package com.twitter.simclustersann.modules

import com.google.inject.Provides

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

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

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

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

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

import com.twitter.storehaus.ReadableStore

import javax.inject.Singleton

import com.twitter.simclustersann.candidate\_source.ApproximateCosineSimilarity

import com.twitter.simclustersann.candidate\_source.ExperimentalApproximateCosineSimilarity

import com.twitter.simclustersann.candidate\_source.OptimizedApproximateCosineSimilarity

import com.twitter.simclustersann.candidate\_source.SimClustersANNCandidateSource

object SimClustersANNCandidateSourceModule extends TwitterModule {

val acsFlag = flag[String](

name = "approximate\_cosine\_similarity",

default = "original",

help =

"Select different implementations of the approximate cosine similarity algorithm, for testing optimizations",

)

@Singleton

@Provides

def provides(

embeddingStore: ReadableStore[SimClustersEmbeddingId, SimClustersEmbedding],

cachedClusterTweetIndexStore: ReadableStore[ClusterId, Seq[(TweetId, Double)]],

statsReceiver: StatsReceiver

): SimClustersANNCandidateSource = {

val approximateCosineSimilarity = acsFlag() match {

case "original" => ApproximateCosineSimilarity

case "optimized" => OptimizedApproximateCosineSimilarity

case "experimental" => ExperimentalApproximateCosineSimilarity

case \_ => ApproximateCosineSimilarity

}

new SimClustersANNCandidateSource(

approximateCosineSimilarity = approximateCosineSimilarity,

clusterTweetCandidatesStore = cachedClusterTweetIndexStore,

simClustersEmbeddingStore = embeddingStore,

statsReceiver = statsReceiver.scope("simClustersANNCandidateSource")

)

}

}