Sparse Linear Algebra
in the Deeplearning4j Framework

Thèse n. 1234 2011
présenté le 12 Mars 2011
à la Faculté des Sciences de Base
laboratoire SuperScience
programme doctoral en SuperScience
École Polytechnique Fédérale de Lausanne

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

pour l'obtention du grade de Docteur ès Sciences
par

Paolino Paperino

acceptée sur proposition du jury:

Prof Name Surname, président du jury
Prof Name Surname, directeur de thèse
Prof Name Surname, rapporteur
Prof Name Surname, rapporteur
Prof Name Surname, rapporteur

Lausanne, EPFL, 2011

Wings are a constraint that makes
it possible to fly.
— Robert Bringhurst

To my parents…

Acknowledgements

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Lausanne, 12 Mars 2011 D. K.

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Key words:

Contents

Acknowledgements **i**

Abstract **iii**

List of figures **vii**

List of tables **ix**

Introduction **1**

1 Sparse Data **3**
1.1 Definition 3
1.2 The Frequency of Sparse Data 3
1.3 Real application 4
1.4 What Sparse linear Algebra resolve and optimize 4
1.5 Formats 4
1.5.1 Matrices 4
1.5.2 Tensors - Multi-dimensional arrays 4

2 Deeplearning4j Library Overview **5**
2.1 Structure of the library 5
2.2 Backend ? 5

3 Sparse Linear Algebra **7**

4 Sparse Formats **9**
4.1 Vectors and Matrices 9
4.1.1 Coordinate format (COO) 9
4.1.2 Compressed Row Format (CSR) 10
4.1.3 Compressed Colum Format (CSC) 10
4.2 Tensors / N-dimensional arrays 10

5 Sparse Matrix and Vector in Dl4J **13**
5.1 Architecture of Arrays in Dl4j 13
5.2 First COO implementation 13
5.2.1 Limitations 13

5.3 CSR format implementation . . . 13
5.3.1 ? . . . 14
5.3.2 Adding a value . . . 14
5.3.3 Retrieve a value . . . 14
5.3.4 Representation . . . 14
5.3.5 Operations . . . 14
5.3.6 Limitations . . . 14

6 Sparse Tensor in Dl4j **15**

7 Mathematics **17**
7.1 Very important formulas . . . 17

A An appendix **19**

Bibliography **21**

List of Figures

4.1 A matrix stored in COO format 9
4.2 A matrix stored in CSR format 10
4.3 A tensor stored in COO format 11

List of Tables

Introduction

A non-numbered chapter...

Nowadays Machine learning is very popular and widely-used to resolve daily life problem

- deeplearning, neural net, self driving car etc

To work correctly and expect a accurate result, those machine learning problems require a huge amount of data. Such datasets are challenging regarding the execution time of the algorithms, the memory space required, the network usage when working imn a distributed environnement, etc

A big part of those problems used sparse datasets. For example a recommender system typically works with a dataset high-sparsity. This dataset contains the rating of movies or products given by the users. But usually the user only rated a very small subset of products.

That means, that if we know that our dataset will be sparse, we can used the sparse linear algebra to resolve the problem to optimized the memory used and the executing time.

Deeplearing4j didn't support any sparse format for vectors, matrices or tensors, neither the operations.

1 Sparse Data

Definition

Data are said sparse when it is contains only a few non-null values. That kind of dataset are really common in Machine Learning application and can be an high influence on the computation.

The sparsity of a dataset is defined by :

$$sparsity = \frac{\text{\# non-null values}}{\text{\# values}} \tag{1.1}$$

Conversely when a dataset has only a few null values, the data are said dense. The density of the dataset is defined by the inverse of the sparsity:

$$density = \frac{1}{\text{sparsity}} \tag{1.2}$$

Using dense methods and data structure with sparse data could have a severe bad impact on the performance

The Frequency of Sparse Data

Sparsity is a very useful property in Machine Learning. Some algorithms can have fast optimization, fast evaluation of the model, statistical robustess or other computational advantages. A lot of machine learning application are using sparse dataset such as recommender system, natural language processing algorithm,

Real application

In 2008 Netflix launched a contest, the Netflix Grand Prize [net()], to improve their recommender system model and to increase the accuracy of predictions and published an sample dataset made with the ratings of anonymous Netflix customers. The dataset had more than 100 millions sampled ratings and it contained about $m = 480'186$ users and $m = 17'770$ movies [Koren(2009)]. If stored as a dense matrix, it would need to store $8'532'905'220$ values in memory. That corresponds to a sparsity $\cong \frac{100'000'000}{8'532'905'220} = 0.011719338$.

Storing more than 8 trillions 64-bit floating-point numbers needs more than 64 gigabyte of memory which quickly become unmanageable even for the world's fastest supercomputers. That is why it is absolutely necessary to encode the matrix into a sparse format to drastically decrease the amount of memory needed.

What Sparse linear Algebra resolve and optimize

Formats

Matrices

Coordinates Format

Compressed Row Format

Compressed Column Format

Tensors - Multi-dimensional arrays

Coordinates Format

Compressed Fiber Format

2 Deeplearning4j Library Overview

Deeplearning4j is a open-source Deep Learning library for the JVM. It runs on distributed CPU's and GPU's.

Structure of the library

The library is composed by several sub-libraries:

Deeplearning4j provides the tools to implement neural networks and build computation graphs

Nd4j is the mathematical back-end of Deeplearning4j. It provides the data structures for the n-dimensional arrays and allow Java to access the native libraries via Libnd4j.

Libnd4j is the computing library that provides native operations on CPU and GPU. It's written in C++ and Cuda.

Datavec provides the operations for the data processing such that data ingestion, normalization and transformation into feature vectors.

Backend ?

3 Sparse Linear Algebra

A sparse matrix is matrix that contains only a very few non-zero element. Conversly, a matrix which contains mostly non-zero elements are dense.

The sparsity coefficient is defined by the number of non-zero element divided by the total number of element in the array.

density = 1 - sparsity

For example the matrix ...

has a sparsity of $\frac{4}{15}$ and a density of $\frac{11}{15}$.

4 Sparse Formats

Vectors and Matrices

There exists several different formats to store a sparse array. The idea behind using a sparse format instead of the classic dense one, is to reduce the memory space and the executing time of the operations. Knowing that a matrix is sparse allows to shortcut some operation steps. For example during a matrix multiplication, we can avoid to perform the multiplication for the zero elements of the sparse matrix.

Coordinate format (COO)

This format is the simplest format to encode a sparse array. The coordinates and the value of each non-zero entry are stored in arrays. Typically each element are encoded in a tuple (row, column, value)

Some implementation variations of the COO format exist. The elements can be sorted along a dimension, or it can be some duplicate indexes.

$$A_{(M\times N)} = \begin{bmatrix} 0 & 2 & 0 \\ 0 & 0 & 3 \\ 1 & 0 & 4 \\ 0 & 0 & 0 \end{bmatrix} \quad \rightarrow \quad \begin{aligned} Values_{(1\times NNZ)} &= \begin{bmatrix} 2 & 3 & 1 & 4 \end{bmatrix} \\ Rows_{(1\times NNZ)} &= \begin{bmatrix} 0 & 1 & 2 & 2 \end{bmatrix} \\ Columns_{(1\times NNZ)} &= \begin{bmatrix} 1 & 2 & 0 & 2 \end{bmatrix} \end{aligned}$$

Figure 4.1: A matrix stored in COO format

With this format it's easy and fast to retrieve the value given an index and to insert a new non-zero element.. It's also fast and simple to convert into a dense format.

But this format don't minimize the memory space. It can be reduced with a compressed format such as CSR or CSC as described below.

$$A_{(N\times M)} = \begin{bmatrix} 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 4 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix} \rightarrow \begin{aligned} Values_{(1\times NNZ)} &= \begin{bmatrix} 2 & 3 & 1 & 4 & 2 & 1 \end{bmatrix} \\ Columns_{(1\times NNZ)} &= \begin{bmatrix} 1 & 2 & 0 & 2 & 2 & 3 \end{bmatrix} \\ pointersB_{(1\times N)} &= \begin{bmatrix} 0 & 1 & 2 & 2 & 4 \end{bmatrix} \\ PointersE_{(1\times N)} &= \begin{bmatrix} 1 & 2 & 2 & 4 & 6 \end{bmatrix} \end{aligned}$$

Figure 4.2: A matrix stored in CSR format

Compressed Row Format (CSR)

The Compressed Row and the Compressed Column formats are the most general format to store a sparse array. They don't store any unnecessary element. But it requires more steps to access the elements than the COO format.

Each non-zero element of a row are stored contiguously in the memory. Each row are also contiguously stored.

The format requires four arrays:

Values All the nonzero values are store contiguously in an array. The array size is NNZ.

Column pointers This array keeps the column position for each values.

Beginning of row pointers Each pointer i points to the first element of the row i in the values array. The array size is the number of rows of the array.

End of row pointers Each pointer i points to the first element in the values array that does not belong to the row i . The array size is the number of rows of the array.

Compressed Colum Format (CSC)

The Compressed Column Format is similar to CSR but it compresses columns instead of rows.

Given a matrix $N \times M$, the pointers arrays will have a size M.

Tensors / N-dimensional arrays

A tensor is a multi-dimensional array. The order of the tensor is the dimensionality of the array needed to represent it. Matrices and vectors can be represented as tensors where the order is equals to 2 and 1 respectively.

This generalization allows a more generic implementation of a n-dimensional array in the Nd4j library.

Coordinate format (COO)

The COO format can easily be extended to encode tensors by storing an array of indexes instead the row and column coordinates.

A array of order $K = 3$ with shape $N \times M \times P$ which has the following non-zero values :

value	indexes
1	0 1 0
2	1 1 2
3	1 2 0
4	2 0 1
5	2 2 0

can be encoded with one values array and one indexes array :

$$Values_{(1\times NNZ)} = \begin{bmatrix} 1, & 2, & 3, & 4, & 5 \end{bmatrix}$$
$$Indexes_{(NNZ\times K)} = \begin{bmatrix} [0,1,0], & [1,1,2], & [1,2,0], & [2,0,1], & [2,2,0] \end{bmatrix}$$

Figure 4.3: A tensor stored in COO format

Compressed Sparse Fiber

5 Sparse Matrix and Vector in Dl4J

Architecture of Arrays in Dl4j

The shape of a array object in Dlj4 is defined through an java interface named INDArray (Interface for N-Dimension Array). This interface contains all the methods needed to put a new values in the array, to retrieve a slice or a value, to compare, to perform operations, etc.

The dense array has a base implementation that implements the interface and which is extended by the cuda and cpu inplementation.

First COO implementation

The first implementation of a sparse format was the COO format for vectors and matrices. I used three databuffers to store the array. One for the values, one for the row coordinates and one for the column coordinates.

Limitations

That implementation cannot be extended to n-dimensional array; separate the coordinates into two arrays limits the number of dimension the array can have. Moreover the implementation of the basic methods was too simple and didn't fit in the interface of the ndarray in Dl4j.

CSR format implementation

For a generalized version of a sparse representation in dl4j, I choose to use the CSR format. This format has the advantages to consume less memory to store a matrix since one dimension is compressed and can be directly use with BLAS operations of Intel MKL Sparse BLAS [mkl()].

?

```
object {
def main() : Unit = { println(new A().foo(-41)); }
}

class A {
def foo(i : Int) : Int = {
var j : Int;
if(i < 0) { j = 0 - i; } else { j = i; }
return j + 1;
}
}
```

Adding a value

Retrieve a value

Representation

Operations

Limitations

6 Sparse Tensor in Dl4j

7 Mathematics

In this chapter we will see some examples of mathematics.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Very important formulas

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} P_0 \\ P_1 \\ P_T \end{bmatrix} = \begin{bmatrix} \frac{P_1}{\tau_{10}} + \frac{P_T}{\tau_T} - \frac{P_0}{\tau_{ex}} \\ -\frac{P_1}{\tau_{10}} - \frac{P_1}{\tau_{isc}} + \frac{P_0}{\tau_{ex}} \\ \frac{P_1}{\tau_{isc}} - \frac{P_T}{\tau_T} \end{bmatrix} \tag{7.1}$$

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

$$\bar{I}_f(\vec{r}) = \gamma(\vec{r})\left(1 - \frac{\tau_T P_T^{eq}\left(1 - \exp\left(-\frac{(T_p - t_p)}{\tau_T}\right)\right)}{1 - \exp\left(-\frac{(T_p - t_p)}{\tau_T} + k_2 t_p\right)} \times \frac{\left(\exp\left(k_2 t_p\right) - 1\right)}{t_p}\right) \tag{7.2}$$

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

A An appendix

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Bibliography

[mkl()] Intel MKL sparse format. https://software.intel.com/en-us/mkl-developer-reference-c-sparse-blas-csr-matrix-storage-format.

[net()] Netflix grand prize. http://www.netflixprize.com.

[Koren(2009)] Yehuda Koren. 1 the bellkor solution to the netflix grand prize, 2009.