Machine Learning at Scale
TensorFlow in the Cloud

Yufeng Guo
Developer Advocate
@YufengG
yufengg.com
Machine Learning is using many examples to answer questions
Training

many examples

Prediction

answer questions
Training  
many examples  

Prediction  
answer questions
- Fast, flexible, and scalable open-source machine learning library
- For research and production
- Distributed training and serving predictions
- Apache 2.0 license

A multidimensional array.

A graph of operations.

TensorFlow

3 + 2 = 5
TensorFlow Supports Many Platforms...

- CPU
- GPU
- Android
- iOS
- Raspberry Pi
- TPU

@YufengG
TensorFlow Distributed Execution Engine

- Python Frontend
- C++ Frontend
- ... more coming

Supported Platforms:
- CPU
- GPU
- Android
- iOS
- ...
Layers

Build models

Python Frontend

C++ Frontend

TensorFlow Distributed Execution Engine

CPU

GPU

Android

iOS

...
TensorFlow Distributed Execution Engine

- CPU
- GPU
- Android
- iOS
- ...

Layers

Build models

Estimator

Keras Model

Train and evaluate models

Python Frontend

C++ Frontend

...

Google Cloud

@YufengG
TensorFlow Distributed Execution Engine

Canned Estimators
- Models in a box. Takes care of optimizer, training loop, learning rate, etc
- Train and evaluate models
- Build models

- Estimator
- Keras Model
- Layers
- Python Frontend
- C++ Frontend
- ...
area = real_valued_column("square_foot"),
rooms = real_valued_column("num_rooms"),
zip_code = sparse_column_with_integerized_feature("zip_code", 10000)

classifier = DNNClassifier(
    feature_columns=[area, rooms, embedding_column(zip_code, 8)],
    hidden_units=[1024, 512, 256, 128])

classifier.fit(train_input_fn)

results = classifier.evaluate(eval_input_fn)

print(results)
classifier = DNNLinearCombinedRegressor(
    linear_feature_columns=[area, rooms, embedding_column(zip_code, 8)],
    linear_optimizer=tf.train.FtrlOptimizer(learning_rate=0.01,
        l2_regularization_strength=0.1),
    dnn_feature_columns=[real_valued_column(area),
        real_valued_column(rooms)]
    dnn_optimizer=tf.train.AdagradOptimizer(learning_rate=0.01,
        initial_accumulator_value=0.1),
    dnn_activation_fn=tf.nn.relu,
    dnn_dropout = 0.5,
    gradient_clip_norm=0.1,
    hidden_units=[1024, 512, 256, 128])

classifier.fit(train_input_fn)
classifier.evaluate(eval_input_fn)
Motivation - a "magical" food app
Just **Launch** and Iterate

- Naive character matching
- Say "Fried chicken"
- Get "Chicken Fried Rice"
- Oops. Now what?
- Machine learning to the rescue!
v2.0: **memorize** all the things!

- Train a linear TF model

- Your app is gaining traction!
Problem: Your users are bored!

- Too many 🍗 & waffles
- Show me similar, but different food
- Your users are picky 🙈
v3.0: More generalized recommendations for all
No good deed goes unpunished

- Some recommendations are "too general"
  - Irrelevant dishes are being sent
- Your users are still picky 😞
No good deed goes unpunished

- 2 types of requests: specific and general
- "iced decaf latte with nonfat milk" != "hot latte with whole milk"
- “seafood” or “italian food” or “fast food”
- How to balance this?
v4.0: Why not both?
Wide & Deep

memorization  generalization
relevance    diversity
Wide and Deep

AND(query="fried chicken", item="chicken fried rice")

query="fried chicken"

item="chicken fried rice"
Meet our dataset: US Census Data

- **Task**: predict whether the household has an annual income of over $50K
- **Over 32k training examples**
- **Extracted from the 1994 US Census by Barry Becker.**
Meet our dataset: US Census Data

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
<td>Continuous</td>
<td>The age of the individual</td>
</tr>
<tr>
<td>workclass</td>
<td>Categorical</td>
<td>The type of employer the individual has (government, military, private, etc.).</td>
</tr>
<tr>
<td>fnlwgt</td>
<td>Continuous</td>
<td>The number of people the census takers believe that observation represents (sample weight). Not used.</td>
</tr>
<tr>
<td>education</td>
<td>Categorical</td>
<td>The highest level of education achieved for that individual.</td>
</tr>
<tr>
<td>education_num</td>
<td>Continuous</td>
<td>The highest level of education in numerical form.</td>
</tr>
<tr>
<td>marital_status</td>
<td>Categorical</td>
<td>Marital status of the individual.</td>
</tr>
</tbody>
</table>
## Meet our dataset: US Census Data

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>occupation</td>
<td>Categorical</td>
<td>The occupation of the individual.</td>
</tr>
<tr>
<td>relationship</td>
<td>Categorical</td>
<td>Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.</td>
</tr>
<tr>
<td>race</td>
<td>Categorical</td>
<td>White, Asian-Pacific Islander, Amer-Indian-Eskimo, Other, Black.</td>
</tr>
<tr>
<td>gender</td>
<td>Categorical</td>
<td>Female, Male.</td>
</tr>
<tr>
<td>capital_gain</td>
<td>Continuous</td>
<td>Capital gains recorded.</td>
</tr>
<tr>
<td>capital_loss</td>
<td>Continuous</td>
<td>Capital Losses recorded.</td>
</tr>
</tbody>
</table>
Meet our dataset: US Census Data

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hours_per_week</td>
<td>Continuous</td>
<td>Hours worked per week.</td>
</tr>
<tr>
<td>native_country</td>
<td>Categorical</td>
<td>Country of origin of the individual.</td>
</tr>
<tr>
<td>income_bracket</td>
<td>Categorical</td>
<td>&quot;&gt;50K&quot; or &quot;&lt;=50K&quot;, meaning whether the person makes more than $50,000 annually.</td>
</tr>
</tbody>
</table>
Wide & Deep

memorization  generalization
relevance  diversity

Sparse  Dense/Real
Categorical  Continuous
To the code!

bit.ly/widendeep-census
tensorboard --logdir=models/
Training many examples

Prediction answer questions
What is Serving?

Data → TensorFlow → Model → RPC Server → App

Data Scientist
What is TensorFlow Serving?

- **C++ Libraries**
  - TensorFlow model save / export formats
  - Generic core platform

- **Binaries**
  - Best practices out of the box
  - Docker containers, K8s tutorial

- **Hosted Service across**
  - Google Cloud ML Engine
  - Internal service
Model Creation
<table>
<thead>
<tr>
<th>Name</th>
<th>Default version</th>
</tr>
</thead>
<tbody>
<tr>
<td>wnd1</td>
<td>vCMD2</td>
</tr>
</tbody>
</table>

[Create Model]
A model is a container for your model versions. After you create your model, train your first version from the command line and add it to Cloud Machine Learning Engine. Learn more

**Model name**
Model names must be unique within each project.

```
my_model
```

[Create]  [Cancel]
my_model2

Versions

This model has no versions yet. Create at least one version to start using your model. Create a version
To create a new version of your model, submit a training job to the Cloud ML API and specify the output below. Learn more

**Name**
Name is permanent.

```
v1|
```

**Source**
Enter the Google Cloud Storage output path you specified in your training job.

```
folder/
```

[Browse]

[Create] [Cancel]
<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Type</th>
<th>Storage class</th>
<th>Last modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>saved_model.pb</td>
<td>733.91 KB</td>
<td>binary/octet-stream</td>
<td>Regional</td>
<td>3/21/17, 7:53 PM</td>
</tr>
<tr>
<td>variables/</td>
<td>-</td>
<td>Folder</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
To create a new version of your model, submit a training job to the Cloud ML API and specify the output below. 

**Name**
Name is permanent.

```
v1
```

**Source**
Enter the Google Cloud Storage output path you specified in your training job.

```
cloudml-engine/wide_n_deep/1490151039088/
```

[Create] [Cancel]
## my_model2

### Recent operations

### Versions

<table>
<thead>
<tr>
<th>Name</th>
<th>Creation time</th>
<th>Last use time</th>
<th>Set as default</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 (default)</td>
<td>May 4, 2017, 3:56:53 PM</td>
<td></td>
<td>Set as default</td>
<td>Delete</td>
</tr>
</tbody>
</table>
export MODEL_NAME='my_model'
gcloud ml-engine models --regions us-central1 create $MODEL_NAME

export MODEL_NAME='cloudwnd'
export VERSION_NAME='learn_runner_standard'
export DEPLOYMENT_SOURCE='gs://cloudml-engine/widendeply_yufengg_20170410_164903/model_WIDE_AND_DEEP_1491857627/export/Servo/1491857907860'

$ gcloud ml-engine versions create $VERSION_NAME --model $MODEL_NAME --origin $DEPLOYMENT_SOURCE
Creating version (this might take a few minutes)......
Instance Prediction
{
    "age": 25,
    "workclass": "Private",
    "education": "11th",
    "education_num": 7,
    "marital_status": "Never-married",
    "occupation": "Machine-op-inspct",
    "relationship": "Own-child",
    "race": "Black",
    "gender": "Male",
    "capital_gain": 0,
    "capital_loss": 0,
    "hours_per_week": 40,
    "native_country": "United-States"
}

{
    "age": 42,
    "workclass": "Self-emp-inc",
    "education": "HS-grad",
    "education_num": 9,
    "marital_status": "Married-civ-spouse",
    "occupation": "Exec-managerial",
    "relationship": "Husband",
    "race": "White",
    "gender": "Male",
    "capital_gain": 5178,
    "capital_loss": 0,
    "hours_per_week": 50,
    "native_country": "United-States"
}
$ gcloud ml-engine predict --model wnd1 --version vCMDZ --json-instances census.json

<table>
<thead>
<tr>
<th>CLASSES</th>
<th>LOGISTIC</th>
<th>LOGITS</th>
<th>PROBABILITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>[0.005143760237842798]</td>
<td>[-5.2648138999938965]</td>
<td>[0.9948562383651733, 0.005143760237842798]</td>
</tr>
<tr>
<td>1</td>
<td>[0.8839852213859558]</td>
<td>[2.0307230949401855]</td>
<td>[0.1160147413611412, 0.8839852213859558]</td>
</tr>
</tbody>
</table>

PROBABILITIES

[0.9948562383651733, 0.005143760237842798]
[0.1160147413611412, 0.8839852213859558]
{
    "probabilities": [
        0.11601490527391434,
        0.8839850425720215
    ],
    "logits": [
        2.030721426010132
    ],
    "classes": 1,
    "logistic": [
        0.8839850425720215
    ]
}

{
    "probabilities": [
        0.9948562383651733,
        0.005143760237842798
    ],
    "logits": [
        -5.2648138999938965
    ],
    "classes": 0,
    "logistic": [
        0.005143760237842798
    ]
}
Training many examples

TensorFlow

Prediction answer questions
Training
many examples

Prediction
answer questions

TensorFlow
Training many examples

Prediction answer questions
Thank you!

@YufengG
yufengg.com

Resources:

Cloud Machine Learning Engine
cloud.google.com/ml-engine

TensorFlow
tensorflow.org

To the code!
bit.ly/widen
bit.ly/wide-deep-census
bit.ly/wide-deep-code
The End