"""

Evaluation job for the heavy ranker of the push notification service.

"""

from datetime import datetime

import twml

from ..libs.metric\_fn\_utils import get\_metric\_fn

from ..libs.model\_utils import read\_config

from .features import get\_feature\_config

from .model\_pools import ALL\_MODELS

from .params import load\_graph\_params

from .run\_args import get\_eval\_arg\_parser

from tensorflow.compat.v1 import logging

def main():

args, \_ = get\_eval\_arg\_parser().parse\_known\_args()

logging.info(f"Parsed args: {args}")

params = load\_graph\_params(args)

logging.info(f"Loaded graph params: {params}")

logging.info(f"Get Feature Config: {args.feature\_list}")

feature\_list = read\_config(args.feature\_list).items()

feature\_config = get\_feature\_config(

data\_spec\_path=args.data\_spec,

params=params,

feature\_list\_provided=feature\_list,

)

logging.info("Build DataRecordTrainer.")

metric\_fn = get\_metric\_fn("OONC\_Engagement" if len(params.tasks) == 2 else "OONC", False)

trainer = twml.trainers.DataRecordTrainer(

name="magic\_recs",

params=args,

build\_graph\_fn=lambda \*args: ALL\_MODELS[params.model.name](params=params)(\*args),

save\_dir=args.save\_dir,

run\_config=None,

feature\_config=feature\_config,

metric\_fn=metric\_fn,

)

logging.info("Run the evaluation.")

start = datetime.now()

trainer.\_estimator.evaluate(

input\_fn=trainer.get\_eval\_input\_fn(repeat=False, shuffle=False),

steps=None if (args.eval\_steps is not None and args.eval\_steps < 0) else args.eval\_steps,

checkpoint\_path=args.eval\_checkpoint,

)

logging.info(f"Evaluating time: {datetime.now() - start}.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

logging.info("Job done.")