Matrices

16/11/2021

Orthonormal Vectors

- Let q_1, q_2, ……. , q_n be vectors, they are said to be orthonormal

$$\text{if } q_i^T q_j = \begin{cases} 0 \;\; if \;\; 1 \neq j \\ 1 \;\; if \;\; 1 = j \end{cases}$$

In other words, the length of each vector is 1

Orthogonal Matrix

- An orthogonal matrix Q is a square matrix with orthonormal columns

$$Q = \begin{bmatrix} | & | & | \\ q_1 & q_i & q_n \\ | & | & | \end{bmatrix}$$

$$Q^T Q = \begin{bmatrix} - & q_1^T & - \\ - & q_j^T & - \\ - & q_n^T & - \end{bmatrix} \begin{bmatrix} | & | & | \\ q_1 & q_i & q_n \\ | & | & | \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathrm{I}$$

When row i of Q^T multiplies column j of Q, the result is

$$q_j^T q_j = 0$$

On the diagonals where i =j, we have $q_j^T q_j = 1$

Orthogonal Matrix

An orthonormal matrix is a type of square matrix whose columns and rows are orthonormal unit vector, eg. Perpendicular and have a length or magnitude of 1.

Then Q^T is Q^{-1}

ie. $Q^T Q = Q Q^T = I$

Computing of Q^T is more time efficient as compared to computing Q^{-1}

- Eg
- $Q = \begin{bmatrix} \cos\theta & -sin\theta \\ sin\theta & \cos\theta \end{bmatrix}$
- $Q^T = Q^{-1} = \begin{bmatrix} \cos\theta & sin\theta \\ -sin\theta & \cos\theta \end{bmatrix}$

Orthogonal Matrix

- An orthonormal matrix is used a lot for linear transformations, such as reflections and permutations.

  A simple 2x2 orthogonal matrix given below is an example of reflection matrix or coordinate matrix.

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

- Computing of Q^T is more time efficient as compared to computing Q^{-1}

Sparse Matrix

- A sparse matrix is a matrix that is comprised of mostly zero values. Sparse matrices are distinct from matrices with mostly non-zero values, which are referred to as dense matrices.
- A matrix is sparse if many of its coefficients are zero. The interest in sparsity arises because its exploitation can lead to enormous computational savings and because many large matrix problems that occur in practice are sparse.

- The sparsity of a matrix can be quantified with a score, which is the number of zero values in the matrix divided by the total number of elements in the matrix.

$$\text{Sparsity} = \frac{\text{count of zero elements}}{\text{total elements}}$$

- Below is an example of a small 3 × 6 sparse matrix.
- $A = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}$
- The example has 13 zero values of the 18 elements in the matrix, giving this matrix a sparsity score of 0.722 or about 72%.

Space Complexity

- In practice, most large matrices are sparse — almost all entries are zeros.
- An example of a very large matrix that is too large to be stored in memory is a link matrix that shows the links from one website to another.
- An example of a smaller sparse matrix might be a word or term occurrence matrix for words in one book against all known words in English.
- In both cases, the matrix contained is sparse with many more zero values than data values. The problem with representing these sparse matrices as dense matrices is that memory is required and must be allocated for each 32-bit or even 64-bit zero value in the matrix. This is clearly a waste of memory resources as those zero values do not contain any information.

Time Complexity

- Assuming a very large sparse matrix can be fit into memory, we will want to perform operations on this matrix. Simply, if the matrix contains mostly zero-values, i.e. no data, then performing operations across this matrix may take a long time where the bulk of the computation performed will involve adding or multiplying zero values together.
- This is a problem of increased time complexity of matrix operations that increases with the size of the matrix. This problem is compounded when we consider that even trivial machine learning methods may require many operations on each row, column, or even across the entire matrix, resulting in vastly longer execution times.

Sparse Matrices in Machine Learning

- Sparse matrices turn up a lot in applied machine learning. Some common examples to motivate you to be aware of the issues of sparsity.

Data

- Sparse matrices come up in some specific types of data, most notably observations that record the occurrence or count of an activity. Three examples include:
- Whether or not a user has watched a movie in a movie catalogue.
- Whether or not a user has purchased a product in a product catalogue.
- Count of the number of listens of a song in a song catalogue.

Data Preparation

- Sparse matrices come up in encoding schemes used in the preparation of data. Three common examples include:
- One hot encoding, used to represent categorical data as sparse binary vectors.
- Count encoding, used to represent the frequency of words in a vocabulary for a document
- TF-IDF encoding, used to represent normalized word frequency scores in a vocabulary.

Areas of Study

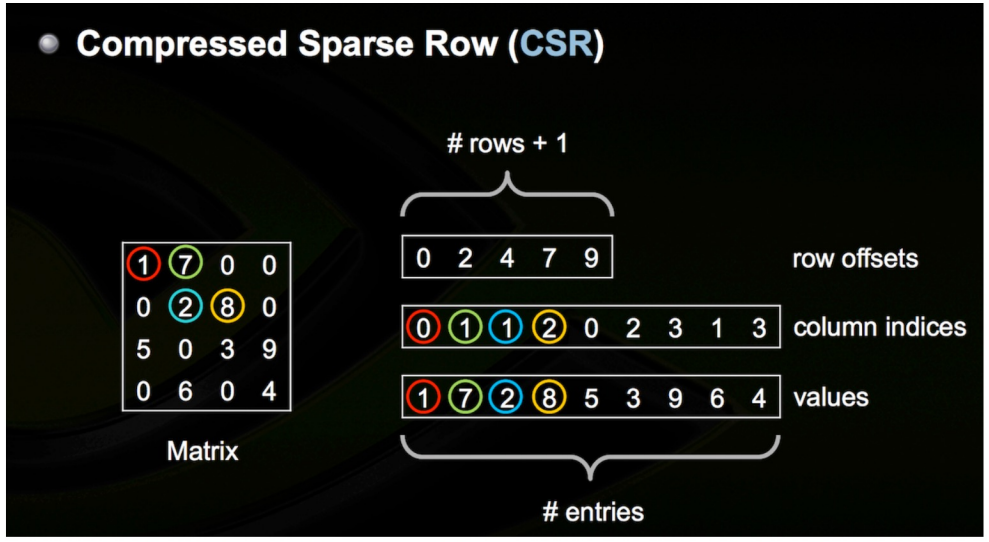
- Some areas of study within machine learning must develop specialized methods to address sparsity directly as the input data is almost always sparse. Three examples include:
- Natural language processing for working with documents of text.
- Recommender systems for working with product usage within a catalog.
- Computer vision when working with images that contain lots of black pixels.
- If there are 100,000 words in the language model, then the feature vector has length 100,000, but for a short email message almost all the features will have count zero.

Working with Sparse Matrices

- The solution to representing and working with sparse matrices is to use an alternate data structure to represent the sparse data. The zero values can be ignored and only the data or non-zero values in the sparse matrix need to be stored or acted upon. There are multiple data structures that can be used to efficiently construct a sparse matrix; three common examples are listed below.
- Dictionary of Keys. A dictionary is used where a row and column index is mapped to a value.
- List of Lists. Each row of the matrix is stored as a list, with each sublist containing the column index and the value.
- Coordinate List. A list of tuples is stored with each tuple containing the row index, column index, and the value.

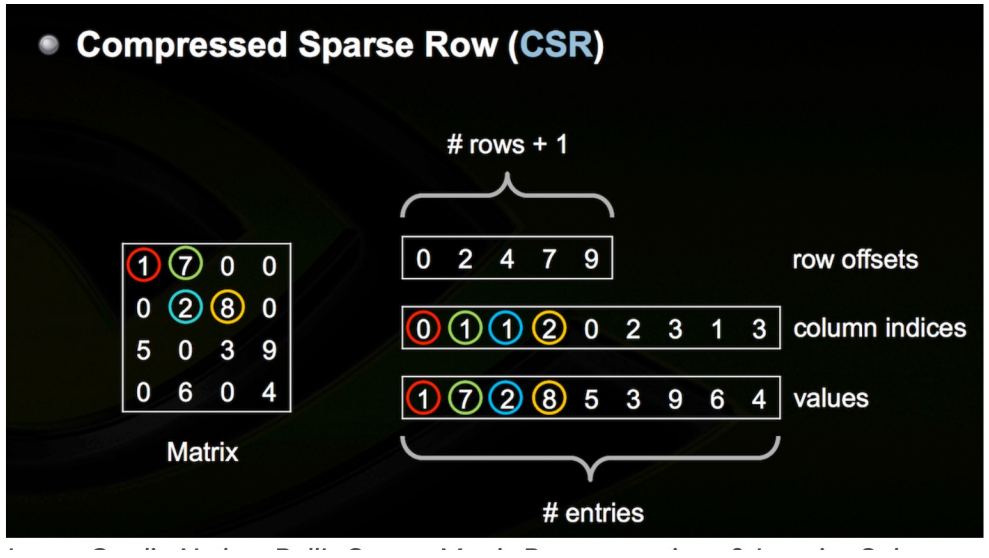
- There are also data structures that are more suitable for performing efficient operations; two commonly used examples are listed below.
  - Compressed Sparse Row. The sparse matrix is represented using three one-dimensional arrays for the non-zero values, the extents of the rows, and the column indexes.
  - Compressed Sparse Column. The same as the Compressed Sparse Row method except the column indices are compressed and read first before the row indices.
  - The Compressed Sparse Row, also called CSR for short, is often used to represent sparse matrices in machine learning given the efficient access and matrix multiplication that it supports.

Compressed Sparse Row



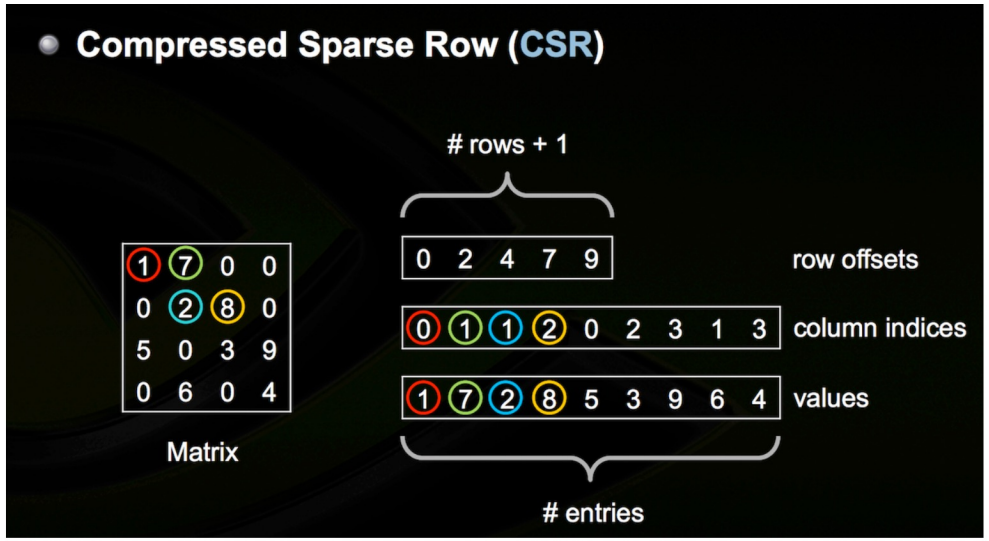
- The first step is to populate the first array. It always starts with 0. Since there are two nonzero values in row 1, we update our array like so [0 2]. There are 2 nonzero values in row 2, so we update our array to [0 2 4]. Doing that for the remaining rows yields [0 2 4 7 9]. By the way, the length of this array should always be the number of rows + 1.

Compressed Sparse Row



- Step two is populating the second array of column indices. Note that the columns are zero-indexed. The first value, 1, is in column 0. The second value, 7, is in column 1. The third value, 2, is in column 1. And so on. The result is the array [0 1 1 2 0 2 3 1 3].

Compressed Sparse Row



- Finally, we populate the third array which looks like this [1 7 2 8 5 3 9 6 4]. Again, we are only storing nonzero values.