Evaluators

Base

paddle.trainer_config_helpers.evaluators.**evaluator_base**(*input*, *type*, *label=None*, *weight=None*, *name=None*, *chunk_scheme=None*, *num_chunk_types=None*, *classification_threshold=None*, *positive_label=None*, *dict_file=None*, *result_file=None*, *num_results=None*, *delimited=None*)

Evaluator will evaluate the network status while training/testing.

User can use evaluator by classify/regression job. For example.

```
classify(prediction, output, evaluator=classification_error_evaluator)
```

And user could define evaluator separately as follow.

```
classification_error_evaluator("ErrorRate", prediction, label)
```

The evaluator often contains a name parameter. It will also be printed when evaluating network. The printed information may look like the following.

```
Batch=200 samples=20000 AvgCost=0.679655 CurrentCost=0.662179 Eval:
classification_error_evaluator=0.4486
CurrentEval: ErrorRate=0.3964
```

Parameters:
- **input** (*list|LayerOutput*) — Input layers, a object of LayerOutput or a list of LayerOutput.
- **label** (*LayerOutput|None*) — An input layer containing the ground truth label.
- **weight** (*LayerOutput.*) — An input layer which is a weight for each sample. Each evaluator may calculate differently to use this weight.

Classification

classification_error_evaluator

paddle.trainer_config_helpers.evaluators.**classification_error_evaluator**(**args*, ***kwargs*)

Classification Error Evaluator. It will print error rate for classification.

The classification error is:

$$classification\_error = \frac{NumOfWrongPredicts}{NumOfAllSamples}$$

The simple usage is:

```
eval =  classification_error_evaluator(input=prob,label=lbl)
```

Parameters:
- **name** (*basestring*) — Evaluator name.

- **input** (*LayerOutput*) — Input Layer name. The output prediction of network.
- **label** (*basestring*) — Label layer name.
- **weight** (*LayerOutput*) — Weight Layer name. It should be a matrix with size [sample_num, 1]. And will just multiply to NumOfWrongPredicts and NumOfAllSamples. So, the elements of weight are all one, then means not set weight. The larger weight it is, the more important this sample is.
- **threshold** (*float*) — The classification threshold.

Returns: None.

auc_evaluator

paddle.trainer_config_helpers.evaluators.**auc_evaluator**(**args*, ***kwargs*)

Auc Evaluator which adapts to binary classification.

The simple usage:

```
eval = auc_evaluator(input, label)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer name. The output prediction of network.
- **label** (*None|basestring*) — Label layer name.
- **weight** (*LayerOutput*) — Weight Layer name. It should be a matrix with size [sample_num, 1].

ctc_error_evaluator

paddle.trainer_config_helpers.evaluators.**ctc_error_evaluator**(**args*, ***kwargs*)

This evaluator is to calculate sequence-to-sequence edit distance.

The simple usage is :

```
eval = ctc_error_evaluator(input=input, label=lbl)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer. Should be the same as the input for ctc_layer.
- **label** (*LayerOutput*) — input label, which is a data_layer. Should be the same as the label for ctc_layer

chunk_evaluator

paddle.trainer_config_helpers.evaluators.**chunk_evaluator**(**args*, ***kwargs*)

Chunk evaluator is used to evaluate segment labelling accuracy for a sequence. It calculates the chunk detection F1 score.

A chunk is correctly detected if its beginning, end and type are correct. Other chunk type is ignored.

For each label in the label sequence, we have:

```
tagType = label % numTagType
chunkType = label / numTagType
otherChunkType = numChunkTypes
```

The total number of different labels is numTagType*numChunkTypes+1. We support 4 labelling scheme. The tag type for each of the scheme is shown as follows:

```
Scheme Begin Inside End   Single
plain  0     -      -     -
IOB    0     1      -     -
IOE    -     0      1     -
IOBES  0     1      2     3
```

'plain' means the whole chunk must contain exactly the same chunk label.

The simple usage is:

```
eval = chunk_evaluator(input)
```

Parameters:
- **input** (*LayerOutput*) — The input layers.
- **name** (*basename|None*) — The Evaluator name, it is not necessary.
- **chunk_scheme** (*basestring*) — The labelling schemes support 4 types. It is one of "IOB", "IOE", "IOBES", "plain".This Evaluator must contain this chunk_scheme.
- **num_chunk_types** — number of chunk types other than "other"

precision_recall_evaluator

paddle.trainer_config_helpers.evaluators.**precision_recall_evaluator**(**args*, ***kwargs*)

An Evaluator to calculate precision and recall, F1-score. It is adapt to the task with multiple labels.

- If positive_label=-1, it will print the average precision, recall, F1-score of all labels.
- If use specify positive_label, it will print the precision, recall, F1-score of this label.

The simple usage:

```
eval = precision_recall_evaluator(input, label)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer name. The output prediction of network.
- **label** (*LayerOutput*) — Label layer name.
- **positive_label** (*LayerOutput.*) — The input label layer.
- **weight** (*LayerOutput*) — Weight Layer name. It should be a matrix with size [sample_num, 1]. (TODO, explaination)

Rank

pnpair_evaluator

paddle.trainer_config_helpers.evaluators.**pnpair_evaluator**(**args*, ***kwargs*)

Positive-negative pair rate Evaluator which adapts to rank task like learning to rank. This evaluator must contain at least three layers.

The simple usage:

```
eval = pnpair_evaluator(input, info, label)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer name. The output prediction of network.
- **label** (*LayerOutput*) — Label layer name.
- **info** (*LayerOutput*) — Label layer name. (TODO, explaination)
- **weight** (*LayerOutput*) — Weight Layer name. It should be a matrix with size [sample_num, 1]. (TODO, explaination)

Utils

sum_evaluator

paddle.trainer_config_helpers.evaluators.**sum_evaluator**(**args*, ***kwargs*)

An Evaluator to sum the result of input.

The simple usage:

```
eval = sum_evaluator(input)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer name.
- **weight** (*LayerOutput*) — Weight Layer name. It should be a matrix with size [sample_num, 1]. (TODO, explaination)

column_sum_evaluator

paddle.trainer_config_helpers.evaluators.**column_sum_evaluator**(**args*, ***kwargs*)

This Evaluator is used to sum the last column of input.

The simple usage is:

```
eval = column_sum_evaluator(input, label)
```

Parameters:
- **name** (*None|basestring*) — Evaluator name.
- **input** (*LayerOutput*) — Input Layer name.

Print

classification_error_printer_evaluator

paddle.trainer_config_helpers.evaluators.**classification_error_printer_evaluator**(**args*, ***kwargs*)

This Evaluator is used to print the classification error of each sample.

The simple usage is:

```
eval = classification_error_printer_evaluator(input)
```

Parameters:
- **input** (*LayerOutput*) — Input layer.
- **label** (*LayerOutput*) — Input label layer.
- **name** (*None|basestring*) — Evaluator name.

gradient_printer_evaluator

paddle.trainer_config_helpers.evaluators.**gradient_printer_evaluator**(**args*, ***kwargs*)

This Evaluator is used to print the gradient of input layers. It contains one or more input layers.

The simple usage is:

```
eval = gradient_printer_evaluator(input)
```

Parameters:
- **input** (*LayerOutput|list*) — One or more input layers.
- **name** (*None|basestring*) — Evaluator name.

maxid_printer_evaluator

paddle.trainer_config_helpers.evaluators.**maxid_printer_evaluator**(**args*, ***kwargs*)

This Evaluator is used to print maximum top k values and their indexes of each row of input layers. It contains one or more input layers. k is specified by num_results.

The simple usage is:

```
eval = maxid_printer_evaluator(input)
```

Parameters:
- **input** (*LayerOutput|list*) — Input Layer name.
- **num_results** (*int.*) — This number is used to specify the top k numbers. It is 1 by default.
- **name** (*None|basestring*) — Evaluator name.

maxframe_printer_evaluator

paddle.trainer_config_helpers.evaluators.**maxframe_printer_evaluator**(**args*, ***kwargs*)

This Evaluator is used to print the top k frames of each input layers. The input layers should contain sequences info or sequences type. k is specified by num_results. It contains one or more input layers.

> **Note:** The width of each frame is 1.

The simple usage is:

```
eval = maxframe_printer_evaluator(input)
```

Parameters:
- **input** (*LayerOutput|list*) — Input Layer name.
- **name** (*None|basestring*) — Evaluator name.

seqtext_printer_evaluator

paddle.trainer_config_helpers.evaluators.**seqtext_printer_evaluator**(**args*, ***kwargs*)

Sequence text printer will print text according to index matrix and a dictionary. There can be multiple input to this layer:

1. If there is no id_input, the input must be a matrix containing the sequence of indices;

2. If there is id_input, it should be ids, and interpreted as sample ids.

The output format will be:

1. sequence without sub-sequence, and there is probability.

```
id      prob space_seperated_tokens_from_dictionary_according_to_seq
```

2. sequence without sub-sequence, and there is not probability.

```
id      space_seperated_tokens_from_dictionary_according_to_seq
```

3. sequence with sub-sequence, and there is not probability.

```
id      space_seperated_tokens_from_dictionary_according_to_sub_seq
                space_seperated_tokens_from_dictionary_according_to_sub_seq
...
```

Typically SequenceTextPrinter layer takes output of maxid or RecurrentGroup with maxid (when generating) as an input.

The simple usage is:

```
eval = seqtext_printer_evaluator(input=maxid_layer,
                                 id_input=sample_id,
                                 dict_file=dict_file,
                                 result_file=result_file)
```

Parameters:
- **input** (*LayerOutput|list*) — Input Layer name.
- **result_file** (*basestring*) — Path of the file to store the generated results.
- **id_input** (*LayerOutput*) — Index of the input sequence, and the specified index will be prited in the gereated results. This an optional parameter.
- **dict_file** (*basestring*) — Path of dictionary. This is an optional parameter. Every line is a word in the dictionary with (line number - 1) as the word index. If this parameter is set to None, or to an empty string, only word index are printed in the generated results.
- **delimited** (*bool*) — Whether to use space to separate output tokens. Default is True. No space is added if set to False.
- **name** (*None|basestring*) — Evaluator name.

Returns: The seq_text_printer that prints the generated sequence to a file.

Return type: evaluator

value_printer_evaluator

paddle.trainer_config_helpers.evaluators.**value_printer_evaluator**(**args*, ***kwargs*)

This Evaluator is used to print the values of input layers. It contains one or more input layers.

The simple usage is:

```
eval = value_printer_evaluator(input)
```

Parameters:
- **input** (*LayerOutput|list*) — One or more input layers.
- **name** (*None|basestring*) — Evaluator name.