# pylint: disable=no-member, invalid-name

"""

Implementing Writer Layer

"""

from .layer import Layer

import libtwml

class BatchPredictionTensorWriter(Layer):

"""

A layer that packages keys and dense tensors into a BatchPredictionResponse.

Typically used at the out of an exported model for use in a the PredictionEngine

(that is, in production) when model predictions are dense tensors.

Arguments:

keys:

keys to hashmap

Output:

output:

a BatchPredictionResponse serialized using Thrift into a uint8 tensor.

"""

def \_\_init\_\_(self, keys, \*\*kwargs): # pylint: disable=useless-super-delegation

super(BatchPredictionTensorWriter, self).\_\_init\_\_(\*\*kwargs)

self.keys = keys

def compute\_output\_shape(self, input\_shape):

"""Computes the output shape of the layer given the input shape.

Args:

input\_shape: A (possibly nested tuple of) `TensorShape`. It need not

be fully defined (e.g. the batch size may be unknown).

Raise NotImplementedError.

"""

raise NotImplementedError

def call(self, values, \*\*kwargs): # pylint: disable=unused-argument, arguments-differ

"""The logic of the layer lives here.

Arguments:

values:

dense tensors corresponding to keys in hashmap

Returns:

The output from the layer

"""

write\_op = libtwml.ops.batch\_prediction\_tensor\_response\_writer(self.keys, values)

return write\_op