The User Behavior Analysis Based on Text Messages Using Parafac and Block Term Decomposition

Bilius Laura Bianca
University "Ștefan cel Mare" of Suceava
Departament of Computers
Faculty of Electrical Engineering and Computer Science
str. Universității nr. 9, 720225 – Suceava, Romania

***Abstract*—Tensor decompositions represent a start for big data analysis and a start in reduction of dimensionality, object detection, clustering and so on. This paper presents a method to study the behavior of users in the online environment and beyond. A beginning for analyzing this type of data is uniting the Parafac Tensor Decomposition and the Block Term Decomposition.**

Keywords—Parafac decomposition; block term decomposition; clustering

I. Introduction

High-order tensors have become very used for applications in big data analysis and signal processing due to tensor decompositions and their unique properties. The big data used in research experiments can be mathematically described with tensors [1].

Various researches have led to the development of tensors in the recent years, one of the reasons being the growth of data over time. This is also the reason why tensors analysis must receive more attention. The purpose of this paper is to present a new method to analyze big data and to efficiently process huge data set in a reasonable timeframe. In this paper, we describe the steps of processing big data by using Parafac because it returns a unique solution and Block Term Decomposition, by using matrices of elements sorted by the group of which they belong.

Nowadays, the technology feels like it is accelerating and people become more and more dependent of digital electronics. People communicate with each other using text messages or phone calls. In this research work we propose a solution to analyze the behavior of users which use text messages at different moments of the day. The purpose is to classify users which use text messages in the same timeframe. The research is based on Parafac Decomposition, clustering and Block Term Decomposition [2, 3].

II. Related Work

Through various researches, tensor models have been successfully applied in various areas such as: factor analysis, video tracking [4], face recognition [5], medical data analysis [6], and fake user detection in social networks [7] and so on.

In [8], the authors studied the tensors network which provides the possibility to analyze big data because of the good compression, parallel processing, establishing statistical connections between cores, factors, components, operation with noisy and missing data.

III. Parafac Decomposition and The Rank-(L_R,L_R,1) Block Term Decomposition

In this section we will provide mathematically definitions of the tensor, Parafac Decomposition and Block Term Decomposition. Tensors provide a compact and natural representation for multidimensional data.

A tensor $T \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is a multidimensional array called N^{th} order tensor or *N*-way tensor, where $N>0$. More exactly, *N* gives us the number of dimensions [9]. A tensor is a generalization of scalars, vectors and matrices. For example, if N=3 [see Fig. 1], then we have a third-order tensor $T \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and each entry of *T* is denoted by $x_{i_1 i_2 i_3}$ [10].

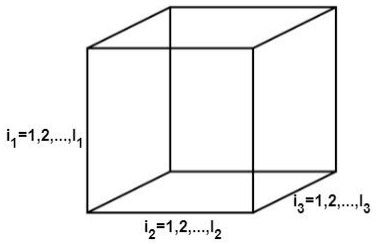

Fig. 1. A third order tensor $T \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ [9]

Canonical Polyadic (also known as CANDECOMP/ Parafac) Decomposition of a higher-order tensor is decomposition in a minimal number of rank-1 tensors [11]. A Parafac decomposition of a third order tensor is given by three loading matrices *A*, *B* and *C* and the sum of squares of the residuals.

The Parafac decomposes a tensor $T \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ as the sum of a finite numbers of rank-one tensors [9]. For a third order tensor $T \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the Parafac decomposition is written as [see Fig. 2]:

$$\mathrm{T} = \sum_{r=1}^{R} a_r \circ b_r \circ c_r + \mathrm{E} \approx [\![\mathrm{A}, \mathrm{B}, \mathrm{C}]\!], [9] \quad (1)$$

Where:

$$\mathrm{A} = [a_1, \ldots, a_R], \mathrm{B} = [b_1, \ldots, b_R], \mathrm{C} = [c_1, \ldots, c_R] [9]. \quad (2)$$

R is a positive integer and the symbol ∘ denotes the outer product of vectors. The Parafac model gives a unique solution if the loading vectors are linear independent in two of the modes. Another condition of uniqueness is given by Kruskal [12]:

$$k_1 + k_2 + k_3 \geq 2R + 2, \text{ [12]} \quad (3)$$

Where k_1, k_2, k_3 are the k-ranks of *A*, *B*, *C* and *R* is the number of Parafac components. The *k-rank* of a matrix *A*, denoted k_A, is defined as the maximum value *k* such that any columns are linearly independent [9].

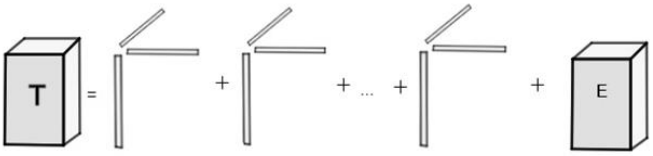

Fig. 2. Parafac decomposition of a N^{th}order tensor [9]

The rank of a tensor *T*, denoted *rank (T)*, is defined as the smallest number of rank-one tensors that generate *T* as their sum. The uniqueness means that it is the only possible combination to sum the tensor with a natural number of the rank-one tensors [9].

The rank-(L_r, L_r, 1) block term decomposition (BTD) is an approximation of a third order tensor by a sum of R terms, each of which is an outer product of a rank L_r matrix and a nonzero vector. Let be *T* a tensor of third order, $A_r \in \mathbb{C}^{I_1 \times L_r}$ and $B_r \in \mathbb{C}^{I_2 \times L_r}$ be rank L_r matrices and let be $c_r \in \mathbb{C}^{I_3}$, c_r nonzero:

$$T \approx \sum_{r=1}^{R} (A_r \cdot B_r^T) \circ c_r \text{ [13]} \quad (4)$$

Is a block tensor decomposition of the tensor T [Fig. 3] [13].

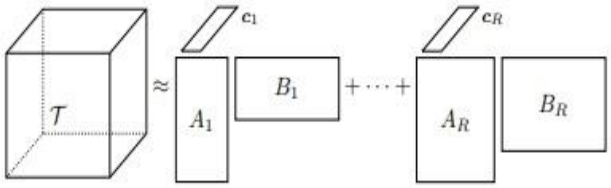

Fig. 3. The rank-(L_r,L_r,1) block term decomposition [13]

In addition, the matrix $A_r \cdot B_r^T \in \mathbb{C}^{I_1 \times I_2}$ has rank L_r. If matrices $[A_1, \ldots, A_R]$ and $[B_1, \ldots, B_R]$ are full column rank and the matrix $[c_1, \ldots, c_R]$ does not contain collinear columns, then the uniqueness of the decomposition is ensured. Also, the rank-(L_r,L_r,1) block term decomposition is a generalization of CPD (Canonical polyadic decomposition) for third order tensors [14].

IV. Problem Formulation

Nowadays, the text messages have become increasingly used. People spend much more time using mobile phones because it represents a quick way of communicating and an easy way to multitask. The interest in studying the behavior of users of using text messages is to see how long users communicate over the day and at what time of day communication is more intense.

Assume that six users, *A*, *B*, *C*, *D*, *E*, and *F*, communicate with each other sending text messages and images. The data can be organized in a tensor of order three [see Fig. 4], where the first two dimensions correspond to users and the third dimension corresponds to time.

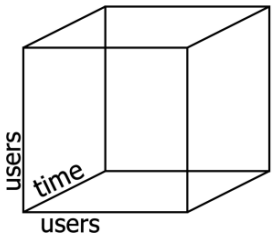

Fig. 4. The communication tensor between users

We used graphs to represent the network communication. As we can observe in the Fig. 5, we have a weighted directed graph. The weighted edges of the graph are given and represent the size of text messages measured in kilobytes. We can observe that users A and B are the most active on using text messages over time and the inactive ones are C and F [15].

The adjacency matrices were done at different moments of the day: first graph 08:00-12:00, second graph 12:00-16:00, third graph 16:00-20:00 and fourth graph 20:00-24:00.

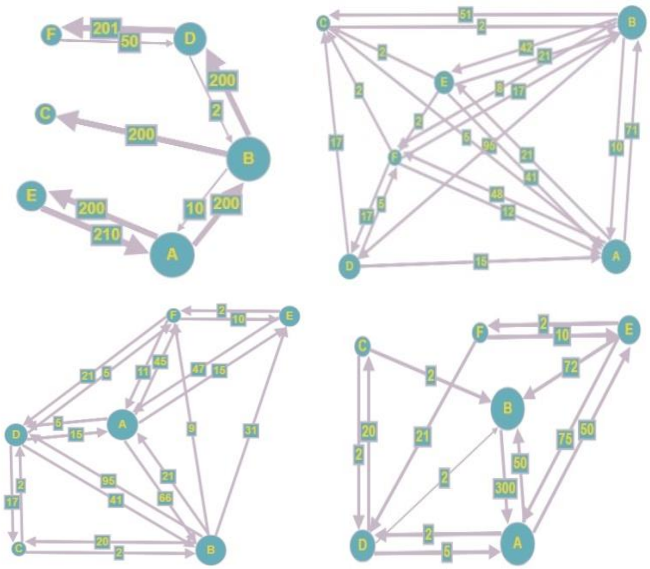

Fig. 5. The weighted directed graphs of the users communication

The purpose is to analyze the change of the communication in time of users and to see if the time of day influence on communication. We used a three dimensional tensor because it preserve a multidimensional structure of data [16]. To analyze the behavior of users of using text messages, we used an empirical data set.

V. Experimental Results

To compute the Parafac decomposition of tensor T, we use the command *parafac (T, fac, Options, const)* from N-Way Tollbox based on MATLAB [17]. The algorithm allows choosing optionally constraints to obtain orthogonal, non-negative or unimodal solutions.

The first two dimensions of tensor $T \in \mathbb{R}^{6\times6\times4}$ contain the adjacency matrices of communication of users and the third dimension is time. Using *parafac* function for tensor *T*, where T is the input array, with 23 components and in all factor matrices had been applied the non-negative constrain. The decomposition explain the tensor in proportion of 100%, converge after 50 iterations and the sum of squares of residuals is 0.108. The Parafac decomposition for 10 components [Fig. 6, 7, 8] explain the tensor in proportion of 99.33%, converge after 50 iterations but the sum of squares of residuals is 2655.84. The data was analyzed for 23 components [18].

The non-negative constrain was used to improve the result and to analyze data more realistic. An unconstrained model will fit the data worse than a constrained model [16].

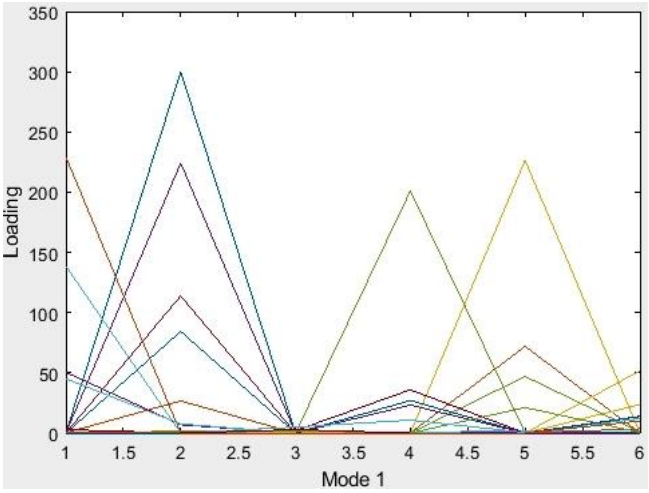

Fig. 6. Graphical representation of the results using 23 components for the first mode

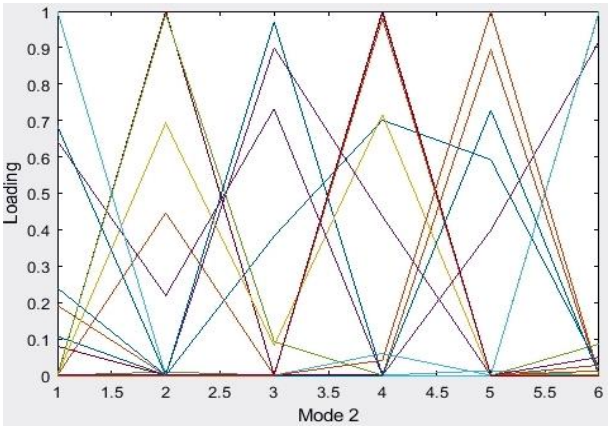

Fig. 7. Graphical representation of the results using 23 components for the second mode

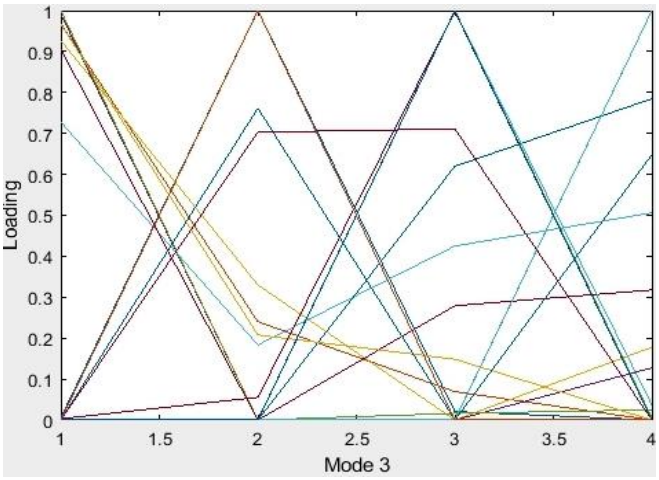

Fig. 8. Graphical representation of the results using 23 components for the third mode

The output of *parafac* function is "Factors" which stores the matrices resulted from applying it. The third matrix contains the degree of communication of users in time.

We want to group the similar vectors to see if there is a similarity between them [see Fig. 8]. If the number of components is big, it will be difficult to be manually handled. In this case, to regroup the similar vectors, it can be used some clustering techniques, like k-means or hierarchical clustering.

Hierarchical clustering groups the data into a multilevel cluster tree or dendrograms and it will help to choose the best level of clustering. To realize a hierarchical clustering must be followed some steps. Firstly, using *pdist* MATLAB function we calculate the distance between objects of C matrix. The second step involves a grouping of objects into a binary, hierarchical cluster tree. Using the information generated by *pdist* function, the *linkage* MATLAB function will link the pairs of objects that are close together into binary clusters. The linkage MATLAB function returns a matrix that encodes a tree containing hierarchical clusters of the rows of the input data matrix. *Linkage* MATLAB function uses distances to determine the order in which it clusters objects.

TABLE I. The Output of Linkage Function

17.0000	21.0000	0
15.0000	24.0000	0
13.0000	18.0000	0.0000
	⋮	
	⋮	
40.0000	43.0000	0.9230
38.0000	44.0000	0.9456

In the Table I, each row identifies a link between objects or clusters. The first two columns identify the objects that have been linked and the last column contains the distance between those objects.

For the sample data set, the *linkage* function of groups objects 38 and 44, which have a distance value of 0.9456. Another example, are objects 17 and 21 which have the closest proximity (distance value is 0).

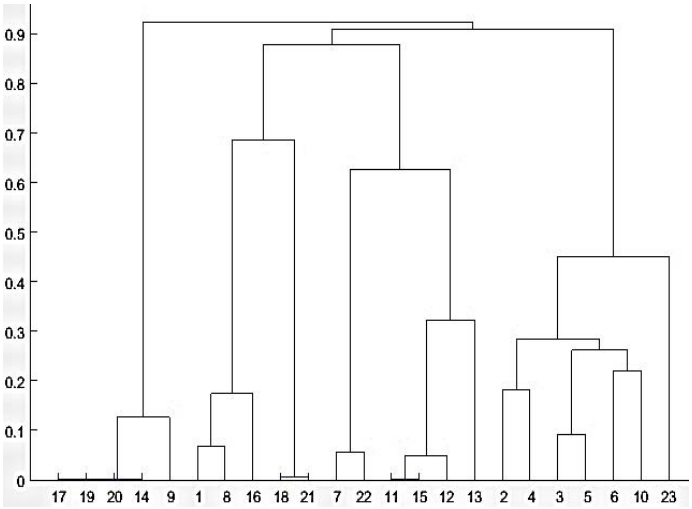

Fig. 9. The dendrogram of hierarchical, binary cluster tree

Using *dendrogram* MATLAB function it is easier to understand when the data is represented graphically [see Fig. 9]. *Dendrogram* MATLAB function generates a dendrogram plot of the hierarchical binary cluster tree.

To verify if the cluster tree was well generated, we can use *cophenet* MATLAB function to compare the datas returned by *linkage* and *pdist* functions. The cophenetic correlation coefficient is unsatisfying, so we used another distances to calculate the distances between objects. Using Euclidian Distance, the cophenetic coefficient was 0.777, the Squared Euclidean distance was 0.458, Standardized Euclidean distance was 0.910, City block distance was 0.581, Minkowski distance was 0.799. Chebychev distance gives the most accurately clustering solution which reflects the data and the cophenetic coefficient was ≈1.000.

The next step is to put the data in clusters. The *cluster* MATLAB function has two ways of clusterization. In natural way, the function allows us to give a threshold, which can be a value from inconsistency coefficients vector (can be found using *inconsistent* MATLAB function).

On my data, we choose the smallest coefficient and it divided them into 15 separate clusters. The second way is to specify your own number of clusters. Firstly, we used the second way. Analyzing the dendrogram, we choose to divide the data into 4 separate clusters [see Table II].

TABLE II. THE OUTPUT OF CLUSTER FUNCTION

IDX = 2 4 4 4 4 1 1 2 4 2 1 1 3 3 3 1 1 2 1 2 1 4 1

The next step is to recreate the three new matrices which contains the columns rearranged of A, B and C, by the group which belongs. After those matrices are created, we can apply Block Term Decomposition. To create a tensor with the original data sorted, we apply *cpdgen* TensorLab function on A, B and C matrices.

Generally, block term decomposition is a regrouping of a decomposed tensor. The Tensorlab function *ll1* can be applied to compute a block term decomposition of a tensor using a multistep approach and it has 2 output formats, *cpd* and *btd* mode. When using the CPD format, the parameter L is required because it provides the necessary information on how the columns are grouped. This function is the best choice because it accepts dense, sparse and incomplete tensors. *ll1* MATLAB function performs a number of steps and provide a good initialization to reduce the computational cost of the decomposition [19].

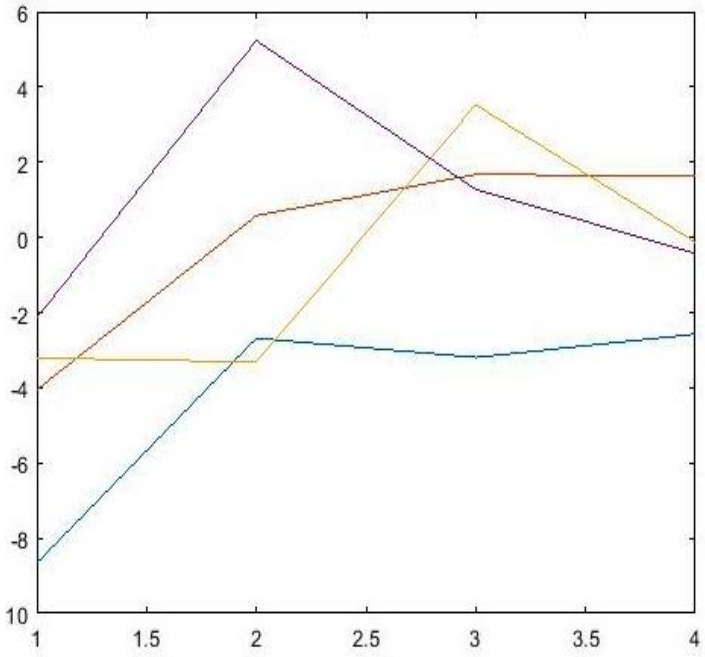

Fig. 10. The output of ll1 function in cpd mode

In Fig. 10, we can see the plot of the matrix C after *ll1* function in *cpd* mode was applied, where L = [9, 5, 3, 6] is the vector which contains the number of elements that each cluster has. In Table III we have the output of the *ll1* function.

TABLE III. THE MATRIX C AFTER LL1 FUNCTION IN CPD MODE WAS APPLIED

-8.6590	-4.0514	-3.2087	-2.1427
-2.6838	0.5798	-3.3145	5.2329
-3.1924	1.6741	3.5252	1.2748
-2.5723	1.6291	-0.1283	-0.4262

In Fig. 10, the blue line belongs to the first group which has a small increase in the use of text messages as time passes. The yellow line belongs to the third group and we can see that users have used the text messages more in the evening. The red line belongs to the second group who uses the text messages more and more throughout the day. The brown line belongs to the second group which have which have a slight increase throughout the day. In conclusion, there is an increase of using text messages in the evening, fact confirmed by the graphs from Fig. 5.

Another analysis of the same dataset where we choose to divide the data into 6 separate clusters [see Table IV] and in Table V we have the output of the *ll1* function.

TABLE IV. THE OUTPUT OF CLUSTER FUNCTION

IDX=4 6 6 6 6 6 2 5 1 6 4 5 4 4 1 2 5 2 1 2 4 1 3

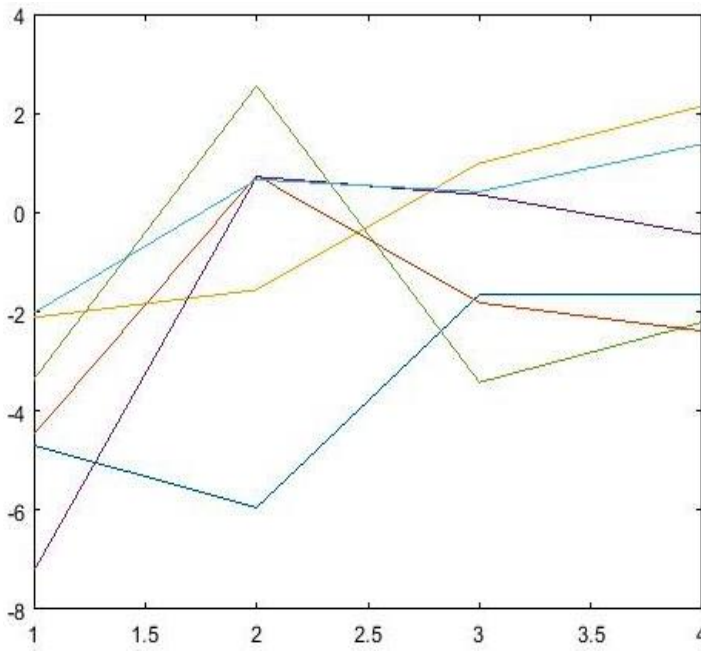

Fig. 11. The output of ll1 function in cpd mode

TABLE V. THE MATRIX C AFTER LL1 FUNCTION IN CPD MODE WAS APPLIED

-4.6948	-4.4791	-2.1156	-7.2278	-3.3653	-2.0216
-5.9477	0.7470	-1.5593	0.7200	2.5560	0.6633
-1.6492	-1.8211	0.9920	0.3592	-3.4240	0.4209
-1.6339	-2.3872	2.1457	-0.4445	-2.2147	1.3810

In Fig. 11, we can see the plot of the matrix C after *ll1* function in *cpd* mode which was applied using 6 clusters and L = [4, 4, 1, 5, 3, 6].

The dark blue line belongs to the first group, which have an increase in the use of text messages as time passes. The yellow line belongs to the third group and we can see that users have used the text messages more in the evening. The green line belongs to the fifth group, which has an increase in the first part of the day and then a decrease in the second part of the day; however, there is a slight increase towards the end of the day. The light blue line belongs to the sixth group, which have a slight increase throughout the day. The purple line belongs to the fourth group, which communicates very little in the first part of the day, but in the second part of the day we can see a huge increase.

In conclusion, using 6 clusters, we can observe that there is an increase of using text messages in the evening, fact confirmed by the graphs from Fig. 5.

In another analysis of the same dataset, we choose to divide the data using the hierarchical clustering in a natural way. The *cluster* function can create clusters by detecting natural groupings in the hierarchical tree [see Table VI]. In Table VII we have the output of the *ll1* function.

TABLE VI. THE OUTPUT OF CLUSTER FUNCTION

IDX= 3 4 4 4 4 4 1 2 5 4 3 2 3 2 1 5 1 3 3 1 2 3 5

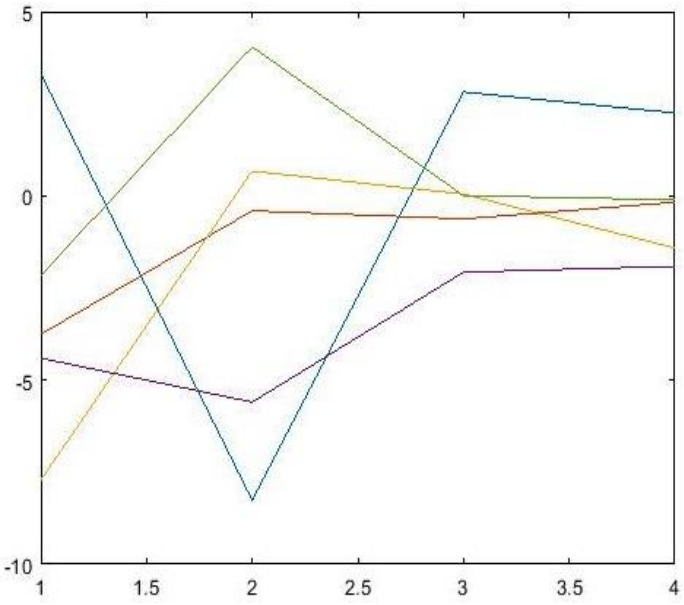

Fig. 12. The output of ll1 function in cpd mode

TABLE VII. THE MATRIX C AFTER LL1 FUNCTION IN CPD MODE WAS APPLIED

3.3097	-3.7560	-7.7130	-4.4060	-2.1640
-8.2634	-0.4053	0.6729	-5.5966	4.0427
2.8190	-0.6237	0.0480	-2.0716	0.0068
2.2518	-0.1741	-1.4155	-1.9127	-0.1062

In Fig. 12, we can see the plot of the matrix C after *ll1* function in *cpd* mode which was applied using 6 clusters and L = [4 4 6 6 3]. The hierarchical clustering was realized in natural way and the inconsistency coefficient was 0.9. It divided them into 5 separate clusters.

After we analyzed the Fig. 12, we can affirm that in the evening the text messages are used by all users at about the same intensity, fact confirmed by the graphs from Fig. 5.

VI. CONCLUSIONS

The existing methods and algorithms become inadequate for processing big data, because it can have huge volume and high complexity. The most important reason for which we started with Parafac decomposition is that the model estimated is easier to analyze, especially when it comes to big data. Adopting the rank-(L_r, L_r, 1) block term decomposition was an important step, because we regrouped the components using a hierarchical clustering. The objective of this paper was to obtain the result of block term decomposition using clustering which helped us to analyze the data.

In summary, big data analysis has potential to be researched because the optimization problems become ineffective for the current data.

REFERENCES

[1] Hemlata, Preeti Gulia, *Big Data Anaytics*, Research Journal of Computer and Information Technology Sciences, Vol. 4(2), 1-4, February(2016).

[2] Qingquan Song, Hancheng Ge, James Caverlee, Xia Hu. (3 May 2018). ArXiv. *Tensor Completion Algorithms in Big Data Analytics*, arXiv:1711.10105 [stat.ML].

[3] Lee J, Choi D, Sael L (2018) CTD: *Fast, accurate, and interpretable method for static and dynamic tensor decompositions.* PLoS ONE

13(7):e0200579. https://doi.org/10.1371/journal.pone.0200579

[4] Xiaqqin Z., Xingchu Shi., Weiming H.,ELSEVIER, Neurocomputing, *Visual tracking via dynamic tensor analysis with mean update*, Vol. 74, Issue 17, October 2011, pp: 3277-3285.

[5] Lee Ying Chong, Lee Ying Chong, Thian Song Ong, Thian Song Ong, Andrew Beng Jin Teoh, Andrew Beng Jin Teoh, } "Tensor manifold-based extreme learning machine for 2.5-D face recognition," Journal of Electronic Imaging 27(1), 013016 (12 February 2018). https://doi.org/10.1117/1.JEI.27.1.013016 . Submission: Received: 23 August 2017; Accepted: 9 January 2018

[6] Joyce C Ho, Joydeep G., Steve R. S., Walter F Stewart, Joshua C Denny, Bradley A M., and Jimeng S. 2014. Limestone: *High-throughput candidate phenotype generation via tensor factorization. Journal of biomedical informatics.*

[7] Cao Q, Sirivianos M, Yang X, Pregueiro T. *Aiding the Detection of Fake Accounts in Large Scale Social Online Services*.2012; p. 197–210.

[8] Andrzej Cichocki,(24 August 2014) *Era of Big Data Processing: A New Approach via Tensor Networks and Tensor Decompositions,* arXiv:1403.2048 [cs.ET].

[9] Tamara Kolda, B. W. (2009). Tensor Decompositions and Applications. *Society for Industrial and Applied Mathematics*, *51*(3), pp. 455-500.

[10] Evrim Acar, B. Y. (2008, June). Unsupervised Multiway Data Analysis: a literature survey. *IEEE Transactions on Knowledge and Data Engineering, Vol. 21*, 6-20.

[11] Ignat Domanov, Lieven de Lathhauter, (2013). SIAM Journal on Matrix Analysis and Applications. *On the uniqueness of the canonical polyadic decomposition of third-order tensors-Part II:Uniqueness of the overall decomposition*, Vol. 34, Issue 3, pp; 876-903.

[12] Xijing Guo, Sebastian Miron, David Brie, and Alwin Stegeman, *Uni-mode and Partial Uniqueness Conditions for CANDECOMP/PARAFAC of Three-Way Arrays with Linearly Dependent Loadings*, SIAM Journal on Matrix Analysis and Applications, *SIAM J. Matrix Anal. & Appl*., 33(1), 111–129. (19 pages) (2012).

[13] Laurent Sorber, Marc Van Barel, Lieven De Lathauter (April, 2013). *Optimization-based algorithms for tensor decompositions: Canonical Polyadic Decomposition, Decomposition In Rank-*(L_r,L_r,1) *Terms And A New Generalization, Vol*. 23, Issue 2, pp 695-720.

[14] Hunyadi, B., Camps, D., Sorber, L. et al. EURASIP J. Adv. Signal Process. Block term decomposition for modelling epileptic seizures (2014) 2014: 139. https://doi.org/10.1186/1687-6180-2014-1397

[15] Nicolas Nisse, *Graph Theory and Optimization Weighted Graphs, Shortest Paths & Spanning Trees*, Université Côte d'Azur, Inria, CNRS, I3S, France, October 2018., URL: https://www.inria.fr.

[16] Dumitrascu Ionut, (May 2014) Block-PARAFAC non-negative decomposition of hyper-spectral images, *Universite de Lorraine, Centre de research en automatique.*

[17] MATLAB MathWorks – MATLAB & Simulink R2015a ,URL: https://www.mathworks.com

[18] Rasmus Bro, Claus A. Andersson. *The N-way toolbox for MATLAB*, 2000

[19] Laurent Sorber, Marc Van Barel and Lieven De Lathauwer. *Tensorlab v3.0*, Available online URL: http://www.tensorlab.net, 2014.