知识点：

1. 通过域名整理graph：tf.name_scope(“xx")

2. 迭代器操作database：tf.data.Dataset.Iterator

```
# create training Dataset and batch it
train_data = tf.data.Dataset.from_tensor_slices(train)
train_data = train_data.shuffle(10000)  # if you want to shuffle your data
train_data = train_data.batch(batch_size)

# create testing Dataset and batch it
test_data = tf.data.Dataset.from_tensor_slices(test)
test_data = test_data.batch(batch_size)

# create one iterator and initialize it with different datasets
iterator = tf.data.Iterator.from_structure(train_data.output_types,
                                           train_data.output_shapes)
img, label = iterator.get_next()

train_init = iterator.make_initializer(train_data)  # initializer for train_data
test_init = iterator.make_initializer(test_data)  # initializer for train_data
```

3. 获取tf.Variable方法：tf.get_variable()

```
with tf.name_scope('embed'):
    self.embed_matrix = tf.get_variable('embed_matrix',
                                        shape=[self.vocab_size, self.embed_size],
                                        initializer=tf.random_uniform_initializer())
```

4. 训练数据断点存储及恢复：tf.train.Saver() & tf.train.get_checkpoint_state()

```
def train(self, num_train_steps):
    saver = tf.train.Saver()  # defaults to saving all variables - in this case
embed_matrix, nce_weight, nce_bias

    initial_step = 0
    utils.safe_mkdir('checkpoints')
    with tf.Session() as sess:
        sess.run(self.iterator.initializer)
        sess.run(tf.global_variables_initializer())
        ckpt = tf.train.get_checkpoint_state(os.path.dirname('checkpoints/checkpoint'))

        # if that checkpoint exists, restore from checkpoint
        if ckpt and ckpt.model_checkpoint_path:
            saver.restore(sess, ckpt.model_checkpoint_path)

        total_loss = 0.0  # we use this to calculate late average loss in the last
SKIP_STEP steps
        writer = tf.summary.FileWriter('graphs/word2vec/lr' + str(self.lr), sess.graph)
        initial_step = self.global_step.eval()
        for index in range(initial_step, num_train_steps):
            try:
                loss_batch, _, summary = sess.run([self.loss, self.optimizer,
self.summary_op])
                writer.add_summary(summary, global_step=index)
                total_loss += loss_batch
                if (index + 1) % self.skip_step == 0:
                    print('Average loss at step {}: {:5.1f}'.format(index, total_loss /
self.skip_step))
                    total_loss = 0.0
                    saver.save(sess, 'checkpoints/skip-gram', index)
            except tf.errors.OutOfRangeError:
                sess.run(self.iterator.initializer)
        writer.close()
```

5. 添加各种数据监测，用于tensorboard显示：tf.summary()

```
def variable_summaries(var):
    with tf.name_scope('summaries'):
        mean = tf.reduce_mean(var)
        tf.summary.scalar('mean', mean)
        with tf.name_scope('stddev'):
            stddev = tf.sqrt(tf.reduce_mean(tf.square(var - mean)))
        tf.summary.scalar('stddev', stddev)
        tf.summary.scalar('max', tf.reduce_max(var))
        tf.summary.scalar('min', tf.reduce_min(var))
        tf.summary.histogram('histogram', var)  ##直方图
```