Distributed Learning of Human Mobility Patterns From Cellular Network Data

Tong Wu*, Raif M. Rustamov† and Colin Goodall†
*Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ, USA
Email: tong.wu.ee@rutgers.edu
†AT&T Labs - Research, Bedminster, NJ, USA
Email: {rustamov, cgoodall}@research.att.com

***Abstract*—The advent of ubiquitous mobile devices has provided us with an abundant spatio-temporal data source that helps us understand human mobility. The big data generated from mobile devices can be distributed at different locations and it is always infeasible to aggregate the data from multiple data collection centers into one location due to communication and privacy considerations. This paper studies human mobility patterns by learning data-adaptive representations for cellular network data that are distributed across a set of interconnected nodes. It proposes a distributed algorithm, termed cloud NN-K-SVD, for collaboratively learning a sparsifying dictionary (i.e., overcomplete basis) from the data without exchanging data samples between different nodes. The effectiveness of cloud NN-K-SVD is demonstrated through experiments on anonymized Call Detail Records from Columbus, OH.**

I. Introduction

Modeling and understanding human movement patterns is of great importance in a number of fields including urban planning, epidemiology, and telecommunications. In urban planning, the design of transportation networks involves understanding of traffic volumes on commuting routes [1]. Likewise, the geographic spread of infectious diseases is often dependent on multiscale human mobility [2]. Furthermore, knowledge of the behavior of populations allows companies and governments to optimize mobile networks.

Location information from wireless cellular networks offers opportunities to the study of human mobility. The data used for mobility modeling originate from anonymized Call Detail Records (CDRs) recorded at the spatial resolution of the deployed cell phone towers (or antennas). These mobile phone location measurements are aggregated on internal servers before any further analysis.[1] In contrast to other location sources such as GPS, these data are coarse both in space and time [3], leading to impossibility of inferring complete trajectories. The ever-growing ubiquity of cellular phones is pushing network operators toward generation of massive amounts of data. In addition, the data can be geographically distributed across many data centers with limitations on the transfer of raw data due to privacy concerns. It is therefore important to develop efficient approaches for collaborative learning of human mobility patterns in this "big, distributed data" regime.

[1]No personally identifiable information (PII) was gathered or used in conducting this study which was in compliance with AT&T's privacy policy.

Sparse representation over redundant dictionaries has drawn extensive attractions and has been shown to be highly effective for many information processing tasks such as image denoising [4], image classification [5], [6], and novel document detection [7]. The basic premise is that natural signals can be well approximated using sparse linear combinations of a few vectors (also called atoms) in some overcomplete basis (or so-called dictionary). Concretely, consider a signal $\mathbf{y} \in \mathbb{R}^m$, we say $\mathbf{y}$ admits a sparse representation over a dictionary $\mathbf{D} \in \mathbb{R}^{m \times K}$ consisting of K atoms, if we can approximate $\mathbf{y}$ as $\mathbf{y} \approx \mathbf{D}\boldsymbol{\theta}$ and the number of nonzero entries in $\boldsymbol{\theta} \in \mathbb{R}^K$ is small compared to K. Intuitively, dictionary learning for sparse representation corresponds to learning a union of low-dimensional subspaces underlying data of interest. While data-adaptive dictionary learning dates back more than a decade [8]–[11], many of these works have been focused on learning a dictionary from the data available at a central location. In recent years, the interest of developing efficient algorithms for distributed dictionary learning has increased because of our move toward an era of data explosion [12], [13].

In this paper, we propose a new distributed dictionary learning algorithm, which we refer to as *cloud NN-K-SVD*, for learning human mobility patterns from anonymized CDRs that are distributed across a number of geographically distant data centers/nodes. The proposed algorithm is a distributed variant of the existing centralized NN-K-SVD algorithm [9] and each node learns a dictionary through computations on its local data and communications with its neighboring nodes. We impose non-negativity on both the dictionary atoms and sparse coefficients in our dictionary learning framework, which renders the interpretability of the resulting dictionaries for mobility patterns and facilitates their further analysis by human experts. Cloud NN-K-SVD can also be regarded as an extension of the cloud K-SVD algorithm [13], and both these works rely on the classical power method for eigenanalysis [14] and consensus averaging [15]. To the best of our knowledge, the work presented here is the first one that explores the use of dictionary learning for characterizing human mobility patterns. In order to demonstrate the validity of our proposed dictionary learning method, we carry out numerical experiments on CDRs from Columbus (OH) area. The results of these experiments show the benefit of collaborative dictionary learning in comparison to local dictionary learning.

978-1-5090-4780-2/17/$31.00 ©2017 IEEE

Notational Convention: We use non-bold letters to represent scalars/sets, bold lower-case letters to denote vectors, and bold upper-case letters to denote matrices. The i-th element of a vector $\mathbf{v}$ is denoted by $v(i)$ and the (i,j)-th element of a matrix $\mathbf{A}$ is denoted by $a_{i,j}$. The i-th column and j-th row of a matrix $\mathbf{A}$ are denoted by $\mathbf{a}_i$ and $\mathbf{a}_{j,T}$, respectively. We use $\mathbf{0}$ to denote the zero vector of appropriate dimension. Given a set Ω, $[\mathbf{A}]_{:,\Omega}$ (resp., $[\mathbf{v}]_\Omega$) denotes the submatrix of $\mathbf{A}$ (resp., subvector of $\mathbf{v}$) corresponding to the columns of $\mathbf{A}$ (resp., entries of $\mathbf{v}$) indexed by Ω. Superscript $(\cdot)^T$ denotes the transpose operation and $\|\cdot\|_0$ counts the number of nonzero entries in a vector. Finally, $\|\mathbf{v}\|_2$ denotes the ℓ_2 norm of a vector $\mathbf{v}$ and $\|\mathbf{A}\|_F$ denotes the Frobenius norm of a matrix $\mathbf{A}$.

II. Problem Formulation

In this paper, we consider a network with P distributed nodes according to an undirected graph $\mathcal{G}(\Pi, \mathcal{E})$, where $\Pi = \{1, \ldots, P\}$ denotes the nodes and $\mathcal{E}$ describes links among the nodes with $(p,q) \in \mathcal{E}$ if there exists a connection between node p and q. Here, each node corresponds to a data center, which can store hundreds of millions of anonymized CDRs, perform computations independently and exchange information with its immediate neighbors. We assume the graph $\mathcal{G}$ is a connected graph.

In order to characterize the human mobility patterns in a geographic region using cellular network data, we use the locations of cell sites with which a phone is associated as the approximated locations of the phone. We first convert the geolocations (latitude, longitude) of the cell sites into slippy tile indexes at a given zoom level h such that every cell site in the region of interest is associated with one tile[2]. To facilitate our analysis, we assume that for every anonymized user, all the CDRs associated with this user are collected in one node/data center of the network. We focus on analyzing the patterns of daily travel. Specifically, for every anonymized user whose CDRs are stored in a data center, we create an m-dimensional binary feature vector $\mathbf{y} \in \mathbb{R}^m$ to indicate the incidence between the user and the slippy tiles in one day (i.e., the vector has ones in the entries where this user has "passed through" the corresponding slippy tiles and zeros everywhere else), where m is the number of slippy tiles in the region. This feature vector describes the travel path of this user and will be regarded as one data sample in the dictionary learning.

Next, consider a collection of m-dimensional training data distributed across these P nodes, where the p-th node has N_p local data samples (which corresponds to N_p anonymized users), given by a matrix $\mathbf{Y}_p = [\mathbf{y}_{p,1}, \mathbf{y}_{p,2}, \ldots, \mathbf{y}_{p,N_p}] \in \mathbb{R}^{m \times N_p}$. We can express all the data samples using a single matrix $\mathbf{Y} = [\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_P] \in \mathbb{R}^{m \times N}$, where $N = \sum_{p=1}^{P} N_p$ denotes the total number of samples distributed across these P nodes. The basic premise in this paper is that all the distributed data samples lie near a union of s-dimensional subspaces with $s \ll m$. To be specific, assuming the distributed data $\mathbf{Y}$ are available at a fusion center, the problem of learning a (centralized) dictionary can be expressed as the following optimization problem [9]:

$$(\mathbf{D}, \mathbf{\Theta}) = \underset{\mathbf{D}, \mathbf{\Theta} \geq \mathbf{0}}{\arg\min} \|\mathbf{Y} - \mathbf{D}\mathbf{\Theta}\|_F^2 \quad \text{s.t.} \quad \forall i, \|\boldsymbol{\theta}_i\|_0 \leq s. \quad (1)$$

Here, $\mathbf{D} \in \mathbb{R}^{m \times K}$ is an overcomplete dictionary that consists of K unit ℓ_2-norm columns (i.e., $K > m$) and $\mathbf{\Theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N] \in \mathbb{R}^{K \times N}$ is the sparse coefficient matrix. In words, (1) aims to learn a dictionary $\mathbf{D}$ such that every path $\mathbf{y}_i$ can be decomposed into no more than s travel routes, where each route corresponds to an atom in the dictionary $\mathbf{D}$. For maintaining the interpretability of path-route associations, we require both $\mathbf{D}$ and $\mathbf{\Theta}$ to be non-negative. This problem is non-convex in $(\mathbf{D}, \mathbf{\Theta})$ and a natural approach for solving it is to alternate between solving (1) for $\mathbf{D}$ using a fixed $\mathbf{\Theta}$ and then solving (1) for $\mathbf{\Theta}$ using a fixed $\mathbf{D}$ [9].

However, due to the extremely high storage and communication costs of big data and privacy concerns, it is always prohibitive to gather the distributed data $\mathbf{Y}$ to a central location. In this regard, we are interested in learning a collection of dictionaries $\{\mathbf{D}_p\}_{p \in \Pi}$ through local computations and communications within the network such that the performance of these collaborative dictionaries approximates the performance of a dictionary $\mathbf{D}$ learned from $\mathbf{Y}$ in a centralized manner. In the following section, we present a decentralized variant of the dictionary learning method proposed in [9].

III. Proposed Algorithm

In this section, we first review the centralized NN-K-SVD algorithm [9], which is followed by our proposed distributed dictionary learning algorithm, termed cloud NN-K-SVD.

A. Review of Centralized NN-K-SVD Algorithm

The NN-K-SVD algorithm starts with a random dictionary $\mathbf{D}$, and solves (1) by iterating between *sparse coding* step and *dictionary update* stage [9]. Specifically, when the dictionary $\mathbf{D}$ is fixed, the sparse coding amounts to solving $\mathbf{\Theta}$ as follows:

$$\forall i, \quad \boldsymbol{\theta}_i = \underset{\boldsymbol{\theta} \in \mathbb{R}^K, \boldsymbol{\theta} \geq \mathbf{0}}{\arg\min} \|\mathbf{y}_i - \mathbf{D}\boldsymbol{\theta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\theta}\|_0 \leq s. \quad (2)$$

This problem can be solved by either convexifying (2) [16] or using greedy algorithms [17]. Afterward, given a fixed $\mathbf{\Theta}$, the dictionary update step in NN-K-SVD involves sequentially updating one atom $\mathbf{d}_k$, $k = 1, \ldots, K$, at a time, while keeping all other atoms in the dictionary fixed. In order to update $\mathbf{d}_k$, we first compute the error matrix $\mathbf{E}_k = \mathbf{Y} - \sum_{\ell \neq k} \mathbf{d}_\ell \boldsymbol{\theta}_{\ell,T}$, and let $\omega_k = \{i : 1 \leq i \leq N, \theta_{k,T}(i) \neq 0\}$. Then the problem of updating $\mathbf{d}_k$ can be expressed as the following positive rank-one optimization problem:

$$(\mathbf{d}_k, [\boldsymbol{\theta}_{k,T}]_{\omega_k}^T) = \underset{\mathbf{u}, \mathbf{v} \geq \mathbf{0}, \|\mathbf{u}\|_2^2 = 1}{\arg\min} \|\widehat{\mathbf{E}}_k - \mathbf{u}\mathbf{v}^T\|_F^2, \quad (3)$$

where $\widehat{\mathbf{E}}_k = [\mathbf{E}_k]_{:,\omega_k}$ is the reduced error matrix by keeping the columns of $\mathbf{E}_k$ indexed by ω_k only. In order to find starting points for $\mathbf{u}$ and $\mathbf{v}$, we apply singular value decomposition (SVD) on $\widehat{\mathbf{E}}_k$ and then cancel out the negative entries. More precisely, we initialize $\mathbf{u}$ and $\mathbf{v}$ by setting $\mathbf{u} = \mathbf{a}$ and $\mathbf{v} = \sigma\mathbf{b}$,

[2]https://wiki.openstreetmap.org/wiki/Slippy_map_tilenames

where $\mathbf{a}$ and $\mathbf{b}$ are the dominant left and right singular vectors of $\widehat{\mathbf{E}}_k$, respectively, while σ denotes the largest singular value of $\widehat{\mathbf{E}}_k$. In order to make both $\mathbf{u}$ and $\mathbf{v}$ positive, we set the negative entries of $\mathbf{u}$ and $\mathbf{v}$ to be zeros by performing the following operations: $\mathbf{u} = \mathbf{u} \odot [\mathbf{u} \geq \mathbf{0}]$ and $\mathbf{v} = \mathbf{v} \odot [\mathbf{v} \geq \mathbf{0}]$, where $\odot$ denotes the element-wise product between two vectors. After this, we adopt an alternate minimization approach [18], which involves iteratively updating $\mathbf{u}$ for a fixed $\mathbf{v}$ using (4) and then updating $\mathbf{v}$ for a fixed $\mathbf{u}$ using (5) [9]:

$$\mathbf{u} = \frac{\widehat{\mathbf{E}}_k \mathbf{v}}{\mathbf{v}^T \mathbf{v}}, \quad \mathbf{u} = \mathbf{u} \odot [\mathbf{u} \geq \mathbf{0}], \tag{4}$$

$$\mathbf{v} = \frac{\widehat{\mathbf{E}}_k^T \mathbf{u}}{\mathbf{u}^T \mathbf{u}}, \quad \mathbf{v} = \mathbf{v} \odot [\mathbf{v} \geq \mathbf{0}]. \tag{5}$$

This process is repeated until convergence is achieved. Finally, we set $\mathbf{d}_k = \frac{\mathbf{u}}{\|\mathbf{u}\|_2}$ and $[\boldsymbol{\theta}_{k,T}]_{\omega_k} = \|\mathbf{u}\|_2 \mathbf{v}^T$. After finishing the update of all the K atoms in $\mathbf{D}$, NN-K-SVD [9] then moves to the sparse coding step and this two-stage iterative process continues until a stopping criterion is satisfied.

B. Distributed Dictionary Learning Using Cloud NN-K-SVD

We now introduce our proposed distributed dictionary learning algorithm based on NN-K-SVD. Similar to the centralized NN-K-SVD, our distributed algorithm starts with a common random dictionary $\mathbf{D}_{\text{init}}$ for all the nodes. In the sparse coding stage, we propose that each node p computes the sparse coefficients of its local data $\mathbf{Y}_p$ using the local dictionary $\mathbf{D}_p$ without collaborating with other nodes by solving

$$\forall i, \quad \boldsymbol{\theta}_{p,i} = \underset{\boldsymbol{\theta} \in \mathbb{R}^K, \boldsymbol{\theta} \geq \mathbf{0}}{\arg\min} \|\mathbf{y}_{p,i} - \mathbf{D}_p \boldsymbol{\theta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\theta}\|_0 \leq s, \tag{6}$$

where $\mathbf{y}_{p,i}$ and $\boldsymbol{\theta}_{p,i}$ denote the i-th sample and its corresponding coefficient vector at node p, respectively. In this paper, we propose to use Nonnegative Matching Pursuit (NMP) [17] to solve (6) because of its greedy nature.

The main challenge in our distributed dictionary learning problem lies in the update of the dictionary atoms. As described in Section III-A, in order to update $\mathbf{d}_k$ in the centralized setting, we need to compute the dominant left and right singular vectors of the reduced error matrix $\widehat{\mathbf{E}}_k$. However, in the distributed scenario, this error matrix is distributed across the network and each node p has its own reduced error matrix $\widehat{\mathbf{E}}_{p,k}$ computed from its local data. Mathematically speaking, we can express the distributed error matrix as $\widehat{\mathbf{E}}_k = [\widehat{\mathbf{E}}_{1,k}, \widehat{\mathbf{E}}_{2,k}, \ldots, \widehat{\mathbf{E}}_{P,k}]$, where $\widehat{\mathbf{E}}_{p,k} = [\mathbf{E}_{p,k}]_{:,\omega_{p,k}}$ with $\mathbf{E}_{p,k} = \mathbf{Y}_p - \sum_{\ell \neq k} \mathbf{d}_{p,\ell} \boldsymbol{\theta}_{p,\ell,T}$ and $\omega_{p,k} = \{i : 1 \leq i \leq N_p, \theta_{p,k,T}(i) \neq 0\}$. Here, $\mathbf{d}_{p,k}$ denotes the k-th atom of the dictionary $\mathbf{D}_p$ and $\boldsymbol{\theta}_{p,k,T}$ denotes the k-th row of the sparse coefficient matrix $\boldsymbol{\Theta}_p$ at node p. In order to estimate the dominant left singular vector of $\widehat{\mathbf{E}}_k$ (again denoted by $\mathbf{a}$) over the network, we perform distributed power method described in [13] by using the distributed error matrices $\widehat{\mathbf{E}}_{p,k}$'s, and we denote the collection of the estimates of $\mathbf{a}$ at different nodes by $\{\mathbf{u}_p\}_{p=1}^P$. We omit the details here in the interest of space. We again use σ and $\mathbf{b}$ to denote the largest singular value and dominant right singular vector of $\widehat{\mathbf{E}}_k$, respectively. Note that

Algorithm 1 Cloud NN-K-SVD for dictionary learning

Input: Distributed data $\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_P$, parameters K and s, and a doubly-stochastic matrix $\mathbf{W}$.

Initialize: Generate a random dictionary $\mathbf{D}_{\text{init}} \in \mathbb{R}^{m \times K}$ and set $\mathbf{D}_p = \mathbf{D}_{\text{init}}$, $p = 1, \ldots, P$.

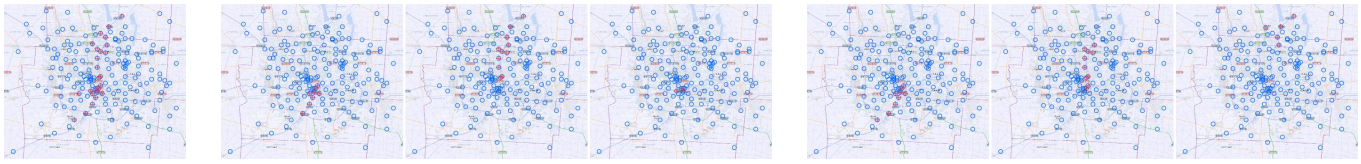
while stopping rule **do**
  The p-th site locally solves (*Sparse Coding*)
  $\forall i$, $\boldsymbol{\theta}_{p,i} = \arg\min_{\boldsymbol{\theta} \geq \mathbf{0}} \|\mathbf{y}_{p,i} - \mathbf{D}_p \boldsymbol{\theta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\theta}\|_0 \leq s.$
  for $k = 1$ to K (*Dictionary Update*) **do**
    $\forall p$, $\mathbf{E}_{p,k} = \mathbf{Y}_p - \sum_{\ell \neq k} \mathbf{d}_{p,\ell} \boldsymbol{\theta}_{p,\ell,T}$, $\omega_{p,k} = \{i : 1 \leq i \leq N_p, \theta_{p,k,T}(i) \neq 0\}$ and $\widehat{\mathbf{E}}_{p,k} = [\mathbf{E}_{p,k}]_{:,\omega_{p,k}}$.
    $\mathbf{F}_p = \widehat{\mathbf{E}}_{p,k} \widehat{\mathbf{E}}_{p,k}^T$.
    Generate $\mathbf{z}_{\text{init}}$ randomly, set $t_b = 0$ and $\forall p$, $\mathbf{z}_p^{(0)} = \mathbf{z}_{\text{init}}$.
    while stopping rule (*Distributed Power Method*) **do**
      $t_b = t_b + 1$, $t_c = 0$ and $\forall p$, $\boldsymbol{\chi}_p^{(0)} = \mathbf{F}_p \mathbf{z}_p^{(t_b - 1)}$.
      while stopping rule (*Consensus Averaging*) **do**
        $t_c = t_c + 1$, $\forall p$, $\boldsymbol{\chi}_p^{(t_c)} = \sum_{q \in \mathcal{N}_p} w_{p,q} \boldsymbol{\chi}_q^{(t_c - 1)}$.
      end while
      $\forall p$, $\mathbf{z}_p^{(t_b)} = \boldsymbol{\chi}_p^{(t_c)} / \|\boldsymbol{\chi}_p^{(t_c)}\|_2$.
    end while
    $\forall p$, $\mathbf{u}_p = \mathbf{z}_p^{(t_b)}$ and $\mathbf{v}_p = \widehat{\mathbf{E}}_{p,k}^T \mathbf{u}_p$.
    $\forall p$, $\mathbf{u}_p = \mathbf{u}_p \odot [\mathbf{u}_p \geq \mathbf{0}]$, $\mathbf{v}_p = \mathbf{v}_p \odot [\mathbf{v}_p \geq \mathbf{0}]$.
    while stopping rule **do**
      $\forall p$, $\mathbf{c}_p = \widehat{\mathbf{E}}_{p,k} \mathbf{v}_p$ and $r_p = \mathbf{v}_p^T \mathbf{v}_p$.
      $t_c = 0$ and $\forall p$, $\boldsymbol{\phi}_p^{(0)} = \mathbf{c}_p$ and $\psi_p^{(0)} = r_p$.
      while stopping rule (*Consensus Averaging*) **do**
        $t_c = t_c + 1$, $\forall p$, $\boldsymbol{\phi}_p^{(t_c)} = \sum_{q \in \mathcal{N}_p} w_{p,q} \boldsymbol{\phi}_q^{(t_c - 1)}$ and $\psi_p^{(t_c)} = \sum_{q \in \mathcal{N}_p} w_{p,q} \psi_q^{(t_c - 1)}$.
      end while
      $\forall p$, $\mathbf{u}_p = \boldsymbol{\phi}_p^{(t_c)} / \psi_p^{(t_c)}$, $\mathbf{u}_p = \mathbf{u}_p \odot [\mathbf{u}_p \geq \mathbf{0}]$.
      $\forall p$, $\mathbf{v}_p = \frac{\widehat{\mathbf{E}}_{p,k}^T \mathbf{u}_p}{\mathbf{u}_p^T \mathbf{u}_p}$, $\mathbf{v}_p = \mathbf{v}_p \odot [\mathbf{v}_p \geq \mathbf{0}]$.
    end while
    $\forall p$, $\mathbf{d}_{p,k} = \frac{\mathbf{u}_p}{\|\mathbf{u}_p\|_2}$ and $[\boldsymbol{\theta}_{p,k,T}]_{\omega_{p,k}} = \|\mathbf{u}_p\|_2 \mathbf{v}_p^T$.
  end for
end while

Output: A collection of dictionaries $\{\mathbf{D}_p \in \mathbb{R}^{m \times K}\}_{p=1}^P$.

$\widehat{\mathbf{E}}_k^T \mathbf{a} = [\widehat{\mathbf{E}}_{1,k}, \widehat{\mathbf{E}}_{2,k}, \ldots, \widehat{\mathbf{E}}_{P,k}]^T \mathbf{a} = \sigma \mathbf{b}$ and we can write the variable $\mathbf{v}$ in (3) as $\mathbf{v}^T = [\mathbf{v}_1^T, \mathbf{v}_2^T, \ldots, \mathbf{v}_P^T]$ in the distributed setting, where $\mathbf{v}_p$ is the variable corresponding to the reduced coefficient vector $[\boldsymbol{\theta}_{p,k,T}]_{\omega_{p,k}}^T$. It then follows that once we have $\mathbf{u}_p$'s available at different nodes, we can initialize $\mathbf{v}_p$'s by setting $\mathbf{v}_p = \widehat{\mathbf{E}}_{p,k}^T \mathbf{u}_p$. In order to make the initial $\mathbf{u}_p$'s and $\mathbf{v}_p$'s positive, we again perform the following operations:

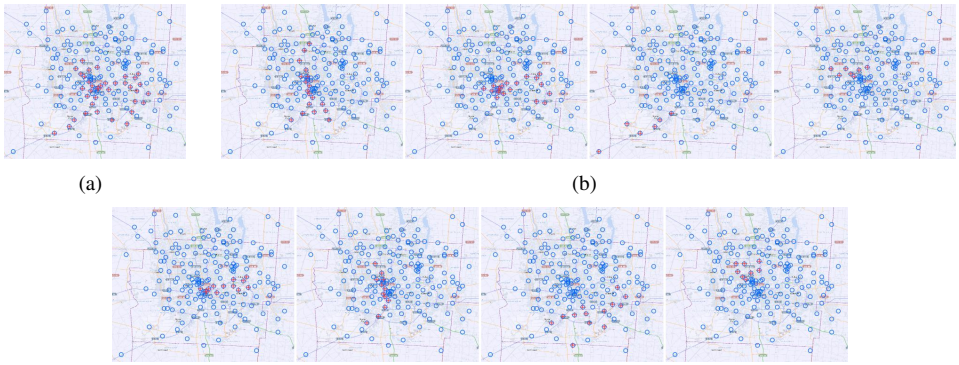
$$\forall p, \quad \mathbf{u}_p = \mathbf{u}_p \odot [\mathbf{u}_p \geq \mathbf{0}], \quad \mathbf{v}_p = \mathbf{v}_p \odot [\mathbf{v}_p \geq \mathbf{0}]. \tag{7}$$

Once we have the initial vectors $\mathbf{u}_p$'s and $\mathbf{v}_p$'s, we move onto the stage where we iteratively update $\mathbf{u}_p$'s and $\mathbf{v}_p$'s using alternate minimization. Notice that in the distributed setting, we can write the first equation in (4) as $\mathbf{u} = \frac{\sum_{p=1}^P \widehat{\mathbf{E}}_{p,k} \mathbf{v}_p}{\sum_{p=1}^P \mathbf{v}_p^T \mathbf{v}_p}$. In this manner, we first compute each $\mathbf{c}_p = \widehat{\mathbf{E}}_{p,k} \mathbf{v}_p$ and $r_p = \mathbf{v}_p^T \mathbf{v}_p$ locally, and then make use of the consensus averaging



(a) (b) (c)

Fig. 1. Comparison between centralized K-SVD and NN-K-SVD for learning the human mobility patterns. (a) represents a test vector/path on May 7, 2016. The associated atoms of this test vector for K-SVD and NN-K-SVD are shown in (b) and (c), respectively.



(a) (b) (c)

Fig. 2. Comparison between centralized K-SVD and NN-K-SVD for learning the human mobility patterns. (a) represents a test vector/path on May 12, 2016. The associated atoms of this test vector for K-SVD and NN-K-SVD are shown in (b) and (c), respectively.

method [15] to compute the summation of these individual vectors/scalars over the network. To be specific, we first design a doubly-stochastic matrix $\mathbf{W}$ according to the topology of the network graph $\mathcal{G}$ [15]. Next, each node is initialized with a vector $\boldsymbol{\phi}_p^{(0)} = \mathbf{c}_p$ (we also set $\psi_p^{(0)} = r_p$, since the computation of $\sum_{p=1}^P r_p$ has the same principle, we focus on describing the procedure of computing $\sum_{p=1}^P \mathbf{c}_p$ here) and we define $\boldsymbol{\Phi}^{(0)} = [\boldsymbol{\phi}_1^{(0)}, \ldots, \boldsymbol{\phi}_P^{(0)}]^T \in \mathbb{R}^{P\times m}$. We also let $\mathcal{N}_p = \{q : (p,q) \in \mathcal{E}\}$ be the set of neighbors of node p. In each iteration, each node updates its local vector through the communications with its neighbors as follows: $\boldsymbol{\phi}_p^{(t_c)} = \sum_{q\in\mathcal{N}_p} w_{p,q}\boldsymbol{\phi}_q^{(t_c-1)}$, where t_c denotes the iteration number. This iteration can be written in vector form as $\boldsymbol{\Phi}^{(t_c)} = \mathbf{W}^{t_c}\boldsymbol{\Phi}^{(0)}$ and it has been shown in [15] that $\lim_{t_c\to\infty}\boldsymbol{\phi}_p^{(t_c)} = (\boldsymbol{\Phi}^{(0)})^T\mathbf{1}/P$ for all p's, where $\mathbf{1} \in \mathbb{R}^P$ is a vector of all ones. But in practice, we can only perform a finite number of iterations for consensus averaging, which we denote by T_c, and the averaging result gets better as T_c grows. In our experiments, the consensus iterations T_c is set to be 10. After finishing consensus iterations for both $\mathbf{c}_p$'s and r_p's, each node p then updates $\mathbf{u}_p$ by setting $\mathbf{u}_p = \boldsymbol{\phi}_p^{(T_c)}/\psi_p^{(T_c)}$ and $\mathbf{u}_p = \mathbf{u}_p \odot [\mathbf{u}_p \geq \mathbf{0}]$. Once we have the updated $\mathbf{u}_p$'s available at different nodes, we then simply update $\mathbf{v}_p$'s locally as $\mathbf{v}_p = \frac{\widehat{\mathbf{E}}_{p,k}^T\mathbf{u}_p}{\mathbf{u}_p^T\mathbf{u}_p}$ and $\mathbf{v}_p = \mathbf{v}_p \odot [\mathbf{v}_p \geq \mathbf{0}]$. The update of $\mathbf{u}_p$'s and $\mathbf{v}_p$'s is repeated until a convergence criteria is met. Finally, we set $\mathbf{d}_{p,k} = \frac{\mathbf{u}_p}{\|\mathbf{u}_p\|_2}$ and $[\boldsymbol{\theta}_{p,k,T}]_{\omega_{p,k}} = \|\mathbf{u}_p\|_2\mathbf{v}_p^T$. This concludes our discussion of the dictionary update step. A complete description of the resulting algorithm, which we term *cloud NN-K-SVD*, is given in Algorithm 1.

IV. Experimental Results

In this section, we present the results of human mobility characterization using centralized/distributed dictionary learning. We focus on analyzing the records of Columbus, OH and we work with three days of data: May 7, 2016 (Saturday), May 11, 2016 (Wednesday), and May 12, 2016 (Thursday). These mobile phone location measurements are anonymized and the locations are discretized to a grid of slippy tiles, each corresponds to a 600m $\times$ 600m area. All the analysis is conducted on internal AT&T servers. There are 175 slippy tiles/grids in the region of interest. We exclude the users who "passed through" less than 20 slippy tiles on weekdays and 9 slippy tiles on weekends since there is little to no mobility of these users. This leaves us with roughly 16000 anonymous users for analysis in each of these three days. Afterwards, we create $m = 175$-dimensional binary feature vectors for all the these users as described in Section II and these vectors are used in the experiments. From all the retained samples, we randomly select 12000 samples for training and the remaining samples (roughly 4000 samples) are used for testing purposes. This random selection is repeated five times. All the training samples are normalized to have unit ℓ_2-norms.

We first examine the performance of learning the mobility patterns using centralized dictionary learning methods. We perform centralized K-SVD/NN-K-SVD with parameters: $m = 175$, $N = 12000$, $K = 200$, and the sparsity level s is set to be 3 for the data on weekends and 4 for the data on weekdays. Then for a test sample $\mathbf{y}$, we apply Orthogonal Matching Pursuit (OMP) [19] and NMP [17] to compute its sparse representation coefficient vector $\boldsymbol{\theta}$ in terms of the

TABLE I
RELATIVE REPRESENTATION ERRORS OF TEST DATA FOR DIFFERENT DICTIONARY LEARNING APPROACHES

	centralized K-SVD	local K-SVD	cloud K-SVD	centralized NN-K-SVD	local NN-K-SVD	cloud NN-K-SVD
May 7	0.240	0.301	0.283	0.237	0.274	0.238
May 11	0.252	0.325	0.277	0.246	0.279	0.246
May 12	0.252	0.326	0.278	0.247	0.280	0.247

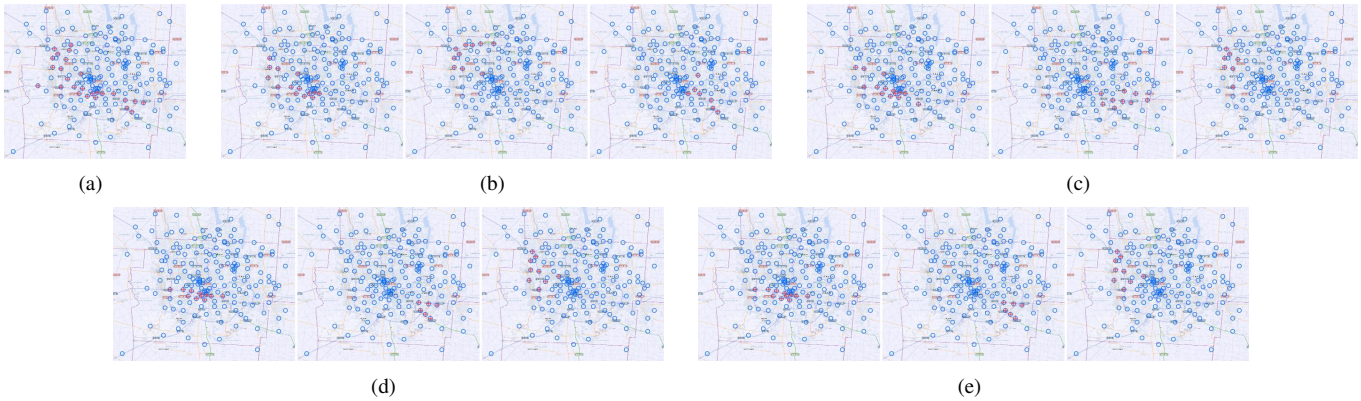


Fig. 3. Comparison between local NN-K-SVD and cloud NN-K-SVD for learning the human mobility patterns. (a) represents a test vector/path on May 7, 2016. The associated atoms of this test vector for local NN-K-SVD at node 4 and node 8 are shown in (b) and (c), respectively. The associated atoms of this test vector for cloud NN-K-SVD at node 4 and node 8 are shown in (d) and (e), respectively.

learned dictionary by K-SVD and NN-K-SVD, respectively. The relative representation error with respect to $\mathbf{y}$ can be defined as $e = \frac{\|\mathbf{y}-\mathbf{D}\boldsymbol{\theta}\|_2^2}{\|\mathbf{y}\|_2^2}$, and we use the mean of relative representation errors of test data to quantitatively evaluate the performance of dictionary learning. As shown in Table I, the relative error of test data using the learned dictionary by NN-K-SVD is slightly less than the one computed using the learned dictionary by K-SVD for all the three days. The visualization of a test path on May 7 and its associated routes/atoms for K-SVD/NN-K-SVD with $s = 3$ is represented in Fig. 1. The blue dots in the figures correspond to the center location of the slippy tiles, and the dots with red cross in Fig. 1(a) correspond to the "active" slippy tiles of this test path and the dots with red cross in each of the plots in Fig. 1(b) and Fig. 1(c) correspond to the slippy tiles whose respective entries in the associated atoms have absolute values greater than 0.1. By comparing Fig. 1(b) with Fig. 1(c), we can see the third associated atom for K-SVD has only few "active" slippy tiles and has large overlap with the first associated one, whereas the three associated atoms for NN-K-SVD do not have much overlap. The main reason for the overlap between the associated atoms is that K-SVD allows negative dictionary atoms and sparse coefficients, and some learned atoms are only used to reduce the representation error without interpretability. We also show a test path on May 12 and its associated routes/atoms with $s = 4$ in Fig. 2, from which we observe that the third associated atom for K-SVD has large overlap with the first associated one. These results confirm the superiority of applying nonnegative dictionary learning methods for learning the human mobility patterns.

Next, we study the performance of distributed dictionary learning on human mobility data. For each set of training and test data, we randomly generate a network with $P = 10$ different nodes using an Erdős-Rényi graph with probability 0.5, and each node has 1200 training samples. The generation of the network is also repeated five times. We perform both cloud NN-K-SVD and cloud K-SVD [13] for collaborative dictionary learning, in contrast with localized dictionary learning approaches, where each node learns a local dictionary from its local data using K-SVD and NN-K-SVD (which we term local K-SVD and local NN-K-SVD, respectively). For a test sample $\mathbf{y}$, we compute its relative representation error at node p using the dictionary $\mathbf{D}_p$ as $e_p = \frac{\|\mathbf{y}-\mathbf{D}_p\boldsymbol{\theta}_p\|_2^2}{\|\mathbf{y}\|_2^2}$, where $\boldsymbol{\theta}_p$ is computed using OMP for cloud/local K-SVD and NMP for cloud/local NN-K-SVD, respectively. Table I summarizes the mean of the relative representation errors of the test data for all the nodes, from which we provide the evidence that cloud NN-K-SVD outperforms the local NN-K-SVD in terms of smaller relative errors of the test data, and the performance of cloud NN-K-SVD is very close to the one for centralized NN-K-SVD. In order to further compare cloud NN-K-SVD with local NN-K-SVD, we again show some test examples and their associated atoms for cloud NN-K-SVD and local NN-K-SVD at some nodes in Fig. 3 and Fig. 4. As can be seen from Fig. 3, the first and the third associated atom for local NN-K-SVD at node 4 (Fig. 3(b)) look similar to the first and the second associated atom at node 8 (Fig. 3(c)), respectively. However, the second associated atom for local NN-K-SVD at node 4 is different from the third associated atom at node 8. In contrast, it can be inferred from Fig. 3(d) and Fig. 3(e) that the associated atoms for cloud NN-K-SVD at node 4 and 8 are very similar and the consistency is another main advantage of using collaborative dictionary learning. We can also observe a similar phenomenon in Fig. 4. Furthermore, some atoms learned by local NN-K-SVD do not have continuous patterns (e.g., the second associated atom at

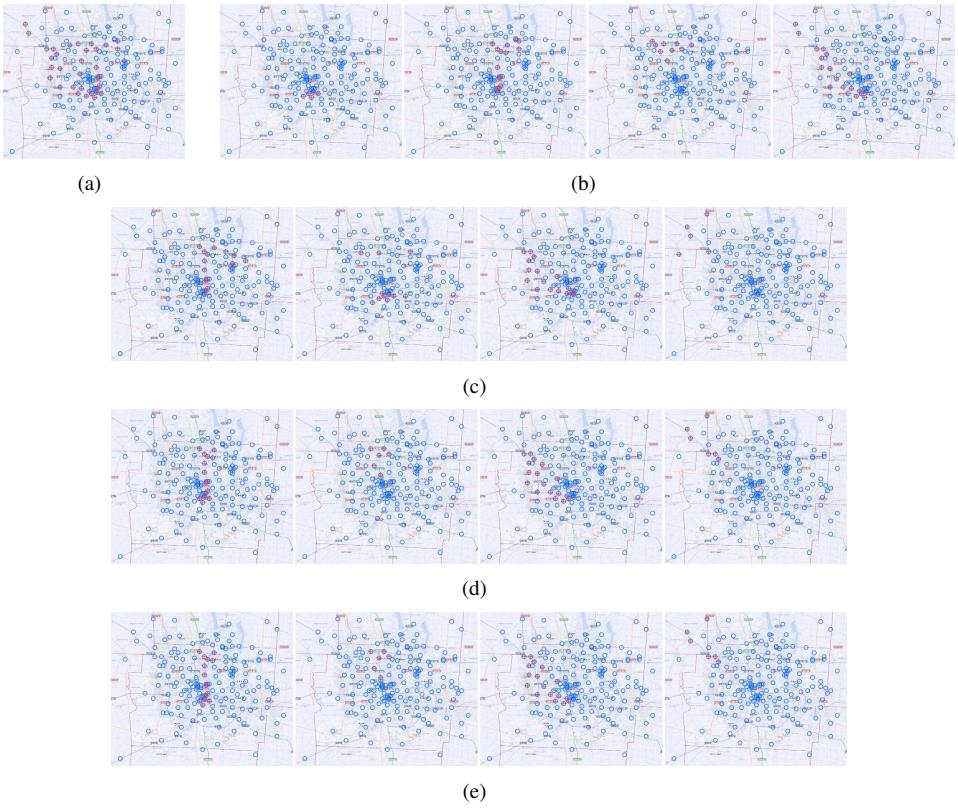


Fig. 4. Comparison between local NN-K-SVD and cloud NN-K-SVD for learning the human mobility patterns. (a) represents a test vector/path on May 11, 2016. The associated atoms of this test vector for local NN-K-SVD at node 3 and node 6 are shown in (b) and (c), respectively. The associated atoms of this test vector for cloud NN-K-SVD at node 3 and node 6 are shown in (d) and (e), respectively.

node 6 in Fig. 4(c)) and these atoms are difficult to interpret. Therefore, our proposed method outperforms the local NN-K-SVD in learning the human mobility patterns.

V. Conclusion

In this paper, we have proposed a new distributed dictionary learning algorithm, termed cloud NN-K-SVD, for collaboratively learning the dictionary from massive data that are distributed across interconnected nodes in the network. The efficacy of the proposed algorithm is demonstrated through numerical experiments on anonymized Call Detail Records with an application of human mobility characterization.

References

[1] "The journey to work: Relation between employment and residence," American Society of Planning Officials, Tech. Rep. 26, 1951.

[2] D. Balcan, V. Colizza, B. Gonçalves, H. Hu, J. J. Ramasco, and A. Vespignani, "Multiscale mobility networks and the spatial spreading of infectious diseases," in *Proc. Natl. Acad. Sci.*, vol. 106, no. 51, 2009, pp. 21 484–21 489.

[3] R. Becker, R. Cáceres, K. Hanson, S. Isaacman, J. M. Loh, M. Martonosi, J. Rowland, S. Urbanek, A. Varshavsky, and C. Volinsky, "Human mobility characterization from cellular network data," *Commun. ACM*, vol. 56, no. 1, pp. 74–82, 2013.

[4] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[5] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach, "Supervised dictionary learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2008, pp. 1033–1040.

[6] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 791–804, 2012.

[7] S. P. Kasiviswanathan, H. Wang, A. Banerjee, and P. Melville, "Online ℓ_1-dictionary learning with application to novel document detection," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 2258–2266.

[8] K. Engan, S. O. Aase, and J. H. Husøy, "Method of optimal directions for frame design," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Process. (ICASSP)*, vol. 5, 1999, pp. 2443–2446.

[9] M. Aharon, M. Elad, and A. M. Bruckstein, "K-SVD and its non-negative variant for dictionary design," in *Proc. SPIE conference wavelets*, vol. 5914, 2005, pp. 327–339.

[10] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[11] R. Rubinstein, M. Zibulevsky, and M. Elad, "Double sparsity: Learning sparse dictionaries for sparse signal approximation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1553–1564, 2010.

[12] J. Liang, M. Zhang, X. Zeng, and G. Yu, "Distributed dictionary learning for sparse representation in sensor networks," *IEEE Trans. Image Process.*, vol. 23, no. 6, pp. 2528–2541, 2014.

[13] H. Raja and W. U. Bajwa, "Cloud K-SVD: A collaborative dictionary learning algorithm for big, distributed data," *IEEE Trans. Signal Process.*, vol. 64, no. 1, pp. 173–188, 2016.

[14] G. H. Golub and C. F. Van Loan, *Matrix computations*. Johns Hopkins University Press, 1996.

[15] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems and Control Letters*, vol. 53, pp. 65–78, 2004.

[16] D. L. Donoho and J. Tanner, "Sparse nonnegative solution of underdetermined linear equations by linear programming," in *Proc. Natl. Acad. Sci.*, vol. 102, no. 27, 2005, pp. 9446–9451.

[17] R. Peharz, M. Stark, and F. Pernkopf, "Sparse nonnegative matrix factorization using ℓ^0-constraints," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, 2010, pp. 83–88.

[18] D. P. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.

[19] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.