import datetime

from absl import logging

import pytz

import tensorflow.compat.v1 as tf

class StopAtTimeHook(tf.train.SessionRunHook):

"""

Hook that stops training at a fixed datetime

"""

def \_\_init\_\_(self, stop\_time):

"""

Arguments:

stop\_time:

a datetime.datetime or a datetime.timedelta specifying when to stop.

For naive datetime.datetime objects (with no time zone specified),

UTC time zone is assumed.

"""

if isinstance(stop\_time, datetime.timedelta):

self.\_stop\_datetime = pytz.utc.localize(datetime.datetime.utcnow() + stop\_time)

elif isinstance(stop\_time, datetime.datetime):

if stop\_time.tzinfo is None:

self.\_stop\_datetime = pytz.utc.localize(stop\_time)

else:

self.\_stop\_datetime = stop\_time.astimezone(pytz.UTC)

else:

raise ValueError("Expecting datetime or timedelta for stop\_time arg")

self.\_stop\_requested = False

def after\_run(self, run\_context, run\_values):

delta = self.\_stop\_datetime - pytz.utc.localize(datetime.datetime.utcnow())

if delta.total\_seconds() <= 0:

logging.info("StopAtTimeHook reached stop\_time; requesting stop")

run\_context.request\_stop()

self.\_stop\_requested = True

@property

def stop\_requested(self):

""" true if this hook requested a stop """

return self.\_stop\_requested