

```

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

    for epoch in range(n_epochs):
        [...]
        for X_batch, y_batch in zip(X_batches, y_batches):
            sess.run(training_op,
                      feed_dict={is_training: True, X: X_batch, y: y_batch})
            accuracy_score = accuracy.eval(
                feed_dict={is_training: False, X: X_test_scaled, y: y_test})
        print(accuracy_score)

```

That's all! In this tiny example with just two layers, it's unlikely that Batch Normalization will have a very positive impact, but for deeper networks it can make a tremendous difference.

Gradient Clipping

A popular technique to lessen the exploding gradients problem is to simply clip the gradients during backpropagation so that they never exceed some threshold (this is mostly useful for recurrent neural networks; see [Chapter 14](#)). This is called *Gradient Clipping*.⁸ In general people now prefer Batch Normalization, but it's still useful to know about Gradient Clipping and how to implement it.

In TensorFlow, the optimizer's `minimize()` function takes care of both computing the gradients and applying them, so you must instead call the optimizer's `compute_gradients()` method first, then create an operation to clip the gradients using the `clip_by_value()` function, and finally create an operation to apply the clipped gradients using the optimizer's `apply_gradients()` method:

```

threshold = 1.0
optimizer = tf.train.GradientDescentOptimizer(learning_rate)
grads_and_vars = optimizer.compute_gradients(loss)
capped_gvs = [(tf.clip_by_value(grad, -threshold, threshold), var)
               for grad, var in grads_and_vars]
training_op = optimizer.apply_gradients(capped_gvs)

```

You would then run this `training_op` at every training step, as usual. It will compute the gradients, clip them between -1.0 and 1.0 , and apply them. The threshold is a hyperparameter you can tune.

Reusing Pretrained Layers

It is generally not a good idea to train a very large DNN from scratch: instead, you should always try to find an existing neural network that accomplishes a similar task

⁸ "On the difficulty of training recurrent neural networks," R. Pascanu et al. (2013).