Deeplearning4j - ND4j方法快速索引

ND4J和ND4S是JVM的科学计算库，并为生产环境设计，亦即例程运行速度快，RAM要求低。

主要特点：

- 多用途多维数组对象
- 多平台功能，包括GPU
- 线性代数和信号处理功能

由于易用性上存在的缺口，Java、Scala和Clojure编程人员无法充分利用NumPy或Matlab等数据分析方面最强大的工具。Breeze等其他库则不支持多维数组或张量，而这却是深度学习和其他任务的关键。ND4J和ND4S正得到国家级实验室的使用，以完成气候建模等任务。这类任务要求完成计算密集的模拟运算。

ND4J在开源、分布式、支持GPU的库内，为JVM带来了符合直觉的、Python编程人员所用的科学计算工具。在结构上，ND4J与SLF4J相似。ND4J让生产环境下的工程师能够轻松将算法和界面移植到Java和Scala体系内的其他库内。

创建ndarray

- 创建值全为0： `Nd4j.zeros(nRows, nCols)` `Nd4j.zeros(int...)`
- 创建值全为1： `Nd4j.ones(nRows, nCols)`
- 复制NDArray： `arr.dup()`
- 创建一个行向量或者列向量： `myRow = Nd4j.create(myDoubleArr)`，
  `myCol = Nd4j.create(myDoubleArr,new int[]{10,1})`
- 使用 double[][] 创建二维 NDArray： `Nd4j.create(double[][])`
- 从行或者列进行 NDArray 堆叠： `Nd4j.hstack(INDArray...)` `Nd4j.vstack(INDArray...)`
- 创建元素服从正太分布的 NDArray： `Nd4j.rand(int,int)` `Nd4j.rand(int[])`
- 普通（0,1）范围的 NDArray: `Nd4j.randn(int,int)` `Nd4j.randn(int[])`

获取 NDArray 的属性

- 获取维度： `rank()`
- 只对二维 NDArray 有用的方法，获取行和列数： `rows()` `columns()`
- 第 i 个维度的长度： `size(i)`

- 获取 NDArray 的形状： `shape()`
- 获取所有元素的个数： `arr.length()`
- 判断 NDArray 的类型： `isMatrix()` `isVector()` `isRowVector()` `isRowVector()`

获取或者设定特定的值

- 获取第 i 行，第 j 列的数值： `arr.getDouble(i,j)`
- 获取超过三维 NDArray 的值： `arr.getDouble(int[])`
- 对特定位置进行赋值： `arr.putScalar(int[],double)`

张量操作

- 加上一个值： `arr1.add(myDouble)`
- 减去一个值： `arr1.sub(myDouble)`
- 乘以一个值： `arr.mul(myDouble)`
- 除以一个值： `arr.div(myDouble)`
- 减法反操作（scalar - arr1）： `arr1.rsub(myDouble)`
- 除法反操作（scalar / arr1）： `arr1.rdiv(myDouble)`

元素（Element-Wise）操作

- 加： `arr1.add(arr2)`
- 减： `arr1.sub(arr2)`
- 乘： `arr1.mul(arr2)`
- 除： `arr1.div(arr2)`
- 赋值： `arr1.assign(arr2)`

规约操作

- 所有元素的和： `arr.sumNumber()`
- 所有元素的乘积： `arr.prod()`
- L1或者L2范数： `arr.norm1()` `arr.norm2()`
- 所有元素的标准差： `arr.stdNumber()`

线性代数操作

- 矩阵乘法：`arr1.mmul(arr2)`
- 矩阵转置：`transpose()`
- 获取对角矩阵：`Nd4j.diag(INDArray)`
- 矩阵求逆：`InvertMatrix.invert(INDArray,boolean)`

获取 NDArray 一部分

- 获取一行（仅用于2维 NDArray）：`getRow(int)`
- 获取多行（仅用于2维 NDArray）：`getRows(int...)`
- 设置一行（仅用于2维 NDArray）：`putRow(int,INDArray)`
- 获取前三行，所有列的值：`Nd4j.create(0).get(NDArrayIndex.interval(0,3),NDArrayIndex.all());`

元素级变换（Tanh, Sigmoid, Sin, Log etc）

- 使用 Transform：`Transforms.sin(INDArray)` `Transforms.log(INDArray)` `Transforms.sigmoid(INDArray)`
- 方法1：`Nd4j.getExecutioner().execAndReturn(new Tanh(INDArray))`
- 方法2：`Nd4j.getExecutioner().execAndReturn(Nd4j.getOpFactory().createTransform("tanh",INDArray))`

参考资料：

1. https://nd4j.org/userguide
2. https://nd4j.org/cn/index
3. 使用Nd4j实现PCA降维：https://github.com/deeplearning4j/nd4j/blob/master/nd4j-backends/nd4j-api-parent/nd4j-api/src/main/java/org/nd4j/linalg/dimensionalityreduction/PCA.java
4. ND4j基本操作代码示例： https://github.com/sjsdfg/dl4j-tutorials

更多文档可以查看 https://github.com/sjsdfg/deeplearning4j-issues。
你的star是我持续分享的动力