import tensorflow.compat.v1 as tf

from twml.trainers import DataRecordTrainer

from twml.contrib.optimizers import PruningOptimizer

class PruningDataRecordTrainer(DataRecordTrainer):

@staticmethod

def get\_train\_op(params, loss):

train\_op = DataRecordTrainer.get\_train\_op(params, loss)

optimizer = PruningOptimizer(learning\_rate=params.get('learning\_rate'))

return optimizer.minimize(

loss=loss,

prune\_every=params.get('pruning\_iter', 5000),

burn\_in=params.get('pruning\_burn\_in', 100000),

decay=params.get('pruning\_decay', .9999),

flops\_target=params.get('pruning\_flops\_target', 250000),

update\_params=train\_op,

global\_step=tf.train.get\_global\_step())

def \_\_init\_\_(self, name, params, build\_graph\_fn, feature\_config=None, \*\*kwargs):

kwargs['optimize\_loss\_fn'] = self.get\_train\_op

super(PruningDataRecordTrainer, self).\_\_init\_\_(

name=name,

params=params,

build\_graph\_fn=build\_graph\_fn,

feature\_config=feature\_config,

\*\*kwargs)

def export\_model(self, \*args, \*\*kwargs):

# TODO: modify graph before exporting to take into account masks

return super(PruningDataRecordTrainer, self).export\_model(\*args, \*\*kwargs)

@staticmethod

def add\_parser\_arguments():

parser = DataRecordTrainer.add\_parser\_arguments()

parser.add\_argument(

"--pruning.iter", "--pruning\_iter", type=int, default=5000,

dest="pruning\_iter",

help="A single feature or feature map is pruned every this many iterations")

parser.add\_argument(

"--pruning.burn\_in", "--pruning\_burn\_in", type=int, default=100000,

dest="pruning\_burn\_in",

help="Only start pruning after collecting statistics for this many training steps")

parser.add\_argument(

"--pruning.flops\_target", "--pruning\_flops\_target", type=int, default=250000,

dest="pruning\_flops\_target",

help="Stop pruning when estimated number of floating point operations reached this target. \

For example, a small feed-forward network might require 250,000 FLOPs to run.")

parser.add\_argument(

"--pruning.decay", "--pruning\_decay", type=float, default=.9999,

dest="pruning\_decay",

help="A float value in [0.0, 1.0) controlling an exponential moving average of pruning \

signal statistics. A value of 0.9999 can be thought of as averaging statistics over 10,000 \

steps.")

return parser