

tf.train.batch_join

```
batch_join(  
    tensors_list,  
    batch_size,  
    capacity=32,  
    enqueue_many=False,  
    shapes=None,  
    dynamic_pad=False,  
    allow_smaller_final_batch=False,  
    shared_name=None,  
    name=None  
)
```

Defined in [tensorflow/python/training/input.py](#).

See the guide: [Inputs and Readers > Input pipeline](#)

Runs a list of tensors to fill a queue to create batches of examples.

The `tensors_list` argument is a list of tuples of tensors, or a list of dictionaries of tensors. Each element in the list is treated similarly to the `tensors` argument of `tf.train.batch()`.

WARNING: This function is nondeterministic, since it starts a separate thread for each tensor.

Enqueues a different list of tensors in different threads. Implemented using a queue – a `QueueRunner` for the queue is added to the current `Graph`'s `QUEUE_RUNNER` collection.

`len(tensors_list)` threads will be started, with thread `i` enqueueing the tensors from `tensors_list[i]`. `tensors_list[i1][j]` must match `tensors_list[i2][j]` in type and shape, except in the first dimension if `enqueue_many` is true.

If `enqueue_many` is `False`, each `tensors_list[i]` is assumed to represent a single example. An input tensor `x` will be output as a tensor with shape `[batch_size] + x.shape`.

If `enqueue_many` is `True`, `tensors_list[i]` is assumed to represent a batch of examples, where the first dimension is indexed by example, and all members of `tensors_list[i]` should have the same size in the first dimension. The slices of any input tensor `x` are treated as examples, and the output tensors will have shape `[batch_size] + x.shape[1:]`.

The `capacity` argument controls the how long the prefetching is allowed to grow the queues.

The returned operation is a dequeue operation and will throw `tf.errors.OutOfRangeError` if the input queue is exhausted. If this operation is feeding another input queue, its queue runner will catch this exception, however, if this operation is used in your main thread you are responsible for catching this yourself.

N.B.: If `dynamic_pad` is `False`, you must ensure that either (i) the `shapes` argument is passed, or (ii) all of the tensors in `tensors_list` must have fully-defined shapes. `ValueError` will be raised if neither of these conditions holds.

If `dynamic_pad` is `True`, it is sufficient that the *rank* of the tensors is known, but individual dimensions may have value `None`. In this case, for each enqueue the dimensions with value `None` may have a variable length; upon dequeue, the output tensors will be padded on the right to the maximum shape of the tensors in the current minibatch. For numbers, this padding takes value 0. For strings, this padding is the empty string. See [PaddingFIFOQueue](#) for more info.

If `allow_smaller_final_batch` is `True`, a smaller batch value than `batch_size` is returned when the queue is closed and

there are not enough elements to fill the batch, otherwise the pending elements are discarded. In addition, all output tensors' static shapes, as accessed via the `shape` property will have a first `Dimension` value of `None`, and operations that depend on fixed `batch_size` would fail.

Args:

- `tensors_list` : A list of tuples or dictionaries of tensors to enqueue.
- `batch_size` : An integer. The new batch size pulled from the queue.
- `capacity` : An integer. The maximum number of elements in the queue.
- `enqueue_many` : Whether each tensor in `tensor_list_list` is a single example.
- `shapes` : (Optional) The shapes for each example. Defaults to the inferred shapes for `tensor_list_list[i]`.
- `dynamic_pad` : Boolean. Allow variable dimensions in input shapes. The given dimensions are padded upon dequeue so that tensors within a batch have the same shapes.
- `allow_smaller_final_batch` : (Optional) Boolean. If `True`, allow the final batch to be smaller if there are insufficient items left in the queue.
- `shared_name` : (Optional) If set, this queue will be shared under the given name across multiple sessions.
- `name` : (Optional) A name for the operations.

Returns:

A list or dictionary of tensors with the same number and types as `tensors_list[i]`.

Raises:

- `ValueError` : If the `shapes` are not specified, and cannot be inferred from the elements of `tensor_list_list`.

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.

Stay Connected

[Blog](#)

[GitHub](#)

[Twitter](#)

Support

[Issue Tracker](#)

[Release Notes](#)

[Stack Overflow](#)

English

[Terms](#) | [Privacy](#)