tf.contrib.timeseries.RandomWindowInputFn

Contents

Class **`RandomWindowInputFn`**

Defined in `tensorflow/contrib/timeseries/python/timeseries/input_pipeline.py`.

Wraps a `TimeSeriesReader` to create random batches of windows.

Tensors are first collected into sequential windows (in a windowing queue created by `tf.train.batch`, based on the order returned from `time_series_reader`), then these windows are randomly batched (in a `RandomShuffleQueue`), the Tensors returned by `create_batch` having shapes prefixed by [`batch_size`, `window_size`].

This `TimeSeriesInputFn` is useful for both training and quantitative evaluation (but be sure to run several epochs for sequential models such as `StructuralEnsembleRegressor` to completely flush stale state left over from training). For qualitative evaluation or when preparing for predictions, use `WholeDatasetInputFn`.

Methods

`__init__`

```
__init__(
    time_series_reader,
    window_size,
    batch_size,
    queue_capacity_multiplier=1000,
    shuffle_min_after_dequeue_multiplier=2,
    discard_out_of_order=True,
    discard_consecutive_batches_limit=1000,
    jitter=True,
    num_threads=2,
    shuffle_seed=None
)
```

Configure the RandomWindowInputFn.

Args:

- `time_series_reader`: A TimeSeriesReader object.
- `window_size`: The number of examples to keep together sequentially. This controls the length of truncated backpropagation: smaller values mean less sequential computation, which can lead to faster training, but create a coarser approximation to the gradient (which would ideally be computed by a forward pass over the entire sequence in order).

- `batch_size`: The number of windows to place together in a batch. Larger values will lead to more stable gradients during training.
- `queue_capacity_multiplier`: The capacity for the queues used to create batches, specified as a multiple of `batch_size` (for RandomShuffleQueue) and `batch_size * window_size` (for the FIFOQueue). Controls the maximum number of windows stored. Should be greater than `shuffle_min_after_dequeue_multiplier`.
- `shuffle_min_after_dequeue_multiplier`: The minimum number of windows in the RandomShuffleQueue after a dequeue, which controls the amount of entropy introduced during batching. Specified as a multiple of `batch_size`.
- `discard_out_of_order`: If True, windows of data which have times which decrease (a higher time followed by a lower time) are discarded. If False, the window and associated features are instead sorted so that times are non-decreasing. Discarding is typically faster, as models do not have to deal with artificial gaps in the data. However, discarding does create a bias where the beginnings and endings of files are under-sampled.
- `discard_consecutive_batches_limit`: Raise an OutOfRangeError if more than this number of batches are discarded without a single non-discarded window (prevents infinite looping when the dataset is too small).
- `jitter`: If True, randomly discards examples between some windows in order to avoid deterministic chunking patterns. This is important for models like AR which may otherwise overfit a fixed chunking.
- `num_threads`: Use this number of threads for queues. Setting a value of 1 removes one source of non-determinism (and in combination with shuffle_seed should provide deterministic windowing).
- `shuffle_seed`: A seed for window shuffling. The default value of None provides random behavior. With `shuffle_seed` set and `num_threads=1`, provides deterministic behavior.

`__call__`

```
__call__()
```

`create_batch`

```
create_batch()
```

Create queues to window and batch time series data.

Returns:

A dictionary of Tensors corresponding to the output of `self._reader` (from the `time_series_reader` constructor argument), each with shapes prefixed by [`batch_size`, `window_size`].

Last updated November 2, 2017.