Topic Modeling with GraphBLAS

Vijay Gadepally

October 2, 2014

Abstract

Topic modeling is a very common tool for individuals working with big data. Topic modeling is a great way to quickly understand common themes in a large dataset and it used widely for a variety of purposes. One of the common techniques for topic modeling is known as non-negative matrix factorization. This article provides an introduction to NMF and lays out the mathematical foundations in terms of matrix operations. Finally, we map these matrix operations to the GraphBLAS set of kernels.

1 Introduction

Topic modeling is a very popular algorithm that provides an intuitive look into the topics that make up data. Very commonly, people make use of

Non-negative matrix factorization (NMF) is a class of tools used to factorize a given matrix into two matrices. Multiplying these two matrices produces the original matrix. Consider a matrix A_{mxn} to be factored into matrices W_{mxk} and H_{kxn} where m corresponds to the number of rows of A, n corresponds to the number of columns in A, and k corresponds to the number of topics. Further, NMF enforces the constraint that none of these matrices contain any negative elements.

By definition,

$$A = W * H \tag{1}$$

1.1 NMF through Iteration

One way to find the matrices W, H such that $A \approx W * H$ is through an iterative technique. An algorithm is provided in Algorithm 1.

Data: Adjacency Matrix A (size mxn), number of topics k
Result: W and H
initialization;
W = random m x k matrix
while $\|A - W * H\|_F > threshold$ **do**
  Solve $W^T * W * H = W * A$ for H
  Set elements in $H < 0$ to 0
  Solve $H * H^T * W^T = H * A^T$ for W
  Set elements in $W < 0$ to 0
end

Algorithm 1: NMF through Iteration

In order to solve the equations in Algorithm 1, it is necessary to find a least squares solution to a system of linear equations for W and H. One way of doing this is by finding finding the matrix inverse of $W^T * W$ and $H * H^T$ (both are square matrices) and multiplying with the RHS of the equations. The matrix inverse may traditionally be found using the SVD, however, in order to use the GraphBLAS bulding blocks, it is necessary we represent the matrix inversion through other GraphBLAS kernels such as SpMV, SpGEMM, etc.

1.2 Matrix Inverse through Iteration

One method to find the matrix inverse is typically done by doing an eigen-decomposition of a matrix. Another way to find the matrix inverse is to use something similar to inverse iteration (used by iterative eigenvalue solvers). In such a system, for iteration k: $X_{k+1} = X_k * (2I - AX_n)$. An algorithm for solving the matrix inverse of a square matrix is given in Algorithm 2.

Data: Matrix A to invert
Result: $X = A^{-1}$
initialization;
$\|A_{row}\| = \max_i(\sum_j A_{ij})$
$\|A_{col}\| = \max_j(\sum_i A_{ij})$
$X_1 = A^T/(\|A_{row}\| * \|A_{col}\|)$
while *for some time* **do**
  $X_{t+1} = X_t * (2 * I_{nxn}) - A * X_t$
end

Algorithm 2: Matrix inverse through Iteration

Algorithm 2 only makes use of SpGEMM, SpRed.

1.3 Combining

Combining Algorithms 1 and 2, we can find the NMF for a matrix A. Where $(W^T * W)^{-1}$ and $(H * H^T)^{-1}$ are determined by using the relation develop in Algorithm 2.

Data: Adjacency Matrix A (size mxn), number of topics k
Result: W and H
W = random m x k matrix
while *Frobenius norm of* $A - W * H > threshold$ **do**
  Solve $H = (W^T * W)^{-1} * W * A$ for H
  Set elements in $H < 0$ to 0
  Solve $W^T = (H * H^T)^{-1} * H * A^T$ for W
  Set elements in $W < 0$ to 0
end

Algorithm 3: NMF and Inverse through Iteration

Using the above algorithm, it is possible to find the non-negative matrix factorization of a given array A using only matrix multiplication, reduction, addition/subtraction and subsref.

2 Mapping NMF to GraphBLAS

In order to compute the non-negative matrix factorization expressed in Algorith 3, we need a few Graph Building Blocks

To compute the NMF, we need for standard (+,*) semiring operations:

1. SpRef/SpAsgn: To assign arrays and results to sparse arrays
2. SpGEMM: To multiply arrays
3. SpEWiseX: To set non-negative portions to 0
4. Transpose Operation

In order to compute the inverse of a square matrix, we need the following operations:

1. SpRef/SpAsgn: To assign arrays and results to sparse arrays
2. SpGEMM: To compute array products in loop
3. Scale: To compute $2 * I$
4. Reduce: To compute the row and column norms of input matrix
5. Addition/Subtraction: Can be realized by SpGEMM over (min,+) semiring

3 Conclusion

The non-negative matrix factorization of a sparse array can give some interesting insight into parts of the array that have affinity with each other. For analysts, it can be a good first step to see what their data contains. As a potential algorithm to be written using GraphBLAS blocks, the algorithm outlined in this article provides a first step implementation using only GraphBLAS kernels.

References

[1] Tim Mattson. Gabb'14: Graph algorithms building blocks workshop. In *IEEE IPDPS*.

[2] Victor Pan and Robert Schreiber. An improved newton iteration for the generalized inverse of a matrix, with applications. *SIAM Journal on Scientific and Statistical Computing*, 12(5):1109–1130, 1991.

[3] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.

[4] Aydın Buluç and John R Gilbert. The combinatorial blas: Design, implementation, and applications. *International Journal of High Performance Computing Applications*, page 1094342011403516, 2011.

[5] David Bader, Aydın Buluç, Joseph Gonzalez UCB, Jeremy Kepner, and Tim Makson. The graph blas effort and its implications for exascale. 2014.