Genomics Proteomics Bioinformatics 12 (2014) 48–51

Genomics Proteomics Bioinformatics

www.elsevier.com/locate/gpb
www.sciencedirect.com

APPLICATION NOTE

CloudNMF: A MapReduce Implementation of Nonnegative Matrix Factorization for Large-scale Biological Datasets

Ruiqi Liao [1], Yifan Zhang [1], Jihong Guan [3], Shuigeng Zhou [1,2,*]

[1] *School of Computer Science, Fudan University, Shanghai 200433, China*
[2] *Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 200433, China*
[3] *Department of Computer Science and Technology, Tongji University, Shanghai 200092, China*

Received 23 May 2013; revised 21 June 2013; accepted 26 June 2013
Available online 8 August 2013

KEYWORDS

Nonnegative matrix factorization;
MapReduce;
Bioinformatics

Abstract In the past decades, advances in high-throughput technologies have led to the generation of huge amounts of biological data that require analysis and interpretation. Recently, nonnegative matrix factorization (NMF) has been introduced as an efficient way to reduce the complexity of data as well as to interpret them, and has been applied to various fields of biological research. In this paper, we present CloudNMF, a distributed open-source implementation of NMF on a MapReduce framework. Experimental evaluation demonstrated that CloudNMF is scalable and can be used to deal with huge amounts of data, which may enable various kinds of a high-throughput biological data analysis in the cloud. CloudNMF is freely accessible at http://admis.fudan.edu.cn/projects/CloudNMF.html.

Introduction

The explosion of biological data brought about by the high-throughput technologies poses a great challenge to bioinformatics research. In order to learn the hidden structures of these high-dimensional data, nonnegative matrix factorization (NMF) [1] was introduced into biological research. NMF was quickly applied to various fields of biological data analysis, such as capturing expression pattern in microarray data [2], discovery of cancer subtypes [3], clustering of gene expression data [4,5], identification of histone modification modules [6], biological text mining [7,8], *etc*. The intrinsic nature of the NMF method makes it very suitable for an integrative analysis of multi-dimensional genomics data [9]. Devarajan presented a comprehensive review of the application of NMF to computational biology [10].

With the increasing dimensionality of biological data, it is foreseeable that the application of NMF to biological research will continue to grow. For example, sequencing technologies are generating terabytes (TBs) or even petabytes (PBs) of data for a multi-dimensional analysis. However, current implementations of NMF in the biology area can only deal with matrices of thousands-by-thousands size. For example, bioNMF [11],

* Corresponding author.
E-mail: sgzhou@fudan.edu.cn (Zhou S).

Peer review under responsibility of Beijing Institute of Genomics, Chinese Academy of Sciences and Genetics Society of China.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.gpb.2013.06.001

Table 1 Algorithm for CloudNMF

Input: nonnegative matrix A, dimension k, iteration number i
Output: nonnegative matrices W and H
1: initiate W and H using random nonnegative values
2: for each iteration:
3: calculate $X_1 = W^T A$ using two MapReduce steps
4: calculate $Y_1 = W^T WH$ using two MapReduce steps
5: update H with $H = H.*X_1/Y_1$ using one MapReduce step
6: calculate $X_2 = AH^T$ using two MapReduce steps
7: calculate $Y_2 = WHH^T$ using two MapReduce steps
8: update W with $W = W.*X_2/Y_2$ using one MapReduce step
9: output W and H

an implementation of NMF for bioinformatics analysis, can only handle matrices of 4096×512 (according to the documentation of bioNMF server), and thus would fail to process data with more attributes or samples. Another implementation using R [12] fails to work when data size reaches gigabytes (GBs) in a standalone machine. In their original papers, both implementations were used to analyze a microarray dataset represented by a 5000×38 gene expression data matrix [13]. However, a much more scalable implementation will be needed to deal with data of a significantly greater size such as protein–protein interaction (PPI) data or sequencing data.

To facilitate biological data analysis in the "Big Data" era [14], we present CloudNMF, an open-source implementation of NMF in MapReduce framework. The implementation was developed on the Hadoop platform and can enable the nonnegative factorization of sparse matrices up to million-by-million size. Furthermore, CloudNMF is provided as a JAR file ready to be deployed anywhere. In particular, CloudNMF can be easily deployed on Amazon Elastic MapReduce to utilize the power of cloud computing for biological data analysis.

Methods

NMF was first introduced by Lee and Seung as a method for learning the substructure of data matrix [1]. It was defined as the factorization of a nonnegative matrix A into the multiplication of two other nonnegative matrices W and H, where A is a $m \times n$ matrix, W and H are $m \times k$ and $k \times n$ matrices, where k is the target dimensionality to be reduced to, which is a number smaller than the minimum of m and n. NMF was aimed at minimizing the Euclidian distance between A and WH, and can be used as an effective technique for dimension reduction and unsupervised clustering. In 2010, Liu et al. proposed an algorithm to perform NMF in the MapReduce framework [15] and showed that the algorithm can be used to factorize huge nonnegative matrices up to millions-by-millions size. However, this algorithm was aimed at Web applications, and no source code of the algorithm is available for public use. Our work is the first open-source implementation of NMF in the MapReduce framework, targeted at dealing with the explosion of biological data.

Our work follows the method previously reported [15], which is based on the well-known iterative updating rule of W and H described by Lee and Seung [16].

$$H \leftarrow H \,.* \frac{W^T A}{W^T WH} \tag{1}$$

$$W \leftarrow W \,.* \frac{AH^T}{WHH^T} \tag{2}$$

Here, $.*$ denotes dot product and ^{T} denotes transpose of matrix.

Similar to the method used in [15], for each iteration, the updating of H and W are both factorized into five MapReduce steps; the computation of each step can be easily distributed into multiple machines to achieve speedup, please see **Table 1** for the details of the algorithm.

The program was implemented using Java and was packaged as a JAR file which can run on local Hadoop clusters (**Figure 1**). We offered a command-line interface for the program; the usage of the command-line interface is also provided in our website (http://admis.fudan.edu.cn/projects/CloudNMF.html). Moreover, Amazon Elastic MapReduce service (http://aws.amazon.com/cn/elasticmapreduce/) offers on-demand computing clusters preinstalled with Hadoop, and provides a web interface to run Hadoop JAR files using only a web browser (see http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/CLI_JobFlowUsingCustomJAR.html). For those inexperienced users who find it hard to build their own Hadoop clusters, it is possible to upload their data and CloudNMF into the cloud and perform their analysis remotely (**Figure 2**).

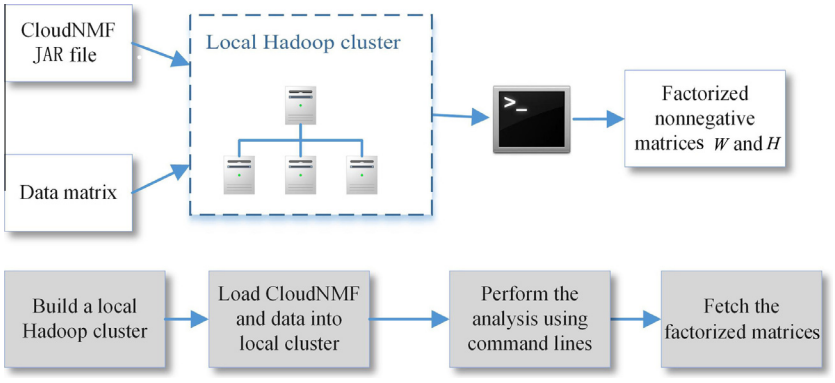

Figure 1 Using CloudNMF with a local Hadoop cluster

Experimental evaluation

In order to test the performance of our program, we applied the program to both real data and simulated matrices. The PPI data matrix from the STRING database [17] was used for performance testing, which includes 108,133,799 protein interactions from 1134 species. The dataset can be represented by a $1{,}349{,}909 \times 1{,}349{,}909$ matrix, where 1,349,909 is the number of distinct proteins in the dataset. Since the interactions between proteins are both nonnegative and sparse, the dataset is quite suitable for the application of NMF.

Based on the STRING dataset, three submatrices of different sizes were generated. The four datasets are described in Table S1 and the performance of CloudNMF for these four datasets is summarized in Table S2. We also generated three simulated matrices of different sizes but containing the same number of nonzero elements to test the impact of matrix size on the performance of the program (Table S3). The experiments were performed on an 8-machine Hadoop cluster, and each machine has a Duo Core CPU and 4 GB memory.

From **Figure 3** we observed a very interesting feature of CloudNMF: the runtime actually increases in proportion to the number of nonzero elements (the number of PPIs) in the matrix (Figure 3A). This may be attributed to the MapReduce implementation of the algorithm: only nonzero elements are stored and distributed for computation. As the size of the matrix grows, the computation time increases logarithmically (Figure 3B). These features make the algorithm better to deal with sparse nonnegative matrices in comparison with the traditional implementations.

Discussion

CloudNMF is the first open-source implementation of MapReduce-based nonnegative matrix factorization, and is capable of handling significantly a greater size of data than existing NMF implementations in bioinformatics. Besides being deployed in local Hadoop clusters, CloudNMF can also be easily used on cloud computing platforms such as Amazon Web Services via only a web browser. Moreover, experimental results show that the algorithm can effectively deal with sparse matrices such as protein–protein interaction networks.

CloudNMF also has some limitations. Although the program achieved considerable performance when dealing with large-size matrices, with the high overhead of MapReduce paradigm, it may be less efficient than existing implementations to deal with small-size matrices. In addition, while bioinformatics analyses using NMF may involve many pre-processing or post-processing steps, we only implemented the basic NMF algorithm. However, the code of CloudNMF is freely accessible at our website; users can integrate the code into their own pipelines to perform more specific analyses.

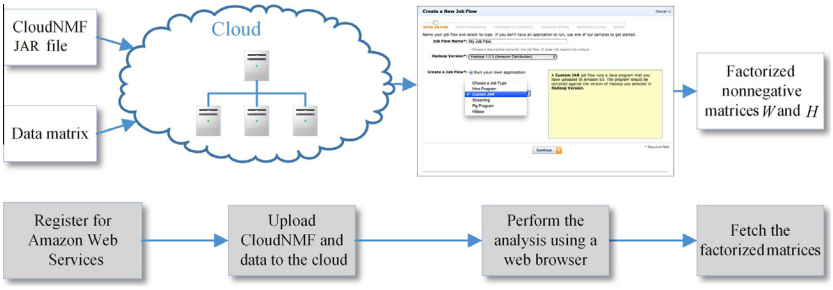

Figure 2 Using CloudNMF with Amazon Web Services

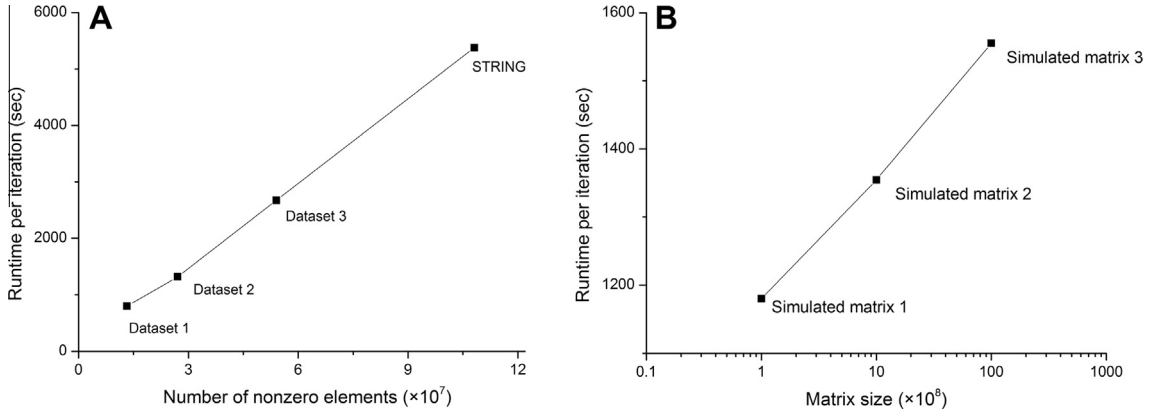

Figure 3 Performance of CloudNMF

A. Performance of CloudNMF on four real datasets shows the linear correlation of runtime per iteration with a number of nonzero elements in the matrix. **B.** Performance of CloudNMF on simulated matrices of different sizes but with the same number of nonzero elements shows that the runtime per iteration is linear to the logarithm of matrix size. Note that the X-axis is on a logarithmic scale.

To sum up, CloudNMF is the first open-source implementation of a MapReduce-based NMF algorithm and can be easily used to process large amounts of data. With the explosion of biological data and the wide application of NMF to biological research, we expect that CloudNMF will play more important roles in bioinformatics in the upcoming "Big Data" era.

Authors' contributions

Ruiqi Liao drafted the manuscript and developed the software. Yifan Zhang participated in the software development. Shuigeng Zhou proposed the idea of the software and revised the manuscript. Jihong Guan revised the manuscript. All authors have read and approved the final manuscript.

Competing interests

The authors have no competing interests to declare.

Acknowledgements

This work is financially supported by National High Technology Research and Development Program of China (863 Program; Grant No. 2012AA020403) and National Natural Science Foundation of China (Grant Nos. 61173118 and 61272380).

Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.gpb.2013.06.001.

References

[1] Lee DD, Seung HS. Learning the parts of objects by non-negative matrix factorization. Nature 1999;401:788–91.
[2] Brunet JP, Tamayo P, Golub TR, Mesirov JP. Metagenes and molecular pattern discovery using matrix factorization. Proc Natl Acad Sci U S A 2004;101:4164–9.
[3] Gao Y, Church G. Improving molecular cancer class discovery through sparse non-negative matrix factorization. Bioinformatics 2005;21:3970–5.
[4] Carmona-Saez P, Pascual-Marqui RD, Tirado F, Carazo JM, Pascual-Montano A. Biclustering of gene expression data by non-smooth non-negative matrix factorization. BMC Bioinformatics 2006;7:78.
[5] Qi Q, Zhao Y, Li M, Simon R. Non-negative matrix factorization of gene expression profiles: a plug-in for BRB-ArrayTools. Bioinformatics 2009;25:545–7.
[6] Jung I, Kim D. LinkNMF: identification of histone modification modules in the human genome using nonnegative matrix factorization. Gene 2013;518:215–21.
[7] Chagoyen M, Carmona-Saez P, Shatkay H, Carazo JM, Pascual-Montano A. Discovering semantic features in the literature: a foundation for building functional associations. BMC Bioinformatics 2006;7:41.
[8] Tjioe E, Berry MW, Homayouni R. Discovering gene functional relationships using FAUN (Feature Annotation Using Nonnegative matrix factorization). BMC Bioinformatics 2010;11:S14.
[9] Zhang S, Liu CC, Li W, Shen H, Laird PW, Zhou XJ. Discovery of multi-dimensional modules by integrative analysis of cancer genomic data. Nucleic Acids Res 2012;40:9379–91.
[10] Devarajan K. Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. PLoS Comput Biol 2008;4:e1000029.
[11] Mejia-Roa E, Carmona-Saez P, Nogales R, Vicente C, Vazquez M, Yang XY, et al. BioNMF: a web-based tool for nonnegative matrix factorization in biology. Nucleic Acids Res 2008;36:W523–8.
[12] Gaujoux R, Seoighe C. A flexible R package for nonnegative matrix factorization. BMC Bioinformatics 2010;11:367.
[13] Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 1999;286:531–7.
[14] Dai L, Gao X, Guo Y, Xiao J, Zhang Z. Bioinformatics clouds for big data manipulation. Biol Direct 2012;7:43.
[15] Liu C, Yang H, Fan J, He LW, Wang YM. Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce. In: Proceedings of the 19th international conference on World Wide Web, Raleigh, North Carolina, USA. New York: ACM; 2010, p. 681–90. http://dx.doi.org/10.1145/1772690.1772760.
[16] Lee D, Seung H. Algorithms for non-negative matrix factorization. Adv Neural Inf Process Syst 2001;13:556–62.
[17] Szklarczyk D, Franceschini A, Kuhn M, Simonovic M, Roth A, Minguez P, et al. The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. Nucleic Acids Res 2011;39:D561–8.