/\* ----------------------------------------------------------------------------

\* This file was automatically generated by SWIG (http://www.swig.org).

\* Version 4.0.2

\*

\* Do not make changes to this file unless you know what you are doing--modify

\* the SWIG interface file instead.

\* ----------------------------------------------------------------------------- \*/

package com.twitter.ann.faiss;

public class PCAMatrix extends LinearTransform {

private transient long swigCPtr;

protected PCAMatrix(long cPtr, boolean cMemoryOwn) {

super(swigfaissJNI.PCAMatrix\_SWIGUpcast(cPtr), cMemoryOwn);

swigCPtr = cPtr;

}

protected static long getCPtr(PCAMatrix obj) {

return (obj == null) ? 0 : obj.swigCPtr;

}

@SuppressWarnings("deprecation")

protected void finalize() {

delete();

}

public synchronized void delete() {

if (swigCPtr != 0) {

if (swigCMemOwn) {

swigCMemOwn = false;

swigfaissJNI.delete\_PCAMatrix(swigCPtr);

}

swigCPtr = 0;

}

super.delete();

}

public void setEigen\_power(float value) {

swigfaissJNI.PCAMatrix\_eigen\_power\_set(swigCPtr, this, value);

}

public float getEigen\_power() {

return swigfaissJNI.PCAMatrix\_eigen\_power\_get(swigCPtr, this);

}

public void setEpsilon(float value) {

swigfaissJNI.PCAMatrix\_epsilon\_set(swigCPtr, this, value);

}

public float getEpsilon() {

return swigfaissJNI.PCAMatrix\_epsilon\_get(swigCPtr, this);

}

public void setRandom\_rotation(boolean value) {

swigfaissJNI.PCAMatrix\_random\_rotation\_set(swigCPtr, this, value);

}

public boolean getRandom\_rotation() {

return swigfaissJNI.PCAMatrix\_random\_rotation\_get(swigCPtr, this);

}

public void setMax\_points\_per\_d(long value) {

swigfaissJNI.PCAMatrix\_max\_points\_per\_d\_set(swigCPtr, this, value);

}

public long getMax\_points\_per\_d() {

return swigfaissJNI.PCAMatrix\_max\_points\_per\_d\_get(swigCPtr, this);

}

public void setBalanced\_bins(int value) {

swigfaissJNI.PCAMatrix\_balanced\_bins\_set(swigCPtr, this, value);

}

public int getBalanced\_bins() {

return swigfaissJNI.PCAMatrix\_balanced\_bins\_get(swigCPtr, this);

}

public void setMean(FloatVector value) {

swigfaissJNI.PCAMatrix\_mean\_set(swigCPtr, this, FloatVector.getCPtr(value), value);

}

public FloatVector getMean() {

long cPtr = swigfaissJNI.PCAMatrix\_mean\_get(swigCPtr, this);

return (cPtr == 0) ? null : new FloatVector(cPtr, false);

}

public void setEigenvalues(FloatVector value) {

swigfaissJNI.PCAMatrix\_eigenvalues\_set(swigCPtr, this, FloatVector.getCPtr(value), value);

}

public FloatVector getEigenvalues() {

long cPtr = swigfaissJNI.PCAMatrix\_eigenvalues\_get(swigCPtr, this);

return (cPtr == 0) ? null : new FloatVector(cPtr, false);

}

public void setPCAMat(FloatVector value) {

swigfaissJNI.PCAMatrix\_PCAMat\_set(swigCPtr, this, FloatVector.getCPtr(value), value);

}

public FloatVector getPCAMat() {

long cPtr = swigfaissJNI.PCAMatrix\_PCAMat\_get(swigCPtr, this);

return (cPtr == 0) ? null : new FloatVector(cPtr, false);

}

public PCAMatrix(int d\_in, int d\_out, float eigen\_power, boolean random\_rotation) {

this(swigfaissJNI.new\_PCAMatrix\_\_SWIG\_0(d\_in, d\_out, eigen\_power, random\_rotation), true);

}

public PCAMatrix(int d\_in, int d\_out, float eigen\_power) {

this(swigfaissJNI.new\_PCAMatrix\_\_SWIG\_1(d\_in, d\_out, eigen\_power), true);

}

public PCAMatrix(int d\_in, int d\_out) {

this(swigfaissJNI.new\_PCAMatrix\_\_SWIG\_2(d\_in, d\_out), true);

}

public PCAMatrix(int d\_in) {

this(swigfaissJNI.new\_PCAMatrix\_\_SWIG\_3(d\_in), true);

}

public PCAMatrix() {

this(swigfaissJNI.new\_PCAMatrix\_\_SWIG\_4(), true);

}

public void train(long n, SWIGTYPE\_p\_float x) {

swigfaissJNI.PCAMatrix\_train(swigCPtr, this, n, SWIGTYPE\_p\_float.getCPtr(x));

}

public void copy\_from(PCAMatrix other) {

swigfaissJNI.PCAMatrix\_copy\_from(swigCPtr, this, PCAMatrix.getCPtr(other), other);

}

public void prepare\_Ab() {

swigfaissJNI.PCAMatrix\_prepare\_Ab(swigCPtr, this);

}

}