1. 입력

```
total_batch = int(mnist.train.num_examples / batch_size)

for epoch in range(training_epoch):
    total_cost = 0

    for i in range(total_batch):
        batch_xs, batch_ys = mnist.train.next_batch(batch_size)
        _, cost_val = sess.run([optimizer, cost], feed_dict={X: batch_xs})
        total_cost += cost_val

    print('Epoch:', '%04d' % (epoch + 1), 'Avg. cost =', '{:.4f}'.format(total_cost / total_batch))
```

```
samples = sess.run(decoder, feed_dict={X: mnist.test.images[:sample_size]})
```

MNIST 사용

2. Layer

- encoder – hidden Layer – decoder

```
# 인코더 생성
W_encode = tf.Variable(tf.random_normal([n_input, n_hidden]))
b_encode = tf.Variable(tf.random_normal([n_hidden]))

encoder = tf.nn.sigmoid(tf.add(tf.matmul(X, W_encode), b_encode))

# 디코더 생성
W_decode = tf.Variable(tf.random_normal([n_hidden, n_input]))
b_decode = tf.Variable(tf.random_normal([n_input]))
decoder = tf.nn.sigmoid(tf.add(tf.matmul(encoder, W_decode), b_decode))
```

- encoder – encoder layer – hidden layer – decoder layer – decoder

```
# 인코더 생성
W1_encode = tf.Variable(tf.random_normal([n_input, encoder_layer]))
b1_encode = tf.Variable(tf.random_normal([encoder_layer]))

encoder1 = tf.nn.sigmoid(tf.add(tf.matmul(X, W1_encode), b1_encode))

W2_encode = tf.Variable(tf.random_normal([encoder_layer, n_hidden]))
b2_encode = tf.Variable(tf.random_normal([n_hidden]))

encoder = tf.nn.sigmoid(tf.add(tf.matmul(encoder1, W2_encode), b2_encode))

# 디코더 생성
W1_decode = tf.Variable(tf.random_normal([n_hidden, decoder_layer]))
b1_decode = tf.Variable(tf.random_normal([decoder_layer]))
decoder1 = tf.nn.sigmoid(tf.add(tf.matmul(encoder, W1_decode), b1_decode))

W2_decode = tf.Variable(tf.random_normal([decoder_layer, n_input]))
b2_decode = tf.Variable(tf.random_normal([n_input]))
decoder = tf.nn.sigmoid(tf.add(tf.matmul(decoder1, W2_decode), b2_decode))
```

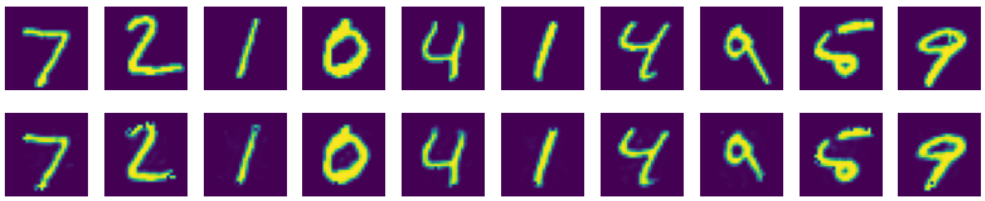
3. 결과

- 비용함수는 mean squared error 를 적용했고, AdamOptimizer 를 이용해 최적화하였다.

```
# 손실 함수 및 최적화 함수 설정
cost = tf.reduce_mean(tf.pow(X - decoder, 2))
optimizer = tf.train.AdamOptimizer(learning_rate).minimize(cost)
```

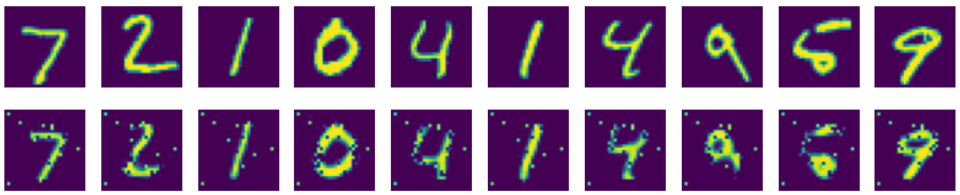
- ① encoder – hidden Layer – decoder 모델

```
Epoch: 0100 Avg. cost = 0.0082
최적화 완료!
```



- ② encoder – encoder layer – hidden layer – decoder layer – decoder 모델

```
Epoch: 0100 Avg. cost = 0.0427
최적화 완료!
```



- ①번 모델과 ②번 모델을 실행한 결과, epoch 100 회를 실행하였을 때 각각 비용함수가 0.0082, 0.0427 로 측정되었고, ①번 모델의 그래프가 더 선명하여 그 차이를 육안으로 확인할 수 있었다. 따라서 ①번 모델의 경우가 더 학습이 잘 진행되었다.