MNIST Image Classification with TensorFlow

This notebook demonstrates how to implement a simple linear image models on MNIST using tf.keras.

This companion notebook extends the basic harness of this notebook to a variety of models including DNN, CNN, dropout, pooling etc.

In [1]:
```
!sudo chown -R jupyter:jupyter /home/jupyter/training-data-analyst
```

In [2]:
```
import numpy as np
import shutil
import os
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
print(tf.__version__)
```

```
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/compat/v2_compat.py:101: disable_resource_variables (from tensorflow.python.op
s.variable_scope) is deprecated and will be removed in a future version.
Instructions for updating:
non-resource variables are not supported in the long term
2.6.0
```

Exploring the data

Let's download MNIST data and examine the shape. We will need these numbers ...

In [3]:
```
HEIGHT = 28
WIDTH = 28
NCLASSES = 10
```

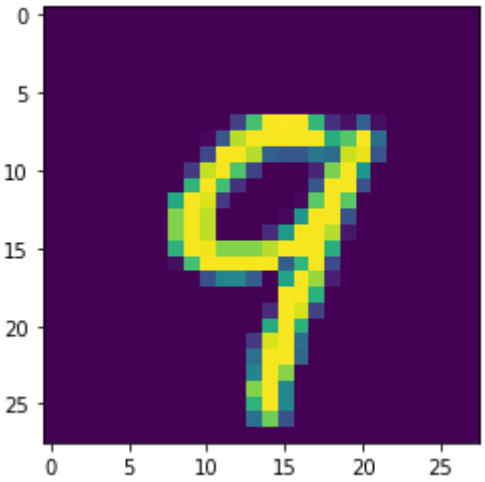
In [4]:
```
# Get mnist data
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Scale our features between 0 and 1
x_train, x_test = x_train / 255.0, x_test / 255.0

# Convert labels to categorical one-hot encoding
y_train = tf.keras.utils.to_categorical(y = y_train, num_classes = NCLASSES)
y_test = tf.keras.utils.to_categorical(y = y_test, num_classes = NCLASSES)

print("x_train.shape = {}".format(x_train.shape))
print("y_train.shape = {}".format(y_train.shape))
print("x_test.shape = {}".format(x_test.shape))
print("y_test.shape = {}".format(y_test.shape))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
```

```
s/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
x_train.shape = (60000, 28, 28)
y_train.shape = (60000, 10)
x_test.shape = (10000, 28, 28)
y_test.shape = (10000, 10)
```

In [5]:

```
import matplotlib.pyplot as plt
IMGNO = 12
plt.imshow(x_test[IMGNO].reshape(HEIGHT, WIDTH));
```



Define the model.

Let's start with a very simple linear classifier. All our models will have this basic interface -- they will take an image and return probabilities.

In [6]:

```
# Build Keras Model Using Keras Sequential API
def linear_model():
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.InputLayer(input_shape = [HEIGHT, WIDTH], name = "
    model.add(tf.keras.layers.Flatten())
    model.add(tf.keras.layers.Dense(units = NCLASSES, activation = tf.nn.softmax
    return model
```

Write Input Functions

As usual, we need to specify input functions for training, evaluation, and predicition.

In [7]:

```
# Create training input function
train_input_fn = tf.estimator.inputs.numpy_input_fn(
    x = {"image": x_train},
    y = y_train,
    batch_size = 100,
    num_epochs = None,
    shuffle = True,
    queue_capacity = 5000
  )
```

```
# Create evaluation input function
eval_input_fn = tf.estimator.inputs.numpy_input_fn(
    x = {"image": x_test},
    y = y_test,
    batch_size = 100,
    num_epochs = 1,
    shuffle = False,
    queue_capacity = 5000
  )

# Create serving input function for inference
def serving_input_fn():
    placeholders = {"image": tf.placeholder(dtype = tf.float32, shape = [None, H
    features = placeholders # as-is
    return tf.estimator.export.ServingInputReceiver(features = features, receive
```

```
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/util/lazy_loader.py:63: The name tf.estimator.inputs is deprecated. Please use
tf.compat.v1.estimator.inputs instead.

WARNING:tensorflow:From /tmp/ipykernel_25512/1739958052.py:2: The name tf.estima
tor.inputs.numpy_input_fn is deprecated. Please use tf.compat.v1.estimator.input
s.numpy_input_fn instead.
```

Create train_and_evaluate function

tf.estimator.train_and_evaluate does distributed training.

In [8]:

```
def train_and_evaluate(output_dir, hparams):
    # Build Keras model
    model = linear_model()

    # Compile Keras model with optimizer, loss function, and eval metrics
    model.compile(
        optimizer = "adam",
        loss = "categorical_crossentropy",
        metrics = ["accuracy"])

    # Convert Keras model to an Estimator
    estimator = tf.keras.estimator.model_to_estimator(
        keras_model = model,
        model_dir = output_dir)

    # Set estimator's train_spec to use train_input_fn and train for so many ste
    train_spec = tf.estimator.TrainSpec(
        input_fn = train_input_fn,
        max_steps = hparams["train_steps"])

    # Create exporter that uses serving_input_fn to create saved_model for servi
    exporter = tf.estimator.LatestExporter(
        name = "exporter",
        serving_input_receiver_fn = serving_input_fn)

    # Set estimator's eval_spec to use eval_input_fn and export saved_model
    eval_spec = tf.estimator.EvalSpec(
        input_fn = eval_input_fn,
        steps = None,
```

```
        exporters = exporter)

    # Run train_and_evaluate loop
    tf.estimator.train_and_evaluate(
        estimator = estimator,
        train_spec = train_spec,
        eval_spec = eval_spec)
```

This is the main() function

In [9]:

```
OUTDIR = "mnist/learned"
shutil.rmtree(OUTDIR, ignore_errors = True) # start fresh each time

hparams = {"train_steps": 1000, "learning_rate": 0.01}
train_and_evaluate(OUTDIR, hparams)
```

```
INFO:tensorflow:Using default config.
INFO:tensorflow:Using the Keras model provided.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_estima
tor/python/estimator/keras_lib.py:703: The name tf.keras.backend.set_session is
deprecated. Please use tf.compat.v1.keras.backend.set_session instead.

WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/ops/init_ops.py:97: calling GlorotUniform.__init__ (from tensorflow.python.op
s.init_ops) with dtype is deprecated and will be removed in a future version.
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the c
onstructor
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/ops/init_ops.py:97: calling Zeros.__init__ (from tensorflow.python.ops.init_op
s) with dtype is deprecated and will be removed in a future version.
Instructions for updating:
Call initializer instance with the dtype argument instead of passing it to the c
onstructor
2021-09-29 12:43:33.549168: I tensorflow/core/common_runtime/process_util.cc:14
6] Creating new thread pool with default inter op setting: 2. Tune using inter_o
p_parallelism_threads for best performance.
/opt/conda/lib/python3.7/site-packages/keras/backend.py:401: UserWarning: `tf.ke
ras.backend.set_learning_phase` is deprecated and will be removed after 2020-10-
11. To update it, simply pass a True/False value to the `training` argument of t
he `__call__` method of your layer or model.
  warnings.warn('`tf.keras.backend.set_learning_phase` is deprecated and '
INFO:tensorflow:Using config: {'_model_dir': 'mnist/learned', '_tf_random_seed':
None, '_save_summary_steps': 100, '_save_checkpoints_steps': None, '_save_checkp
oints_secs': 600, '_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000, '_log_step
_count_steps': 100, '_train_distribute': None, '_device_fn': None, '_protocol':
None, '_eval_distribute': None, '_experimental_distribute': None, '_experimental
_max_worker_delay_secs': None, '_session_creation_timeout_secs': 7200, '_checkpo
int_save_graph_def': True, '_service': None, '_cluster_spec': ClusterSpec({}),
'_task_type': 'worker', '_task_id': 0, '_global_id_in_cluster': 0, '_master':
'', '_evaluation_master': '', '_is_chief': True, '_num_ps_replicas': 0, '_num_wo
rker_replicas': 1}
INFO:tensorflow:Not using Distribute Coordinator.
```

```
INFO:tensorflow:Running training and evaluation locally (non-distributed).
INFO:tensorflow:Start train and evaluate loop. The evaluate will happen after ev
ery checkpoint. Checkpoint frequency is determined based on RunConfig arguments:
save_checkpoints_steps None or save_checkpoints_secs 600.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/training/training_util.py:236: Variable.initialized_value (from tensorflow.pyt
hon.ops.variables) is deprecated and will be removed in a future version.
Instructions for updating:
Use Variable.read_value. Variables in 2.X are initialized automatically both in
eager and graph (inside tf.defun) contexts.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_estima
tor/python/estimator/inputs/queues/feeding_queue_runner.py:65: QueueRunner.__ini
t__ (from tensorflow.python.training.queue_runner_impl) is deprecated and will b
e removed in a future version.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow_estima
tor/python/estimator/inputs/queues/feeding_functions.py:491: add_queue_runner (f
rom tensorflow.python.training.queue_runner_impl) is deprecated and will be remo
ved in a future version.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Warm-starting with WarmStartSettings: WarmStartSettings(ckpt_to_
initialize_from='mnist/learned/keras/keras_model.ckpt', vars_to_warm_start='.*',
var_name_to_vocab_info={}, var_name_to_prev_var_name={})
INFO:tensorflow:Warm-starting from: mnist/learned/keras/keras_model.ckpt
INFO:tensorflow:Warm-starting variables only in TRAINABLE_VARIABLES.
INFO:tensorflow:Warm-started 2 variables.
INFO:tensorflow:Create CheckpointSaverHook.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/training/monitored_session.py:907: start_queue_runners (from tensorflow.pytho
n.training.queue_runner_impl) is deprecated and will be removed in a future vers
ion.
Instructions for updating:
To construct input pipelines, use the `tf.data` module.
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 0...
INFO:tensorflow:Saving checkpoints for 0 into mnist/learned/model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 0...
INFO:tensorflow:loss = 2.3271546, step = 1
INFO:tensorflow:global_step/sec: 233.52
INFO:tensorflow:loss = 0.77984947, step = 101 (0.430 sec)
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not
been increased. Current value (could be stable): 147 vs previous value: 147. You
could increase the global step by passing tf.train.get_global_step() to Optimize
r.apply_gradients or Optimizer.minimize.
INFO:tensorflow:global_step/sec: 243.909
INFO:tensorflow:loss = 0.5936929, step = 201 (0.412 sec)
INFO:tensorflow:global_step/sec: 143.971
INFO:tensorflow:loss = 0.5222031, step = 301 (0.692 sec)
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not
been increased. Current value (could be stable): 324 vs previous value: 324. You
could increase the global step by passing tf.train.get_global_step() to Optimize
r.apply_gradients or Optimizer.minimize.
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not
been increased. Current value (could be stable): 326 vs previous value: 326. You
could increase the global step by passing tf.train.get_global_step() to Optimize
```

```
r.apply_gradients or Optimizer.minimize.
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not
been increased. Current value (could be stable): 350 vs previous value: 350. You
could increase the global step by passing tf.train.get_global_step() to Optimize
r.apply_gradients or Optimizer.minimize.
WARNING:tensorflow:It seems that global step (tf.train.get_global_step) has not
been increased. Current value (could be stable): 352 vs previous value: 352. You
could increase the global step by passing tf.train.get_global_step() to Optimize
r.apply_gradients or Optimizer.minimize.
INFO:tensorflow:global_step/sec: 125.158
INFO:tensorflow:loss = 0.40985405, step = 401 (0.804 sec)
INFO:tensorflow:global_step/sec: 172.705
INFO:tensorflow:loss = 0.33293733, step = 501 (0.574 sec)
INFO:tensorflow:global_step/sec: 318.762
INFO:tensorflow:loss = 0.3173626, step = 601 (0.314 sec)
INFO:tensorflow:global_step/sec: 323.965
INFO:tensorflow:loss = 0.44926682, step = 701 (0.309 sec)
INFO:tensorflow:global_step/sec: 323.071
INFO:tensorflow:loss = 0.39262304, step = 801 (0.309 sec)
INFO:tensorflow:global_step/sec: 320.293
INFO:tensorflow:loss = 0.5216224, step = 901 (0.312 sec)
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 1000...
INFO:tensorflow:Saving checkpoints for 1000 into mnist/learned/model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 1000...
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Starting evaluation at 2021-09-29T12:43:39
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from mnist/learned/model.ckpt-1000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
/opt/conda/lib/python3.7/site-packages/keras/engine/training.py:2470: UserWarnin
g: `Model.state_updates` will be removed in a future version. This property shou
ld not be used in TensorFlow 2.0, as `updates` are applied automatically.
  warnings.warn('`Model.state_updates` will be removed in a future version. '
INFO:tensorflow:Inference Time : 0.40821s
INFO:tensorflow:Finished evaluation at 2021-09-29-12:43:39
INFO:tensorflow:Saving dict for global step 1000: acc = 0.914, global_step = 100
0, loss = 0.31924975
INFO:tensorflow:Saving 'checkpoint_path' summary for global step 1000: mnist/lea
rned/model.ckpt-1000
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
WARNING:tensorflow:From /opt/conda/lib/python3.7/site-packages/tensorflow/pytho
n/saved_model/signature_def_utils_impl.py:201: build_tensor_info (from tensorflo
w.python.saved_model.utils_impl) is deprecated and will be removed in a future v
ersion.
Instructions for updating:
This function will only be available through the v1 compatibility library as tf.
compat.v1.saved_model.utils.build_tensor_info or tf.compat.v1.saved_model.build_
tensor_info.
INFO:tensorflow:Signatures INCLUDED in export for Classify: None
INFO:tensorflow:Signatures INCLUDED in export for Regress: None
INFO:tensorflow:Signatures INCLUDED in export for Predict: ['serving_default']
INFO:tensorflow:Signatures INCLUDED in export for Train: None
INFO:tensorflow:Signatures INCLUDED in export for Eval: None
INFO:tensorflow:Restoring parameters from mnist/learned/model.ckpt-1000
INFO:tensorflow:Assets added to graph.
INFO:tensorflow:No assets to write.
INFO:tensorflow:SavedModel written to: mnist/learned/export/exporter/temp-163291
```

```
9419/saved_model.pb
INFO:tensorflow:Loss for final step: 0.31267023.
```

I got:

```
Saving dict for global step 1000: categorical_accuracy = 0.9112,
global_step = 1000, loss = 0.32516304
```

In other words, we achieved 91.12% accuracy with the simple linear model!

```
# Copyright 2020 Google Inc. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#      http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing,
software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
# See the License for the specific language governing permissions
and
# limitations under the License.
```