Module: tf.contrib.training

Contents

Defined in `tensorflow/contrib/training/__init__.py`.

Training and input utilities.

See Training (contrib) guide.

Classes

`class FeedingQueueRunner`: A queue runner that allows the feeding of values such as numpy arrays.

`class GreedyLoadBalancingStrategy`: Returns the least-loaded ps task for op placement.

`class HParams`: Class to hold a set of hyperparameters as name-value pairs.

`class NextQueuedSequenceBatch`: NextQueuedSequenceBatch stores deferred SequenceQueueingStateSaver data.

`class RandomStrategy`: Returns a random PS task for op placement.

`class SequenceQueueingStateSaver`: SequenceQueueingStateSaver provides access to stateful values from input.

`class StopAfterNEvalsHook`: Run hook used by the evaluation routines to run the `eval_ops` N times.

`class SummaryAtEndHook`: A run hook that saves a summary with the results of evaluation.

Functions

`add_gradients_summaries(...)`: Add summaries to gradients.

`batch_sequences_with_states(...)`: Creates batches of segments of sequential input.

`bucket(...)`: Lazy bucketing of input tensors according to `which_bucket`.

`bucket_by_sequence_length(...)`: Lazy bucketing of inputs according to their length.

`byte_size_load_fn(...)`: Load function that computes the byte size of a single-output `Operation`.

`checkpoints_iterator(...)`: Continuously yield new checkpoint files as they appear.

`clip_gradient_norms(...)`: Clips the gradients by the given value.

`clip_gradient_norms_fn(...)`: Returns a `transform_grads_fn` function for gradient clipping.

`create_train_op(...)`: Creates an `Operation` that evaluates the gradients and returns the loss.

`evaluate_once(...)`: Evaluates the model at the given checkpoint path.

`evaluate_repeatedly(...)`: Repeatedly searches for a checkpoint in `checkpoint_dir` and evaluates it.

`get_or_create_eval_step(...)` : Gets or creates the eval step `Tensor`.

`multiply_gradients(...)` : Multiply specified gradients.

`parse_values(...)` : Parses hyperparameter values from a string into a python map..

`rejection_sample(...)` : Stochastically creates batches by rejection sampling.

`resample_at_rate(...)` : Given `inputs` tensors, stochastically resamples each at a given rate.

`stratified_sample(...)` : Stochastically creates batches based on per-class probabilities.

`train(...)` : Runs the training loop.

`wait_for_new_checkpoint(...)` : Waits until a new checkpoint file is found.

`weighted_resample(...)` : Performs an approximate weighted resampling of `inputs`.

Last updated November 2, 2017.