package com.twitter.ann.faiss

import com.twitter.ann.common.Queryable

import com.twitter.ann.common.\_

import com.twitter.search.common.file.AbstractFile

import com.twitter.util.logging.Logging

case class FaissParams(

nprobe: Option[Int],

quantizerEf: Option[Int],

quantizerKFactorRF: Option[Int],

quantizerNprobe: Option[Int],

ht: Option[Int])

extends RuntimeParams {

override def toString: String = s"FaissParams(${toLibraryString})"

def toLibraryString: String =

Seq(

nprobe.map { n => s"nprobe=${n}" },

quantizerEf.map { ef => s"quantizer\_efSearch=${ef}" },

quantizerKFactorRF.map { k => s"quantizer\_k\_factor\_rf=${k}" },

quantizerNprobe.map { n => s"quantizer\_nprobe=${n}" },

ht.map { ht => s"ht=${ht}" },

).flatten.mkString(",")

}

object FaissIndex {

def loadIndex[T, D <: Distance[D]](

outerDimension: Int,

outerMetric: Metric[D],

directory: AbstractFile

): Queryable[T, FaissParams, D] = {

new QueryableIndexAdapter[T, D] with Logging {

protected val metric: Metric[D] = outerMetric

protected val dimension: Int = outerDimension

protected val index: Index = {

info(s"Loading faiss with ${swigfaiss.get\_compile\_options()}")

QueryableIndexAdapter.loadJavaIndex(directory)

}

}

}

}