A TENSOR BASED FRAMEWORK FOR COMMUNITY DETECTION IN DYNAMIC NETWORKS

Esraa Al-Sharoa[1]*, Mahmood Al-khassaweneh*[1,2] *and Selin Aviyente*[1]

[1]Department of Electrical and Computer Engineering, Michigan State University, USA
[2] Department of Computer Engineering, Yarmouk University, Jordan

ABSTRACT

Many systems from human brain to the networks on social media, can be modeled as graphs. The network structure helps us understand, predict and optimize the behavior of dynamical systems. One of the important tools in understanding network topology is community detection. Even though community detection methods are well developed for static networks, the extensions to the dynamic case are more limited. In this paper, we introduce two tensor based frameworks, windowed and running time, for identifying and tracking the network community structure across time. The frameworks take the history of the networks into account. The proposed approach relies on determining the subspace across time using the Tucker decomposition of a tensor constructed from networks across time. We also propose a computationally efficient way to update the subspace estimates across time to track changes in community structure. The proposed approach is evaluated on both simulated and real dynamic networks.

***Index Terms*—** Tensor decomposition, Dynamic networks, Spectral Clustering

1. INTRODUCTION

Networks are commonly used to model the relationships and interactions between people or objects such as in social, biological, and communication networks. A lot of research has been done in detecting the community structure of networks, especially for static networks [1]. In many applications, the community structure changes over time. The topology of dynamic networks change over time, since nodes and edges might come and go. Standard spectral clustering can be used to analyze these networks at each time point separately, but this approach is very sensitive to instantaneous noise.

Early work in dynamic network clustering extended the existing static clustering algorithms with an added constraint to take the historical data into account for temporal smoothness. An extension of the k-means and agglomerative hierarchical clustering approaches was developed in [2] for dynamic networks. A cost function that considers both the cluster membership at the current time and the deviation from the previous time point was introduced. In [3] two frameworks were introduced: preserving cluster quality (PCQ) and preserving cluster membership (PCM). In both frameworks the clustering depends on a cost function to guarantee temporal smoothness. This cost function requires a *priori* knowledge about the number of clusters and depends on the choice of a changing parameter. In [4], an evolutionary clustering approach based on a statistical model of the adjacency matrix with an adaptive forgetting factor for evolutionary clustering and tracking (AFFECT) was introduced. In AFFECT, static clustering methods were used after smoothing the proximity between objects over time. This was accomplished by adaptively estimating a forgetting factor to estimate new proximity. This estimation increases the computation cost compared to static clustering algorithms. Authors in [5] introduced a constrained tensor factorization approach using PARAFAC to identify the community structure in static networks. This was achieved by constructing a 3-way tensor with its slices defined as the subgraphs or egonet that represent each node and its single hop neighbors in the network. In [6], a non-negative tensor factorization approach was introduced using PARAFAC to detect the community structure in temporal networks. A single 3-way tensor was constructed to represent all the successive adjacency matrices in the temporal network. The approach was verified by recreating the class structure of a temporal interaction network of a school.

In this paper, we introduce a model-free framework that can take the past history into account with non-arbitrary optimized weighting coefficients. The proposed framework relies on tensor representation of networks across time and borrows ideas from tensor based clustering of multi-layer networks [7]. However, unlike the work in [7], we consider streaming tensors whose dimension changes with time. Moreover, the proposed framework both tracks and identifies the network community structure across time and does not make any assumptions about the number of clusters and the change points like PCM and PCQ. In the proposed work, we develop two algorithms: the windowed and running time tensor analysis. A tensor is constructed at each time point by taking the history of networks into account using a sliding window unlike the work in [6] that constructs a single tensor for all time points. Our approach identifies the common subspace across time using Tucker decomposition. Moreover, we introduced a cost function to track the changes in the community structure over time.

2. BACKGROUND ON TENSOR DECOMPOSITION

A tensor is a multidimensional array that is commonly used to represent data with multiple variables or modes. An N-mode tensor is denoted by $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Matrix factorization methods such as SVD have been extended to tensors through PARAFAC and Tucker decompositions [8]. Tucker decomposition provides orthogonal subspace information along each mode of the tensor,

$$\mathcal{X} = \mathcal{C} \times_1 U_{(1)} \times_2 U_{(2)} \cdots \times_N U_{(N)}, \tag{1}$$

where $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the core tensor and $U_{(n)}$ is the orthogonal component matrix along the nth mode. Tucker decomposition is commonly implemented through high-order SVD (HOSVD) [9]

This work was in part supported by NSF CCF-1422262 and CCF-1615489 and the Schlumberger Foundation, Faculty for the Future.

or high-order orthogonal iteration (HOOI) [10]. HOSVD performs matrix SVD on every unfolding matrix of the tensor independently. On the other hand, HOOI performs alternating optimization to find the best projection matrices.

3. TENSOR BASED TEMPORAL CLUSTERING

For static graphs, spectral clustering is commonly used to detect the community structure in the graph, since it represents the relationship between the nodes in the graph in a lower dimensional subspace. Spectral clustering solves the following trace optimization problem [11, 12],

$$\min_{U\in\mathbb{R}^{m\times k}} tr\left(U^T L_N U\right), \qquad s.t\ \ U^T U = I, \tag{2}$$

where, $L_N = I - A_N$ is the normalized Laplacian matrix, $A_N = D^{\frac{-1}{2}} A_N D^{\frac{-1}{2}}$ is the normalized adjacency matrix, and D is the degree matrix which is defined as $D_{ii} = \sum_j a_{ij}$. Spectral clustering problem can also be written as,

$$\max_{U\in\mathbb{R}^{m\times k}} tr\left(U^T A_N U\right), \qquad s.t\ \ U^T U = I, \tag{3}$$

or

$$\max_{U\in\mathbb{R}^{m\times k}} \| U^T A_N U \|_F^2, \qquad s.t\ \ U^T U = I. \tag{4}$$

The solution to the optimization problems in (2), (3) and (4) is found by choosing U as the matrix that contains the k eigenvectors that correspond to the smallest eigenvalues of L_N or the largest eigenvalues of A_N. The community structure is then determined by applying k-means to matrix U [11].

Assume that dynamic networks are represented by a sequence of weighted and undirected graphs $\{G(t)\}_{t_1,\cdots,t_M}$, where $G(t)$ is an $m\times m$ graph representing the network at time step t, m is the total number of nodes and M is the total number of time points. Each graph is defined through the adjacency matrix $A^{(t)} \in \mathbb{R}^{m\times m}$ where $a_{ij} = 0$ if the nodes i and j are not connected, otherwise $a_{ij} = a_{ji}$ [11].

In the dynamic clustering case, we would like to find the community structure at time t based on the past history. One way to accomplish this is to employ a weighted average of the adjacency matrices within a time window of length L and then optimize the conventional spectral clustering cost function over this new weighted average. This results in an optimization over both the subspace vectors matrix, U, and the weight vector, W. The optimization problem in (4) can then be written as,

$$\max_{U,W} \| U^T \left(\sum_{l=1}^{L} w_l A_N^{(l)}\right) U \|_F^2 \ \ s.t\ U^T U = I,\ W \geq 0, \| W \|_F = 1, \tag{5}$$

where w_l are the weights for the adjacency matrices within a time window of length L. In the proposed approaches, we build a third order tensor, $\mathcal{X}(i,j,k)$. where the k^{th} frontal slice of the tensor $\mathcal{X}$ is the adjacency matrix $A_N^{(k)}$. The dimension of the third mode of this tensor may either be fixed as in the case of the windowed tensor approach or grow with time as in the running time tensor approach. In the windowed tensor approach, we build a sliding fixed size tensor over all time points. The running time tensor approach, on the other hand, uses a tensor size that changes over time. In both approaches, (5) can be reformulated in terms of the Tucker decomposition as follows,

$$\max_{U,W} \| \mathcal{X} \times_1 U^T \times_2 U^T \times_3 W^T \|_F^2,\ U^T U = I, \| W \|_F = 1, \tag{6}$$

where U is the basis along the first and second modes and W corresponds to the weighting vector across time. Since the networks are undirected, the components along the first two modes are the same. The problem in (6) can be solved by higher-order orthogonal iteration ($HOOI$), where in each iteration either the matrix U or the vector W is fixed to optimize the other one.

3.1. Tracking the Community structure over time

For both algorithms, we define the cost function through $\| \mathcal{X} \times_1 U^T \times_2 U^T \times_3 W^T \|_F$. This cost function satisfies the following inequality,

$$Cost = \| \mathcal{X} \times_1 U^T \times_2 U^T \times_3 W^T \|_F \leq \| \mathcal{X} \|_F \| U \|_2^2 \| W \|_2, \tag{7}$$

which can be further simplified to,

$$Cost = \| \mathcal{X} \times_1 U^T \times_2 U^T \times_3 W^T \|_F \leq \| \mathcal{X} \|_F, \tag{8}$$

by noting that $\| W \|_2 = 1$ and $\| U \|_2^2 = \lambda_{max}(U^* U)$, where λ_{max} is the largest eigenvalue of $U^* U$, which equals to 1 since $U^* U = I$. Therefore, by dividing both sides in (8) by $\| \mathcal{X} \|_F$ we get a normalized cost function which is always between 0 and 1,

$$Normalized\ Cost = \frac{\| \mathcal{X} \times_1 U^T \times_2 U^T \times_3 W^T \|_F^2}{\| \mathcal{X} \|_F^2}. \tag{9}$$

Changes in the normalized cost function in (9) reflect changes in the community structure. Once the change points are determined, the new number of clusters at that time point will be determined using the eigengap criterion [11].

3.2. Windowed Tensor Approach

In this approach, we use a sliding window of fixed size α to build the tensor at time point t as $\mathcal{X}^{(t)} \in \mathbb{R}^{m\times m\times\alpha} = \left[A_N^{(t-\alpha+1)} \cdots A_N^{(t)}\right]$. The constructed tensors have the same size for all time points. At each time point, the constructed tensor adds a new adjacency matrix and excludes the first adjacency of the previous tensor. The algorithm is summarized in the pseudo code in Algorithm 1. In this approach, we do not need to initialize the eigenvectors matrix using HOSVD, except for the first tensor. For the rest of the time points, the estimated optimal eigenvectors matrix at time point $(t-1)$, U^{t-1} is used to initialize for time point t.

3.3. Running Time Tensor Approach

In this approach, the tensor at time point t is built by taking all history into account, where $\mathcal{X}^{(t)} = \left[A_N^{(1)} \cdots A_N^{(t)}\right]$. This approach estimates the optimal eigenspace that represents the network at time t in a similar fashion to WTA. The main differences are in the tensor size and initialization of the matrix U. This approach uses a window size that grows with time. Consequently, the tensor constructed at each time point has a different size. Moreover, at each time point the eigenvectors matrix is an updated version of the previous one. This update was motivated by the fact that the tensor at time t is the same as the tensor at time $(t-1)$ except for one time slice. Therefore, the matrix U is adaptively updated based on the new adjacency matrix. We used TrackU algorithm [13, 14] to update the eigenvectors matrix. The RTTA is described in Algorithm 2, and the eigenvectors matrix update is described in Algorithm 3.

Algorithm 1 Windowed Tensor Approach (WTA)

Input: $A_N^{(1)} \cdots A_N^{(M)}, \alpha$
Output: Clustering Labels, Change points

for $t = \alpha : M$ **do**
  Construct a similarity tensor $\mathcal{X}^{(t)} = \left[A_N^{(t-\alpha+1)} \cdots A_N^{(t)}\right]$
  Matricize the tensor along mode 3 to get $X_{(3)}^{(t)}$
  if $t = \alpha$ **then**
    initialize U_0 by HOSVD of $\mathcal{X}_1^{(t)}$
  else
    initialize U_0 by U^{t-1}
  end if
  $i = 1$
  while $\| U_{i+1}^{(t)} - U_i^{(t)} \|_F^2 > \varepsilon$ **do**
    calculate W_{i+1} as the dominant left singular vector of $X_{(3)}^{(t)} \left(U_i^{(t)} \otimes U_i^{(t)}\right)$
    compute $\bar{A} = \sum_{l=t-\alpha+1}^{t} (W_{i+1})_l A_N^{(l)}$
    obtain U_{i+1} by eigenvalue decomposition of $\bar{A}$
    $i = i+1$
  end while
  track the change points using (9)
  use eigen gap criterion to determine the number of clusters
  normalize the columns of $U^{(t)}$
  apply $k-$means on $U^{(t)}$ to obtain clustering labels
end for

Algorithm 2 Running Time Tensor Approach (RTTA)

Input: $A_N^{(1)} \cdots A_N^{(M)}, starttime, forgettingfactor = \beta$
Output: Clustering Labels, Change points

for $t = starttime : M$ **do**
  Construct a similarity tensor $\mathcal{X}^{(t)} = \left[A_N^{(1)} \cdots A_N^{(t)}\right]$
  Matricize the tensor along mode 3 to get $X_{(3)}^{(t)}$
  if $t = starttime$ **then**
    initialize U_0 by HOSVD of $\mathcal{X}_1^{(t)}$
  else
    initialize U_0=**TrackU**$(U^{(t-1)}, S^{(t-1)}, A_N^{(t)}, \beta)$
  end if
  $i = 1$
  while $\| U_{i+1}^{(t)} - U_i^{(t)} \|_F^2 > \varepsilon$ **do**
    calculate W_{i+1} as the dominant left singular vector of $X_{(3)}^{(t)} \left(U_i^{(t)} \otimes U_i^{(t)}\right)$
    compute $\bar{A} = \sum_{l=1}^{t} (W_{i+1})_l A_N^{(l)}$
    obtain U_{i+1} and S_{i+1} by eigenvalue decomposition of $\bar{A} = U_{i+1}^{(t)} S_{i+1}^{(t)} \left(U_{i+1}^{(t)}\right)^T$
    $i = i+1$
  end while
  track the change points using the normalized cost function given in (9)
  use eigen gap criterion to determine the number of clusters
  normalize the columns of $U^{(t)}$
  apply $k-$means on $U^{(t)}$ to obtain clustering labels
end for

4. RESULTS

4.1. Simulated Temporal Networks

A dynamic network is generated for 60 time points with 64 nodes. Intra and inter-cluster edges were selected from a truncated Gaussian distribution in the range of $[0,1]$. The simulated networks were created with the first 20 networks having 4 clusters and edges selected from a truncated Gaussian distribution, with $\mu_{intra} = 0.3$, $\sigma_{intra} = 0.3$ and $\mu_{inter} = 0.2$, $\sigma_{inter} = 0.2$. The number of clusters decreases to 2 for the next 20 networks, with the edges taken from a truncated Gaussian distribution, with $\mu_{intra} = 0.2$, $\sigma_{intra} = 0.3$ and $\mu_{inter} = 0.2$, $\sigma_{inter} = 0.2$. From time point 41 until point 60, the number of clusters increases to 4, with edges drawn from truncated Gaussian distribution with $\mu_{intra} = 0.3$, $\sigma_{intra} = 0.3$ and $\mu_{inter} = 0.2$, $\sigma_{inter} = 0.2$. For the windowed tensor approach, a 3-mode tensor was constructed at each time point, with a window size of 6. We ran this experiment 100 times. The number of clusters at each time point was detected using the eigengap criterion. The performance of the proposed approach (WTA), standard spectral clustering (SC), (PCQ), (PCM) and (AFFECT) is compared using Rand index [15]. The algorithm was able to detect the correct community structure over time and the change points using the normalized cost function in (9). The results in Fig.1 indicate that the proposed algorithm outperforms the other algorithms in terms of Rand index and detecting the correct community structure.

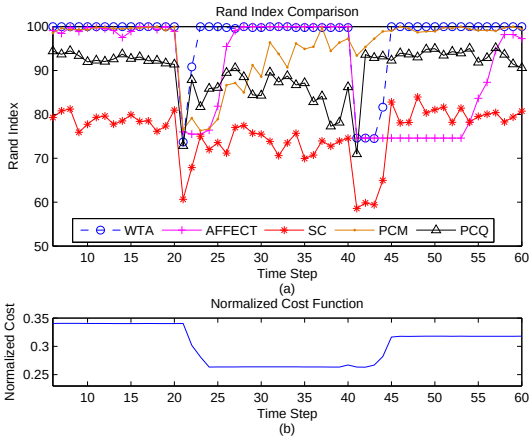

Fig. 1: (a) Rand index comparison between WTA, AFFECT, SC, PCM and PCQ for the synthetic data set, the number of clusters changes from 4 to 2 at the time point 21, and from 2 to 4 at time point 41. (b) The normalized cost function.

4.2. Reality Data Mining

This data set represents the cell phone activity for 94 students and staff in MIT Media Laboratory. It was collected between August/2004 and June/2005 [16]. The data set is represented by a dynamic network across 46 time points. Each point refers to a week during the school year. Each time point was represented by a 3-mode tensor. Both algorithms were applied to this data to track the changes over time.We assumed that there were two dominant clusters at each time point, and we used the normalized cost function in (9) to track the changes in the community structure over time. The detected change points corresponds to end of the Fall semester,

start of the Spring semester, spring break and the end of the Spring semester. The change points shown in Fig.2 were detected by the WTA with $\alpha = 11$.

Algorithm 3 TrackU

Input: Eigenvector matrix: U, Eigenvalues matrix: S, adjacency matrix at time point t: A_N, forgetting factor: β
Output: updated eigenvector matrix

```
C_A =size(A_N,2), C_U =size(U,2), s=diag(S)
for i = 1 : C_A do
    b̄ = A_N(:,i)
    for j = 1 : C_U do
        y_j = u_j^T b̄_j
        s_j^(t) = β s_j + y_j^2
        compute the error: e_j = b̄_j − y_j u_j
        u_j = u_j + (1/s_j) y_j e_j
        b̄_{j+1} = b̄_j − y_j u_j
    end for
end for
orthogonalize U using Gram-Schmidt
```

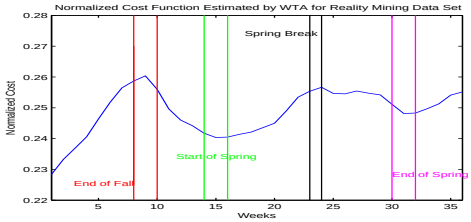

Fig. 2: Detected Change Points for Reality Mining Data.

4.3. Dynamic Functional Connectivity Networks (dFCNs)

The proposed tensor tracking approach is applied to a set of connectivity graphs constructed from EEG data containing the error-related negativity (ERN). The ERN is a brain potential response that occurs following performance errors in a speeded reaction time task usually 25-75 ms after the response. Previous work [17] indicates that there is increased coordination between the lateral prefrontal cortex (lPFC) and medial prefrontal cortex (mPFC) within the theta frequency band (4-8 Hz) and ERN time window. EEG data from 63-channels was collected in accordance with the 10/20 system on a Neuroscan Synamps2 system (Neuroscan, Inc.) sampled at 128 Hz from 91 subjects. The task was a common speeded response letter (H/S) flanker, where error response locked trials from each subject were utilized. The EEG data are pre-processed by the spherical spline current source density (CSD) waveforms to sharpen event-related potential (ERP) scalp topographies and reduce volume conduction [18].

We constructed average connectivity networks at each time point across subjects to represent the adjacency matrix of the connectivity graphs. Individual connectivity networks were constructed by using a previously introduced phase synchrony measure [19]. This results in a dynamic network with 63 nodes and 256 time points. We applied the proposed RTTA to detect the community structure over time. Starting from time point 30, a 3-mode tensor was built at each time point.The change points were detected by the normalized cost function in (9) as shown in Fig.3. As it can be seen in Fig.3, there is an increase in the cost function around -78 ms right before the response time. Similarly, there is a decrease in the cost function around 400 ms corresponding well with the end of the ERN waveform. The number of clusters was determined by eigengap criterion and the detected community structure corresponding to these three time periods are given in Fig.4. While the community structures for pre- and post- ERN intervals are similar, there is increased segregation during ERN especially in the frontal central brain regions. This is aligned with the increased coordination between lPFC and mPFC during this time.

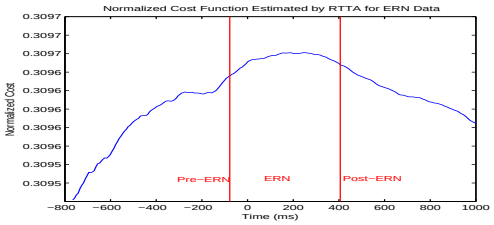

Fig. 3: Detected Change Points for ERN Data.

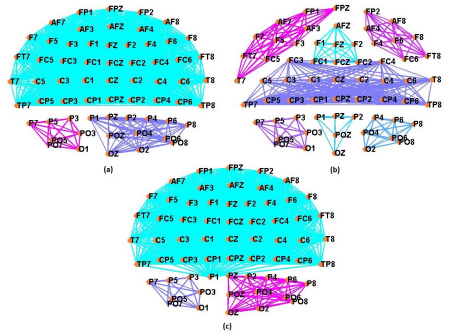

Fig. 4: Community structure for the ERN networks obtained by RTTA: (a) Pre-ERN, (b) ERN, (c) Post-ERN.

5. CONCLUSIONS

In this paper, we introduced two frameworks to detect and track community structure in dynamic networks: windowed and running time tensor analysis. A tensor is constructed at each time point by taking the past history into account. Both frameworks detect the community structure by finding a common subspace through Tucker decomposition of the constructed tensors. In addition, we introduced a cost function to track changes in the dynamic network. The algorithms were tested on multiple simulated and real data sets and compared to the existing state of the art algorithms. The simulation results show that the proposed algorithms are able to detect the correct community structure and changes in the number of clusters over time even in noisy networks. The application to functional connectivity brain network revealed changes in the community structure across time that aligned well with the experimental setting and prior work on cognitive control. Future work will consider the extension of this framework to higher order tensor type data clustering across different modes.

6. REFERENCES

[1] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3, pp. 75–174, 2010.

[2] D. Chackrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, vol. II, pp. 554–560.

[3] Y. Chi, X. Song, D. Zhou, K. Hino, and B. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, vol. II, pp. 153–162.

[4] K. Xu, M. Kliger, and A. Hero, "Adaptive evolutionary clustering," *Data Mining and Knowledge Discovery*, vol. 28, no. 2, pp. 304–336, 2014.

[5] F. Sheikholeslami, B. Baingana, G. B. Giannakisand, and N. D. Sidiropoulos, "Egonet tensor decomposition for community identification," in *Proceedings of Globalsip 2016*, 2016.

[6] L. Gauvin, A. Panisson, and C. Cattuto, "Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach," *PloS one*, vol. 9, no. 1, 2014.

[7] X. Liu, S. Ji, W. Glanzel, and B. De Moor, "Multiview partitioning via tensor methods," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, pp. 1056–1069, January 2013.

[8] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[9] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM journal on Matrix Analysis and Applications*, vol. 21, pp. 12531278, January 2000.

[10] L. De Lathauwer, B. De Moor, and J. Vandewalle, "On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, pp. 1324–1342, January 2000.

[11] U. V. Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[12] N. Andrew and J. Michael Y. Weiss, "On spectral clustering:analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[13] S. Papadimitriou, S. Jimeng, and F. Christos, "Streaming pattern discovery in multiple time-series," in *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 2005, pp. 697–708.

[14] Sun J, D. Tao, S. Papadimitriou, P. Yu, S. Philip, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," *ACM Transactions on Knowledge Discovery from Data*, vol. 2, no. 3, pp. 11, 2008.

[15] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[16] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Personal and ubiquitous computing*, vol. 10, no. 4, pp. 255–268, 2006.

[17] J. F. Cavanagh, M. X. Cohen, and J. J. Allen, "Prelude to and resolution of an error: Eeg phase synchrony reveals cognitive control dynamics during action monitoring," *The Journal of Neuroscience*, vol. 29, pp. 98–105, 2009.

[18] J. Kayser and C. E.Tenke, "Principal components analysis of laplacian waveforms as a generic method for identifying erp generator patterns: I. evaluation with auditory oddball tasks," *Clinical neurophysiology*, vol. 117, no. 2, pp. 348–368, 2006.

[19] S. Aviyente, E. Bernat, W. Evans, and S. Sponheim, "A phase synchrony measure for quantifying dynamic functional integration in the brain," *Human brain mapping*, vol. 32, no. 1, pp. 80–93, 2011.