tf.contrib.learn.read_keyed_batch_features_shared_queue

```
read_keyed_batch_features_shared_queue(
    file_pattern,
    batch_size,
    features,
    reader,
    randomize_input=True,
    num_epochs=None,
    queue_capacity=10000,
    reader_num_threads=1,
    feature_queue_capacity=100,
    num_queue_runners=2,
    parse_fn=None,
    name=None
)
```

Defined in `tensorflow/contrib/learn/python/learn/learn_io/graph_io.py`.

Adds operations to read, queue, batch and parse `Example` protos.

Given file pattern (or list of files), will setup a shared queue for file names, setup a worker queue that gets filenames from the shared queue, read `Example` proto using provided `reader`, use batch queue to create batches of examples of size `batch_size` and parse example given `features` specification.

All queue runners are added to the queue runners collection, and may be started via `start_queue_runners`.

All ops are added to the default graph.

Args:

- `file_pattern`: List of files or patterns of file paths containing `Example` records. See `tf.gfile.Glob` for pattern rules.
- `batch_size`: An int or scalar `Tensor` specifying the batch size to use.
- `features`: A `dict` mapping feature keys to `FixedLenFeature` or `VarLenFeature` values.
- `reader`: A function or class that returns an object with `read` method, (filename tensor) -> (example tensor).
- `randomize_input`: Whether the input should be randomized.
- `num_epochs`: Integer specifying the number of times to read through the dataset. If None, cycles through the dataset forever. NOTE - If specified, creates a variable that must be initialized, so call tf.local_variables_initializer() and run the op in a session.
- `queue_capacity`: Capacity for input queue.
- `reader_num_threads`: The number of threads to read examples.
- `feature_queue_capacity`: Capacity of the parsed features queue.
- `num_queue_runners`: Number of threads to enqueue the parsed example queue. Using multiple threads to enqueue the parsed example queue helps maintain a full queue when the subsequent computations overall are cheaper than parsing.
- `parse_fn`: Parsing function, takes `Example` Tensor returns parsed representation. If `None`, no parsing is done.
- `name`: Name of resulting op.

Returns:

Returns tuple of: - `Tensor` of string keys. - A dict of `Tensor` or `SparseTensor` objects for each in `features`.

Raises:

- `ValueError`: for invalid inputs.

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 3.0 License, and code samples are licensed under the Apache 2.0 License. For details, see our Site Policies. Java is a registered trademark of Oracle and/or its affiliates.

Last updated November 2, 2017.