Lecture 12: Introduction to TensorFlow II

In [2]:

```
# To support both python 2 and python 3
from __future__ import division, print_function, unicode_literals

# Common imports
import numpy as np
import os

# To make this notebook's output stable across runs
def reset_graph(seed=42):
    tf.reset_default_graph()
    tf.set_random_seed(seed)
    np.random.seed(seed)

# To plot pretty figures
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
plt.rcParams['axes.labelsize'] = 14
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12
```

In [3]:
```
import tensorflow as tf
import numpy as np
from sklearn.datasets import fetch_california_housing
reset_graph()

housing = fetch_california_housing()
m, n = housing.data.shape
housing_data_plus_bias = np.c_[np.ones((m, 1)), housing.data]
housing_data_target = housing.target.reshape(-1, 1)
```

```
/Users/mcewen/anaconda3/envs/tensorflow_py35/lib/python3.5/site-packages/h5py/
__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype
from `float` to `np.floating` is deprecated. In future, it will be treated as
`np.float64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
```

In [4]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_housing_data = scaler.fit_transform(housing.data)
scaled_housing_data_plus_bias = np.c_[np.ones((m, 1)), scaled_housing_data]
```

Saving and restoring models

Models and parameters can be saved easily.

Good to not only save the final model but to also checkpoint (i.e. save intermediate models) as training performed.

In [6]:
```
with tf.Session() as sess:
    sess.run(init)

    for epoch in range(n_epochs):
        if epoch % 100 == 0:
            print("Epoch", epoch, "MSE =", mse.eval())

            save_path = saver.save(sess, "./my_model.ckpt")
        sess.run(training_op)

    best_theta = theta.eval()
    save_path = saver.save(sess, "./my_model_final.ckpt")
```

```
Epoch 0 MSE = 9.161543
Epoch 100 MSE = 0.7145006
Epoch 200 MSE = 0.56670463
Epoch 300 MSE = 0.5555716
Epoch 400 MSE = 0.5488117
Epoch 500 MSE = 0.5436362
Epoch 600 MSE = 0.53962916
Epoch 700 MSE = 0.53650916
Epoch 800 MSE = 0.5340678
Epoch 900 MSE = 0.53214717
```

In [7]:
```
best_theta
```

Out[7]:
```
array([[ 2.0685525 ],
       [ 0.8874027 ],
       [ 0.14401658],
       [-0.34770882],
       [ 0.36178368],
       [ 0.00393811],
       [-0.04269556],
       [-0.6614528 ],
       [-0.6375277 ]], dtype=float32)
```

Models can then be restored easily.

In [8]:
```
with tf.Session() as sess:
    saver.restore(sess, "./my_model_final.ckpt")
    best_theta_restored = theta.eval()
```

```
INFO:tensorflow:Restoring parameters from ./my_model_final.ckpt
```

In [9]:
```
best_theta_restored
```

Out[9]:
```
array([[ 2.0685525 ],
       [ 0.8874027 ],
       [ 0.14401658],
       [-0.34770882],
       [ 0.36178368],
       [ 0.00393811],
       [-0.04269556],
       [-0.6614528 ],
       [-0.6375277 ]], dtype=float32)
```

In [10]:
```
np.allclose(best_theta, best_theta_restored)
```

Out[10]:
```
True
```

Computational graph definition is saved in file with `.meta` extension.

Can also load graphs.

In [11]:
```
reset_graph()  # start with an empty graph.

saver = tf.train.import_meta_graph("./my_model_final.ckpt.meta")  # load the graph
 structure
theta = tf.get_default_graph().get_tensor_by_name("theta:0")

with tf.Session() as sess:
    saver.restore(sess, "./my_model_final.ckpt")  # restores the graph's state
    best_theta_restored = theta.eval()
```

```
INFO:tensorflow:Restoring parameters from ./my_model_final.ckpt
```

In [12]:
```
np.allclose(best_theta, best_theta_restored)
```

Out[12]:
```
True
```

Visualising computational graphs with TensorBoard

TensorBoard provides functionality to visualise computational graphs and training statistics.

Logging

In [13]:
```
reset_graph()

from datetime import datetime

now = datetime.utcnow().strftime("%Y%m%d%H%M%S")
root_logdir = "tf_logs"
logdir = "{}/run-{}/".format(root_logdir, now)
```

In [14]:
```
n_epochs = 1000
learning_rate = 0.01

X = tf.placeholder(tf.float32, shape=(None, n + 1), name="X")
y = tf.placeholder(tf.float32, shape=(None, 1), name="y")
theta = tf.Variable(tf.random_uniform([n + 1, 1], -1.0, 1.0, seed=42), name="thet
a")
y_pred = tf.matmul(X, theta, name="predictions")
error = y_pred - y
mse = tf.reduce_mean(tf.square(error), name="mse")
optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
training_op = optimizer.minimize(mse)

init = tf.global_variables_initializer()
```

Create summary and file writer:

In [15]:
```
mse_summary = tf.summary.scalar('MSE', mse)
file_writer = tf.summary.FileWriter(logdir, tf.get_default_graph())
```

In [16]:
```
n_epochs = 10
batch_size = 100
n_batches = int(np.ceil(m / batch_size))
```

In [17]:
```
def fetch_batch(epoch, batch_index, batch_size):
    np.random.seed(epoch * n_batches + batch_index)
    indices = np.random.randint(m, size=batch_size)
    X_batch = scaled_housing_data_plus_bias[indices]
    y_batch = housing_data_target[indices]
    return X_batch, y_batch
```

In [18]:
```
with tf.Session() as sess:
    sess.run(init)

    for epoch in range(n_epochs):
        for batch_index in range(n_batches):
            X_batch, y_batch = fetch_batch(epoch, batch_index, batch_size)
            if batch_index % 10 == 0:
                summary_str = mse_summary.eval(feed_dict={X: X_batch, y: y_batch})
                step = epoch * n_batches + batch_index
                file_writer.add_summary(summary_str, step)
            sess.run(training_op, feed_dict={X: X_batch, y: y_batch})

    best_theta = theta.eval()
```

In [19]:
```
file_writer.close()
```

In [20]:
```
best_theta
```

Out[20]:
```
array([[ 2.0703337 ],
       [ 0.8637145 ],
       [ 0.12255151],
       [-0.31211874],
       [ 0.38510373],
       [ 0.00434168],
       [-0.01232954],
       [-0.83376896],
       [-0.8030471 ]], dtype=float32)
```

TensorBoard

Now we can inspect the logs in TensorBoard.

At the command line run:

```
tensorboard --logdir tf_logs
```

Open in a browser on port 6006, i.e. localhost:6006...

More complex models

Name scopes

Large complex models can easily become cluttered with many nodes making then difficult to visualise directly.

To avoid this problem *name scopes* can be created to group nodes.

In [21]:

```
reset_graph()

now = datetime.utcnow().strftime("%Y%m%d%H%M%S")
root_logdir = "tf_logs"
logdir = "{}/run-{}/".format(root_logdir, now)

n_epochs = 1000
learning_rate = 0.01

X = tf.placeholder(tf.float32, shape=(None, n + 1), name="X")
y = tf.placeholder(tf.float32, shape=(None, 1), name="y")
theta = tf.Variable(tf.random_uniform([n + 1, 1], -1.0, 1.0, seed=42), name="thet
a")
y_pred = tf.matmul(X, theta, name="predictions")
```

Create `error` and `mse` inside `loss` name scope:

In [22]:
```
with tf.name_scope("loss") as scope:
    error = y_pred - y
    mse = tf.reduce_mean(tf.square(error), name="mse")
```

Set up optimizer:

In [23]:
```
optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
training_op = optimizer.minimize(mse)

init = tf.global_variables_initializer()

mse_summary = tf.summary.scalar('MSE', mse)
file_writer = tf.summary.FileWriter(logdir, tf.get_default_graph())
```

Run:

In [24]:
```
n_epochs = 10
batch_size = 100
n_batches = int(np.ceil(m / batch_size))

with tf.Session() as sess:
    sess.run(init)

    for epoch in range(n_epochs):
        for batch_index in range(n_batches):
            X_batch, y_batch = fetch_batch(epoch, batch_index, batch_size)
            if batch_index % 10 == 0:
                summary_str = mse_summary.eval(feed_dict={X: X_batch, y: y_batch})
                step = epoch * n_batches + batch_index
                file_writer.add_summary(summary_str, step)
            sess.run(training_op, feed_dict={X: X_batch, y: y_batch})

    best_theta = theta.eval()
```

In [25]:

```
file_writer.flush()
file_writer.close()
print("Best theta:")
print(best_theta)
```

```
Best theta:
[[ 2.0703337 ]
 [ 0.8637145 ]
 [ 0.12255151]
 [-0.31211874]
 [ 0.38510373]
 [ 0.00434168]
 [-0.01232954]
 [-0.83376896]
 [-0.8030471 ]]
```

View in TensorBoard...

In [26]:
```
print(error.op.name)
```

```
loss/sub
```

In [27]:
```
print(mse.op.name)
```

```
loss/mse
```

Modularity

Many graph components are often repeated.

For example, let's create a graph with two rectified linear units (ReLUs).

Could just repeat code but this is not efficient coding and is prone to errors.

In [28]:
```
reset_graph()

n_features = 3
X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")

w1 = tf.Variable(tf.random_normal((n_features, 1)), name="weights1")
w2 = tf.Variable(tf.random_normal((n_features, 1)), name="weights2")
b1 = tf.Variable(0.0, name="bias1")
b2 = tf.Variable(0.0, name="bias2")

z1 = tf.add(tf.matmul(X, w1), b1, name="z1")
z2 = tf.add(tf.matmul(X, w2), b2, name="z2")

relu1 = tf.maximum(z1, 0., name="relu1")
relu2 = tf.maximum(z2, 0., name="relu2")

output = tf.add(relu1, relu2, name="output")
```

Better approach is to create a function defining a ReLU.

In [29]:
```
reset_graph()

def relu(X):
    w_shape = (int(X.get_shape()[1]), 1)
    w = tf.Variable(tf.random_normal(w_shape), name="weights")
    b = tf.Variable(0.0, name="bias")
    z = tf.add(tf.matmul(X, w), b, name="z")
    return tf.maximum(z, 0., name="relu")

n_features = 3
X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")
relus = [relu(X) for i in range(5)]
output = tf.add_n(relus, name="output")
```

In [30]:
```
file_writer = tf.summary.FileWriter("tf_logs/relu1", tf.get_default_graph())
```

View in TensorBoard...

Still somewhat confusing.

Use name scopes to make more clear.

In [31]:
```
reset_graph()

def relu(X):
    with tf.name_scope("relu"):
        w_shape = (int(X.get_shape()[1]), 1)
        w = tf.Variable(tf.random_normal(w_shape), name="weights")
        b = tf.Variable(0.0, name="bias")
        z = tf.add(tf.matmul(X, w), b, name="z")
        return tf.maximum(z, 0., name="max")
```

In [32]:
```
n_features = 3
X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")
relus = [relu(X) for i in range(5)]
output = tf.add_n(relus, name="output")

file_writer = tf.summary.FileWriter("tf_logs/relu2", tf.get_default_graph())
file_writer.close()
```

Sharing variables

Often want to share variables between nodes.

For example, say we want to consider ReLUs with non-zero thresholds.

One way is to simply pass parameters around as variables.

In [33]:
```
reset_graph()

def relu(X, threshold):
    with tf.name_scope("relu"):
        w_shape = (int(X.get_shape()[1]), 1)
        w = tf.Variable(tf.random_normal(w_shape), name="weights")
        b = tf.Variable(0.0, name="bias")
        z = tf.add(tf.matmul(X, w), b, name="z")
        return tf.maximum(z, threshold, name="max")

threshold = tf.Variable(0.0, name="threshold")
X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")
relus = [relu(X, threshold) for i in range(5)]
output = tf.add_n(relus, name="output")
```

While this works perfectly fine, TensorFlow offers an alternative approach that is cleaner and more modular.

Use TensorFlow `tf.get_variable` function to create (if it doesn't yet exist) or reuse variables.

In [34]:
```
reset_graph()

with tf.variable_scope("relu"):
    threshold = tf.get_variable("threshold", shape=(),
                                initializer=tf.constant_initializer(0.0))
```

This will actually raise an exception if the variable has already be created by an earlier call to `tf.get_variable`.

Need to explicitly specify that can reuse variables.

Can be performed by setting variable scope's `reuse` attribute to True, either by:

In [35]:
```
with tf.variable_scope("relu", reuse=True):
    threshold = tf.get_variable("threshold")
```

or by setting explicitly:

In [36]:
```
with tf.variable_scope("relu") as scope:
    scope.reuse_variables()
    threshold = tf.get_variable("threshold")
```

Using the shared threshold variable:

In [37]:
```
reset_graph()

def relu(X):
    with tf.variable_scope("relu", reuse=True):
        threshold = tf.get_variable("threshold")
        w_shape = int(X.get_shape()[1]), 1
        w = tf.Variable(tf.random_normal(w_shape), name="weights")
        b = tf.Variable(0.0, name="bias")
        z = tf.add(tf.matmul(X, w), b, name="z")
        return tf.maximum(z, threshold, name="max")

X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")
with tf.variable_scope("relu"):
    threshold = tf.get_variable("threshold", shape=(),
                                initializer=tf.constant_initializer(0.0))
relus = [relu(X) for relu_index in range(5)]
output = tf.add_n(relus, name="output")
```

In [38]:
```
file_writer = tf.summary.FileWriter("tf_logs/relu3", tf.get_default_graph())
file_writer.close()
```

View in TensorBoard...

Could also put the `threshold` inside the first ReLU:

In [39]:
```
reset_graph()

def relu(X):
    threshold = tf.get_variable("threshold", shape=(),
                                initializer=tf.constant_initializer(0.0))
    w_shape = (int(X.get_shape()[1]), 1)
    w = tf.Variable(tf.random_normal(w_shape), name="weights")
    b = tf.Variable(0.0, name="bias")
    z = tf.add(tf.matmul(X, w), b, name="z")
    return tf.maximum(z, threshold, name="max")

X = tf.placeholder(tf.float32, shape=(None, n_features), name="X")
relus = []
for relu_index in range(5):
    with tf.variable_scope("relu", reuse=(relu_index >= 1)) as scope:
        relus.append(relu(X))
output = tf.add_n(relus, name="output")
```

In [40]:
```
file_writer = tf.summary.FileWriter("tf_logs/relu4", tf.get_default_graph())
file_writer.close()
```

View in TensorBoard...

Neural networks in TensorFlow

Using TF.Learn (high-level API)

TF.Learn provides a high-level TensorFlow API in Python that is compatible with SciKit-Learn.

In [41]:
```
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("/tmp/data/")
```

```
Extracting /tmp/data/train-images-idx3-ubyte.gz
Extracting /tmp/data/train-labels-idx1-ubyte.gz
Extracting /tmp/data/t10k-images-idx3-ubyte.gz
Extracting /tmp/data/t10k-labels-idx1-ubyte.gz
```

In [42]:
```
X_train = mnist.train.images
X_test = mnist.test.images
y_train = mnist.train.labels.astype("int")
y_test = mnist.test.labels.astype("int")
```

In [43]:
```
import tensorflow as tf

config = tf.contrib.learn.RunConfig(tf_random_seed=42)

feature_cols = tf.contrib.learn.infer_real_valued_columns_from_input(X_train)
dnn_clf = tf.contrib.learn.DNNClassifier(hidden_units=[300,100], n_classes=10,
                                         feature_columns=feature_cols, config=conf
ig)
dnn_clf = tf.contrib.learn.SKCompat(dnn_clf)
dnn_clf.fit(X_train, y_train, batch_size=50, steps=40000)
```

```
WARNING:tensorflow:Using temporary folder as model directory: /tmp/tmpe69c9rh9
INFO:tensorflow:Using config: {'_evaluation_master': '', '_keep_checkpoint_eve
ry_n_hours': 10000, '_num_ps_replicas': 0, '_log_step_count_steps': 100, '_tf_
random_seed': 42, '_task_type': None, '_tf_config': gpu_options {
  per_process_gpu_memory_fraction: 1.0
}
, '_save_checkpoints_steps': None, '_environment': 'local', '_cluster_spec': <
tensorflow.python.training.server_lib.ClusterSpec object at 0x10abc1f98>, '_ma
ster': '', '_save_checkpoints_secs': 600, '_model_dir': '/tmp/tmpe69c9rh9', '_
task_id': 0, '_session_config': None, '_save_summary_steps': 100, '_keep_check
point_max': 5, '_is_chief': True, '_num_worker_replicas': 0}
WARNING:tensorflow:From /Users/mcewen/anaconda3/envs/tensorflow_py35/lib/pytho
n3.5/site-packages/tensorflow/contrib/learn/python/learn/estimators/dnn.py:19
2: get_global_step (from tensorflow.contrib.framework.python.ops.variables) is
deprecated and will be removed in a future version.
Instructions for updating:
Please switch to tf.train.get_global_step
INFO:tensorflow:Create CheckpointSaverHook.
INFO:tensorflow:Saving checkpoints for 1 into /tmp/tmpe69c9rh9/model.ckpt.
INFO:tensorflow:loss = 2.4005778, step = 1
INFO:tensorflow:global_step/sec: 264.728
INFO:tensorflow:loss = 0.31267586, step = 101 (0.379 sec)
INFO:tensorflow:global_step/sec: 234.107
INFO:tensorflow:loss = 0.29636884, step = 201 (0.427 sec)
INFO:tensorflow:global_step/sec: 220.965
```

```
INFO:tensorflow:loss = 0.4080768, step = 301 (0.452 sec)
INFO:tensorflow:global_step/sec: 275.559
INFO:tensorflow:loss = 0.23435771, step = 401 (0.363 sec)
INFO:tensorflow:global_step/sec: 230.905
INFO:tensorflow:loss = 0.2913079, step = 501 (0.433 sec)
INFO:tensorflow:global_step/sec: 162.635
INFO:tensorflow:loss = 0.07169643, step = 601 (0.615 sec)
INFO:tensorflow:global_step/sec: 236.966
INFO:tensorflow:loss = 0.14009716, step = 701 (0.422 sec)
INFO:tensorflow:global_step/sec: 239.115
INFO:tensorflow:loss = 0.19815028, step = 801 (0.418 sec)
INFO:tensorflow:global_step/sec: 246.523
INFO:tensorflow:loss = 0.11660824, step = 901 (0.406 sec)
INFO:tensorflow:global_step/sec: 273.111
INFO:tensorflow:loss = 0.24129894, step = 1001 (0.366 sec)
INFO:tensorflow:global_step/sec: 243.832
INFO:tensorflow:loss = 0.19023083, step = 1101 (0.410 sec)
INFO:tensorflow:global_step/sec: 282.785
INFO:tensorflow:loss = 0.14697056, step = 1201 (0.353 sec)
INFO:tensorflow:global_step/sec: 273.273
INFO:tensorflow:loss = 0.1890896, step = 1301 (0.366 sec)
INFO:tensorflow:global_step/sec: 275.319
INFO:tensorflow:loss = 0.06954378, step = 1401 (0.364 sec)
INFO:tensorflow:global_step/sec: 242.192
INFO:tensorflow:loss = 0.11753181, step = 1501 (0.413 sec)
INFO:tensorflow:global_step/sec: 276.011
INFO:tensorflow:loss = 0.09583752, step = 1601 (0.362 sec)
INFO:tensorflow:global_step/sec: 271.31
INFO:tensorflow:loss = 0.03823237, step = 1701 (0.369 sec)
INFO:tensorflow:global_step/sec: 258.147
INFO:tensorflow:loss = 0.15002182, step = 1801 (0.387 sec)
INFO:tensorflow:global_step/sec: 265.109
INFO:tensorflow:loss = 0.112414055, step = 1901 (0.377 sec)
INFO:tensorflow:global_step/sec: 269.362
INFO:tensorflow:loss = 0.11133062, step = 2001 (0.371 sec)
INFO:tensorflow:global_step/sec: 247.38
INFO:tensorflow:loss = 0.021828175, step = 2101 (0.404 sec)
INFO:tensorflow:global_step/sec: 256.207
```

```
INFO:tensorflow:loss = 0.022309918, step = 2201 (0.390 sec)
INFO:tensorflow:global_step/sec: 259.115
INFO:tensorflow:loss = 0.05583959, step = 2301 (0.386 sec)
INFO:tensorflow:global_step/sec: 267.849
INFO:tensorflow:loss = 0.052905086, step = 2401 (0.373 sec)
INFO:tensorflow:global_step/sec: 277.507
INFO:tensorflow:loss = 0.09949044, step = 2501 (0.360 sec)
INFO:tensorflow:global_step/sec: 287.202
INFO:tensorflow:loss = 0.05044488, step = 2601 (0.348 sec)
INFO:tensorflow:global_step/sec: 258.488
INFO:tensorflow:loss = 0.011002336, step = 2701 (0.387 sec)
INFO:tensorflow:global_step/sec: 256.5
INFO:tensorflow:loss = 0.06067799, step = 2801 (0.390 sec)
INFO:tensorflow:global_step/sec: 287.312
INFO:tensorflow:loss = 0.17532235, step = 2901 (0.348 sec)
INFO:tensorflow:global_step/sec: 268.433
INFO:tensorflow:loss = 0.016599359, step = 3001 (0.373 sec)
INFO:tensorflow:global_step/sec: 266.434
INFO:tensorflow:loss = 0.050229833, step = 3101 (0.375 sec)
INFO:tensorflow:global_step/sec: 282.718
INFO:tensorflow:loss = 0.009811628, step = 3201 (0.354 sec)
INFO:tensorflow:global_step/sec: 268.309
INFO:tensorflow:loss = 0.040984496, step = 3301 (0.373 sec)
INFO:tensorflow:global_step/sec: 264.772
INFO:tensorflow:loss = 0.22640564, step = 3401 (0.378 sec)
INFO:tensorflow:global_step/sec: 269.333
INFO:tensorflow:loss = 0.11032523, step = 3501 (0.371 sec)
INFO:tensorflow:global_step/sec: 288.665
INFO:tensorflow:loss = 0.18904746, step = 3601 (0.346 sec)
INFO:tensorflow:global_step/sec: 269.212
INFO:tensorflow:loss = 0.030300684, step = 3701 (0.372 sec)
INFO:tensorflow:global_step/sec: 288.321
INFO:tensorflow:loss = 0.009619746, step = 3801 (0.347 sec)
INFO:tensorflow:global_step/sec: 254.618
INFO:tensorflow:loss = 0.057061974, step = 3901 (0.393 sec)
INFO:tensorflow:global_step/sec: 265.451
INFO:tensorflow:loss = 0.11478825, step = 4001 (0.377 sec)
INFO:tensorflow:global_step/sec: 284.962
```

```
INFO:tensorflow:loss = 0.030539632, step = 4101 (0.351 sec)
INFO:tensorflow:global_step/sec: 284.729
INFO:tensorflow:loss = 0.054942384, step = 4201 (0.351 sec)
INFO:tensorflow:global_step/sec: 259.206
INFO:tensorflow:loss = 0.18744163, step = 4301 (0.386 sec)
INFO:tensorflow:global_step/sec: 266.298
INFO:tensorflow:loss = 0.1380551, step = 4401 (0.376 sec)
INFO:tensorflow:global_step/sec: 286.556
INFO:tensorflow:loss = 0.0155159235, step = 4501 (0.349 sec)
INFO:tensorflow:global_step/sec: 261.465
INFO:tensorflow:loss = 0.01754779, step = 4601 (0.382 sec)
INFO:tensorflow:global_step/sec: 286.819
INFO:tensorflow:loss = 0.009509798, step = 4701 (0.349 sec)
INFO:tensorflow:global_step/sec: 295.747
INFO:tensorflow:loss = 0.02467858, step = 4801 (0.338 sec)
INFO:tensorflow:global_step/sec: 293.854
INFO:tensorflow:loss = 0.10251294, step = 4901 (0.340 sec)
INFO:tensorflow:global_step/sec: 283.519
INFO:tensorflow:loss = 0.08789965, step = 5001 (0.353 sec)
INFO:tensorflow:global_step/sec: 274.044
INFO:tensorflow:loss = 0.010380711, step = 5101 (0.365 sec)
INFO:tensorflow:global_step/sec: 286.238
INFO:tensorflow:loss = 0.028989911, step = 5201 (0.349 sec)
INFO:tensorflow:global_step/sec: 297.931
INFO:tensorflow:loss = 0.026102195, step = 5301 (0.336 sec)
INFO:tensorflow:global_step/sec: 297.045
INFO:tensorflow:loss = 0.026997259, step = 5401 (0.337 sec)
INFO:tensorflow:global_step/sec: 270.202
INFO:tensorflow:loss = 0.018675137, step = 5501 (0.370 sec)
INFO:tensorflow:global_step/sec: 289.067
INFO:tensorflow:loss = 0.047280237, step = 5601 (0.346 sec)
INFO:tensorflow:global_step/sec: 288.248
INFO:tensorflow:loss = 0.008423564, step = 5701 (0.347 sec)
INFO:tensorflow:global_step/sec: 297.896
INFO:tensorflow:loss = 0.007797351, step = 5801 (0.336 sec)
INFO:tensorflow:global_step/sec: 265.633
INFO:tensorflow:loss = 0.06397591, step = 5901 (0.376 sec)
INFO:tensorflow:global_step/sec: 291.179
```

```
INFO:tensorflow:loss = 0.090881824, step = 6001 (0.343 sec)
INFO:tensorflow:global_step/sec: 282.704
INFO:tensorflow:loss = 0.014260262, step = 6101 (0.354 sec)
INFO:tensorflow:global_step/sec: 295.998
INFO:tensorflow:loss = 0.016228803, step = 6201 (0.338 sec)
INFO:tensorflow:global_step/sec: 288.335
INFO:tensorflow:loss = 0.048095588, step = 6301 (0.347 sec)
INFO:tensorflow:global_step/sec: 284.125
INFO:tensorflow:loss = 0.040203236, step = 6401 (0.352 sec)
INFO:tensorflow:global_step/sec: 294.58
INFO:tensorflow:loss = 0.012072418, step = 6501 (0.339 sec)
INFO:tensorflow:global_step/sec: 288.468
INFO:tensorflow:loss = 0.013664221, step = 6601 (0.347 sec)
INFO:tensorflow:global_step/sec: 250.1
INFO:tensorflow:loss = 0.01638243, step = 6701 (0.400 sec)
INFO:tensorflow:global_step/sec: 293.678
INFO:tensorflow:loss = 0.008239056, step = 6801 (0.340 sec)
INFO:tensorflow:global_step/sec: 295.132
INFO:tensorflow:loss = 0.012790847, step = 6901 (0.339 sec)
INFO:tensorflow:global_step/sec: 283.542
INFO:tensorflow:loss = 0.021387834, step = 7001 (0.353 sec)
INFO:tensorflow:global_step/sec: 295.925
INFO:tensorflow:loss = 0.008367192, step = 7101 (0.338 sec)
INFO:tensorflow:global_step/sec: 290.367
INFO:tensorflow:loss = 0.04883387, step = 7201 (0.344 sec)
INFO:tensorflow:global_step/sec: 298.618
INFO:tensorflow:loss = 0.009504036, step = 7301 (0.335 sec)
INFO:tensorflow:global_step/sec: 291.725
INFO:tensorflow:loss = 0.016978893, step = 7401 (0.343 sec)
INFO:tensorflow:global_step/sec: 286.606
INFO:tensorflow:loss = 0.010757222, step = 7501 (0.349 sec)
INFO:tensorflow:global_step/sec: 291.264
INFO:tensorflow:loss = 0.010587335, step = 7601 (0.343 sec)
INFO:tensorflow:global_step/sec: 293.598
INFO:tensorflow:loss = 0.005931136, step = 7701 (0.341 sec)
INFO:tensorflow:global_step/sec: 296.404
INFO:tensorflow:loss = 0.005197429, step = 7801 (0.337 sec)
INFO:tensorflow:global_step/sec: 293.652
```

```
INFO:tensorflow:loss = 0.006449755, step = 7901 (0.341 sec)
INFO:tensorflow:global_step/sec: 288.311
INFO:tensorflow:loss = 0.0020574287, step = 8001 (0.347 sec)
INFO:tensorflow:global_step/sec: 295.886
INFO:tensorflow:loss = 0.006619092, step = 8101 (0.338 sec)
INFO:tensorflow:global_step/sec: 295.087
INFO:tensorflow:loss = 0.03500336, step = 8201 (0.339 sec)
INFO:tensorflow:global_step/sec: 290.589
INFO:tensorflow:loss = 0.025590505, step = 8301 (0.344 sec)
INFO:tensorflow:global_step/sec: 292.153
INFO:tensorflow:loss = 0.005352935, step = 8401 (0.342 sec)
INFO:tensorflow:global_step/sec: 290.158
INFO:tensorflow:loss = 0.0057409974, step = 8501 (0.345 sec)
INFO:tensorflow:global_step/sec: 293.607
INFO:tensorflow:loss = 0.005307211, step = 8601 (0.340 sec)
INFO:tensorflow:global_step/sec: 292.729
INFO:tensorflow:loss = 0.0039272737, step = 8701 (0.342 sec)
INFO:tensorflow:global_step/sec: 291.595
INFO:tensorflow:loss = 0.008358784, step = 8801 (0.343 sec)
INFO:tensorflow:global_step/sec: 290.065
INFO:tensorflow:loss = 0.0027265372, step = 8901 (0.345 sec)
INFO:tensorflow:global_step/sec: 241.939
INFO:tensorflow:loss = 0.0041564466, step = 9001 (0.414 sec)
INFO:tensorflow:global_step/sec: 248.57
INFO:tensorflow:loss = 0.008153919, step = 9101 (0.402 sec)
INFO:tensorflow:global_step/sec: 277.052
INFO:tensorflow:loss = 0.00431604, step = 9201 (0.361 sec)
INFO:tensorflow:global_step/sec: 274.892
INFO:tensorflow:loss = 0.0053035887, step = 9301 (0.364 sec)
INFO:tensorflow:global_step/sec: 278.628
INFO:tensorflow:loss = 0.04991601, step = 9401 (0.359 sec)
INFO:tensorflow:global_step/sec: 229.064
INFO:tensorflow:loss = 0.003466385, step = 9501 (0.436 sec)
INFO:tensorflow:global_step/sec: 288.426
INFO:tensorflow:loss = 0.022571037, step = 9601 (0.347 sec)
INFO:tensorflow:global_step/sec: 263.924
INFO:tensorflow:loss = 0.009801387, step = 9701 (0.380 sec)
INFO:tensorflow:global_step/sec: 248.803
```

```
INFO:tensorflow:loss = 0.0025435286, step = 9801 (0.400 sec)
INFO:tensorflow:global_step/sec: 240.351
INFO:tensorflow:loss = 0.012849368, step = 9901 (0.416 sec)
INFO:tensorflow:global_step/sec: 244.567
INFO:tensorflow:loss = 0.0025271494, step = 10001 (0.409 sec)
INFO:tensorflow:global_step/sec: 245.434
INFO:tensorflow:loss = 0.0050374116, step = 10101 (0.408 sec)
INFO:tensorflow:global_step/sec: 232.404
INFO:tensorflow:loss = 0.009774216, step = 10201 (0.430 sec)
INFO:tensorflow:global_step/sec: 246.513
INFO:tensorflow:loss = 0.0021551389, step = 10301 (0.406 sec)
INFO:tensorflow:global_step/sec: 221.703
INFO:tensorflow:loss = 0.003974612, step = 10401 (0.450 sec)
INFO:tensorflow:global_step/sec: 231.503
INFO:tensorflow:loss = 0.0045803287, step = 10501 (0.432 sec)
INFO:tensorflow:global_step/sec: 244.281
INFO:tensorflow:loss = 0.010262251, step = 10601 (0.409 sec)
INFO:tensorflow:global_step/sec: 256.124
INFO:tensorflow:loss = 0.026278675, step = 10701 (0.390 sec)
INFO:tensorflow:global_step/sec: 269.944
INFO:tensorflow:loss = 0.004861756, step = 10801 (0.371 sec)
INFO:tensorflow:global_step/sec: 264.498
INFO:tensorflow:loss = 0.0015780836, step = 10901 (0.378 sec)
INFO:tensorflow:global_step/sec: 235.682
INFO:tensorflow:loss = 0.0211305, step = 11001 (0.424 sec)
INFO:tensorflow:global_step/sec: 227.749
INFO:tensorflow:loss = 0.0032920483, step = 11101 (0.439 sec)
INFO:tensorflow:global_step/sec: 243.491
INFO:tensorflow:loss = 0.0009200088, step = 11201 (0.411 sec)
INFO:tensorflow:global_step/sec: 251.464
INFO:tensorflow:loss = 0.0067240377, step = 11301 (0.398 sec)
INFO:tensorflow:global_step/sec: 225.963
INFO:tensorflow:loss = 0.0052556265, step = 11401 (0.443 sec)
INFO:tensorflow:global_step/sec: 204.695
INFO:tensorflow:loss = 0.014270798, step = 11501 (0.488 sec)
INFO:tensorflow:global_step/sec: 251.689
INFO:tensorflow:loss = 0.0010388009, step = 11601 (0.397 sec)
INFO:tensorflow:global_step/sec: 268.567
```

```
INFO:tensorflow:loss = 0.0033234973, step = 11701 (0.373 sec)
INFO:tensorflow:global_step/sec: 238.592
INFO:tensorflow:loss = 0.0006890038, step = 11801 (0.419 sec)
INFO:tensorflow:global_step/sec: 220.398
INFO:tensorflow:loss = 0.00561754, step = 11901 (0.454 sec)
INFO:tensorflow:global_step/sec: 202.065
INFO:tensorflow:loss = 0.00064186467, step = 12001 (0.494 sec)
INFO:tensorflow:global_step/sec: 252.02
INFO:tensorflow:loss = 0.0026009055, step = 12101 (0.396 sec)
INFO:tensorflow:global_step/sec: 263.789
INFO:tensorflow:loss = 0.0028248266, step = 12201 (0.379 sec)
INFO:tensorflow:global_step/sec: 250.232
INFO:tensorflow:loss = 0.005670745, step = 12301 (0.400 sec)
INFO:tensorflow:global_step/sec: 255.048
INFO:tensorflow:loss = 0.00037431132, step = 12401 (0.392 sec)
INFO:tensorflow:global_step/sec: 230.846
INFO:tensorflow:loss = 0.0009724629, step = 12501 (0.434 sec)
INFO:tensorflow:global_step/sec: 246.041
INFO:tensorflow:loss = 0.002971116, step = 12601 (0.406 sec)
INFO:tensorflow:global_step/sec: 238.799
INFO:tensorflow:loss = 0.0025933585, step = 12701 (0.419 sec)
INFO:tensorflow:global_step/sec: 244.936
INFO:tensorflow:loss = 0.013733688, step = 12801 (0.408 sec)
INFO:tensorflow:global_step/sec: 212.487
INFO:tensorflow:loss = 0.0031792028, step = 12901 (0.471 sec)
INFO:tensorflow:global_step/sec: 224.235
INFO:tensorflow:loss = 0.006552681, step = 13001 (0.446 sec)
INFO:tensorflow:global_step/sec: 210.939
INFO:tensorflow:loss = 0.0034216207, step = 13101 (0.474 sec)
INFO:tensorflow:global_step/sec: 244.237
INFO:tensorflow:loss = 0.0016324178, step = 13201 (0.409 sec)
INFO:tensorflow:global_step/sec: 243.963
INFO:tensorflow:loss = 0.011358909, step = 13301 (0.410 sec)
INFO:tensorflow:global_step/sec: 233.174
INFO:tensorflow:loss = 0.005848825, step = 13401 (0.430 sec)
INFO:tensorflow:global_step/sec: 218.895
INFO:tensorflow:loss = 0.0029444648, step = 13501 (0.457 sec)
INFO:tensorflow:global_step/sec: 239.313
```

```
INFO:tensorflow:loss = 0.0054142918, step = 13601 (0.417 sec)
INFO:tensorflow:global_step/sec: 215.349
INFO:tensorflow:loss = 0.0021757944, step = 13701 (0.464 sec)
INFO:tensorflow:global_step/sec: 259.738
INFO:tensorflow:loss = 0.0069082426, step = 13801 (0.386 sec)
INFO:tensorflow:global_step/sec: 225.309
INFO:tensorflow:loss = 0.0046300385, step = 13901 (0.443 sec)
INFO:tensorflow:global_step/sec: 274.909
INFO:tensorflow:loss = 0.0021858108, step = 14001 (0.364 sec)
INFO:tensorflow:global_step/sec: 253.946
INFO:tensorflow:loss = 0.011085386, step = 14101 (0.394 sec)
INFO:tensorflow:global_step/sec: 257.61
INFO:tensorflow:loss = 0.006658467, step = 14201 (0.388 sec)
INFO:tensorflow:global_step/sec: 265.824
INFO:tensorflow:loss = 0.0007703595, step = 14301 (0.376 sec)
INFO:tensorflow:global_step/sec: 197.294
INFO:tensorflow:loss = 0.0011539387, step = 14401 (0.508 sec)
INFO:tensorflow:global_step/sec: 227.607
INFO:tensorflow:loss = 0.0010545183, step = 14501 (0.439 sec)
INFO:tensorflow:global_step/sec: 216.513
INFO:tensorflow:loss = 0.0048172306, step = 14601 (0.463 sec)
INFO:tensorflow:global_step/sec: 196.147
INFO:tensorflow:loss = 0.0013384329, step = 14701 (0.509 sec)
INFO:tensorflow:global_step/sec: 261.452
INFO:tensorflow:loss = 0.0011707735, step = 14801 (0.382 sec)
INFO:tensorflow:global_step/sec: 263.525
INFO:tensorflow:loss = 0.002439394, step = 14901 (0.380 sec)
INFO:tensorflow:global_step/sec: 245.842
INFO:tensorflow:loss = 0.0016568727, step = 15001 (0.406 sec)
INFO:tensorflow:global_step/sec: 230.473
INFO:tensorflow:loss = 0.0023113445, step = 15101 (0.434 sec)
INFO:tensorflow:global_step/sec: 260.202
INFO:tensorflow:loss = 0.001015892, step = 15201 (0.384 sec)
INFO:tensorflow:global_step/sec: 264.948
INFO:tensorflow:loss = 0.0022939122, step = 15301 (0.377 sec)
INFO:tensorflow:global_step/sec: 186.205
INFO:tensorflow:loss = 0.0028404044, step = 15401 (0.537 sec)
INFO:tensorflow:global_step/sec: 227.866
```

```
INFO:tensorflow:loss = 0.005412121, step = 15501 (0.439 sec)
INFO:tensorflow:global_step/sec: 230.416
INFO:tensorflow:loss = 0.0035895878, step = 15601 (0.434 sec)
INFO:tensorflow:global_step/sec: 205.284
INFO:tensorflow:loss = 0.0058891536, step = 15701 (0.487 sec)
INFO:tensorflow:global_step/sec: 215.393
INFO:tensorflow:loss = 0.0008186471, step = 15801 (0.465 sec)
INFO:tensorflow:global_step/sec: 241.404
INFO:tensorflow:loss = 0.00085781404, step = 15901 (0.414 sec)
INFO:tensorflow:global_step/sec: 234.486
INFO:tensorflow:loss = 0.0065113665, step = 16001 (0.426 sec)
INFO:tensorflow:global_step/sec: 242.968
INFO:tensorflow:loss = 0.0030622215, step = 16101 (0.412 sec)
INFO:tensorflow:global_step/sec: 209.894
INFO:tensorflow:loss = 0.0001887983, step = 16201 (0.476 sec)
INFO:tensorflow:global_step/sec: 222.774
INFO:tensorflow:loss = 0.0025533608, step = 16301 (0.450 sec)
INFO:tensorflow:global_step/sec: 219.212
INFO:tensorflow:loss = 0.0011763193, step = 16401 (0.456 sec)
INFO:tensorflow:global_step/sec: 227.725
INFO:tensorflow:loss = 0.0017928132, step = 16501 (0.439 sec)
INFO:tensorflow:global_step/sec: 252.458
INFO:tensorflow:loss = 0.0024241805, step = 16601 (0.396 sec)
INFO:tensorflow:global_step/sec: 260.79
INFO:tensorflow:loss = 0.0024024954, step = 16701 (0.384 sec)
INFO:tensorflow:global_step/sec: 252.153
INFO:tensorflow:loss = 0.002722007, step = 16801 (0.397 sec)
INFO:tensorflow:global_step/sec: 170.35
INFO:tensorflow:loss = 0.001503015, step = 16901 (0.586 sec)
INFO:tensorflow:global_step/sec: 224.773
INFO:tensorflow:loss = 0.003879984, step = 17001 (0.445 sec)
INFO:tensorflow:global_step/sec: 241.66
INFO:tensorflow:loss = 0.0028237433, step = 17101 (0.414 sec)
INFO:tensorflow:global_step/sec: 237.813
INFO:tensorflow:loss = 0.0022598186, step = 17201 (0.421 sec)
INFO:tensorflow:global_step/sec: 255.464
INFO:tensorflow:loss = 0.0011070793, step = 17301 (0.391 sec)
INFO:tensorflow:global_step/sec: 233.826
```

```
INFO:tensorflow:loss = 0.0013992988, step = 17401 (0.428 sec)
INFO:tensorflow:global_step/sec: 220.61
INFO:tensorflow:loss = 0.0016118982, step = 17501 (0.464 sec)
INFO:tensorflow:global_step/sec: 176.551
INFO:tensorflow:loss = 0.0008287414, step = 17601 (0.556 sec)
INFO:tensorflow:global_step/sec: 254.058
INFO:tensorflow:loss = 0.0010201752, step = 17701 (0.394 sec)
INFO:tensorflow:global_step/sec: 242.954
INFO:tensorflow:loss = 0.00018981178, step = 17801 (0.413 sec)
INFO:tensorflow:global_step/sec: 178.664
INFO:tensorflow:loss = 0.0020044565, step = 17901 (0.560 sec)
INFO:tensorflow:global_step/sec: 198.222
INFO:tensorflow:loss = 0.00046436657, step = 18001 (0.504 sec)
INFO:tensorflow:global_step/sec: 172.62
INFO:tensorflow:loss = 0.00092771056, step = 18101 (0.580 sec)
INFO:tensorflow:global_step/sec: 167.684
INFO:tensorflow:loss = 0.0036528863, step = 18201 (0.596 sec)
INFO:tensorflow:global_step/sec: 144.41
INFO:tensorflow:loss = 0.005607925, step = 18301 (0.692 sec)
INFO:tensorflow:global_step/sec: 200.663
INFO:tensorflow:loss = 0.0026073065, step = 18401 (0.498 sec)
INFO:tensorflow:global_step/sec: 178.654
INFO:tensorflow:loss = 0.0052851615, step = 18501 (0.561 sec)
INFO:tensorflow:global_step/sec: 202.498
INFO:tensorflow:loss = 0.0038767403, step = 18601 (0.494 sec)
INFO:tensorflow:global_step/sec: 193.43
INFO:tensorflow:loss = 0.0017031265, step = 18701 (0.517 sec)
INFO:tensorflow:global_step/sec: 207.831
INFO:tensorflow:loss = 0.0020837628, step = 18801 (0.480 sec)
INFO:tensorflow:global_step/sec: 211.753
INFO:tensorflow:loss = 0.0026109123, step = 18901 (0.473 sec)
INFO:tensorflow:global_step/sec: 191.338
INFO:tensorflow:loss = 0.0014386168, step = 19001 (0.524 sec)
INFO:tensorflow:global_step/sec: 209.703
INFO:tensorflow:loss = 0.0005016105, step = 19101 (0.476 sec)
INFO:tensorflow:global_step/sec: 183.617
INFO:tensorflow:loss = 0.0011798079, step = 19201 (0.545 sec)
INFO:tensorflow:global_step/sec: 147.608
```

```
INFO:tensorflow:loss = 0.0056989426, step = 19301 (0.677 sec)
INFO:tensorflow:global_step/sec: 167.479
INFO:tensorflow:loss = 0.0014174287, step = 19401 (0.597 sec)
INFO:tensorflow:global_step/sec: 195.298
INFO:tensorflow:loss = 0.00181431, step = 19501 (0.511 sec)
INFO:tensorflow:global_step/sec: 263.274
INFO:tensorflow:loss = 0.00072336383, step = 19601 (0.380 sec)
INFO:tensorflow:global_step/sec: 264.501
INFO:tensorflow:loss = 9.478824e-05, step = 19701 (0.378 sec)
INFO:tensorflow:global_step/sec: 269.693
INFO:tensorflow:loss = 0.00046725423, step = 19801 (0.371 sec)
INFO:tensorflow:global_step/sec: 283.992
INFO:tensorflow:loss = 0.0019611202, step = 19901 (0.352 sec)
INFO:tensorflow:global_step/sec: 286.151
INFO:tensorflow:loss = 0.0025471402, step = 20001 (0.350 sec)
INFO:tensorflow:global_step/sec: 285.848
INFO:tensorflow:loss = 0.00043874516, step = 20101 (0.350 sec)
INFO:tensorflow:global_step/sec: 293.607
INFO:tensorflow:loss = 0.0025391192, step = 20201 (0.341 sec)
INFO:tensorflow:global_step/sec: 265.989
INFO:tensorflow:loss = 0.0010329548, step = 20301 (0.377 sec)
INFO:tensorflow:global_step/sec: 295.86
INFO:tensorflow:loss = 0.00016640488, step = 20401 (0.338 sec)
INFO:tensorflow:global_step/sec: 289.613
INFO:tensorflow:loss = 0.0009935172, step = 20501 (0.345 sec)
INFO:tensorflow:global_step/sec: 284.252
INFO:tensorflow:loss = 0.0008639615, step = 20601 (0.352 sec)
INFO:tensorflow:global_step/sec: 283.753
INFO:tensorflow:loss = 0.00084766647, step = 20701 (0.352 sec)
INFO:tensorflow:global_step/sec: 295.259
INFO:tensorflow:loss = 0.0010883757, step = 20801 (0.339 sec)
INFO:tensorflow:global_step/sec: 283.963
INFO:tensorflow:loss = 0.0015062927, step = 20901 (0.352 sec)
INFO:tensorflow:global_step/sec: 295.286
INFO:tensorflow:loss = 0.0022610624, step = 21001 (0.339 sec)
INFO:tensorflow:global_step/sec: 284.517
INFO:tensorflow:loss = 0.0010078625, step = 21101 (0.352 sec)
INFO:tensorflow:global_step/sec: 283.938
```

```
INFO:tensorflow:loss = 0.00453955, step = 21201 (0.352 sec)
INFO:tensorflow:global_step/sec: 287.58
INFO:tensorflow:loss = 0.0005150675, step = 21301 (0.348 sec)
INFO:tensorflow:global_step/sec: 288.968
INFO:tensorflow:loss = 0.002108911, step = 21401 (0.346 sec)
INFO:tensorflow:global_step/sec: 285.26
INFO:tensorflow:loss = 0.0003527573, step = 21501 (0.350 sec)
INFO:tensorflow:global_step/sec: 292.023
INFO:tensorflow:loss = 0.0007121074, step = 21601 (0.343 sec)
INFO:tensorflow:global_step/sec: 283.626
INFO:tensorflow:loss = 0.0009194807, step = 21701 (0.353 sec)
INFO:tensorflow:global_step/sec: 285.981
INFO:tensorflow:loss = 0.000383486, step = 21801 (0.349 sec)
INFO:tensorflow:global_step/sec: 292.029
INFO:tensorflow:loss = 0.0004972108, step = 21901 (0.342 sec)
INFO:tensorflow:global_step/sec: 284.274
INFO:tensorflow:loss = 6.2377825e-05, step = 22001 (0.352 sec)
INFO:tensorflow:global_step/sec: 277.149
INFO:tensorflow:loss = 0.0002139326, step = 22101 (0.361 sec)
INFO:tensorflow:global_step/sec: 286.635
INFO:tensorflow:loss = 0.0011587589, step = 22201 (0.349 sec)
INFO:tensorflow:global_step/sec: 294.074
INFO:tensorflow:loss = 0.0015314096, step = 22301 (0.340 sec)
INFO:tensorflow:global_step/sec: 285.986
INFO:tensorflow:loss = 0.0019228004, step = 22401 (0.350 sec)
INFO:tensorflow:global_step/sec: 287.245
INFO:tensorflow:loss = 0.0013467915, step = 22501 (0.348 sec)
INFO:tensorflow:global_step/sec: 288.158
INFO:tensorflow:loss = 0.0023154982, step = 22601 (0.347 sec)
INFO:tensorflow:global_step/sec: 288.681
INFO:tensorflow:loss = 0.00090699946, step = 22701 (0.346 sec)
INFO:tensorflow:global_step/sec: 296.74
INFO:tensorflow:loss = 0.0008416923, step = 22801 (0.337 sec)
INFO:tensorflow:global_step/sec: 290.268
INFO:tensorflow:loss = 0.0020299358, step = 22901 (0.345 sec)
INFO:tensorflow:global_step/sec: 285.795
INFO:tensorflow:loss = 0.00051947305, step = 23001 (0.350 sec)
INFO:tensorflow:global_step/sec: 297.033
```

```
INFO:tensorflow:loss = 0.0023677696, step = 23101 (0.336 sec)
INFO:tensorflow:global_step/sec: 288.996
INFO:tensorflow:loss = 0.0018610213, step = 23201 (0.346 sec)
INFO:tensorflow:global_step/sec: 291.823
INFO:tensorflow:loss = 0.0018711936, step = 23301 (0.343 sec)
INFO:tensorflow:global_step/sec: 293.561
INFO:tensorflow:loss = 0.0005051533, step = 23401 (0.340 sec)
INFO:tensorflow:global_step/sec: 290.037
INFO:tensorflow:loss = 0.00076665386, step = 23501 (0.345 sec)
INFO:tensorflow:global_step/sec: 292.423
INFO:tensorflow:loss = 0.000556651, step = 23601 (0.342 sec)
INFO:tensorflow:global_step/sec: 294.917
INFO:tensorflow:loss = 0.0001052184, step = 23701 (0.339 sec)
INFO:tensorflow:global_step/sec: 293.751
INFO:tensorflow:loss = 0.00091980194, step = 23801 (0.340 sec)
INFO:tensorflow:global_step/sec: 294.418
INFO:tensorflow:loss = 0.0017980053, step = 23901 (0.340 sec)
INFO:tensorflow:global_step/sec: 295.052
INFO:tensorflow:loss = 0.0014662343, step = 24001 (0.339 sec)
INFO:tensorflow:global_step/sec: 294.872
INFO:tensorflow:loss = 0.0006457205, step = 24101 (0.339 sec)
INFO:tensorflow:global_step/sec: 291.595
INFO:tensorflow:loss = 0.0017085895, step = 24201 (0.343 sec)
INFO:tensorflow:global_step/sec: 291.751
INFO:tensorflow:loss = 0.00019012339, step = 24301 (0.343 sec)
INFO:tensorflow:global_step/sec: 293.751
INFO:tensorflow:loss = 0.0018188378, step = 24401 (0.340 sec)
INFO:tensorflow:global_step/sec: 290.217
INFO:tensorflow:loss = 0.0012226698, step = 24501 (0.345 sec)
INFO:tensorflow:global_step/sec: 289.85
INFO:tensorflow:loss = 0.00072973536, step = 24601 (0.345 sec)
INFO:tensorflow:global_step/sec: 296.724
INFO:tensorflow:loss = 0.0011833684, step = 24701 (0.337 sec)
INFO:tensorflow:global_step/sec: 298.927
INFO:tensorflow:loss = 0.0011878891, step = 24801 (0.334 sec)
INFO:tensorflow:global_step/sec: 291.026
INFO:tensorflow:loss = 0.0013818216, step = 24901 (0.344 sec)
INFO:tensorflow:global_step/sec: 290.64
```

```
INFO:tensorflow:loss = 0.00048778497, step = 25001 (0.344 sec)
INFO:tensorflow:global_step/sec: 290.145
INFO:tensorflow:loss = 0.00084742083, step = 25101 (0.344 sec)
INFO:tensorflow:global_step/sec: 278.89
INFO:tensorflow:loss = 0.0016339379, step = 25201 (0.359 sec)
INFO:tensorflow:global_step/sec: 282.105
INFO:tensorflow:loss = 0.00015762541, step = 25301 (0.354 sec)
INFO:tensorflow:global_step/sec: 284.355
INFO:tensorflow:loss = 0.00057926326, step = 25401 (0.352 sec)
INFO:tensorflow:global_step/sec: 283.233
INFO:tensorflow:loss = 0.00096197583, step = 25501 (0.353 sec)
INFO:tensorflow:global_step/sec: 277.772
INFO:tensorflow:loss = 0.00072867685, step = 25601 (0.360 sec)
INFO:tensorflow:global_step/sec: 280.53
INFO:tensorflow:loss = 0.0003659475, step = 25701 (0.356 sec)
INFO:tensorflow:global_step/sec: 286.732
INFO:tensorflow:loss = 0.0006396459, step = 25801 (0.349 sec)
INFO:tensorflow:global_step/sec: 296.271
INFO:tensorflow:loss = 0.0014290272, step = 25901 (0.338 sec)
INFO:tensorflow:global_step/sec: 295.946
INFO:tensorflow:loss = 0.00010238056, step = 26001 (0.338 sec)
INFO:tensorflow:global_step/sec: 290.226
INFO:tensorflow:loss = 0.0007521265, step = 26101 (0.345 sec)
INFO:tensorflow:global_step/sec: 291.662
INFO:tensorflow:loss = 0.0011646885, step = 26201 (0.343 sec)
INFO:tensorflow:global_step/sec: 291.21
INFO:tensorflow:loss = 0.0007397069, step = 26301 (0.343 sec)
INFO:tensorflow:global_step/sec: 296.281
INFO:tensorflow:loss = 0.0011290793, step = 26401 (0.338 sec)
INFO:tensorflow:global_step/sec: 288.817
INFO:tensorflow:loss = 0.00079500815, step = 26501 (0.346 sec)
INFO:tensorflow:global_step/sec: 291.029
INFO:tensorflow:loss = 0.00047347747, step = 26601 (0.344 sec)
INFO:tensorflow:global_step/sec: 288.322
INFO:tensorflow:loss = 3.1328407e-05, step = 26701 (0.347 sec)
INFO:tensorflow:global_step/sec: 290.106
INFO:tensorflow:loss = 0.00035584634, step = 26801 (0.345 sec)
INFO:tensorflow:global_step/sec: 296.826
```

```
INFO:tensorflow:loss = 0.0007996074, step = 26901 (0.337 sec)
INFO:tensorflow:global_step/sec: 293.92
INFO:tensorflow:loss = 0.00077521073, step = 27001 (0.340 sec)
INFO:tensorflow:global_step/sec: 293.024
INFO:tensorflow:loss = 0.00047619123, step = 27101 (0.341 sec)
INFO:tensorflow:global_step/sec: 281.852
INFO:tensorflow:loss = 0.00039861762, step = 27201 (0.355 sec)
INFO:tensorflow:global_step/sec: 264.348
INFO:tensorflow:loss = 0.0011550712, step = 27301 (0.378 sec)
INFO:tensorflow:global_step/sec: 250.965
INFO:tensorflow:loss = 0.0005360307, step = 27401 (0.398 sec)
INFO:tensorflow:global_step/sec: 282.026
INFO:tensorflow:loss = 0.0010285559, step = 27501 (0.355 sec)
INFO:tensorflow:global_step/sec: 261.956
INFO:tensorflow:loss = 0.0011170513, step = 27601 (0.382 sec)
INFO:tensorflow:global_step/sec: 281.406
INFO:tensorflow:loss = 0.00024562303, step = 27701 (0.355 sec)
INFO:tensorflow:global_step/sec: 275.104
INFO:tensorflow:loss = 0.00017662946, step = 27801 (0.363 sec)
INFO:tensorflow:global_step/sec: 281.947
INFO:tensorflow:loss = 0.00060662144, step = 27901 (0.355 sec)
INFO:tensorflow:global_step/sec: 230.119
INFO:tensorflow:loss = 0.0014573914, step = 28001 (0.435 sec)
INFO:tensorflow:global_step/sec: 222.396
INFO:tensorflow:loss = 0.00066597166, step = 28101 (0.451 sec)
INFO:tensorflow:global_step/sec: 255.755
INFO:tensorflow:loss = 0.0007559935, step = 28201 (0.390 sec)
INFO:tensorflow:global_step/sec: 291.408
INFO:tensorflow:loss = 0.00081949023, step = 28301 (0.343 sec)
INFO:tensorflow:global_step/sec: 251.939
INFO:tensorflow:loss = 0.0012622013, step = 28401 (0.397 sec)
INFO:tensorflow:global_step/sec: 263.2
INFO:tensorflow:loss = 0.00015271838, step = 28501 (0.380 sec)
INFO:tensorflow:global_step/sec: 270.376
INFO:tensorflow:loss = 0.0005445875, step = 28601 (0.370 sec)
INFO:tensorflow:global_step/sec: 248.966
INFO:tensorflow:loss = 0.001242336, step = 28701 (0.402 sec)
INFO:tensorflow:global_step/sec: 278.546
```

```
INFO:tensorflow:loss = 0.0011522959, step = 28801 (0.359 sec)
INFO:tensorflow:global_step/sec: 277.429
INFO:tensorflow:loss = 0.00029501267, step = 28901 (0.361 sec)
INFO:tensorflow:global_step/sec: 258.641
INFO:tensorflow:loss = 0.0012439629, step = 29001 (0.386 sec)
INFO:tensorflow:global_step/sec: 271.343
INFO:tensorflow:loss = 0.0011318299, step = 29101 (0.369 sec)
INFO:tensorflow:global_step/sec: 272.929
INFO:tensorflow:loss = 0.0011576376, step = 29201 (0.366 sec)
INFO:tensorflow:global_step/sec: 282.675
INFO:tensorflow:loss = 0.0012464767, step = 29301 (0.354 sec)
INFO:tensorflow:global_step/sec: 213.543
INFO:tensorflow:loss = 0.0011650472, step = 29401 (0.469 sec)
INFO:tensorflow:global_step/sec: 228.147
INFO:tensorflow:loss = 0.00089854986, step = 29501 (0.437 sec)
INFO:tensorflow:global_step/sec: 270.826
INFO:tensorflow:loss = 0.0003493056, step = 29601 (0.369 sec)
INFO:tensorflow:global_step/sec: 284.395
INFO:tensorflow:loss = 0.0001967982, step = 29701 (0.351 sec)
INFO:tensorflow:global_step/sec: 294.581
INFO:tensorflow:loss = 0.00042677126, step = 29801 (0.339 sec)
INFO:tensorflow:global_step/sec: 267.014
INFO:tensorflow:loss = 0.00096111797, step = 29901 (0.375 sec)
INFO:tensorflow:global_step/sec: 280.362
INFO:tensorflow:loss = 0.00018180076, step = 30001 (0.357 sec)
INFO:tensorflow:global_step/sec: 293.301
INFO:tensorflow:loss = 0.0005855758, step = 30101 (0.341 sec)
INFO:tensorflow:global_step/sec: 292.846
INFO:tensorflow:loss = 0.00021347596, step = 30201 (0.342 sec)
INFO:tensorflow:global_step/sec: 282.455
INFO:tensorflow:loss = 0.00061550434, step = 30301 (0.354 sec)
INFO:tensorflow:global_step/sec: 286.051
INFO:tensorflow:loss = 0.0015167601, step = 30401 (0.349 sec)
INFO:tensorflow:global_step/sec: 291.989
INFO:tensorflow:loss = 0.0007232324, step = 30501 (0.343 sec)
INFO:tensorflow:global_step/sec: 297.897
INFO:tensorflow:loss = 0.00081983523, step = 30601 (0.336 sec)
INFO:tensorflow:global_step/sec: 283.147
```

```
INFO:tensorflow:loss = 0.0006647414, step = 30701 (0.353 sec)
INFO:tensorflow:global_step/sec: 293.197
INFO:tensorflow:loss = 0.0010431712, step = 30801 (0.341 sec)
INFO:tensorflow:global_step/sec: 286.485
INFO:tensorflow:loss = 0.0005624868, step = 30901 (0.349 sec)
INFO:tensorflow:global_step/sec: 287.587
INFO:tensorflow:loss = 0.00092834665, step = 31001 (0.348 sec)
INFO:tensorflow:global_step/sec: 276.7
INFO:tensorflow:loss = 0.0011234849, step = 31101 (0.361 sec)
INFO:tensorflow:global_step/sec: 290.911
INFO:tensorflow:loss = 0.00031862652, step = 31201 (0.344 sec)
INFO:tensorflow:global_step/sec: 293.03
INFO:tensorflow:loss = 0.0004291339, step = 31301 (0.341 sec)
INFO:tensorflow:global_step/sec: 291.187
INFO:tensorflow:loss = 0.0013877174, step = 31401 (0.343 sec)
INFO:tensorflow:global_step/sec: 297.709
INFO:tensorflow:loss = 0.00020001482, step = 31501 (0.336 sec)
INFO:tensorflow:global_step/sec: 299.657
INFO:tensorflow:loss = 0.00032674294, step = 31601 (0.334 sec)
INFO:tensorflow:global_step/sec: 275.276
INFO:tensorflow:loss = 0.0006365755, step = 31701 (0.364 sec)
INFO:tensorflow:global_step/sec: 240.812
INFO:tensorflow:loss = 0.00016916303, step = 31801 (0.415 sec)
INFO:tensorflow:global_step/sec: 290.958
INFO:tensorflow:loss = 0.0006920603, step = 31901 (0.344 sec)
INFO:tensorflow:global_step/sec: 295.674
INFO:tensorflow:loss = 0.00012926888, step = 32001 (0.338 sec)
INFO:tensorflow:global_step/sec: 284.034
INFO:tensorflow:loss = 0.00039280686, step = 32101 (0.352 sec)
INFO:tensorflow:global_step/sec: 289.241
INFO:tensorflow:loss = 0.0011558913, step = 32201 (0.345 sec)
INFO:tensorflow:global_step/sec: 300.043
INFO:tensorflow:loss = 0.00056561973, step = 32301 (0.333 sec)
INFO:tensorflow:global_step/sec: 289.739
INFO:tensorflow:loss = 0.00017675917, step = 32401 (0.345 sec)
INFO:tensorflow:global_step/sec: 289.39
INFO:tensorflow:loss = 0.00040773995, step = 32501 (0.346 sec)
INFO:tensorflow:global_step/sec: 290.326
```

```
INFO:tensorflow:loss = 0.00021993961, step = 32601 (0.344 sec)
INFO:tensorflow:global_step/sec: 291.508
INFO:tensorflow:loss = 0.00079787505, step = 32701 (0.343 sec)
INFO:tensorflow:global_step/sec: 296.095
INFO:tensorflow:loss = 0.00090279884, step = 32801 (0.338 sec)
INFO:tensorflow:global_step/sec: 293.661
INFO:tensorflow:loss = 0.0005482138, step = 32901 (0.341 sec)
INFO:tensorflow:global_step/sec: 290.595
INFO:tensorflow:loss = 0.0006554589, step = 33001 (0.344 sec)
INFO:tensorflow:global_step/sec: 288.215
INFO:tensorflow:loss = 0.0018582876, step = 33101 (0.347 sec)
INFO:tensorflow:global_step/sec: 294.181
INFO:tensorflow:loss = 0.00076735055, step = 33201 (0.340 sec)
INFO:tensorflow:global_step/sec: 292.27
INFO:tensorflow:loss = 0.00038163952, step = 33301 (0.342 sec)
INFO:tensorflow:global_step/sec: 286.823
INFO:tensorflow:loss = 0.001371027, step = 33401 (0.349 sec)
INFO:tensorflow:global_step/sec: 285.081
INFO:tensorflow:loss = 0.00029575542, step = 33501 (0.351 sec)
INFO:tensorflow:global_step/sec: 280.163
INFO:tensorflow:loss = 0.00088878634, step = 33601 (0.357 sec)
INFO:tensorflow:global_step/sec: 297.187
INFO:tensorflow:loss = 0.0009937927, step = 33701 (0.336 sec)
INFO:tensorflow:global_step/sec: 297.895
INFO:tensorflow:loss = 0.0005908036, step = 33801 (0.336 sec)
INFO:tensorflow:global_step/sec: 289.873
INFO:tensorflow:loss = 0.0009054421, step = 33901 (0.345 sec)
INFO:tensorflow:global_step/sec: 284.707
INFO:tensorflow:loss = 0.00071158016, step = 34001 (0.351 sec)
INFO:tensorflow:global_step/sec: 288.914
INFO:tensorflow:loss = 0.00053563213, step = 34101 (0.346 sec)
INFO:tensorflow:global_step/sec: 293.284
INFO:tensorflow:loss = 0.0011523266, step = 34201 (0.341 sec)
INFO:tensorflow:global_step/sec: 289.047
INFO:tensorflow:loss = 0.00016720987, step = 34301 (0.346 sec)
INFO:tensorflow:global_step/sec: 290.33
INFO:tensorflow:loss = 0.0004537905, step = 34401 (0.345 sec)
INFO:tensorflow:global_step/sec: 292.24
```

```
INFO:tensorflow:loss = 0.00044780105, step = 34501 (0.342 sec)
INFO:tensorflow:global_step/sec: 288.94
INFO:tensorflow:loss = 0.00041601766, step = 34601 (0.346 sec)
INFO:tensorflow:global_step/sec: 296.694
INFO:tensorflow:loss = 0.0011050355, step = 34701 (0.337 sec)
INFO:tensorflow:global_step/sec: 274.09
INFO:tensorflow:loss = 0.00052697194, step = 34801 (0.365 sec)
INFO:tensorflow:global_step/sec: 185.778
INFO:tensorflow:loss = 0.00071083516, step = 34901 (0.540 sec)
INFO:tensorflow:global_step/sec: 246.85
INFO:tensorflow:loss = 0.0002585916, step = 35001 (0.403 sec)
INFO:tensorflow:global_step/sec: 260.571
INFO:tensorflow:loss = 0.0008163878, step = 35101 (0.383 sec)
INFO:tensorflow:global_step/sec: 190.92
INFO:tensorflow:loss = 0.00018969526, step = 35201 (0.525 sec)
INFO:tensorflow:global_step/sec: 193.444
INFO:tensorflow:loss = 0.0001349323, step = 35301 (0.517 sec)
INFO:tensorflow:global_step/sec: 184.108
INFO:tensorflow:loss = 0.0010902109, step = 35401 (0.542 sec)
INFO:tensorflow:global_step/sec: 203.988
INFO:tensorflow:loss = 0.00035070805, step = 35501 (0.490 sec)
INFO:tensorflow:global_step/sec: 225.803
INFO:tensorflow:loss = 0.0002819493, step = 35601 (0.443 sec)
INFO:tensorflow:global_step/sec: 247.427
INFO:tensorflow:loss = 0.0008391325, step = 35701 (0.403 sec)
INFO:tensorflow:global_step/sec: 294.864
INFO:tensorflow:loss = 0.000632369, step = 35801 (0.339 sec)
INFO:tensorflow:global_step/sec: 289.384
INFO:tensorflow:loss = 0.00037967376, step = 35901 (0.346 sec)
INFO:tensorflow:global_step/sec: 277.089
INFO:tensorflow:loss = 0.0003320363, step = 36001 (0.361 sec)
INFO:tensorflow:global_step/sec: 290.439
INFO:tensorflow:loss = 0.0011039884, step = 36101 (0.344 sec)
INFO:tensorflow:global_step/sec: 277.872
INFO:tensorflow:loss = 0.0008606437, step = 36201 (0.360 sec)
INFO:tensorflow:global_step/sec: 297.916
INFO:tensorflow:loss = 0.0002927651, step = 36301 (0.336 sec)
INFO:tensorflow:global_step/sec: 293.495
```

```
INFO:tensorflow:loss = 0.00069891097, step = 36401 (0.341 sec)
INFO:tensorflow:global_step/sec: 279.758
INFO:tensorflow:loss = 0.00015776746, step = 36501 (0.358 sec)
INFO:tensorflow:global_step/sec: 271.591
INFO:tensorflow:loss = 0.00033270434, step = 36601 (0.368 sec)
INFO:tensorflow:global_step/sec: 282.735
INFO:tensorflow:loss = 0.00028696892, step = 36701 (0.354 sec)
INFO:tensorflow:global_step/sec: 263.932
INFO:tensorflow:loss = 0.0008025733, step = 36801 (0.379 sec)
INFO:tensorflow:global_step/sec: 284.206
INFO:tensorflow:loss = 0.0005028858, step = 36901 (0.352 sec)
INFO:tensorflow:global_step/sec: 264.347
INFO:tensorflow:loss = 0.00062359317, step = 37001 (0.378 sec)
INFO:tensorflow:global_step/sec: 286.358
INFO:tensorflow:loss = 0.00019955178, step = 37101 (0.349 sec)
INFO:tensorflow:global_step/sec: 296.903
INFO:tensorflow:loss = 0.00016318852, step = 37201 (0.337 sec)
INFO:tensorflow:global_step/sec: 289.063
INFO:tensorflow:loss = 0.0006694493, step = 37301 (0.346 sec)
INFO:tensorflow:global_step/sec: 246.168
INFO:tensorflow:loss = 0.0003708102, step = 37401 (0.406 sec)
INFO:tensorflow:global_step/sec: 245.622
INFO:tensorflow:loss = 0.00025077042, step = 37501 (0.407 sec)
INFO:tensorflow:global_step/sec: 245.825
INFO:tensorflow:loss = 0.000264987, step = 37601 (0.407 sec)
INFO:tensorflow:global_step/sec: 261.187
INFO:tensorflow:loss = 0.00010086442, step = 37701 (0.383 sec)
INFO:tensorflow:global_step/sec: 205.151
INFO:tensorflow:loss = 0.0006153757, step = 37801 (0.488 sec)
INFO:tensorflow:global_step/sec: 224.999
INFO:tensorflow:loss = 0.0011952921, step = 37901 (0.444 sec)
INFO:tensorflow:global_step/sec: 251.428
INFO:tensorflow:loss = 0.00019914961, step = 38001 (0.398 sec)
INFO:tensorflow:global_step/sec: 251.182
INFO:tensorflow:loss = 0.0008874108, step = 38101 (0.398 sec)
INFO:tensorflow:global_step/sec: 246.062
INFO:tensorflow:loss = 0.00077106326, step = 38201 (0.406 sec)
INFO:tensorflow:global_step/sec: 247.689
```

```
INFO:tensorflow:loss = 5.9410097e-05, step = 38301 (0.404 sec)
INFO:tensorflow:global_step/sec: 244.082
INFO:tensorflow:loss = 0.000106180356, step = 38401 (0.410 sec)
INFO:tensorflow:global_step/sec: 238.792
INFO:tensorflow:loss = 0.0004074581, step = 38501 (0.418 sec)
INFO:tensorflow:global_step/sec: 237.921
INFO:tensorflow:loss = 0.0006691146, step = 38601 (0.422 sec)
INFO:tensorflow:global_step/sec: 230.154
INFO:tensorflow:loss = 0.00045461592, step = 38701 (0.433 sec)
INFO:tensorflow:global_step/sec: 223.848
INFO:tensorflow:loss = 0.00013856331, step = 38801 (0.447 sec)
INFO:tensorflow:global_step/sec: 222.868
INFO:tensorflow:loss = 0.0014249012, step = 38901 (0.449 sec)
INFO:tensorflow:global_step/sec: 237.969
INFO:tensorflow:loss = 0.0001628006, step = 39001 (0.420 sec)
INFO:tensorflow:global_step/sec: 245.218
INFO:tensorflow:loss = 0.0007085502, step = 39101 (0.408 sec)
INFO:tensorflow:global_step/sec: 250.064
INFO:tensorflow:loss = 0.00033850258, step = 39201 (0.400 sec)
INFO:tensorflow:global_step/sec: 225.221
INFO:tensorflow:loss = 0.00038915616, step = 39301 (0.444 sec)
INFO:tensorflow:global_step/sec: 247.8
INFO:tensorflow:loss = 0.0003968734, step = 39401 (0.404 sec)
INFO:tensorflow:global_step/sec: 261.599
INFO:tensorflow:loss = 0.00015025295, step = 39501 (0.382 sec)
INFO:tensorflow:global_step/sec: 245.64
INFO:tensorflow:loss = 0.00062390114, step = 39601 (0.408 sec)
INFO:tensorflow:global_step/sec: 230.621
INFO:tensorflow:loss = 0.00014465627, step = 39701 (0.433 sec)
INFO:tensorflow:global_step/sec: 216.937
INFO:tensorflow:loss = 0.0011388173, step = 39801 (0.461 sec)
INFO:tensorflow:global_step/sec: 232.34
INFO:tensorflow:loss = 0.0007998731, step = 39901 (0.430 sec)
INFO:tensorflow:Saving checkpoints for 40000 into /tmp/tmpe69c9rh9/model.ckpt.
INFO:tensorflow:Loss for final step: 0.00044029002.
```

Out[43]:
```
SKCompat()
```

In [44]:
```
from sklearn.metrics import accuracy_score

y_pred = dnn_clf.predict(X_test)
accuracy_score(y_test, y_pred['classes'])
```

```
INFO:tensorflow:Restoring parameters from /tmp/tmpe69c9rh9/model.ckpt-40000
```

Out[44]:
```
0.9835
```

Better accuracy than we achieved with Scitkit-Learn!

Using plain TensorFlow

The standard API, as we've focused on previously, provides much more control in constructing and training the network architecture.

Construction of the computational graph

In [45]:
```
import tensorflow as tf

n_inputs = 28*28  # MNIST
n_hidden1 = 300
n_hidden2 = 100
n_outputs = 10
```

In [46]:
```
reset_graph()

X = tf.placeholder(tf.float32, shape=(None, n_inputs), name="X")
y = tf.placeholder(tf.int64, shape=(None), name="y")
```

Each layer in the network is similar so let's define a general layer that we can reuse.

In [47]:
```
def neuron_layer(X, n_neurons, name, activation=None):
    with tf.name_scope(name):
        n_inputs = int(X.get_shape()[1])
        stddev = 2 / np.sqrt(n_inputs) # More on this in next lecture
        init = tf.truncated_normal((n_inputs, n_neurons), stddev=stddev)
        W = tf.Variable(init, name="kernel")
        b = tf.Variable(tf.zeros([n_neurons]), name="bias")
        Z = tf.matmul(X, W) + b
        if activation is not None:
            return activation(Z)
        else:
            return Z
```

Each layer in the network is similar so let's define a general layer that we can reuse.

In [47]:
```
def neuron_layer(X, n_neurons, name, activation=None):
    with tf.name_scope(name):
        n_inputs = int(X.get_shape()[1])
        stddev = 2 / np.sqrt(n_inputs) # More on this in next lecture
        init = tf.truncated_normal((n_inputs, n_neurons), stddev=stddev)
        W = tf.Variable(init, name="kernel")
        b = tf.Variable(tf.zeros([n_neurons]), name="bias")
        Z = tf.matmul(X, W) + b
        if activation is not None:
            return activation(Z)
        else:
            return Z
```

In practice, TensorFlow contains many built-in functions so it's generally not necessary to define layers like this (see, e.g., `tf.layers.dense`).

Now let's construct 3 layers.

In [48]:
```
with tf.name_scope("dnn"):
    hidden1 = neuron_layer(X, n_hidden1, name="hidden1",
                           activation=tf.nn.relu)
    hidden2 = neuron_layer(hidden1, n_hidden2, name="hidden2",
                           activation=tf.nn.relu)
    logits = neuron_layer(hidden2, n_outputs, name="outputs")
```

Train using cross-entropy, where softmax applied when defining cross-entropy rather than in network construction.

In [49]:
```
with tf.name_scope("loss"):
    xentropy = tf.nn.sparse_softmax_cross_entropy_with_logits(labels=y,
                                                              logits=logits)
    loss = tf.reduce_mean(xentropy, name="loss")
```

Train using cross-entropy, where softmax applied when defining cross-entropy rather than in network construction.

In [49]:
```
with tf.name_scope("loss"):
    xentropy = tf.nn.sparse_softmax_cross_entropy_with_logits(labels=y,
                                                              logits=logits)
    loss = tf.reduce_mean(xentropy, name="loss")
```

Define gradient descent optimizer.

In [50]:
```
learning_rate = 0.01

with tf.name_scope("train"):
    optimizer = tf.train.GradientDescentOptimizer(learning_rate)
    training_op = optimizer.minimize(loss)
```

Define nodes to evaluate accuracy.

The function `in_top_k` (https://www.tensorflow.org/api_docs/python/tf/nn/in_top_k) checks whether the targets (`y`) are in the top k predictions (`logits`).

In [51]:
```
with tf.name_scope("eval"):
    correct = tf.nn.in_top_k(logits, y, 1)
    accuracy = tf.reduce_mean(tf.cast(correct, tf.float32))
```

Define nodes to evaluate accuracy.

The function `in_top_k` (https://www.tensorflow.org/api_docs/python/tf/nn/in_top_k) checks whether the targets (`y`) are in the top k predictions (`logits`).

In [51]:
```
with tf.name_scope("eval"):
    correct = tf.nn.in_top_k(logits, y, 1)
    accuracy = tf.reduce_mean(tf.cast(correct, tf.float32))
```

Finally, define initializer.

In [52]:
```
init = tf.global_variables_initializer()
saver = tf.train.Saver()
```

Execution (training)

In [53]:
```
n_epochs = 10
batch_size = 50
```

In [54]:
```
with tf.Session() as sess:
    init.run()
    for epoch in range(n_epochs):
        for iteration in range(mnist.train.num_examples // batch_size):
            X_batch, y_batch = mnist.train.next_batch(batch_size)
            sess.run(training_op, feed_dict={X: X_batch, y: y_batch})
        acc_train = accuracy.eval(feed_dict={X: X_batch, y: y_batch})
        acc_val = accuracy.eval(feed_dict={X: mnist.validation.images,
                                           y: mnist.validation.labels})
        print(epoch, "Train accuracy:", acc_train, "Val accuracy:", acc_val)

    save_path = saver.save(sess, "./my_model_final.ckpt")
```

```
0 Train accuracy: 0.9 Val accuracy: 0.9146
1 Train accuracy: 0.94 Val accuracy: 0.9348
2 Train accuracy: 0.92 Val accuracy: 0.9466
3 Train accuracy: 0.96 Val accuracy: 0.9508
4 Train accuracy: 0.92 Val accuracy: 0.9586
5 Train accuracy: 0.94 Val accuracy: 0.9584
6 Train accuracy: 0.98 Val accuracy: 0.9608
7 Train accuracy: 0.96 Val accuracy: 0.9636
8 Train accuracy: 0.92 Val accuracy: 0.9638
9 Train accuracy: 0.96 Val accuracy: 0.965
```

Using the trained network to make predictions

In [55]:
```
with tf.Session() as sess:
    saver.restore(sess, "./my_model_final.ckpt")
    X_new_scaled = mnist.test.images[:20]
    Z = logits.eval(feed_dict={X: X_new_scaled})
    y_pred = np.argmax(Z, axis=1)
```

```
INFO:tensorflow:Restoring parameters from ./my_model_final.ckpt
```

Recall that we only defined the network to compute the logits. If require class probabilities then need to apply softmax function. But that is not needed if just want to make single prediction (i.e. just pick class with largest logit value).

In [56]:
```
print("Predicted classes:", y_pred)
print("Actual classes:   ", mnist.test.labels[:20])
```

```
Predicted classes: [7 2 1 0 4 1 4 9 6 9 0 6 9 0 1 5 9 7 3 4]
Actual classes:    [7 2 1 0 4 1 4 9 5 9 0 6 9 0 1 5 9 7 3 4]
```