package com.twitter.simclusters\_v2.summingbird.common

import com.twitter.cuad.ner.thriftscala.WholeEntityType

import com.twitter.simclusters\_v2.summingbird.common.Implicits.thriftDecayedValueMonoid

import com.twitter.simclusters\_v2.thriftscala.{Scores, SimClusterEntity, TweetTextEntity}

import scala.collection.Map

private[summingbird] object EntityUtil {

def updateScoreWithLatestTimestamp[K](

scoresMapOption: Option[Map[K, Scores]],

timeInMs: Long

): Option[Map[K, Scores]] = {

scoresMapOption map { scoresMap =>

scoresMap.mapValues(score => updateScoreWithLatestTimestamp(score, timeInMs))

}

}

def updateScoreWithLatestTimestamp(score: Scores, timeInMs: Long): Scores = {

score.copy(

favClusterNormalized8HrHalfLifeScore = score.favClusterNormalized8HrHalfLifeScore.map {

decayedValue => thriftDecayedValueMonoid.decayToTimestamp(decayedValue, timeInMs)

},

followClusterNormalized8HrHalfLifeScore = score.followClusterNormalized8HrHalfLifeScore.map {

decayedValue => thriftDecayedValueMonoid.decayToTimestamp(decayedValue, timeInMs)

}

)

}

def entityToString(entity: SimClusterEntity): String = {

entity match {

case SimClusterEntity.TweetId(id) => s"t\_id:$id"

case SimClusterEntity.SpaceId(id) => s"space\_id:$id"

case SimClusterEntity.TweetEntity(textEntity) =>

textEntity match {

case TweetTextEntity.Hashtag(str) => s"$str[h\_tag]"

case TweetTextEntity.Penguin(penguin) =>

s"${penguin.textEntity}[penguin]"

case TweetTextEntity.Ner(ner) =>

s"${ner.textEntity}[ner\_${WholeEntityType(ner.wholeEntityType)}]"

case TweetTextEntity.SemanticCore(semanticCore) =>

s"[sc:${semanticCore.entityId}]"

}

}

}

}