SWIFT: Scalable Wasserstein Factorization for Sparse Nonnegative Tensors

Ardavan Afshar,[1] Kejing Yin,[2] Sherry Yan,[3] Cheng Qian,[4] Joyce Ho,[5] Haesun Park,[1] Jimeng Sun[6]
[1] Georgia Institute of Technology [2] Hong Kong Baptist University [3] Sutter Health [4] IQVIA
[5] Emory University [6] University of Illinois at Urbana-Champaign
aafshar8@gatech.edu, cskjyin@comp.hkbu.edu.hk, YanSX@sutterhealth.org, alextoqc@gmail.com, joyce.c.ho@emory.edu,
hpark@cc.gatech.edu, jimeng@illinois.edu

Abstract

Existing tensor factorization methods assume that the input tensor follows some specific distribution (i.e. Poisson, Bernoulli, and Gaussian), and solve the factorization by minimizing some empirical loss functions defined based on the corresponding distribution. However, it suffers from several drawbacks: 1) In reality, the underlying distributions are complicated and unknown, making it infeasible to be approximated by a simple distribution. 2) The correlation across dimensions of the input tensor is not well utilized, leading to sub-optimal performance. Although heuristics were proposed to incorporate such correlation as side information under Gaussian distribution, they can not easily be generalized to other distributions. Thus, a more principled way of utilizing the correlation in tensor factorization models is still an open challenge. Without assuming any explicit distribution, we formulate the tensor factorization as an *optimal transport* problem with Wasserstein distance, which can handle non-negative inputs.

We introduce SWIFT, which minimizes the Wasserstein distance that measures the distance between the input tensor and that of the reconstruction. In particular, we define the N-th order tensor Wasserstein loss for the widely used tensor CP factorization and derive the optimization algorithm that minimizes it. By leveraging sparsity structure and different equivalent formulations for optimizing computational efficiency, SWIFT is as scalable as other well-known CP algorithms. Using the factor matrices as features, SWIFT achieves up to 9.65% and 11.31% relative improvement over baselines for downstream prediction tasks. Under the noisy conditions, SWIFT achieves up to 15% and 17% relative improvements over the best competitors for the prediction tasks.

1 Introduction

Tensor factorization techniques are effective and powerful tools for analyzing multi-modal data and have been shown tremendous success in a wide range of applications including spatio-temporal analysis (Bahadori, Yu, and Liu 2014; Fanaee-T and Gama 2016), graph analysis (Gujral, Pasricha, and Papalexakis 2020), and health informatics (Yin et al. 2019; He, Henderson, and Ho 2019) applications. Many constraints such as non-negativity (Kim, He, and Park 2014), sparsity (Henderson et al. 2017), orthogonality (Wang et al. 2015), and smoothness (Afshar et al. 2018) are imposed on tensor methods in order to improve the performance both quantitatively and qualitatively. Moreover, depending on the nature of input data, tensor methods fit different distributions on data including Gaussian (Bader and Kolda 2007), Poisson (Chi and Kolda 2012), Bernoulli (Hong, Kolda, and Duersch 2020) distributions and minimize various empirical loss functions such as sum square loss, KL-divergence, and log-loss. However, there are several limitations with these techniques. 1) Existing factorization models often assume some specific data distributions, yet in practice, the underlying distributions are complicated and often unknown. 2) The nature of these factorization models neglects correlation relations within each tensor mode (such as external knowledge about similarity among those features). Although there are several extensions to tensor factorization approaches that consider these similarity matrices as side information (Acar, Kolda, and Dunlavy 2011; Kim et al. 2017), they are derived under Gaussian distribution and are not directly generalizable to unknown distributions.

Recent success of *Wasserstein distance or loss* (*a.k.a. earth mover's distance* or *optimal transport* distance) shows its potential as a better measure of the difference between two distributions (Arjovsky, Chintala, and Bottou 2017; Cuturi 2013; Frogner et al. 2015). This distance metric provides a natural measure of the distance between two distributions via a ground metric of choice and can be defined as the cost of the optimal transport plan for moving the mass in the source distribution to match that in the target one. Recently, Wasserstein distance has been applied to matrix factorization and dictionary learning problems with great success (Sandler and Lindenbaum 2009; Rolet, Cuturi, and Peyré 2016; Qian et al. 2016; Schmitz et al. 2018; Varol, Nejatbakhsh, and McGrory 2019). To the best of our knowledge, its extension to tensor factorization was never studied and is in fact non-trivial due to the following challenges:

- **Wasserstein loss is not well-defined for tensors:** The Wasserstein loss is originally defined over vectors, where each entry of the vector represents one physical location and the cost of transporting from one location to another is used. Existing matrix-based Wasserstein loss are defined by the sum of the vector-based Wasserstein loss over the columns of the matrices (Rolet, Cuturi, and Peyré 2016). Unfortunately, this definition is not applicable to tensors of multiple modes.

- **Wasserstein loss is difficult to scale:** Learning with Wasserstein loss generally requires solving the optimal transport problem in each iteration, which is extremely time-consuming.
- **Large and sparse nonnegative input:** Existing works on Wasserstein matrix factorization are developed based on dense input with relatively small size. In reality, tensors can often be very large are are often sparse. Efficient learning algorithms are possible only when the sparsity structures of the large input tensor are properly identified and fully utilized.

To overcome these challenges, we propose SWIFT, a tensor factorization method which efficiently minimizes Wasserstein distance for sparse nonnegative tensors. The main contributions of SWIFT include:

- **Defining Optimal Transport for Tensors that Handles Nonnegative Input:** SWIFT is the first technique that minimizes optimal transport (OT) distance for tensor factorization approaches. The benefit of OT is that it does not assume any specific distribution in the data. SWIFT is able to handle nonnegative inputs such as binary, counts and probability measures, which are common input in real-world tensor data.
- **Full Utilization of Data Sparsity and Parallelism:** By fully exploring and utilizing the sparsity structure of the input data, SWIFT significantly reduces the number of times required to compute OT and enables parallelism. SWIFT obtains up to $16\times$ faster OT computation than direct implementation of Wasserstein tensor factorization.
- **Efficient Computation:** SWIFT reduces the amount of computations by smartly rearranging the objective function for solving each factor matrix. Scalability of SWIFT is comparable with well-known CP algorithms. Moreover, SWIFT achieves up to $921\times$ speed up over a direct implementation of Wasserstein tensor factorization without our speedup strategies, as shown in the appendix.

2 Notations and Background

2.1 Basic Notations and Tensor Operations

We denote vectors by bold lowercase letters (*e.g.* $\mathbf{u}$), matrices by bold uppercase letters (*e.g.* $\mathbf{A}$), and tensors by Euler script letters (*e.g.* $\mathcal{X}$). The entropy E for a nonnegative matrix $\mathbf{A} \in \mathbb{R}_+^{M\times N}$ is defined as $E(\mathbf{A}) = -\sum_{i,j=1}^{M,N} \mathbf{A}(i,j)\log(\mathbf{A}(i,j))$. $KL(\mathbf{A}||\mathbf{B})$ is the generalized KL-divergence between two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{M\times N}$ is defined as $KL(\mathbf{A}||\mathbf{B}) = \sum_{i,j=1}^{M,N} \mathbf{A}(i,j)\log(\frac{\mathbf{A}(i,j)}{\mathbf{B}(i,j)}) - \mathbf{A}(i,j) + \mathbf{B}(i,j)$.

Mode-n Matricization. Matricization (Kolda and Bader 2009) is the process of reordering the entries of a tensor into a matrix. Specifically, the mode-n matricization is the concatenation of all the mode-n fibers obtained by fixing the indices for every but the n^{th} mode. It transforms the tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\ldots\times I_N}$ into matrix $\mathbf{X}_{(\mathbf{n})}$, and the size of the resulting matrix is I_n by $I_1\ldots I_{n-1}I_{n+1}\ldots I_N$. To ease the notation, we define $I_{(-n)} = I_1\ldots I_{n-1}I_{n+1}\ldots I_N$.

Khatri-Rao Product. The Khatri-Rao product (Kolda and Bader 2009) of two matrices $\mathbf{A} \in \mathbb{R}^{I\times R}$ and $\mathbf{B} \in \mathbb{R}^{J\times R}$ is the column-wise Kronecker product $\mathbf{C} = \mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \cdots \quad \mathbf{a}_R \otimes \mathbf{b}_R]$, where $\mathbf{a}_i, \mathbf{b}_i$ are the column-i of matrices $\mathbf{A}$ and $\mathbf{B}$, $\otimes$ denotes the Kronecker product, and $\mathbf{C} \in \mathbb{R}^{IJ\times K}$. We denote the Khatri-Rao product of all factor matrices except the n-th mode as

$$\mathbf{A}_{\odot}^{(-n)} = (\mathbf{A}_N \odot \ldots \odot \mathbf{A}_{n+1} \odot \mathbf{A}_{n-1} \odot \ldots \odot \mathbf{A}_1) \in \mathbb{R}^{I_{(-n)}\times R}, \quad (1)$$

where $\mathbf{A}_n \in \mathbb{R}^{I_n\times R}$ indicates the n-th factor matrix.

Canonical/Polyadic (CP) decomposition. The CP factorization (Kolda and Bader 2009) approximates a tensor $\mathcal{X}$ as the sum of rank-one tensors ($\hat{\mathcal{X}} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!] = \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \ldots \circ \mathbf{a}_r^{(N)}$), where $\hat{\mathcal{X}}$ is a reconstructed tensor, $\mathbf{a}_r^{(n)}$ is the r-th column of factor matrix $\mathbf{A}^{(n)}$, $\circ$ denotes the outer product of vectors, and R is the number of the rank-one tensors to approximate the input tensor, *i.e.*, the target rank. The mode-n matricization of reconstructed tensor $\hat{\mathcal{X}}$ is $\hat{\mathbf{X}}_{(n)} = \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T \in \mathbb{R}^{I_n\times I_{(-n)}}$.

2.2 Preliminaries & Related Work

Wasserstein Distance and Optimal Transport. Wasserstein distance (*a.k.a.* earth mover's distance or optimal transport distance) computes the distance between two probability vectors[1]. Given two vectors $\mathbf{a} \in \mathbb{R}_+^n$, $\mathbf{b} \in \mathbb{R}_+^m$ and cost matrix $\mathbf{C} \in \mathbb{R}_+^{n\times m}$, the Wasserstein distance between $\mathbf{a}$, $\mathbf{b}$ is shown by $W(\mathbf{a}, \mathbf{b})$ and minimizes $\langle \mathbf{C}, \mathbf{T}\rangle$ where $\langle ., .\rangle$ indicates the Frobenius inner product and $\mathbf{C} \in \mathbb{R}_+^{n\times m}$ is a symmetric input cost matrix where $\mathbf{C}(i,j)$ represents the cost of moving $\mathbf{a}[i]$ to $\mathbf{b}[j]$. $\mathbf{T} \in U(\mathbf{a}, \mathbf{b})$ where $\mathbf{T}$ is an optimal transport solution between probability vectors $\mathbf{a}$ and $\mathbf{b}$ and $U(\mathbf{a}, \mathbf{b}) = \{\mathbf{T} \in \mathbb{R}_+^{n\times m} | \mathbf{T}\mathbf{1}_m = \mathbf{a}, \mathbf{T}^T\mathbf{1}_n = \mathbf{b}\}$ is a set of all non-negative $n \times m$ matrices with row and column sums $\mathbf{a}$, $\mathbf{b}$ respectively. $\mathbf{1}_m$ represents m dimensional vector of ones. The aforementioned problem has complexity $O(n^3)$ (assuming $m = n$) (Peyré, Cuturi et al. 2019). However, computing this distance metric comes with a heavy computational price (Pele and Werman 2009). In order to reduce the complexity of computation, Cuturi et.al. (Cuturi 2013) propose an entropy regularized optimal transport problem between vectors $\mathbf{a}$, $\mathbf{b}$:

$$W_V(\mathbf{a}, \mathbf{b}) = \underset{\mathbf{T}\in U(\mathbf{a},\mathbf{b})}{\text{minimize}} \quad \langle \mathbf{C}, \mathbf{T}\rangle - \frac{1}{\rho}E(\mathbf{T}), \quad (2)$$

where $E(\mathbf{T})$ is an entropy function and ρ is a regularization parameter. When $\rho \geq 0$, the solution of (2) is called the *Sinkhorn divergence* (*a.k.a.* entropy regularized Wasserstein distance) between probability vectors $\mathbf{a}$ and $\mathbf{b}$. Eq. (2) is a strictly convex problem with a unique solution and can be computed with vectors $\mathbf{u} \in \mathbb{R}_+^n, \mathbf{v} \in \mathbb{R}_+^m$ such that $\text{diag}(\mathbf{u})\mathbf{K}\,\text{diag}(\mathbf{v}) \in U(\mathbf{a}, \mathbf{b})$. Here, $\mathbf{K} = \exp(-\rho\mathbf{C}) \in \mathbb{R}_+^{n\times m}$. Finding the optimal $\mathbf{u}$ and $\mathbf{v}$ can be computed via the Sinkhorn's algorithm (Sinkhorn and Knopp 1967).

Wasserstein Dictionary Learning. There are several techniques for minimizing Wasserstein loss for dictionary learning problem (Sandler and Lindenbaum 2009; Rolet, Cuturi, and Peyré 2016; Qian et al. 2016; Schmitz et al. 2018;

[1] Vector $\mathbf{a}$ is a probability vector if $\|\mathbf{a}\|_1 = 1$ and all elements in $\mathbf{a}$ are non-negative.

Varol, Nejatbakhsh, and McGrory 2019; Xu 2020). Sandler et.al. (Sandler and Lindenbaum 2009) introduces the first non-negative matrix factorization problem that minimizes Wasserstein loss by proposing a linear programming problem, however, their method needs heavy computation. Cuturi (Rolet, Cuturi, and Peyré 2016) proposed a Wasserstein dictionary learning problem based on entropy regularization. (Qian et al. 2016) proposes a similar method by exploiting knowledge in both data manifold and features correlation. Xu (Xu 2020) introduces a nonlinear matrix factorization approach for graphs that considers topological structures. Unfortunately, these approaches cannot be directly generalized to tensor inputs due to the challenges mentioned in Section 1.

3 SWIFT Framework

We define and solve optimal transport problem for tensor input by proposing Scalable WassersteIn FacTorization (SWIFT) for sparse nonnegative tensors. First we define the input and output for SWIFT. Our proposed method requires the N-th order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ and N cost matrices $\mathbf{C}_n \in \mathbb{R}_+^{I_n \times I_n}$ $(n = 1, ..., N)$ capturing the relations between dimensions along each tensor mode as the **input** to SWIFT. Here, $\mathbf{C}_n$ is a cost matrix for mode n and can be computed directly from the tensor input, or derived from external knowledge. It can also be an identity matrix, meaning that the correlation among features are ignored if the cost matrix is not available. SWIFT is based on CP decomposition and computes N non-negative factor matrices $\mathbf{A}_n \in \mathbb{R}_+^{I_n \times R}$ $(n = 1, ..., N)$ as the **output**. These factor matrices can then be used for downstream tasks, such as clustering and classification.

3.1 Wasserstein Distance for Tensors

Definition 1 Wasserstein Matrix Distance: *Given a cost matrix $\mathbf{C} \in \mathbb{R}_+^{M \times M}$, the Wasserstein distance between two matrices $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_P] \in \mathbb{R}_+^{M \times P}$ and $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_P] \in \mathbb{R}_+^{M \times P}$ is denoted by $W_M(\mathbf{A}, \mathbf{B})$, and given by:*

$$\begin{aligned} W_M(\mathbf{A}, \mathbf{B}) &= \sum_{p=1}^{P} W_V(\mathbf{a}_p, \mathbf{b}_p) \\ &= \underset{\mathbf{T}_p \in U(\mathbf{a}_p, \mathbf{b}_p)}{\text{minimize}} \sum_{p=1}^{P} \langle \mathbf{C}, \mathbf{T}_p \rangle - \frac{1}{\rho} E(\mathbf{T}_p) \\ &= \underset{\overline{\mathbf{T}} \in U(\mathbf{A}, \mathbf{B})}{\text{minimize}} \langle \overline{\mathbf{C}}, \overline{\mathbf{T}} \rangle - \frac{1}{\rho} E(\overline{\mathbf{T}}), \end{aligned} \tag{3}$$

Note that sum of the minimization equals minimization of sums since each $\mathbf{T}_p$ is independent of others. Here, ρ is a regularization parameter, $\overline{\mathbf{C}} = \underbrace{[\mathbf{C}, ..., \mathbf{C}]}_{P \text{ times}}$ and $\overline{\mathbf{T}} = [\mathbf{T}_1, ...\mathbf{T}_p, ..., \mathbf{T}_P]$ are concatenations of the cost matrices and the transport matrices for the P optimal transport problems, respectively. Note that $U(\mathbf{A}, \mathbf{B})$ is the feasible region of the transport matrix $\overline{\mathbf{T}}$ and is given by:

$$\begin{aligned} U(\mathbf{A}, \mathbf{B}) &= \left\{ \overline{\mathbf{T}} \in \mathbb{R}_+^{M \times MP} \mid \mathbf{T}_p \mathbf{1}_M = \mathbf{a}_p, \mathbf{T}_p^T \mathbf{1}_M = \mathbf{b}_p \quad \forall p \right\} \\ &= \left\{ \overline{\mathbf{T}} \in \mathbb{R}_+^{M \times MP} \mid \Delta(\overline{\mathbf{T}}) = \mathbf{A}, \Psi(\overline{\mathbf{T}}) = \mathbf{B} \right\} \end{aligned} \tag{4}$$

where $\Delta(\overline{\mathbf{T}}) = [\mathbf{T}_1 \mathbf{1}_M, ..., \mathbf{T}_P \mathbf{1}_M] = \overline{\mathbf{T}}(\mathbf{I}_P \otimes \mathbf{1}_M)$, $\Psi(\overline{\mathbf{T}}) = [\mathbf{T}_1^T \mathbf{1}_M, ..., \mathbf{T}_P^T \mathbf{1}_M]$ and $\mathbf{1}_M$ is a one vector with length M.

Definition 2 Wasserstein Tensor Distance: *The Wasserstein distance between N-th order tensor $\mathcal{X} \in \mathbb{R}_+^{I_1 \times \ldots \times I_N}$ and its reconstruction $\hat{\mathcal{X}} \in \mathbb{R}_+^{I_1 \times \ldots \times I_N}$ is denoted by $W_T(\hat{\mathcal{X}}, \mathcal{X})$:*

$$\begin{aligned} W_T(\hat{\mathcal{X}}, \mathcal{X}) &= \sum_{n=1}^{N} W_M\left(\widehat{\mathbf{X}}_{(n)}, \mathbf{X}_{(n)}\right) \\ &\equiv \sum_{n=1}^{N} \left\{ \underset{\overline{\mathbf{T}}_n \in U(\widehat{\mathbf{X}}_{(n)}, \mathbf{X}_{(n)})}{\text{minimize}} \langle \overline{\mathbf{C}}_n, \overline{\mathbf{T}}_n \rangle - \frac{1}{\rho} E(\overline{\mathbf{T}}_n) \right\}, \end{aligned} \tag{5}$$

where $\overline{\mathbf{C}}_n = [\mathbf{C}_n, \mathbf{C}_n, ..., \mathbf{C}_n] \in \mathbb{R}_+^{I_n \times I_n I_{(-n)}}$ is obtained by repeating the cost matrix of the n-th mode for $I_{(-n)}$ times and horizontally concatenating them. $\overline{\mathbf{T}}_n = [\mathbf{T}_{n1}, ..., \mathbf{T}_{nj}, ..., \mathbf{T}_{nI_{(-n)}}] \in \mathbb{R}_+^{I_n \times I_n I_{(-n)}}$ and $\mathbf{T}_{nj} \in \mathbb{R}_+^{I_n \times I_n}$ is the transport matrix between the columns $\widehat{\mathbf{X}}_{(n)}(:, j) \in \mathbb{R}_+^{I_n}$ and $\mathbf{X}_{(n)}(:, j) \in \mathbb{R}_+^{I_n}$.

Note that $\overline{\mathbf{C}}$ and $\overline{\mathbf{C}}_n$ are for notation convenience and we do not keep multiple copies of $\mathbf{C}$ and $\mathbf{C}_n$ in implementation.

Proposition 1 *The Wasserstein distance between tensors $\mathcal{X}$ and $\mathcal{Y}$ denoted by $W_T(\mathcal{X}, \mathcal{Y})$ is a valid distance and satisfies the metric axioms as follows:*

1. *Positivity: $W_T(\mathcal{X}, \mathcal{Y}) \geq 0$*
2. *Symmetry: $W_T(\mathcal{X}, \mathcal{Y}) = W_T(\mathcal{Y}, \mathcal{X})$*
3. *Triangle Inequality: $\forall \mathcal{X}, \mathcal{Y}, \mathcal{Z} \quad W_T(\mathcal{X}, \mathcal{Y}) \leq W_T(\mathcal{X}, \mathcal{Z}) + W_T(\mathcal{Z}, \mathcal{Y})$*

We provide the proof in the appendix.

3.2 Wasserstein Tensor Factorization

Given an input tensor $\mathcal{X}$, SWIFT aims to find the low-rank approximation $\widehat{\mathcal{X}}$ such that their Wasserstein distance in (5) is minimized. Formally, we solve for $\widehat{\mathcal{X}}$ by minimizing $W_T(\widehat{\mathcal{X}}, \mathcal{X})$, where $\widehat{\mathcal{X}} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$ is the CP factorization of $\mathcal{X}$. Together with Definitions 1 and 2, we have the following optimization problem:

$$\begin{aligned} \underset{\{\mathbf{A}_n \geq 0, \overline{\mathbf{T}}_n\}_{n=1}^N}{\text{minimize}} \quad & \sum_{n=1}^{N} \left(\langle \overline{\mathbf{C}}_n, \overline{\mathbf{T}}_n \rangle - \frac{1}{\rho} E(\overline{\mathbf{T}}_n) \right) \\ \text{subject to} \quad & \widehat{\mathcal{X}} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!] \\ & \overline{\mathbf{T}}_n \in U(\widehat{\mathbf{X}}_{(n)}, \mathbf{X}_{(n)}),\ n = 1, \ldots, N \end{aligned} \tag{6}$$

where the first constraint enforces a low-rank CP approximation, the second one ensures that the transport matrices are inside the feasible region. We also interested in imposing non-negativity constraint on the CP factor matrices for

both well-definedness of the optimal transport problem and interpretability of the factor matrices. Similar to prior works on vector-based, and matrix-based Wasserstein distance minimization problems (Frogner et al. 2015; Qian et al. 2016), in order to handle non-probability inputs, we convert the second hard constraint in (6) to soft regularizations by Lagrangian method using the generalized KL-divergence. Together with the fact that $\widehat{\mathbf{X}}_{(n)} = \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T$ for CP factorization, we convert (6) into the following objective function:

$$\underset{\{\mathbf{A}_n \geq 0, \overline{\mathbf{T}}_n\}_{n=1}^N}{\text{minimize}} \sum_{n=1}^{N} \Bigg(\underbrace{\langle \overline{\mathbf{C}}_n, \overline{\mathbf{T}}_n \rangle - \frac{1}{\rho} E(\overline{\mathbf{T}}_n)}_{\text{Part } P_1} + \lambda \Big(\underbrace{KL(\Delta(\overline{\mathbf{T}}_n) || \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T)}_{\text{Part } P_2} + \underbrace{KL(\Psi(\overline{\mathbf{T}}_n) || \mathbf{X}_{(n)})}_{\text{Part } P_3} \Big) \Bigg) \tag{7}$$

where $\Delta(\overline{\mathbf{T}}_n) = [\mathbf{T}_{n1}\mathbf{1}, ..., \mathbf{T}_{nj}\mathbf{1}, ..., \mathbf{T}_{nI_{(-n)}}\mathbf{1}]$, $\Psi(\overline{\mathbf{T}}_n) = [\mathbf{T}_{n1}^T\mathbf{1}, ..., \mathbf{T}_{nj}^T\mathbf{1}, ..., \mathbf{T}_{nI_{(-n)}}^T\mathbf{1}]$, and λ is the weighting parameter for generalized KL-divergence regularization.

We use the alternating minimization to solve (7). SWIFT iteratively updates: 1) **Optimal Transports Problems.** For each mode-n matricization, SWIFT computes at most a set of $I_{(-n)}$ optimal transport problems. By exploiting the sparsity structure and avoiding explicitly computing transport matrices we can significantly reduce the computation cost. 2) **Factor Matrices.** The CP factor matrices are involved inside the Khatri-Rao product and needs excessive amount of computation. However, by rearranging the terms involved in (7), we can efficiently update each factor matrix. Next, we provide efficient ways to update optimal transport and factor matrices in more details.

3.3 Solution for Optimal Transport Problems

For mode-n, $\overline{\mathbf{T}}_n = [\mathbf{T}_{n1}, ..., \mathbf{T}_{nj}, ..., \mathbf{T}_{nI_{(-n)}}] \in \mathbb{R}_+^{I_n \times I_n I_{(-n)}}$ includes a set of $I_{(-n)}$ different optimal transport problems. The optimal solution of j-th optimal transport problem for mode n is $\mathbf{T}_{nj}^* = \text{diag}(\mathbf{u}_j)\mathbf{K}_n\text{diag}(\mathbf{v}_j)$, where $\mathbf{K}_n = e^{(-\rho \mathbf{C}_n - 1)} \in \mathbb{R}_+^{I_n \times I_n}, \mathbf{u}_j, \mathbf{v}_j \in \mathbb{R}_+^{I_n}$. This requires computing $I_{(-n)}$ transport matrices with size $I_n \times I_n$ in (7), which is extremely time-consuming. To reduce the amount of computation, SWIFT proposes the following three strategies:

1) Never explicitly computing transport matrices ($\overline{\mathbf{T}}_n^*$): Although we are minimizing (7) with respect to $\overline{\mathbf{T}}_n$, we never explicitly compute $\overline{\mathbf{T}}_n^*$. In stead of directly computing the optimal transport matrices $\overline{\mathbf{T}}_n^*$, we make use of the constraint $\mathbf{T}_{nj}^*\mathbf{1} = \text{diag}(\mathbf{u}_j)\mathbf{K}_n\mathbf{v}_j = \mathbf{u}_j * (\mathbf{K}_n\mathbf{v}_j)$ where $*$ denotes element-wise product. As a result, the following proposition effectively updates objective function 7.

Proposition 2 *$\Delta(\overline{\mathbf{T}}_n) = [\mathbf{T}_{n1}\mathbf{1}, ..., \mathbf{T}_{nj}\mathbf{1}, ..., \mathbf{T}_{nI_{(-n)}}\mathbf{1}] = \mathbf{U}_n * (\mathbf{K}_n\mathbf{V}_n)$ minimizes (7) where $\mathbf{U}_n = (\widehat{\mathbf{X}}_{(n)})^{\Phi} \oslash (\mathbf{K}_n(\mathbf{X}_{(n)} \oslash (\mathbf{K}_n^T\mathbf{U}_n))^{\Phi})^{\Phi}$, $\mathbf{V}_n = (\mathbf{X}_{(n)} \oslash (\mathbf{K}_n^T\mathbf{U}_n))^{\Phi}$, $\Phi = \frac{\lambda\rho}{\lambda\rho+1}$, and $\oslash$ indicates element-wise division. See Section appendix for proof.*

2) Exploiting Sparsity Structure in $\mathbf{X}_{(n)} \in \mathbb{R}_+^{I_n \times I_{(-n)}}$: We observe that there are many columns with all zero elements in $\mathbf{X}_{(n)}$ due to the sparsity structure in the input data. There is no need to compute transport matrix for those zero columns, therefore, we can easily drop zero value columns in $\mathbf{X}_{(n)}$ and its corresponding columns in $\mathbf{U}_n, \mathbf{V}_n$, and $\widehat{\mathbf{X}}_{(n)}$ from our computations. We use NNZ_n to denote the number of non-zero columns in $\mathbf{X}_{(n)}$. By utilizing this observation, we reduce the number of times to solve the optimal transport problems from $I_{(-n)}$ to NNZ_n, where we usually have $NNZ_n \ll I_{(-n)}$ for sparse input.

3) Parallelization of the optimal transport computation: The NNZ_n optimal transport problems for each factor matrix $\mathbf{X}_{(n)}$ can be solved independently. Therefore, parallelization on multiple processes is straightforward for SWIFT.

3.4 Solution for Factor Matrices

All the factor matrices are involved in Part P_2 of Objective (7) and present in N different KL-divergence terms. The objective function with respect to factor matrix $\mathbf{A}_n$ for mode n is:

$$\underset{\mathbf{A}_n \geq 0}{\text{minimize}} \quad \sum_{i=1}^{N} KL\Big(\Delta(\overline{\mathbf{T}}_i) \;||\; \mathbf{A}_i(\mathbf{A}_{\odot}^{(-i)})^T\Big) \tag{8}$$

where $\Delta(\overline{\mathbf{T}}_i) \in \mathbb{R}_+^{I_i \times I_{(-i)}}$, $\mathbf{A}_i \in \mathbb{R}_+^{I_i \times R}$, $\mathbf{A}_{\odot}^{(-i)} \in \mathbb{R}_+^{I_{(-i)} \times R}$. Updating factor matrix $\mathbf{A}_n$ in (8) is expensive due to varying positions of $\mathbf{A}_n$ in the N KL-divergence terms. Specifically, $\mathbf{A}_n$ is involved in the Khatri-Rao product $\mathbf{A}_{\odot}^{(-i)}$, as defined in (1), for every $i \neq n$. On the other hand, when $i = n$, $\mathbf{A}_n$ is not involved in $\mathbf{A}_{\odot}^{(-i)}$.

Efficient rearranging operations. In order to solve (8) efficiently, we introduce operator Π, which performs a sequence of reshape, permute and another reshape operations, such that, when applied to the right-hand side of (8), $\mathbf{A}_n$ is no longer involved inside the Khatri-Rao product for all $i \neq n$. Formally,

$$\Pi(\mathbf{A}_i(\mathbf{A}_{\odot}^{(-i)})^T, n) = \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T \in \mathbb{R}_+^{I_n \times I_{(-n)}} \;\; \forall\, i \neq n. \tag{9}$$

To maintain equivalence to (8), we apply the same operation to the left-hand side of (8), which leads us to the following formulation:

$$\underset{\mathbf{A}_n \geq 0}{\text{minimize}} \quad KL\left(\begin{bmatrix} \Pi(\Delta(\overline{\mathbf{T}}_1), n) \\ \vdots \\ \Pi(\Delta(\overline{\mathbf{T}}_i), n) \\ \vdots \\ \Pi(\Delta(\overline{\mathbf{T}}_N), n) \end{bmatrix} \Bigg|\Bigg| \begin{bmatrix} \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T \\ \vdots \\ \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T \\ \vdots \\ \mathbf{A}_n(\mathbf{A}_{\odot}^{(-n)})^T \end{bmatrix} \right) \tag{10}$$

Where $\Pi(\Delta(\overline{\mathbf{T}}_i), n) \in \mathbb{R}_+^{I_n \times I_{(-n)}}$ for all i. Due to the fact that KL-divergence is computed point-wisely, the above formula is equivalent to (8), with the major difference that $\mathbf{A}_n$ is at the same position in every KL-divergence term in (10), and is no longer involved inside the Khatri-Rao product terms; therefore, it can be much more efficiently updated via multiplicative update rules (Lee and Seung 2001). More details

regarding operator Π, and multiplicative update rules for $\mathbf{A}_n$ are provided in the appendix.

In every iteration of SWIFT, we first update N different optimal transport problems and then update N factor matrices. Algorithm 1 summarizes the optimization procedure in SWIFT.

Proposition 3 *SWIFT is based on Block Coordinate Descent (BCD) algorithm and guarantees convergence to a stationary point. See detailed proofs in the appendix.*

Details regarding complexity analysis are provided in appendix.

Algorithm 1: SWIFT

Input : $\mathcal{X} \in \mathbb{R}_+^{I_1 \times I_2 \times \ldots \times I_N}$, $\mathbf{C}_n$, NNZ_n $n = 1, ..., N$, target rank R, λ, and ρ

Output: $\mathbf{A}_n \in \mathbb{R}_+^{I_n \times R}$ $n = 1, ..., N$

$\mathbf{K}_n = e^{(-\rho \mathbf{C}_n - 1)}$ $n = 1, ..., N$;
Initialize $\mathbf{A}_n$ $n = 1, ..., N$ randomly;
while *stopping criterion is not met* **do**
  for *n = 1,...,N* **do**
    // Optimal Transport Update (Section 3.3);
    $\Phi = \frac{\lambda\rho}{\lambda\rho+1}$;
    $\mathbf{U}_n = ones(I_n, NNZ_n) \oslash I_n$;
    for *s=1,...,Sinkhorn Iteration* **do**
      $\mathbf{U}_n = (\widehat{\mathbf{X}}_{(n)})^{\Phi} \oslash \left(\mathbf{K}_n \left(\mathbf{X}_{(n)} \oslash (\mathbf{K}_n^T \mathbf{U}_n)\right)^{\Phi}\right)^{\Phi}$;
    end
    $\mathbf{V}_n = \left(\mathbf{X}_{(n)} \oslash (\mathbf{K}_n^T \mathbf{U}_n)\right)^{\Phi}$;
    $\Delta(\overline{\mathbf{T}}_n) = \mathbf{U}_n * (\mathbf{K}_n \mathbf{V}_n)$;
  end
  for n=1, ..., N **do**
    // Factor Matrix Update (Section 3.4);
    Update $\mathbf{A}_n$ based on (10);
  end
end

4 Experimental Results

4.1 Experimental Setup

Datasets and Evaluation Metrics

1) **BBC News** (Greene and Cunningham 2006) is a publicly available dataset from the BBC News Agency for text classification task. A third-order count tensor is constructed with the size of 400 articles by 100 words by 100 words. $\mathcal{X}(i, j, k)$ is the number of co-occurrences of the j-th and the k-th words in every sentence of the i-th article. We use the pair-wise cosine distance as the word-by-word and article-by-article cost matrices with details provided in the appendix. The downstream task is to predict the category (from business, entertainment, politics, sport or tech) of each article and we use *accuracy* as the evaluation metric.

2) **Sutter** is a dataset collected from a large real-world health provider network containing the electronic health records (EHRs) of patients. A third-order binary tensor is constructed with the size of 1000 patients by 100 diagnoses by 100 medications. The downstream task is to predict the onset of heart failure (HF) for the patients (200 out of the 1000 patients are diagnosed with HF) and use PR-AUC (Area Under the Precision-Recall Curve) to evaluate the HF prediction task.

We chose these two datasets because of different data types (count in BBC, binary in Sutter). Note that **the cost matrices are derived from the original input by cosine similarity without any additional external knowledge**. Hence the comparisons are fair since the input are the same.

Baselines We compare the performance of SWIFT with different tensor factorization methods with different loss functions and their variants:

- The first loss function minimizes sum square loss and has 4 variants: 1) **CP-ALS** (Bader and Kolda 2007); 2) **CP-NMU** (Bader and Kolda 2007); 3) **Supervised CP** (Kim et al. 2017); and 4) **Similarity based CP** (Kim et al. 2017). The first one is unconstrained, the second one incorporates non-negativity constraint, the third one utilizes label information, and the last one uses similarity information among features (similar to SWIFT).
- The second loss function is Gamma loss (**CP-Continuous** (Hong, Kolda, and Duersch 2020)) which is the start of the art method (SOTA) for non-negative continuous tensor.
- The third loss function is Log-loss (**CP-Binary** (Hong, Kolda, and Duersch 2020)) which is SOTA binary tensor factorization by fitting Bernoulli distribution.
- The fourth loss function is Kullback-Leibler Loss (**CP-APR** (Chi and Kolda 2012)) which fits Poisson distribution on the input data and is suitable for count data.

4.2 Classification Performance of SWIFT

To evaluate low-rank factor matrices, we utilize the downstream prediction tasks as a proxy to assess the performance of SWIFT and the baselines, similar to the existing works (Ho, Ghosh, and Sun 2014; Yin et al. 2019; Afshar et al. 2020; Yin et al. 2020a). We performed 5-fold cross validation and split the data into training, validation, and test sets by a ratio of 3:1:1.

Outperforming various tensor factorizations: Table 1 summarizes the classification performance using the factor matrices obtained by SWIFT and the baselines with varying target rank ($R \in \{5, 10, 20, 30, 40\}$). We report the mean and standard deviation of *accuracy* for BBC News, and that of *PR-AUC Score* for Sutter dataset over the test set. For BBC News, SWIFT outperforms all baselines for different target ranks with relative improvement ranging from 1.69% to 9.65%. For Sutter, SWIFT significantly outperforms all baselines for all values of R with relative improvement ranging from 5.10% to 11.31%.

Outperforming various classifiers: We further compare the performance of SWIFT against widely-adopted classifiers, including Lasso Logistic Regression, Random Forest, Multi-Layer Perceptron, and K-Nearest Neighbor, using raw data. The input for BBC and Sutter to these classifiers are obtained by matricizing the input tensors along the article

Table 1: The first part reports the average and standard deviation of *accuracy* on the test set as for different value of R on BBC NEWS data. The second part depicts the average and standard deviation of *PR-AUC Score* on test data for Sutter dataset. Both experiments are based on five-fold cross validation. We used Lasso Logistic Regression as a classifier.

		R=5	R=10	R=20	R=30	R=40
BBC News Dataset	CP-ALS	.521 ± .033	.571 ± .072	.675 ± .063	.671 ± .028	.671 ± .040
	CP-NMU	.484 ± .039	.493 ± .048	.581 ± .064	.600 ± .050	.650 ± .031
	Supervised CP	.506 ± .051	.625 ± .073	.631 ± .050	.665 ± .024	.662± .012
	Similarity Based CP	.518 ± .032	.648 ± .043	.638 ± .021	.662 ± .034	.673 ± .043
	CP-Continuous	.403 ± .051	.481 ± .056	.528 ± .022	.559 ± .024	.543 ± .043
	CP-Binary	.746 ± .058	.743 ± .027	.737 ± .008	.756 ± .062	.743 ± .044
	CP-APR	.675 ± .059	.768 ± .033	.753 ± .035	.743 ± .033	.746 ± .043
	SWIFT	**.759 ± .013**	**.781 ± .013**	**.803 ± .010**	**.815 ± .005**	**.818 ± .022**
Sutter Data	CP-ALS	.327 ± .072	.333 ± .064	.311 ± .068	.306 ± .065	.332 ± .098
	CP-NMU	.300 ± .054	.294 ± .064	.325 ± .085	.344 ± .068	.302 ± .071
	Supervised CP	.301 ± .044	.305 ± .036	.309 ± .054	.291 ± .037	.293 ± .051
	Similarity Based CP	.304 ± .042	.315 ± .041	.319 ± .063	.296 ± .041	.303 ± .032
	CP-Continuous	.252 ± .059	.237 ± .043	.263 ± .065	.244 ± .053	.256 ± .077
	CP-Binary	.301 ± .061	.325 ± .079	.328 ± .080	.267 ± .074	.296 ± .063
	CP-APR	.305 ± .075	.301 ± .068	.290 ± .052	.313 ± .082	.304 ± .086
	SWIFT	**.364 ± .063**	**.350 ± .031**	**.350 ± .040**	**.369 ± .066**	**.374 ± .044**

Table 2: Average and standard deviation of accuracy on BBC NEWS and PR-AUC score on Sutter data sets by performing Lasso LR, RF, MLP, and KNN on raw data sets.

	Accuracy on BBC	PR-AUC on Sutter
Lasso LR	.728 ± .013	.308 ± .033
RF	.6281 ± .049	.318 ± .083
MLP	.690 ± .052	.305 ± .054
KNN	.5956 ± .067	.259 ± .067
SWIFT (R=5)	.759 ± .013	.364 ± .063
SWIFT (R=40)	**.818 ± .020**	**.374 ± .044**

mode and the patient mode, respectively. Table 2 summarizes the results, and it clearly shows that SWIFT using Lasso LR classifier even with R=5 outperforms all the other classifiers compared.

4.3 Classification Performance on Noisy Data

To measure the performance of SWIFT against noisy input, we inject noise to the raw input tensor to construct the noisy input tensor. For the binary tensor input, we add Bernoulli noise. Given the noise level (p), we randomly choose zero elements, such that the total number of selected zero elements equals to the number of non-zero elements. Then, we flip the selected zero elements to one with probability p. We follow the similar procedure for the count tensor input, except that we add the noise by flipping the selected zero value to a count value with probability p, and the added noise value is selected uniformly at random between 1 and maximum value in the tensor input.

Performance on BBC data with Noise: Figure 1 presents the average and standard deviation of categorizing the articles on the test data with respect to different levels of noise ($p \in \{0.05, 0.1, 0.15, 0.20, 0.25, 0.30\}$). For all levels of noise, SWIFT outperforms all baselines by improving accuracy up to 15% over the best baseline especially for medium and high noise levels. Similar results are also achieved on Sutter dataset, as shown in the appendix.

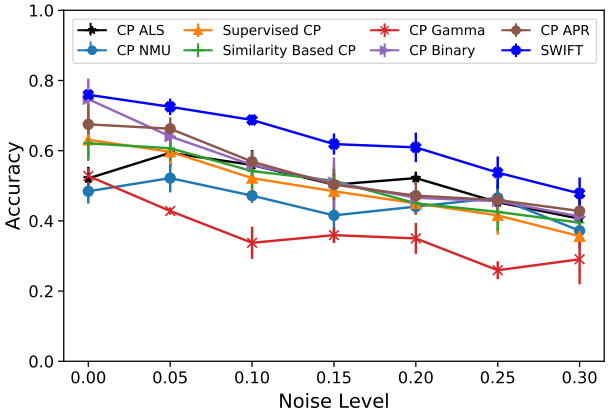

Figure 1: The average and standard deviation of accuracy of different baselines as a function of the noise level on BBC NEWS. SWIFT outperforms other baselines by improving accuracy up to 15%.

4.4 Scalability of SWIFT

In this section, we assess the scalability of SWIFT in comparison to the other 7 CP algorithms introduced earlier. Figure 2 depicts the average and standard deviation of running time of one iteration in seconds. In this experiment, we set $R = 40$. In order to have fair comparisons, we switched off the par-

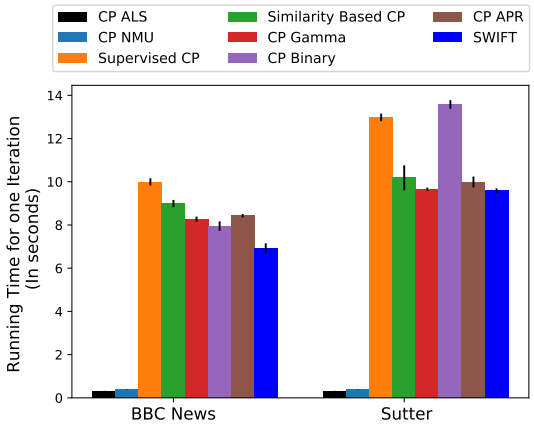

Figure 2: The running time of one iteration in seconds on BBC News and Sutter with $R = 40$.

allelization of the optimal transport computation for SWIFT since none of the baselines can be run with parallelization. As shown in Figure 2, SWIFT is as scalable as other baselines, suggesting that the great improvement of the performance are achieved without sacrificing the efficiency. We further compare against a direct implementation of Wasserstein tensor factorization. Results are summarized in the appendix, which show that SWIFT is up to $293x$ faster than the direct implementation. This verifies that the strategies introduced in Section 3 are the keys to SWIFT being scalable and performant.

4.5 Interpretability of SWIFT

To demonstrate the interpretability of results produced by SWIFT we perform computational phenotyping using Sutter dataset.

Computational phenotyping is a fundamental task in healthcare. It refers to extracting meaningful and interpretable medical concepts (patient clusters) from noisy electronic health records (EHRs) (Richesson et al. 2016). Tensor factorization techniques are powerful tools for extracting meaningful phenotyping (Ho, Ghosh, and Sun 2014; Yin et al. 2018, 2020b; Zhao et al. 2019; Afshar et al. 2020; Yin et al. 2020a). Here we extract heart failure (HF) phenotypes using Sutter dataset. We use the same tensor as we described in the previous section and run SWIFT by selecting $R = 40$ since it achieves the highest PR-AUC in comparison to other values of R. $\mathbf{A}_2(:, r), \mathbf{A}_3(:, r)$ represent the membership value of diagnosis and medication features in r-th phenotype.

Results: Table 3 lists three examples of the discovered phenotypes. The weight next to phenotype index indicates the lasso logistic regression coefficient for heart failure prediction (i.e.,21.93, 19.58 and -16.22 in Table 3). "Dx" indicates diagnoses and "Rx" represents medications. All phenotypes are clinically meaningful, endorsed and annotated by a medical expert. The first phenotype is about Atrial Fibrillation which captures patients with hypertension and cardiac dysrhythmias that are associated high blood pressure medications. Cardiometabolic disease is another phenotype that captures

Table 3: Three phenotypes examples learned on Sutter dataset with SWIFT. The weight value for each phenotype represents the lasso logistic regression coefficient for the heart failure prediction task. "Dx" represents for diagnoses and "Rx" indicates for medications. All phenotypes are considered clinically meaningful by a clinical expert.

Atrial Fibrillation (Weight= 21.93)
Dx-Essential hypertension [98.]
Dx-Disorders of lipid metabolism [53.]
Dx-Cardiac dysrhythmias [106.]
Rx-Calcium Channel Blockers
Rx-Alpha-Beta Blockers
Rx-Angiotensin II Receptor Antagonists
Cardiometablic Disease (Weight= 19.58)
Dx-Diabetes mellitus without complication [49.]
Dx-Essential hypertension [98.]
Dx-Disorders of lipid metabolism [53.]
Rx-Diagnostic Tests
Rx-Biguanides
Rx-Diabetic Supplies
Mental Disorder (Weight= -16.22)
Dx-Anxiety disorders [651]
Dx-Menopausal disorders [173.]
Dx-Depressive disorders [6572]
Rx-Benzodiazepines
Rx-Selective Serotonin Reuptake Inhibitors (SSRIs)
Rx-Serotonin Modulators

diabetes patients with hypertension. These two phenotypes have high positive weights (21.93 and 19.58 respectively) for predicting HF diagnosis. The third phenotype is depressive and menopausal disorders, Serotonin and Benzodiazepines appeared in this phenotype are medications that are commonly prescribed to patients with these conditions in clinical practice. This phenotype has a negative association with HF (weight = -16.22). The remaining phenotypes positively associated with HF are listed in the appendix.

5 Conclusion

In this paper, we define the Wasserstein distance between two tensors and propose SWIFT, an effective tensor factorization model based on the defined tensor Wasserstein distance. To efficiently learn the Wasserstein tensor factorization, we introduce introduce several techniques, including exploitation of the sparsity structure of the input tensor, efficient rearrangement by utilizing tensor properties, and parallelization of the optimal transport computation. Experimental results depict that SWIFT consistently outperforms baselines in downstream classification tasks for both binary and count tensor inputs. In the presence of noise, SWIFT also outperforms baselines by a large margin.

Acknowledgments

This work is in part supported by National Science Foundation award SCH-2014438, IIS-1418511, CCF-1533768,

IIS-1838042, the National Institute of Health award NIH R01 1R01NS107291-01 and R56HL138415.

References

Acar, E.; Kolda, T. G.; and Dunlavy, D. M. 2011. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422* .

Afshar, A.; Perros, I.; Papalexakis, E. E.; Searles, E.; Ho, J.; and Sun, J. 2018. COPA: Constrained PARAFAC2 for sparse & large datasets. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 793–802.

Afshar, A.; Perros, I.; Park, H.; deFilippi, C.; Yan, X.; Stewart, W.; Ho, J.; and Sun, J. 2020. TASTE: temporal and static tensor factorization for phenotyping electronic health records. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, 193–203.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein Generative Adversarial Networks. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 214–223. International Convention Centre, Sydney, Australia: PMLR. URL http://proceedings.mlr.press/v70/arjovsky17a.html.

Bader, B. W.; and Kolda, T. G. 2007. Efficient MATLAB computations with sparse and factored tensors. *SIAM Journal on Scientific Computing* 30(1): 205–231.

Bahadori, M. T.; Yu, Q. R.; and Liu, Y. 2014. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Advances in neural information processing systems*, 3491–3499.

Chi, E. C.; and Kolda, T. G. 2012. On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications* 33(4): 1272–1299.

Cuturi, M. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, 2292–2300.

Fanaee-T, H.; and Gama, J. 2016. Event detection from traffic tensors: A hybrid model. *Neurocomputing* 203: 22–33.

Frogner, C.; Zhang, C.; Mobahi, H.; Araya, M.; and Poggio, T. A. 2015. Learning with a Wasserstein loss. In *Advances in Neural Information Processing Systems*, 2053–2061.

Greene, D.; and Cunningham, P. 2006. Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*, 377–384. ACM Press.

Gujral, E.; Pasricha, R.; and Papalexakis, E. 2020. Beyond Rank-1: Discovering Rich Community Structure in Multi-Aspect Graphs. In *Proceedings of The Web Conference 2020*, 452–462.

He, H.; Henderson, J.; and Ho, J. C. 2019. Distributed Tensor Decomposition for Large Scale Health Analytics. In *The World Wide Web Conference*, 659–669.

Henderson, J.; Ho, J. C.; Kho, A. N.; Denny, J. C.; Malin, B. A.; Sun, J.; and Ghosh, J. 2017. Granite: Diversified, sparse tensor factorization for electronic health record-based phenotyping. In *2017 IEEE international conference on healthcare informatics (ICHI)*, 214–223. IEEE.

Ho, J. C.; Ghosh, J.; and Sun, J. 2014. Marble: high-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 115–124.

Hong, D.; Kolda, T. G.; and Duersch, J. A. 2020. Generalized canonical polyadic tensor decomposition. *SIAM Review* 62(1): 133–163.

Kim, J.; He, Y.; and Park, H. 2014. Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework. *Journal of Global Optimization* 58(2): 285–319.

Kim, Y.; El-Kareh, R.; Sun, J.; Yu, H.; and Jiang, X. 2017. Discriminative and distinct phenotyping by constrained tensor factorization. *Scientific reports* 7(1): 1114.

Kolda, T. G.; and Bader, B. W. 2009. Tensor decompositions and applications. *SIAM Review* 51(3): 455–500.

Lee, D. D.; and Seung, H. S. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, 556–562.

Pele, O.; and Werman, M. 2009. Fast and robust earth mover's distances. In *2009 IEEE 12th International Conference on Computer Vision*, 460–467. IEEE.

Peyré, G.; Cuturi, M.; et al. 2019. Computational optimal transport. *Foundations and Trends® in Machine Learning* 11(5-6): 355–607.

Qian, W.; Hong, B.; Cai, D.; He, X.; Li, X.; et al. 2016. Non-Negative Matrix Factorization with Sinkhorn Distance. In *IJCAI*, 1960–1966.

Richesson, R. L.; Sun, J.; Pathak, J.; Kho, A. N.; and Denny, J. C. 2016. Clinical phenotyping in selected national networks: demonstrating the need for high-throughput, portable, and computational methods. *Artificial intelligence in medicine* 71: 57–61.

Rolet, A.; Cuturi, M.; and Peyré, G. 2016. Fast dictionary learning with a smoothed Wasserstein loss. In *Artificial Intelligence and Statistics*, 630–638.

Sandler, R.; and Lindenbaum, M. 2009. Nonnegative matrix factorization with earth mover's distance metric. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1873–1880. IEEE.

Schmitz, M. A.; Heitz, M.; Bonneel, N.; Ngole, F.; Coeurjolly, D.; Cuturi, M.; Peyré, G.; and Starck, J.-L. 2018. Wasserstein dictionary learning: Optimal transport-based unsupervised nonlinear dictionary learning. *SIAM Journal on Imaging Sciences* 11(1): 643–678.

Sinkhorn, R.; and Knopp, P. 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics* 21(2): 343–348.

Varol, E.; Nejatbakhsh, A.; and McGrory, C. 2019. Temporal Wasserstein non-negative matrix factorization for non-rigid motion segmentation and spatiotemporal deconvolution. *arXiv preprint arXiv:1912.03463* .

Wang, Y.; Chen, R.; Ghosh, J.; Denny, J. C.; Kho, A.; Chen, Y.; Malin, B. A.; and Sun, J. 2015. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1265–1274.

Xu, H. 2020. Gromov-Wasserstein Factorization Models for Graph Clustering. In *AAAI*, 6478–6485.

Yin, K.; Afshar, A.; Ho, J. C.; Cheung, W. K.; Zhang, C.; and Sun, J. 2020a. LogPar: Logistic PARAFAC2 Factorization for Temporal Binary Data with Missing Values. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1625–1635.

Yin, K.; Cheung, W. K.; Fung, B. C.; and Poon, J. 2020b. Learning Inter-Modal Correspondence and Phenotypes from Multi-Modal Electronic Health Records. *IEEE Transactions on Knowledge and Data Engineering* doi:10.1109/TKDE.2020.3038211.

Yin, K.; Cheung, W. K.; Liu, Y.; Fung, B. C.; and Poon, J. 2018. Joint Learning of Phenotypes and Diagnosis-Medication Correspondence via Hidden Interaction Tensor Factorization. In *IJCAI*, 3627–3633.

Yin, K.; Qian, D.; Cheung, W. K.; Fung, B. C.; and Poon, J. 2019. Learning phenotypes and dynamic patient representations via rnn regularized collective non-negative tensor factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1246–1253.

Zhao, J.; Zhang, Y.; Schlueter, D. J.; Wu, P.; Kerchberger, V. E.; Rosenbloom, S. T.; Wells, Q. S.; Feng, Q.; Denny, J. C.; and Wei, W.-Q. 2019. Detecting time-evolving phenotypic topics via tensor factorization on electronic health records: Cardiovascular disease case study. *Journal of biomedical informatics* 98: 103270.