstractions can be coupled together.

Nodes and tensors in TensorFlow are Python objects, and TensorFlow applications are themselves Python applications.

It uses Python to provide a convenient front-end API for building applications with the framework, while executing those applications in high-performance C++.

The actual math operations, however, are not performed in Python.

The libraries of transformations that are available through TensorFlow are written as high-performance C++ binaries.

Python just directs traffic between the pieces, and provides high-level programming abstractions to hook them together.

TensorFlow applications can be run on most any target that’s convenient: a local machine, a cluster in the cloud, iOS and Android devices, CPUs or GPUs.

If you use Google’s own cloud, you can run TensorFlow on Google’s custom TensorFlow Processing Unit (TPU) silicon for further acceleration.

The resulting models created by TensorFlow, though, can be deployed on most any device where they will be used to serve predictions.

4

TensorFlow has grown to become one of the most loved and widely adopted ML platforms in the world

Let ‘s watch a video to understand more about TensorFlow

5

Now let’s look at the structure of TensorFlow 2.0.

TensorFlow 2.0 focus on simplicity and ease of use, featuring updates like:  
Easy model building with Keras and eager execution.  
Robust model deployment in production on any platform.  
Powerful experimentation for research.

the APIs has been packaged together into a comprehensive platform that supports machine learning workflows from training through deployment.

6

[Keras](https://www.tensorflow.org/guide/keras), a user-friendly API standard for machine learning, will be the central high-level API used to build and train models.

The Keras API makes it easy to get started with TensorFlow.

Importantly, Keras provides several model-building APIs (Sequential, Functional, and Subclassing), so you can choose the right level of abstraction for your project.

TensorFlow2.0’s implementation contains enhancements including

1） eager execution, for immediate iteration

2） Intuitive debugging,

3） Tf.data, for building scalable input pipelines.

7

Let’s take a look at the new architecture of TensorFlow 2.0 using a simplified, conceptual diagram as shown

**Easy model building-**Here’s an example workflow:

* Load your data using **[tf.data](https://www.tensorflow.org/guide/datasets)**.
* Training data is read using input pipelines which are created using **tf.data**.
* Feature characteristics, for example bucketing and feature crosses are described using tf.feature\_column.
* Convenient input from in-memory data (for example, NumPy) is also supported.

8

Build, train and validate your model with [tf.keras](https://www.tensorflow.org/guide/keras), or use [Premade Estimators](https://www.tensorflow.org/guide/premade_estimators).

Keras integrates tightly with the rest of TensorFlow so you can access TensorFlow’s features whenever you want.

A set of standard packaged models (for example, linear or logistic regression, gradient boosted trees, random forests) are also available to use directly (implemented using the tf.estimator API).

If you’re not looking to train a model from scratch, you’ll soon be able to use transfer learning to train a Keras or Estimator model using modules from [TensorFlow Hub](https://www.tensorflow.org/hub/).  
Run and debug with [eager execution](https://www.tensorflow.org/guide/eager), then use [tf.function](https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/r2/guide/autograph.ipynb) for the benefits of graphs.

TensorFlow 2.0 runs with eager execution by default for ease of use and smooth debugging.

Additionally, the tf.function annotation transparently translates your Python programs into TensorFlow graphs.

This process retains all the advantages of 1.x TensorFlow graph-based execution: Performance optimizations,

remote execution and the ability to serialize, export and deploy easily, while adding the flexibility and ease of use of expressing programs in simple Python.

9

Use Distribution Strategies for distributed training.

For large ML training tasks, the [Distribution Strategy API](https://www.tensorflow.org/guide/distribute_strategy) makes it easy to distribute and train models on different hardware configurations without changing the model definition.

Since TensorFlow provides support for a range of hardware accelerators like CPUs, GPUs, and TPUs, （Tensor Processing Unit）

you can enable training workloads to be distributed to single-node/multi-accelerator as well as multi-node/multi-accelerator configurations, including [TPU Pods](https://cloud.google.com/blog/products/ai-machine-learning/now-you-can-train-ml-models-faster-and-lower-cost-cloud-tpu-pods).

Although this API supports a variety of cluster configurations, [templates](https://github.com/tensorflow/ecosystem/tree/master/distribution_strategy) to deploy training on [Kubernetes clusters](https://kubernetes.io/) in on-prem or cloud environments are provided.

10

Export to **SavedModel.**

TensorFlow will standardize on SavedModel as an interchange format for TensorFlow Serving, TensorFlow Lite, TensorFlow.js, TensorFlow Hub, and more.

11

**Robust model deployment in production on any platform**

TensorFlow has always provided a direct path to production. Whether it’s on servers, edge devices, or the web,

TensorFlow lets you train and deploy your model easily, no matter what language or platform you use.

In TensorFlow 2.0, compatibility and parity across platforms and components are improved by standardizing exchange formats and aligning APIs.

Once you’ve trained and saved your model, you can execute it directly in your application or serve it using one of the deployment libraries:

[TensorFlow Serving](https://www.tensorflow.org/tfx/serving/): A TensorFlow library allowing models to be served over HTTP/REST or gRPC/Protocol Buffers.  
[TensorFlow Lite](https://www.tensorflow.org/mobile/tflite/): TensorFlow’s lightweight solution for mobile and embedded devices provides the capability to deploy models on Android, iOS and embedded systems like a Raspberry Pi and Edge TPUs.  
[TensorFlow.js](https://js.tensorflow.org/): Enables deploying models in JavaScript environments, such as in a web browser or server side through Node.js. TensorFlow.js also supports defining models in JavaScript and training directly in the web browser using a Keras-like API.

TensorFlow also has support for additional languages (some maintained by the broader community), including [C](https://www.tensorflow.org/install/lang_c), [Java](https://www.tensorflow.org/install/lang_java), [Go](https://www.tensorflow.org/install/lang_go), [C#](https://github.com/migueldeicaza/TensorFlowSharp), [Rust](https://github.com/tensorflow/rust), [Julia](https://github.com/malmaud/TensorFlow.jl), [R](https://tensorflow.rstudio.com/), and others.

12

**This picture shows typical platforms for the general-purpose computation,** machine learning **and** Deep Learning, and Tensorflow can be used both for ML and DL.

TensorFlow is **Powerful experimentation for research**

TensorFlow makes it easy to take new ideas from concept to code, and from model to publication.

TensorFlow 2.0 incorporates a number of features that enables the definition and training of state-of-the-art models without sacrificing speed or performance:

* [Keras Functional API](https://www.tensorflow.org/guide/keras) and [Model Subclassing API](https://www.tensorflow.org/guide/keras): Allows for creation of complex topologies including using residual layers, custom multi-input/-output models, and imperatively written forward passes.
* Custom Training Logic: Fine-grained control on gradient computations with [tf.GradientTape](https://www.tensorflow.org/api_docs/python/tf/GradientTape) and [tf.custom\_gradient](https://www.tensorflow.org/api_docs/python/tf/custom_gradient).
* And for even more flexibility and control, the low-level TensorFlow API is always available and working in conjunction with the higher-level abstractions for fully customizable logic.

TensorFlow 2.0 brings several new additions that allow researchers and advanced users to experiment, using rich extensions

like [Ragged Tensors](https://medium.com/tensorflow/introducing-ragged-tensors-ac301c31fd38), [TensorFlow Probability](https://medium.com/tensorflow/introducing-tensorflow-probability-dca4c304e245), [Tensor2Tensor](https://ai.googleblog.com/2017/06/accelerating-deep-learning-research.html), and more to be announced.

Along with these capabilities, TensorFlow provides eager execution for easy prototyping & debugging, Distribution Strategy API and AutoGraph to train at scale, and support for [TPUs](https://cloud.google.com/tpu/), making TensorFlow 2.0 an easy to use, customizable, and highly scalable platform for conducting state of the art ML research and translating that research into production pipelines.

13

To manage using TensorFlow, a serious experiments were designed,

which included Boston Housing Prediction using linear regression, Objection Detection and **Variational AutoEncoder.**

**The steps of each experiment showed in the diagram.**

Boston Housing Prediction included preparation, import data, Data visualization, build Model, Train the Model and result visualization.

**Objection Detection** included build the environment of TensorFlow on PC, build computation graph by using TensorFlow and Using computational graph to compute computation.

**Variational AutoEncoder include two stages Network construction and** Solving gradient disappearing problem using VAE, which consist of Learn the concept of layers, build a model that encapsulates layers by using the function sequential, set the loss function, calculate the gradient of the loss function to perform Back Propagation to adjust the parameters and add some noise into the validation set to solve the gradient disappearing problem.

all the experiments material including the manual and codes are provided on the MOOC platform, which can help you to do the hands-on.

14

In this session, we learned concepts and mechanism of TensorFlow

thank you for your attention, if you have any question, feel free to contact me.