from reader import EventBusPipedBinaryRecordReader

import tensorflow.compat.v1 as tf

import twml

"""

This module provides input function for DeepBird v2 training.

The training data records are loaded from an EventBus reader.

"""

def get\_eventbus\_data\_record\_generator(eventbus\_reader):

"""

This module provides a data record generater from EventBus reader.

Args:

eventbus\_reader: EventBus reader

Returns:

gen: Data record generater

"""

eventbus\_reader.initialize()

counter = [0]

def gen():

while True:

record = eventbus\_reader.read()

if eventbus\_reader.debug:

tf.logging.warn("counter: {}".format(counter[0]))

with open('tmp\_record\_{}.bin'.format(counter[0]), 'wb') as f:

f.write(record)

counter[0] = counter[0] + 1

yield record

return gen

def get\_eventbus\_data\_record\_dataset(eventbus\_reader, parse\_fn, batch\_size):

"""

This module generates batch data for training from a data record generator.

"""

dataset = tf.data.Dataset.from\_generator(

get\_eventbus\_data\_record\_generator(eventbus\_reader), tf.string, tf.TensorShape([]))

return dataset.batch(batch\_size).map(parse\_fn, num\_parallel\_calls=4).prefetch(buffer\_size=10)

def get\_train\_input\_fn(feature\_config, params, parse\_fn=None):

"""

This module provides input function for DeepBird v2 training.

It gets batched training data from data record generator.

"""

eventbus\_reader = EventBusPipedBinaryRecordReader(

params.jar\_file, params.num\_eb\_threads, params.subscriber\_id,

filter\_str=params.filter\_str, debug=params.debug)

train\_parse\_fn = parse\_fn or twml.parsers.get\_sparse\_parse\_fn(

feature\_config, ["ids", "keys", "values", "batch\_size", "weights"])

return lambda: get\_eventbus\_data\_record\_dataset(

eventbus\_reader, train\_parse\_fn, params.train\_batch\_size)