package com.twitter.simclusters\_v2.scalding.update\_known\_for

import com.twitter.bijection.scrooge.BinaryScalaCodec

import com.twitter.hermit.candidate.thriftscala.Candidates

import com.twitter.logging.Logger

import com.twitter.pluck.source.cassowary.FollowingsCosineSimilaritiesManhattanSource

import com.twitter.scalding.typed.TypedPipe

import com.twitter.scalding.DateOps

import com.twitter.scalding.DateParser

import com.twitter.scalding.Days

import com.twitter.scalding.Execution

import com.twitter.scalding.RichDate

import com.twitter.scalding.TypedTsv

import com.twitter.scalding.UniqueID

import com.twitter.scalding.\_

import com.twitter.scalding\_internal.dalv2.DAL

import com.twitter.scalding\_internal.dalv2.DALWrite.D

import com.twitter.scalding\_internal.dalv2.DALWrite.\_

import com.twitter.scalding\_internal.dalv2.remote\_access.AllowCrossClusterSameDC

import com.twitter.scalding\_internal.job.TwitterExecutionApp

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

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

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

import com.twitter.simclusters\_v2.hdfs\_sources.AdhocKeyValSources

import com.twitter.simclusters\_v2.hdfs\_sources.InternalDataPaths

import com.twitter.simclusters\_v2.hdfs\_sources.SimclustersV2KnownFor20M145KDec11ScalaDataset

import com.twitter.simclusters\_v2.hdfs\_sources.SimclustersV2KnownFor20M145KUpdatedScalaDataset

import com.twitter.simclusters\_v2.hdfs\_sources.SimclustersV2RawKnownFor20M145K2020ScalaDataset

import com.twitter.simclusters\_v2.scalding.KnownForSources

import com.twitter.simclusters\_v2.scalding.KnownForSources.fromKeyVal

import com.twitter.simclusters\_v2.scalding.common.Util

import com.twitter.wtf.scalding.jobs.common.ScheduledExecutionApp

import java.util.TimeZone

/\*\*

\* Scheduled job

\*

\* capesospy-v2 update --build\_locally --start\_cron update\_known\_for\_20m\_145k\_2020 \

\* src/scala/com/twitter/simclusters\_v2/capesos\_config/atla\_proc.yaml

\*/

object UpdateKnownFor20M145K2020 extends ScheduledExecutionApp {

override val firstTime: RichDate = RichDate("2020-10-04")

override val batchIncrement: Duration = Days(7)

private val tempLocationPath = "/user/cassowary/temp/simclusters\_v2/known\_for\_20m\_145k\_2020"

private val simsGraphPath =

"/atla/proc/user/cassowary/manhattan\_sequence\_files/approximate\_cosine\_similarity\_follow"

override def runOnDateRange(

args: Args

)(

implicit dateRange: DateRange,

timeZone: TimeZone,

uniqueID: UniqueID

): Execution[Unit] = {

Execution.getConfigMode.flatMap {

case (\_, mode) =>

implicit def valueCodec: BinaryScalaCodec[Candidates] = BinaryScalaCodec(Candidates)

// Step - 1 (DataProcessing): Parameters for getting mapped indices for user-ids

val minActiveFollowers = args.int("minActiveFollowers", 400)

val topK = args.int("topK", 20000000)

// Step - 2 (DataProcessing): Parameters to remove users not in the topK most followed users from simsGraph

val maxNeighbors = args.int("maxNeighbors", 400)

// Step - 3 (Final Clustering): Parameters to run the clustering algorithm

/\* squareWeightEnable is a boolean flag that changes the edge weights obtained from the

underlying sims graph

a) If false - edge weight between two neighbors is just their cosine similarity.

b) If true - edge weight = cosine\_sim \* cosine\_sim \* 10. The squaring makes the higher

weight edges relatively more important; this is based on the intuition that a neighbor

with cosine similarity of 0.1 is more than 2x important compared to a neighbor with

cosine similarity of 0.05. The multiplication with 10 brings the weights back into a

'nicer' range since squaring will reduce their absolute value.

\*/

val squareWeightsEnable = args.boolean("squareWeightsEnable")

val maxEpochsForClustering = args.int("maxEpochs", 3)

val wtCoeff = args.double("wtCoeff", 10.0)

val previousKnownFor: TypedPipe[(UserId, Array[(ClusterId, Float)])] =

fromKeyVal(

DAL

.readMostRecentSnapshot(

SimclustersV2RawKnownFor20M145K2020ScalaDataset,

dateRange.embiggen(Days(30)))

.withRemoteReadPolicy(AllowCrossClusterSameDC)

.toTypedPipe,

ModelVersions.Model20M145K2020

)

UpdateKnownForSBFRunner

.runUpdateKnownFor(

TypedPipe

.from(FollowingsCosineSimilaritiesManhattanSource(simsGraphPath))

.map(\_.\_2),

minActiveFollowers,

topK,

maxNeighbors,

tempLocationPath,

previousKnownFor,

maxEpochsForClustering,

squareWeightsEnable,

wtCoeff,

mode

)

.flatMap { updateKnownFor =>

Execution

.zip(

KnownForSources

.toKeyVal(updateKnownFor, ModelVersions.Model20M145K2020)

.writeDALVersionedKeyValExecution(

SimclustersV2RawKnownFor20M145K2020ScalaDataset,

D.Suffix(InternalDataPaths.RawKnownFor2020Path)

),

UpdateKnownForSBFRunner

.evaluateUpdatedKnownFor(updateKnownFor, previousKnownFor)

.flatMap { emailText =>

Util

.sendEmail(

emailText,

s"Change in cluster assignments for new KnownFor ModelVersion: 20M145K2020",

"no-reply@twitter.com")

Execution.unit

}

).unit

}

}

}

}

/\*

knownFor Week-1:

scalding remote run \

--target src/scala/com/twitter/simclusters\_v2/scalding/update\_known\_for:update\_known\_for\_20m\_145k\_2020-adhoc \

--main-class com.twitter.simclusters\_v2.scalding.update\_known\_for.UpdateKnownFor20M145K2020Adhoc \

--submitter atla-aor-08-sr1 --user cassowary \

--submitter-memory 128192.megabyte --hadoop-properties "mapreduce.map.memory.mb=8192 mapreduce.map.java.opts='-Xmx7618M' mapreduce.reduce.memory.mb=8192 mapreduce.reduce.java.opts='-Xmx7618M'" \

-- \

--date 2020-08-30 --maxNeighbors 100 --minActiveFollowers 400 --topK 20000000 --numNodesPerCommunity 200 --maxEpochs 4 --squareWeightsEnable --wtCoeff 10.0 \

--inputSimsDir /atla/proc/user/cassowary/manhattan\_sequence\_files/approximate\_cosine\_similarity\_follow \

--outputClusterDir /user/cassowary/adhoc/your\_ldap/simclusters/clustering\_outputs/output\_clustering\_assignments\_2020\_readAgain\_v4\_week\_1

knownFor Week-2:

scalding remote run \

--target src/scala/com/twitter/simclusters\_v2/scalding/update\_known\_for:update\_known\_for\_20m\_145k\_2020-adhoc \

--main-class com.twitter.simclusters\_v2.scalding.update\_known\_for.UpdateKnownFor20M145K2020Adhoc \

--submitter atla-aor-08-sr1 --user cassowary \

--submitter-memory 128192.megabyte --hadoop-properties "mapreduce.map.memory.mb=8192 mapreduce.map.java.opts='-Xmx7618M' mapreduce.reduce.memory.mb=8192 mapreduce.reduce.java.opts='-Xmx7618M'" \

-- \

--date 2020-08-30 --maxNeighbors 100 --minActiveFollowers 400 --topK 20000000 --numNodesPerCommunity 200 --maxEpochs 4 --squareWeightsEnable --wtCoeff 10.0 \

--inputSimsDir /atla/proc/user/cassowary/manhattan\_sequence\_files/approximate\_cosine\_similarity\_follow \

--inputPreviousKnownForDataSet /user/cassowary/adhoc/your\_ldap/simclusters/clustering\_outputs/output\_clustering\_assignments\_2020\_readAgain\_v4\_week\_1\_KeyVal \

--outputClusterDir /user/cassowary/adhoc/your\_ldap/simclusters/clustering\_outputs/output\_clustering\_assignments\_2020\_readAgain\_v4\_week\_2

\*/

object UpdateKnownFor20M145K2020Adhoc extends TwitterExecutionApp {

implicit val tz: java.util.TimeZone = DateOps.UTC

implicit val dp = DateParser.default

val log = Logger()

def job: Execution[Unit] =

Execution.getConfigMode.flatMap {

case (config, mode) =>

Execution.withId { implicit uniqueId =>

val args = config.getArgs

implicit def valueCodec: BinaryScalaCodec[Candidates] = BinaryScalaCodec(Candidates)

// Step - 1 (DataProcessing): Parameters for getting mapped indices for user-ids

val minActiveFollowers = args.int("minActiveFollowers", 400)

val topK = args.int("topK", 20000000)

// Step - 2 (DataProcessing): Parameters to remove users not in the topK most followed users from simsGraph

val clusterAssignmentOutput = args("outputClusterDir")

val maxNeighbors = args.int("maxNeighbors", 400)

// Step - 3 (Final Clustering): Parameters to run the clustering algorithm

val squareWeightsEnable = args.boolean("squareWeightsEnable")

val maxEpochsForClustering = args.int("maxEpochs", 3)

val wtCoeff = args.double("wtCoeff", 10.0)

val simsGraphPath =

"/atla/proc/user/cassowary/manhattan\_sequence\_files/approximate\_cosine\_similarity\_follow"

// Read in the knownFor dataset, that can be used to initialize the clusters for this week.

val inputPreviousKnownFor: TypedPipe[(Long, Array[(Int, Float)])] =

args.optional("inputPreviousKnownForDataSet") match {

case Some(inputKnownForDir) =>

println(

"Input knownFors provided, using these as the initial cluster assignments for users")

TypedPipe

.from(AdhocKeyValSources.knownForSBFResultsDevelSource(inputKnownForDir))

case None =>

println(

"Using knownFor Assignments from prod as no previous assignment was provided in the input")

if (args.boolean("dec11")) {

KnownForSources

.fromKeyVal(

DAL

.readMostRecentSnapshotNoOlderThan(

SimclustersV2KnownFor20M145KDec11ScalaDataset,

Days(30)).withRemoteReadPolicy(AllowCrossClusterSameDC).toTypedPipe,

ModelVersions.Model20M145KDec11

)

} else {

KnownForSources

.fromKeyVal(

DAL

.readMostRecentSnapshotNoOlderThan(

SimclustersV2KnownFor20M145KUpdatedScalaDataset,

Days(30)).withRemoteReadPolicy(AllowCrossClusterSameDC).toTypedPipe,

ModelVersions.Model20M145KUpdated

)

}

}

UpdateKnownForSBFRunner

.runUpdateKnownFor(

TypedPipe

.from(FollowingsCosineSimilaritiesManhattanSource(simsGraphPath))

.map(\_.\_2),

minActiveFollowers,

topK,

maxNeighbors,

clusterAssignmentOutput,

inputPreviousKnownFor,

maxEpochsForClustering,

squareWeightsEnable,

wtCoeff,

mode

)

.flatMap { updateKnownFor =>

Execution

.zip(

updateKnownFor

.mapValues(\_.toList).writeExecution(TypedTsv(clusterAssignmentOutput)),

updateKnownFor.writeExecution(AdhocKeyValSources.knownForSBFResultsDevelSource(

clusterAssignmentOutput + "\_KeyVal")),

UpdateKnownForSBFRunner

.evaluateUpdatedKnownFor(updateKnownFor, inputPreviousKnownFor)

.flatMap { emailText =>

Util

.sendEmail(

emailText,

s"Change in cluster assignments for new KnownFor ModelVersion: 20M145K2020" + clusterAssignmentOutput,

"no-reply@twitter.com")

Execution.unit

}

).unit

}

}

}

}