Input Pipeline

Reading data

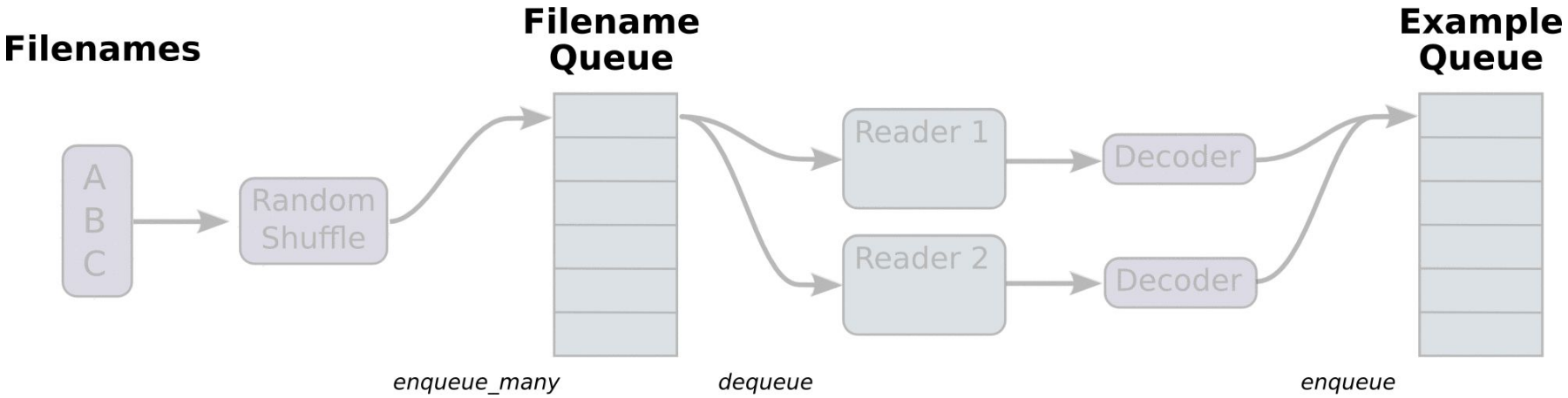
- Feeding: Python code provides the data when running each step
- Reading from files: as input pipeline reads the data from files at the beginning of a TensorFlow graph
- Preloaded data: a constant or variable in TensorFlow graph holds all the data(just for small data set)

Feeding

- Define tf.placeholder node and feed them while running a session.

```
with tf.Session():
    input = tf.placeholder(tf.float32)
    classifier = …
    print(classifier.eval(feed_dict={input: my_python_preprocessing_fn()}))
```

Reading from files



Why input pipeline?

- Improve bottleneck of loading data while training step is executing
- Do not need to occupy main memory a lot.

Reading from files

1. The list of filenames
   a. *Optional* filename shuffling
   b. *Optional* epoch limit
2. Filename queue
3. A Reader for the file format(csv, tsv, tfrecord, ……)
4. A decoder for a record read by the reader
   a. *Optional* preprocessing
5. Example queue

The list of filenames

- Get list of filenames with os modules (or other modules)
  - ex) ["file0", "file1"] or [("file%d" % i) for i in range(2)]
- Pass the list of filenames to the tf.train.string_input_producer function. string_input_producer creates a FIFO queue for holding the filenames until the reader needs them.
- Use tf.parse_single_example to read example protocol buffers.

Problem with data formats

- csv, tsv, ……
  - Save values by delimiter(comma or tab)
  - Inefficient for saving large data
    - 1.99378119 -> requires 10 char vs 32 float bit (binary format)

Protocol buffer?

- Tensorflow provides serialization through protocol buffer(protobuf).
- Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler.
- You define how you want your data to be structured once, then you can use special generated source code to easily write and read your structured data to and from a variety of data streams and using a variety of languages.

Pickle vs Protobuf

Python already provide serialization through pickle module, however, most of tensorflow implemented based on protobuf. For example, graphdef, operations, etc are structured through protobuf.

Tensorflow provides Example and Feature protobuf type.

Let’s navigate Example and Feature protobuf type.

Overall Code Structure

```
filename_queue = tf.train.string_input_producer(['data/COCO_samples.tfrecords'],
num_epochs=5)
image = read_and_decode(filename_queue)
init_op = tf.group(tf.global_variables_initializer(),
tf.local_variables_initializer())
with tf.Session() as sess:
   sess.run(init_op)
   coord = tf.train.Coordinator()
   threads = tf.train.start_queue_runners(coord=coord)
   for i in range(num_iterations):
       # retrieve a single image by running the operation  ‘image’
       img = sess.run(image)
       # do training here

   coord.request_stop()
   coord.join(threads)
```

Overall Code Structure

```
def read_and_decode(filename_queue, read_labels = False):
   reader = tf.TFRecordReader()
   _, serialized_example = reader.read(filename_queue)
   features = tf.parse_single_example(serialized_example,
           features={'height':tf.FixedLenFeature([],tf.int64),
                     'width':tf.FixedLenFeature([],tf.int64),
                     'data':tf.FixedLenFeature([],tf.string),
                     'label':tf.FixedLenFeature([],tf.int64)})
   # Convert from a scalar string tensor (whose single string has
   height = tf.cast(features['height'], tf.int32)
   width = tf.cast(features['width'], tf.int32)
   image = tf.decode_raw(features['data'], tf.uint8)
   # reshape
   image_shape = (224,224,3)
   image = tf.reshape(image, image_shape)

   # Convert label from a scalar uint8 tensor to an int32 scalar.
   label = tf.cast(features['label'], tf.int32)
   images, labels = tf.train.shuffle_batch([image, label],batch_size=2, capacity=30,
                                           num_threads=4, min_after_dequeue=10)
   return images, labels
```

Functions and classes to know

- Filelist
  - Tf.train.string_input_producer
- Reader: There are other readers available. Check in TF website.
  - tf.TFRecordReader
  - Tf.parse_single_example
- Threading Manager
  - tf.train.Coordinator
- Thread
  - tf.train.start_queue_runners

Do exercise

Run 005_input_pipeline/input_pipeline.ipynb