package com.twitter.simclusters\_v2.scalding.embedding.common

import com.twitter.recos.entities.thriftscala.Entity

import com.twitter.scalding.Args

import com.twitter.scalding.TypedPipe

import com.twitter.simclusters\_v2.common.ModelVersions

import com.twitter.simclusters\_v2.scalding.embedding.common.EmbeddingUtil.UserId

import com.twitter.simclusters\_v2.thriftscala.ModelVersion

import com.twitter.wtf.entity\_real\_graph.common.EntityUtil

import com.twitter.wtf.entity\_real\_graph.thriftscala.Edge

import com.twitter.wtf.entity\_real\_graph.thriftscala.EntityType

import com.twitter.wtf.entity\_real\_graph.thriftscala.FeatureName

object EntityEmbeddingUtil {

def getEntityUserMatrix(

entityRealGraphSource: TypedPipe[Edge],

halfLife: HalfLifeScores.HalfLifeScoresType,

entityType: EntityType

): TypedPipe[(Entity, (UserId, Double))] = {

entityRealGraphSource

.flatMap {

case Edge(userId, entity, consumerFeatures, \_, \_)

if consumerFeatures.exists(\_.exists(\_.featureName == FeatureName.Favorites)) &&

EntityUtil.getEntityType(entity) == entityType =>

for {

features <- consumerFeatures

favFeatures <- features.find(\_.featureName == FeatureName.Favorites)

ewmaMap <- favFeatures.featureValues.ewmaMap

favScore <- ewmaMap.get(halfLife.id)

} yield (entity, (userId, favScore))

case \_ => None

}

}

object HalfLifeScores extends Enumeration {

type HalfLifeScoresType = Value

val OneDay: Value = Value(1)

val SevenDays: Value = Value(7)

val FourteenDays: Value = Value(14)

val ThirtyDays: Value = Value(30)

val SixtyDays: Value = Value(60)

}

case class EntityEmbeddingsJobConfig(

topK: Int,

halfLife: HalfLifeScores.HalfLifeScoresType,

modelVersion: ModelVersion,

entityType: EntityType,

isAdhoc: Boolean)

object EntityEmbeddingsJobConfig {

def apply(args: Args, isAdhoc: Boolean): EntityEmbeddingsJobConfig = {

val entityTypeArg =

EntityType.valueOf(args.getOrElse("entity-type", default = "")) match {

case Some(entityType) => entityType

case \_ =>

throw new IllegalArgumentException(

s"Argument [--entity-type] must be provided. Supported options [" +

s"${EntityType.SemanticCore.name}, ${EntityType.Hashtag.name}]")

}

EntityEmbeddingsJobConfig(

topK = args.getOrElse("top-k", default = "100").toInt,

halfLife = HalfLifeScores(args.getOrElse("half-life", default = "14").toInt),

// Fail fast if there is no correct model-version argument

modelVersion = ModelVersions.toModelVersion(

args.getOrElse("model-version", ModelVersions.Model20M145K2020)

),

// Fail fast if there is no correct entity-type argument

entityType = entityTypeArg,

isAdhoc = isAdhoc

)

}

}

}