ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230561

Multiplicative Sparse Tensor Factorization for Multi-View Multi-Task Learning

Xinyi Wang[1], **Lu Sun**[1], **Canh Hao Nguyen**[2] **and Hiroshi Mamitsuka**[2]

[1]ShanghaiTech University, {wangxy6, sunlu1}@shanghaitech.edu.cn
[2]Kyoto University, {canhhao, mami}@kuicr.kyoto-u.ac.jp

Abstract. Multi-View Multi-Task Learning (MVMTL) aims to make predictions on dual-heterogeneous data. Such data contains features from multiple views, and multiple tasks in the data are related with each other through common views. Existing MVMTL methods usually face two major challenges: 1) to save the predictive information from full-order interactions between views efficiently. 2) to learn a parsimonious and highly interpretable model such that the target is related to the features through a subset of interactions. To deal with the challenges, we propose a novel MVMTL method based on multiplicative sparse tensor factorization. For 1), we represent full-order interactions between views as a tensor, that enables to capture the complex correlations in dual-heterogeneous data by a concise model. For 2), we decompose the interaction tensor into a product of two components: one being shared with all tasks and the other being specific to individual tasks. Moreover, tensor factorization is applied to control the model complexity and learn a consensus latent representation shared by multiple tasks. Theoretical analysis reveals the equivalence between our method and a family of models with a joint but more general form of regularizers. Experiments on both synthetic and real-world datasets prove its effectiveness.

1 Introduction

Nowadays, many real-word applications involve data collected from multiple sources and views (*feature heterogeneity*). For example, in image annotation, each image has features from multiple views, such as color histogram, wavelet texture and edge direction. To utilize the information from different views, Multi-View Learning (MVL) is proposed to improve the performance by saving view consistency, i.e., the predictions of different views should be mutually consistent [7, 3, 23]. In addition, some practical applications exhibit the relationship between different tasks (*task heterogeneity*). For example, in protein function prediction, each protein may associated with multiple classes, such as transcription, cellular organization and metabolism. Rather than learning tasks independently, Multi-Task Learning (MTL) learns related tasks together to improve its generalization ability by saving task correlations, i.e., different tasks may share useful related information [29, 17, 32].

MVL and MTL enable to deal with feature and task heterogeneity, respectively, but neither of them can handle *dual-heterogeneity*, where multiple tasks are correlated through common views. A representative example is the classification of web pages. Each web page contains information from at least three views: image, text and hyperlink, and can be annotated with multiple topics, such as sports, news and education. To solve this kind of problem, Muli-View Multi-Task Learning (MVMTL) is proposed to capture both task correlations and view consistency, based on graph model [6], joint regularization [31], multilinear model [19], deep learning [33] and feature learning [24].

Experimental results [19, 24] on various real-world applications have shown that MVMTL has achieved much success on dual-heterogeneous data. However, existing MVMTL models still suffer from two main problems: 1) lack of ability to efficiently save predictive information from full-order interactions[1] among views and tasks. Previous methods usually require views to be consistent, which may be too restrictive in practice, since different views also provide complementary information. Take web page classification as an example, text often provides complementary information to image and hyperlink. As a result, considering full-order interactions between views might capture all the complex hidden relationships. However, it typically results in exponentially growth of model complexity. 2) incapable to learn parsimonious and highly interpretable models that removes irrelevant interactions, leading to degraded performance in practice. Even though full-order interactions easily blow up the dimensionality, only a few interactions may be useful for training the model, including a small subset of interactions shared among multiple tasks and the own specific interactions of eack task. For example, a topic of a web page is probably correlated with a subset of interactions among texts, images and hyperlink.

To overcome the aforementioned two problems, we propose a novel MVMTL method based on **M**ultiplicative **S**parse **T**ensor **F**actorization (**MSTF**). Specifically, in order to capture the complex relationship in dual-heterogeneous data by a concise model, we use a tensor structure to represent full-order MVMT interactions, in which different orders collectively provide distinct and complementary information (for Problem 1). To make the model parsimonious and interpretable, we propose the vanilla MSTF model by decomposing the model parameters into an element-wise product of two components: one is sparse and shared across all the tasks and the other one is specific for each task (for Problem 2). To prevent from overparameterization, we further apply CANDECOMP / PARAFAC (CP) decomposition [12] and Tucker decomposition [27] to the shared component and the specific one, respectively, and propose the MSTF model. We illustrate the framework of vanilla MSTF and MSTF in Figure 1. Theoretical analysis shows the equivalence between MSTF

[1] Full-order interactions include first-order and higher-order interactions, where first-order interactions represent contributions of the original features in each view and higher-order interactions represent contributions of the tensor product of multi-view features.

Please check ArXiv or contact the authors for any appendices or supplementary material mentioned in the paper.

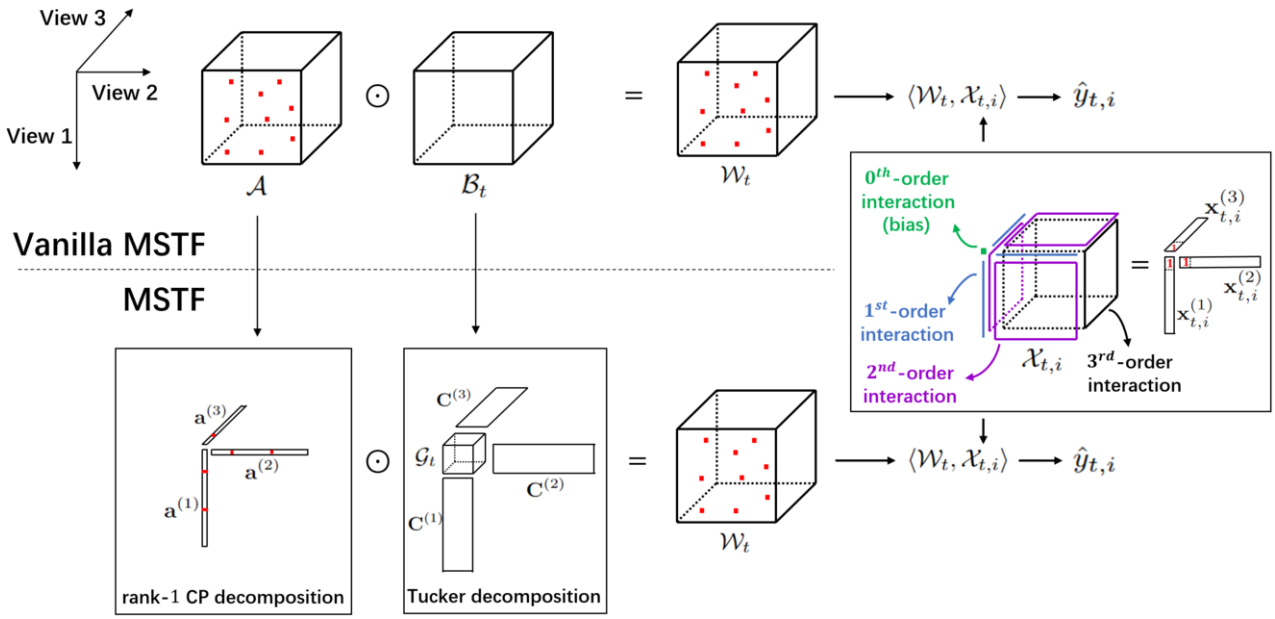


Figure 1. Illustration of the proposed framework. Given three views, the weight tensor of the t-th task $\mathcal{W}_t$ in vanilla MSTF is decomposed into an element-wise product of a task-shared component $\mathcal{A}$ and a task-specific one $\mathcal{B}_t$, where red points represent nonzero interactions shared across multiple tasks. In MSTF, tensor factorization is applied to $\mathcal{A}$ and $\mathcal{B}_t$. The i-th sample $\mathcal{X}_{t,i}$ in the t-th task is represented as a tensor by the tensor product of multi-view features, i.e., $\mathcal{X}_{t,i} = [1; \mathbf{x}_{t,i}^{(1)}] \circ [1; \mathbf{x}_{t,i}^{(2)}] \circ [1; \mathbf{x}_{t,i}^{(3)}]$, to save full-order feature interactions. The predicted output is calculated by $\hat{y}_{t,i} = \langle \mathcal{W}_t, \mathcal{X}_{t,i} \rangle$.

and a family of methods that are jointly regularized with a general form of regularizers. We develop an efficient alternating algorithm to optimize the proposed method. Experimental results on both synthetic and real-world datasets show the effectiveness of MSTF. The contributions in this work can be summarized as follows:

- We propose a novel MVMTL method, MSTF, that collectively models full-order interactions by a concise and multiplicatively decomposable tensor, where only a few interactions are active and shared across tasks.
- The equivalence between MSTF and a general family of jointly regularized models has been proved, leading to several novel MVMTL formulations.
- We develop an efficient alternating algorithm to optimize the problem, and experimental results show its effectiveness on both synthetic and real-word datasets.

2 Related Works

For *Multi-Task Learning (MTL)*, by assuming that multiple tasks share a common latent representation, different regularizers are used to promote low-rankness [22] or sparsity [4, 16] of the model. KMSV [2] uses a new tight approximation for rank constraint. Group structures have also been studied [25, 1]. [30] uses a generalized block-diagonal structure, while VSTG [9] performs variable selection and learns an overlapping group structure. Some other methods are proposed to capture both task correlation and task specificity based on model decomposition by summation [8, 5] or multiplication [18, 28]. In order to exclude useless features, the additive methods need the corresponding elements in all components to be zeros, while the multiplicative methods only need the entries in one component to be zeros, which shows its superiority in feature selection [28].

For *Multi-View Multi-Task Learning (MVMTL)*, IteM^2 [6] constructs a bi-partite graph to capture the relations among tasks and views. CSL-MVMT [10] is proposed by learning a shared latent space of different tasks with common views. MAMUDA [11] learns task-view correlations in the transformed discriminant feature space. SPLIT [24] learns view-wise weights and saves task correlations by multiplicatively decomposing the basis matrix. MTMVCSF [20] captures both consistent and complementary information by latent feature representation of multiple views. These methods can be considered as matrix-based methods, because they are limited to model first-order feature interactions by organizing model weights in a simple matrix form.

Different from the aforementioned matrix-based methods, recently several *Tensor-based MVMTL* methods have been proposed to capture higher-order interactions by using the flexible and expressive tensor structure. In [14], aptMTVL uses an adaptive basis multilinear factor analyzers to handle task-view-label relations. In [15], racMVMT captures the task-view interactions based on asymmetric bilinear factor analyzers and rank constraints. However, they are not sparse models and both ignore full-order interactions among features. MFM [19] learns both task-specific features and task-view shared full-order feature interactions by using multilinear factorization machines, and uses CP decomposition to reduce the complexity. However, it is not flexible enough to learn a parsimonious and interpretable model and fails to directly model task correlations. In contrast, the proposed MSTF uses a tensor structure to represent full-order interactions and efficiently captures a subset of useful interactions shared across tasks by multiplicative sparse tensor factorization.

3 Preliminaries

3.1 Notations and Tensor Basics

In this paper, we use lowercase letters, bold lowercase letters, bold uppercase letters and caligraphic letters to denote scalars, vectors, matrices and tensors, respectively. Here we introduce some basic concepts of tensor based on [13]. An N-order tensor is denoted by $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and element $(i_1, ..., i_N)$ of $\mathcal{X}$ is denoted by

$x_{i_1,\ldots,i_N}$. For two same-sized tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, the inner product is denoted by $\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1,\ldots,i_N} x_{i_1,\ldots,i_N} y_{i_1,\ldots,i_N}$ and the element-wise product is denoted by $\mathcal{X} \odot \mathcal{Y} = \mathcal{Z}$ with element $(i_1 \ldots, i_N)$ being $z_{i_1,\ldots,i_N} = x_{i_1,\ldots,i_N} y_{i_1,\ldots,i_N}$. The tensor (outer) product of N vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}$ is denoted element-wised by $(\mathbf{x}^{(1)} \circ \cdots \circ \mathbf{x}^{(N)})_{i_1,i_2,\ldots,i_N} = x^{(1)}_{i_1} \cdots x^{(N)}_{i_N}$. The n-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is denoted by $(\mathcal{X} \times_n \mathbf{U}) \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$ with its element being $(\mathcal{X} \times_n \mathbf{U})_{i_1 \times \cdots \times i_{n-1} \times j \times i_{n+1} \times \cdots \times I_N} = \sum_{i_n=1}^{I_n} x_{i_1,i_2,\ldots,i_N} u_{j i_n}$. Let $\| \cdot \|_k$ denote the tensor l_k-norm with $\|\mathcal{A}\|_k = \sqrt[k]{\sum_{i_1,\ldots,i_V} |a_{i_1 \ldots i_V}|^k}$, where k is a positive integer. The tensor Frobenius norm $\| \cdot \|_F$ and the tensor l_1-norm are obtained by $k = 2$ and $k = 1$, respectively. For a three-order tensor $\mathcal{X}$, its CP decomposition is defined as

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r, \tag{1}$$

where $\mathbf{a}_r \in \mathbb{R}^{I_1}, \mathbf{b}_r \in \mathbb{R}^{I_2}$ and $\mathbf{c}_r \in \mathbb{R}^{I_3}$. R is a positive integer that is often called CP-rank. Its Tucker decomposition is defined as

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{r_1,r_2,r_3} g_{r_1 r_2 r_3} \mathbf{a}_{r_1} \circ \mathbf{b}_{r_2} \circ \mathbf{c}_{r_3}, \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{I_1 \times R_1}, \mathbf{B} \in \mathbb{R}^{I_2 \times R_2}$ and $\mathbf{C} \in \mathbb{R}^{I_3 \times R_3}$ are factor matrices and $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is the core tensor. In (2), $\mathbf{a}_{r_1}, \mathbf{b}_{r_2}$ and $\mathbf{c}_{r_3}$ represent the r_1-th, r_2-th and r_3-th columns of $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$, respectively. In fact, CP decomposition is the special case of Tucker decomposition when the core tensor is super-diagonal.

3.2 General MVMTL Formulation

Given the MVMTL problem with V views and m tasks, let $(\mathbf{X}_t^{(1)}, \mathbf{X}_t^{(2)}, \ldots, \mathbf{X}_t^{(V)})$ denote the data of the t-th task. $\mathbf{X}_t^{(v)} \in \mathbb{R}^{n_t \times d_v}$ is the data matrix of the v-th view in the t-th task, where n_t denotes the number of samples in the t-th task and d_v denotes the dimensionality of the v-th view. The i-th row $\mathbf{x}_{t,i}^{(v)T}$ of $\mathbf{X}_t^{(v)}$ is the i-th sample of the v-th view in the t-th task. A classic way for MVMTL is to use a linear model to minimize the regularized empirical loss between predicted output and real target $y_{t,i}$, i.e.,

$$\min_{\mathbf{W}} \sum_{t=1}^{m} \frac{1}{n_t} \sum_{i=1}^{n_t} L(y_{t,i}, \sum_{v=1}^{V} \langle \mathbf{w}_t^{(v)}, \mathbf{x}_{t,i}^{(v)} \rangle) + \lambda \Omega(\mathbf{W}), \tag{3}$$

where L is the loss function, Ω is the regularizer and $\lambda > 0$ is the hyperarameter. In (3), $\mathbf{w}_t^{(v)}$ denotes the weight vector for the v-th view in the t-th task, which is the t-th column of the v-th view weight matrix $\mathbf{W}^{(v)} \in \mathbb{R}^{d_v \times m}$. $\mathbf{W} = [\mathbf{W}^{(1)}; \cdots; \mathbf{W}^{(v)}] \in \mathbb{R}^{d \times m} (d = \sum_v d_v)$ is the total weight matrix.

4 Methodology

4.1 Multilinear Models with Full-Order Interactions

The linear model in (3) only considers original features and ignores higher-order interactions among views. Therefore, inspired by MFM [19], we construct a tensor-based data structure for each data sample to capture full-order interactions. Specifically, the i-th sample $\mathcal{X}_{t,i}$ in the t-th task is formulated in tensor form by $\mathcal{X}_{t,i} = [1; \mathbf{x}_{t,i}^{(1)}] \circ \cdots \circ [1; \mathbf{x}_{t,i}^{(V)}]$. Then, we introduce the multilinear predictive model for the predicted output $\hat{y}_{t,i}$:

$$\hat{y}_{t,i} = \langle \mathcal{W}_t, [1; \mathbf{x}_{t,i}^{(1)}] \circ \cdots \circ [1; \mathbf{x}_{t,i}^{(V)}] \rangle = \langle \mathcal{W}_t, \mathcal{X}_{t,i} \rangle, \tag{4}$$

where $\mathcal{W}_t \in \mathbb{R}^{(d_1+1) \times \cdots \times (d_V+1)}$ denotes the weight tensor of the t-th task, with each entry in $\mathcal{W}_t$ saving the weight coefficient of corresponding multi-view feature interaction. An illustrative example is shown in Figure 1. Furthermore, considering both first-order and higher-order interactions can easily handle incomplete and even missing views in MVMTL problems.

4.2 The Vanilla MSTF Method

Although we consider full-order MVMT interactions in (4), in practice, only a subset of interactions may be related with the target and multiple tasks probably share a common subset of interactions. For example, in web page classification, distinct topics of web pages are probably related to text-image interactions via a small number of common pathways. Therefore, to select such interactions and learn a interpretable model, we decompose the weight tensor $\mathcal{W}_t$ of the t-th task into an element-wise product of two components $\mathcal{A}$ and $\mathcal{B}_t$, i.e.,

$$\mathcal{W}_t = \mathcal{A} \odot \mathcal{B}_t, \qquad t = 1, 2, \ldots, m, \tag{5}$$

where $\mathcal{A}$ is shared by all tasks and controls the sparsity across tasks, while $\mathcal{B}_t$ is specific for each task and models task-specificity. Typically, $\mathcal{A}$ is treated as a binary tensor, with 1 indicating the relevance of the corresponding interaction and 0 otherwise. To avoid the difficulty of combinational optimization, the constraint on $\mathcal{A}$ is relaxed to be non-negativity, i.e., $\mathcal{A} \geq 0$. Hence, to promote across-task sparsity by $\mathcal{A}$ and meanwhile learn task-specific weights by $\mathcal{B}_t$, we propose the optimization problem of vanilla MSTF (**vMSTF**):

$$\min_{\mathcal{A}, \mathcal{B}_t} \sum_{t=1}^{m} \frac{1}{n_t} \sum_{i=1}^{n_t} L(y_{t,i}, \langle \mathcal{W}_t, \mathcal{X}_{t,i} \rangle) + \lambda_1 \|\mathcal{A}\|_1 + \lambda_2 \sum_{t=1}^{m} \|\mathcal{B}_t\|_F^2,$$
$$s.t.\ \mathcal{W}_t = \mathcal{A} \odot \mathcal{B}_t, \ \ \mathcal{A} \geq 0, \quad \forall t, \tag{6}$$

where λ_1 and λ_2 are positive hyperparameters. If $(\mathcal{A})_{i_1 \ldots i_V} = 0$, then corresponding interaction in $(\mathcal{W}_t)_{i_1 \ldots i_V}$ is removed for all tasks, regardless of $(\mathcal{B}_t)_{i_1 \ldots i_V}$'s value. Therefore, vMSTF enables to rule out irrelevant interactions by learning a sparse $\mathcal{A}$, and save task-specific weights in $\mathcal{B}_t$, $t = 1, 2, \ldots, m$.

4.3 The MSTF method

The proposed vMSTF in (6) successfully builds a sparse and interpretable model as the target is correlated to interactions via a few pathways, each of which probably involve a small subset of features. However, since $\mathcal{A}, \mathcal{B}_t \in \mathbb{R}^{(d_1+1) \times \cdots \times (d_V+1)}$, there will be $(m+1) \prod_v (d_v + 1)$ parameters to learn, and the number of parameters will increase exponentially with the number of views, making it intractable for large-scale problems. In addition, multiple tasks and views are usually correlated with each other through common interactions, indicating shared low-rank subspace. Thus, in order to further control model complexity and capture task-view correlations, we apply tensor factorization to $\mathcal{A}$ and $\mathcal{B}_t$. Specifically, since $\mathcal{A}$ can be simply treated as a sparse indicator across all tasks, we apply rank-1 CP decomposition[2] to it, i.e., $\mathcal{A} = \mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(V)}$, where

[2] Rank-1 CP decomposition works well in practice, as shown in Table 3 and Figure 4. In addition, it helps to derive Theorem 1 and 2, that generalizes a family of models.

$\mathbf{a}^{(v)} \in \mathbb{R}^{(d_v+1)}$. For $\mathcal{B}_t$, as it models task-specific weights of interactions, which typically contains more complex and rich information than the sparsity structure saved in $\mathcal{A}$, we then apply Tucker decomposition to control its complexity and learn the latent representation, i.e., $\mathcal{B}_t = \mathcal{G}_t \times_1 \mathbf{C}^{(1)} \cdots \times_V \mathbf{C}^{(V)}$, where $\mathbf{C}^{(v)} \in \mathbb{R}^{(d_v+1)\times r_v}$ and $\mathcal{G}_t \in \mathbb{R}^{r_1\times r_2\times\cdots\times r_V}$. Note that $\mathbf{C}^{(v)}$ $(\forall v)$ are shared across all tasks, that saves task correlations and further reduces the model size. Then, based on (5), we have:

$$\begin{aligned}\mathcal{W}_t &= (\mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(V)}) \odot (\mathcal{G}_t \times_1 \mathbf{C}^{(1)} \cdots \times_V \mathbf{C}^{(V)}) \\ &= \mathcal{G}_t \times_1 \mathbf{C}^{(1)} \cdots \times_V \mathbf{C}^{(V)} \times_1 diag(\mathbf{a}^{(1)}) \ldots \times_V diag(\mathbf{a}^{(V)}) \\ &= \mathcal{G}_t \times_1 (diag(\mathbf{a}^{(1)})\mathbf{C}^{(1)}) \cdots \times_V (diag(\mathbf{a}^{(V)})\mathbf{C}^{(V)}) \\ &= \mathcal{G}_t \times_1 \mathbf{F}^{(1)} \times_2 \mathbf{F}^{(2)} \cdots \times_V \mathbf{F}^{(V)}. \end{aligned} \tag{7}$$

where $\mathbf{F}^{(v)} \in \mathbb{R}^{(d_v+1)\times r_v} = diag(\mathbf{a}^{(v)})\mathbf{C}^{(v)}$ and $diag(\mathbf{a}^{(v)})$ represents a diagonal matrix with the elements of $\mathbf{a}^{(v)}$ on the main diagonal. Eq.(7) shows that applying tensor factorization to $\mathcal{A}$ and $\mathcal{B}_t$ in (5) is equivalent to directly applying Tucker decomposition to $\mathcal{W}_t$ and then promoting sparse $\mathbf{F}^{(v)}$ by imposing sparsity constraint on $\mathbf{a}^{(v)}$. Therefore, we propose the optimization problem of **MSTF**:

$$\begin{aligned}\min_{\mathbf{a}^{(v)}\geq 0, \mathbf{C}^{(v)}, \mathcal{G}_t} & \sum_{t=1}^{m} \frac{1}{n_t} \sum_{i=1}^{n_t} L(y_{t,i}, \langle \mathcal{W}_t, \mathcal{X}_{t,i}\rangle) + \\ & \sum_{v=1}^{V} (\lambda_1 \|\mathbf{a}^{(v)}\|_1 + \lambda_2 \|\mathbf{C}^{(v)}\|_F^2) + \lambda_3 \sum_{t=1}^{m} \|\mathcal{G}_t\|_F^2, \\ s.t. \mathcal{W}_t = \mathcal{G}_t \times_1 \mathbf{F}^{(1)} \ldots \times_V \mathbf{F}^{(V)}, & \mathbf{F}^{(v)} = diag(\mathbf{a}^{(v)})\mathbf{C}^{(v)}, \ \forall t, v, \end{aligned} \tag{8}$$

where λ_1, λ_2 and λ_3 are positive hyperparameters. The l_1-norm on $\mathbf{a}^{(v)}$ brings sparsity to $\mathbf{F}^{(v)}$ and thus leads to sparse $\mathcal{W}_t$. Compared with vMSTF, two more advantages are achieved by MSTF. First, the number of parameters decreases from $(m+1)\prod_v (d_v+1)$ to $\sum_v (d_v+1)(r_v+1) + m\prod_v r_v (r_v \ll d_v, \forall v)$ by tensor factorization. Second, tensor factorization brings deeper insight into relationship among views and tasks, which can improve model performance without physically building the weight tensor.

5 Theoretical Analysis

In this section, we will give the theoretical analysis[3] for both vMSTF and MSTF that reveals the equivalence between the two models and a family of joint regularization with a general form of regularizers. For vMSTF in (6), we introduce a general optimization problem:

$$\begin{aligned}\min_{\mathcal{A}\geq 0, \mathcal{B}_t} & \sum_{t=1}^{m} L(\mathcal{W}_t) + \lambda_1 \|\mathcal{A}\|_k^k + \lambda_2 \sum_{t=1}^{m} \|\mathcal{B}_t\|_p^p, \\ s.t. \ & \mathcal{W}_t = \mathcal{A} \odot \mathcal{B}_t, \quad t = 1, 2, ..., m, \end{aligned} \tag{9}$$

where k, p are two positive integers. In this paper, we mainly focus on $k, p \in \{1, 2\}$. When $k = 1, p = 2$, (9) becomes (6). In this section, we denote $\frac{1}{n_t}\sum_{i=1}^{n_t} L(y_{t,i}, \langle \mathcal{W}_t, \mathcal{X}_{t,i}\rangle)$ by $L(\mathcal{W}_t)$ for brevity. Then we have the following theorem:

Theorem 1 *Let $(\hat{\mathcal{A}}, \hat{\mathcal{B}}_t)$ be the optimal solution of problem (9) and $\hat{\mathcal{W}}$ be the optimal solution of the following optimization problem,*

$$\min_{\mathcal{W}} \sum_{t=1}^{m} L(\mathcal{W}_t) + \gamma \sum_{i_1=1}^{d_1+1} \cdots \sum_{i_V=1}^{d_V+1} \sqrt{\|\mathbf{w}_{i_1\ldots i_V:}\|_p^{p/q}}, \tag{10}$$

[3] The proofs for Theorems 1 and 2 are provided in the supplement.

Table 1. A summary of conclusions from Theorems 1 and 2 with $k, p \in \{1, 2\}$. Ω_1 and Ω_2 denote the corresponding regularizers on $\mathbf{w}_{i_1\ldots i_V:}$ in (10) and $(\mathbf{F}^{(v)})_{i:}$ in (13), respectively.

(k,p)	γ	vanilla MSTF		MSTF	
		$\Omega_1(\mathcal{W})$	$(\mathcal{A})_{i_1\ldots i_V}$	$\Omega_2(\mathbf{F}^{(v)})$	$(\mathbf{a}^{(v)})_i$
(1,1)	$2\lambda_1^{\frac{1}{2}}\lambda_2^{\frac{1}{2}}$	$\|\mathbf{w}_{i_1\ldots i_V:}\|_1^{1/2}$	$\frac{\lambda_2}{\lambda_1}\sum_{t=1}^{m}\|(\mathcal{B}_t)_{i_1\ldots i_V}\|$	$\|(\mathbf{F}^{(v)})_{i:}\|_1^{1/2}$	$\frac{\lambda_2}{\lambda_1}\sum_{j=1}^{r_v}\|(\mathbf{C}^{(v)})_{ij}\|$
(1,2)	$2\lambda_1^{\frac{2}{3}}\lambda_2^{\frac{1}{3}}$	$\|\mathbf{w}_{i_1\ldots i_V:}\|_2^{2/3}$	$\frac{\lambda_2}{\lambda_1}\sum_{t=1}^{m}\|(\mathcal{B}_t)_{i_1\ldots i_V}\|^2$	$\|(\mathbf{F}^{(v)})_{i:}\|_2^{2/3}$	$\frac{\lambda_2}{\lambda_1}\sum_{j=1}^{r_v}\|(\mathbf{C}^{(v)})_{ij}\|^2$
(2,1)	$2\lambda_1^{\frac{1}{3}}\lambda_2^{\frac{2}{3}}$	$\|\mathbf{w}_{i_1\ldots i_V:}\|_1^{2/3}$	$(\frac{\lambda_2}{\lambda_1})^{\frac{1}{2}}\sqrt{\sum_{t=1}^{m}\|(\mathcal{B}_t)_{i_1\ldots i_V}\|}$	$\|(\mathbf{F}^{(v)})_{i:}\|_1^{2/3}$	$(\frac{\lambda_2}{\lambda_1})^{\frac{1}{2}}\sqrt{\sum_{j=1}^{r_v}\|(\mathbf{C}^{(v)})_{ij}\|}$
(2,2)	$2\lambda_1^{\frac{1}{2}}\lambda_2^{\frac{1}{2}}$	$\|\mathbf{w}_{i_1\ldots i_V:}\|_2$	$(\frac{\lambda_2}{\lambda_1})^{\frac{1}{2}}\sqrt{\sum_{t=1}^{m}\|(\mathcal{B}_t)_{i_1\ldots i_V}\|^2}$	$\|(\mathbf{F}^{(v)})_{i:}\|_2$	$(\frac{\lambda_2}{\lambda_1})^{\frac{1}{2}}\sqrt{\sum_{j=1}^{r_v}\|(\mathbf{C}^{(v)})_{ij}\|^2}$

where $\mathbf{w}_{i_1\ldots i_V:} \in \mathbb{R}^m$ is the mode-$(V+1)$ fiber of $\mathcal{W} = [\mathcal{W}_1; ...; \mathcal{W}_m] \in \mathbb{R}^{(d_1+1)\times\cdots\times(d_V+1)\times m}$. When $\gamma = 2\sqrt{\lambda_1^{p/kq}\lambda_2^{2-p/kq}}$ and $q = (k+p)/2k$, we have $\hat{\mathcal{W}}_t = \hat{\mathcal{A}} \odot \hat{\mathcal{B}}_t$. In addition, task-shared $\hat{\mathcal{A}}$ is related to task-specific $\hat{\mathcal{B}}_t$ by:

$$(\mathcal{A})_{i_1\ldots i_V} = (\frac{\lambda_2}{\lambda_1})^{\frac{1}{k}} \sqrt[k]{\sum_{t=1}^{m} |(\mathcal{B}_t)_{i_1\ldots i_V}|^p}, \quad \forall i_1, ..., i_V. \tag{11}$$

The above theorem shows that under some circumstances, (9) is equivalent to (10), which leads to group sparsity of $\mathcal{W}$ by treating the $(i_1, ...i_V)$-th interactions across tasks as a group and promote sparsity at the inter-group level. In addition, the direct relationship in (11) between $\mathcal{A}$ and $\mathcal{B}_t$ helps us to develop an efficient optimization algorithm.

Similarly, we introduce another general optimization problem for MSTF in (8):

$$\begin{aligned}\min_{\substack{\mathbf{a}^{(v)}\geq 0,\\ \mathbf{C}^{(v)}, \mathcal{G}_t}} & \sum_{t=1}^{m} L(\mathcal{W}_t) + \sum_{v=1}^{V} (\lambda_1 \|\mathbf{a}^{(v)}\|_k^k + \lambda_2 \|\mathbf{C}^{(v)}\|_p^p) + \lambda_3 \sum_{t=1}^{m} \|\mathcal{G}_t\|_F^2, \\ s.t. \mathcal{W}_t = \mathcal{G}_t \times_1 \mathbf{F}^{(1)} \ldots \times_V \mathbf{F}^{(V)}, & \mathbf{F}^{(v)} = diag(\mathbf{a}^{(v)})\mathbf{C}^{(v)}, \ \forall t, v. \end{aligned} \tag{12}$$

When $k = 1, p = 2$, (12) becomes (8). Then we have the following theorem:

Theorem 2 *Let $(\hat{\mathbf{a}}^{(v)}, \hat{\mathbf{C}}^{(v)}, \hat{\mathcal{G}}_t)$ be the optimal solution of problem (12) and $(\hat{\mathbf{F}}^{(v)}, \hat{\mathcal{G}}_t)$ be the optimal solution of the following problem,*

$$\begin{aligned}\min_{\mathbf{F}^{(v)}, \mathcal{G}_t} & \sum_{t=1}^{m} L(\mathcal{W}_t) + \gamma \sum_{v=1}^{V} \sum_{i=1}^{d_v+1} \sqrt{\|(\mathbf{F}^{(v)})_{i:}\|_p^{p/q}} + \lambda_3 \sum_{t=1}^{m} \|\mathcal{G}_t\|_F^2, \\ s.t. \ & \mathcal{W}_t = \mathcal{G}_t \times_1 \mathbf{F}^{(1)} \times_2 \mathbf{F}^{(2)} \cdots \times_V \mathbf{F}^{(V)}, \quad \forall t, \end{aligned} \tag{13}$$

where $(\mathbf{F}^{(v)})_{i:} \in \mathbb{R}^{r_v}$ represents the i-th row of $\mathbf{F}^{(v)}$. When $\gamma = 2\sqrt{\lambda_1^{p/kq}\lambda_2^{2-p/kq}}$ and $q = (k+p)/2k$, we have $\mathbf{F}^{(v)} = diag(\mathbf{a}^{(v)})\mathbf{C}^{(v)}$. In addition, $\hat{\mathbf{a}}^{(v)}$ is related to $\hat{\mathbf{C}}^{(v)}$ according to:

$$(\mathbf{a}^{(v)})_i = (\frac{\lambda_2}{\lambda_1})^{\frac{1}{k}} \sqrt[k]{\sum_{j=1}^{r_v} |(\mathbf{C}^{(v)})_{ij}|^p}, \quad \forall i. \tag{14}$$

The above theorem shows that (12) is equivalent to (13) under some circumstances and $\mathbf{a}^{(v)}$ is relative to $\mathbf{C}^{(v)}$. A summary of conclusions from Theorems 1 and 2 is shown in Table 1.

6 Optimization

Since (9) and (12) are general forms of vMSTF in (6) and MSTF in (8), respectively, here we develop optimization algorithms for (9) and (12). We show the algorithm for MSTF in this section and present the algorithm for vMSTF in the supplement.

6.1 Algorithm for MSTF

The optimization problem in (12) is convex w.r.t $\mathbf{a}^{(v)}$, $\mathbf{C}^{(v)}$ and $\mathcal{G}_t$, respectively, so we can use an alternating algorithm to solve it. Least squared loss is used here and algorithms for other losses can be easily extended. For clarity, we reformulate the loss in a matrix form. The algorithm repeats following three steps until convergence.

1. Update $\mathbf{C}^{(v)}$ with fixed $\mathbf{a}^{(v)}$ and $\mathcal{G}_t$. Let $(\mathbf{X}_{t,i})_{(v)}$, $(\mathbf{W}_t)_{(v)}$ and $(\mathbf{G}_t)_{(v)}$ be the mode-v matricization of tensor $\mathcal{X}_{t,i}$, $\mathcal{W}_t$ and $\mathcal{G}_t$, respectively. Then we get the subproblem w.r.t. $\mathbf{C}^{(v)}$:

$$\min_{\mathbf{C}^{(v)}} \sum_{t=1}^{m} \frac{1}{n_t} \|\mathbf{y}_t - \mathbf{X}_t[(\mathbf{F}^{(-v)}(\mathbf{G}_t)_{(v)}{}^T) \otimes diag(\mathbf{a}_v)] vec(\mathbf{C}^{(v)})\|_2^2 + \lambda_2 \|vec(\mathbf{C}^{(v)})\|_p^p, \tag{15}$$

   where $\mathbf{X}_t$ denotes the data matrix of the t-th task with the i-th row being $vec((\mathbf{X}_{t,i})_{(v)})^T$, $\mathbf{F}^{(-v)} = \mathbf{F}^{(V)} \cdots \otimes \mathbf{F}^{(v+1)} \otimes \mathbf{F}^{(v-1)} \cdots \otimes \mathbf{F}^{(1)}$, and $\otimes$ represents the kronecker product. When $p = 1$, (15) is a lasso problem which can be solved by accelerated proximal gradient descent [21]. When $p = 2$, it is a ridge regression problem which is solved by a closed-form solution.
2. Update $\mathbf{a}^{(v)}$ with fixed $\mathbf{C}^{(v)}$ and $\mathcal{G}_t$. According to Table 1, $\mathbf{a}^{(v)}$ can be solved by a closed-form solution.
3. Update $\mathcal{G}_t$ with fixed $\mathbf{a}^{(v)}$ and $\mathbf{C}^{(v)}$. Let $\mathbf{X}_t$ with the i-th row being $vec(\mathcal{X}_{t,i})^T$, then each task can be independently optimized and the subproblem is a ridge regression problem:

$$\min_{\mathcal{G}_t} \frac{1}{n_t} \|\mathbf{y}_t - \mathbf{X}_t(\mathbf{F}^{(V)} \otimes \cdots \otimes \mathbf{F}^{(1)}) vec(\mathcal{G}_t)\|_2^2 + \lambda_3 \|vec(\mathcal{G}_t)\|_2^2, \tag{16}$$

   which and can be solved by a closed-form solution.

Proposition 1 *The proposed alternating algorithm does not increase the objective value of (12) at each iteration, i.e.,*

$$J(\mathbf{a}^{(v)(i+1)}, \mathbf{C}^{(v)(i+1)}, \mathcal{G}_t^{(i+1)}) \leq J(\mathbf{a}^{(v)(i)}, \mathbf{C}^{(v)(i)}, \mathcal{G}_t^{(i)}), \tag{17}$$

in the $(i+1)$-th iteration, with J denoting its objective value.

It guarantees that the objective value of (12) will not increase in each iteration. The proof of Proposition 1 and the Matlab code of vMSTF and MSTF are given in the supplement.

6.2 Computational Analysis

For vMSTF, in each iteration, it has a linear time complexity in the number of samples and tasks, which is $\mathcal{O}(mn(\prod_v (d_v + 1)))$, where $n = \sum_t n_t$. Details of vMSTF are given in the supplement. For MSTF, updating $\{\mathbf{C}^{(v)}\}_{v=1}^V$ and $\{\mathcal{G}_t\}_{t=1}^m$ require $\mathcal{O}(mn(\sum_v r_v(d_v+1)))$ and $\mathcal{O}(mn\prod_v r_V)$, respectively. Calculating $\{\mathbf{a}^{(v)}\}_{v=1}^V$ requires $\mathcal{O}(m(\sum_v (d_v + 1)))$. Therefore, MSTF has a total time complexity of $\mathcal{O}(mn\prod_v r_v + mn(\sum_v r_v(d_v + 1)))$ in each iteration, which is linear in the number of features, samples and tasks. Since $r_v \ll d_v$, MSTF has much superiority in time complexity compared to vMSTF, especially when the number V of views and the dimension d_v of each views are large.

Table 2. The statistics of used seven real-world datasets

Datasets	V	m	d_v	n_t	Domain
FOX	2	4	747∼2711	1523	Text categorization
Emotions	2	6	8∼64	593	Music emotion detection
Yeast	2	14	24∼79	2417	Bioinformatics
NUS-Object	5	7	64∼225	10370	Image annotation
NUS-Scene	5	15	63∼224	16406	Image annotation
Mirflickr	6	38	100∼4096	25000	Image annotation
Corel5k	6	260	100∼4096	4999	Image annotation

7 Experiments

7.1 Experimental Setting

7.1.1 Synthetic Datasets

For synthetic datasets, we set the number of tasks and views as $m = 5$ and $V = 2$, respectively. Each task has the same number of samples $(n = 100)$ and each view has the same number of features $(d = 19)$. For $\mathcal{A}$, $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(2)}$ are randomly sampled from uniform distribution $\mathcal{U}(0, 1)$. For $\mathcal{B}_t$, $\mathbf{C}^{(v)} \in \mathbb{R}^{(d+1)\times r_v}$ and $\mathcal{G}_t \in \mathbb{R}^{r_1 \times r_2}$ are randomly sampled from normal distribution $\mathcal{N}(0, 6)$, with $r_1 = r_2 = 4$. Then $\mathcal{A}$ and $\mathcal{B}_t \in \mathbb{R}^{(d+1)\times(d+1)}$ can be generated by $\mathcal{A} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)}$ and $\mathcal{B}_t = \mathcal{G}_t \times_1 \mathbf{C}^{(1)} \times_2 \mathbf{C}^{(2)}$, respectively, and the weight tensor $\mathcal{W}_t$ for each task is calculated by $\mathcal{A} \odot \mathcal{B}_t$. To make $\mathcal{W}_t$ sparse,we assign zero values to selected entries of $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(2)}$ according to the indices $\{1 : 3, 7 : 9, 13 : 15, 19 : 20\}$ and $\{1 : 3, 18 : 20\}$, respectively. Finally, the target of each sample is calculated by $y_{t,i} = \langle \mathcal{W}_t, \mathcal{X}_{t,i} \rangle + \xi_{t,i}$, where $\mathcal{X}_{t,i} \in \mathbb{R}^{(d+1)\times(d+1)} = [1; \mathbf{x}_{t,i}^{(1)}] \circ [1; \mathbf{x}_{t,i}^{(2)}]$ and $\mathbf{x}_{t,i}^{(v)}$ is randomly sampled from normal distribution $\mathcal{N}(0, 5)$ and $\xi_{t,i} \sim \mathcal{N}(0, 0.01)$ denotes the stochastic noise.

7.1.2 Real-World Datasets

Experiments are conducted on seven real-world datasets: FOX[4], Emotions[5], Yeast[5], NUS-Object[6], NUS-Scene[6], Mirflickr[7] and Corel5k[8]. Their statistics are summarized in Table 2. More details are shown in the supplement.

7.1.3 Comparing Methods

We compare MSTF[9] with three types of methods: MTL, MVMTL and baseline. For MTL, we use MMTFL [28], VSTG [9] and KMSV [2] as the comparing methods. MMTFL uses multiplication method to do feature selection, VSTG performs variable selection and learns group structure, and KMSV applies a new tight approximation for rank constraint. For MVMTL, we use MFM [19], racMVMT [15] and SPLIT [24] as the comparing methods. Both MFM and racMVMT are tensor-based MVMTL methods, while SPLIT is a multiplicative matrix-based model. Lasso [26] is used as the baseline. The proposed vMSTF and MSTF are implemented based on the specific choices of k and p in (9) and (12): vMSTF uses $k = 1$ and $p = 2$, and MSTF uses $k = 2$ and $p = 2$.

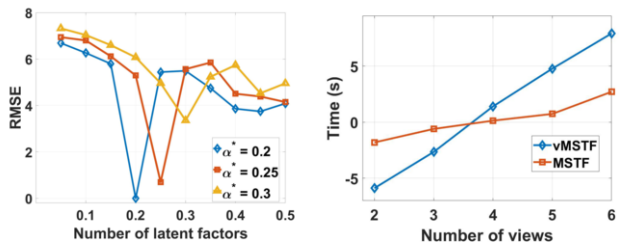
[4] https://sites.google.com/site/qianmingjie/home/datasets/
[5] https://sourceforge.net/projects/mulan/files/datasets/
[6] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
[7] https://press.liacs.nl/mirflickr/
[8] https://github.com/watersink/Corel5K/
[9] We provide the MATLAB code of vMSTF and MSTF at: https://github.com/xywang27/MSTF

Table 3. Experimental results on seven real-world datasets. The best results of each dataset are highlighted in boldface.

Dataset	metric	Lasso	MMTFL	VSTG	KMSV	MFM	racMVMT	SPLIT	vMSTF	MSTF
FOX	ACC	.920(.007)	.909(.006)	.915(.007)	.977(.004)	**.979(.002)**	.889(.019)	.916(.008)	.976(.003)	**.979(.003)**
	AUC	.996(.001)	.996(.001)	.996(.001)	.996(.000)	.996(.002)	.994(.001)	**.997(.001)**	.996(.001)	**.997(.001)**
Emotions	ACC	.732(.018)	.716(.011)	.726(.016)	.796(.013)	.783(.013)	.729(.016)	.732(.015)	**.799(.010)**	.785(.007)
	AUC	.835(.010)	.813(.015)	.841(.015)	.836(.006)	.841(.011)	.833(.016)	.839(.011)	**.842(.014)**	.820(.008)
Yeast	ACC	.603(.004)	.571(.013)	.581(.017)	.800(.003)	.805(.006)	.576(.005)	.591(.009)	.805(.003)	**.807(.002)**
	AUC	.685(.008)	.646(.011)	.688(.007)	.701(.003)	**.741(.015)**	.685(.017)	.691(.007)	.707(.008)	.708(.012)
NUS-Object	ACC	.678(.007)	.661(.010)	.644(.012)	.884(.005)	.889(.008)	.655(.011)	.646(.006)	.885(.006)	**.892(.004)**
	AUC	.884(.005)	.883(.007)	.846(.017)	.885(.006)	.886(.003)	.883(.008)	.857(.017)	.884(.011)	**.891(.012)**
NUS-Scene	ACC	.633(.005)	.598(.005)	.593(.019)	.903(.003)	.902(.004)	.592(.008)	.601(.006)	.903(.002)	**.905(.003)**
	AUC	.815(.006)	.807(.006)	.779(.013)	.812(.005)	.800(.004)	.804(.009)	.775(.011)	.809(.007)	**.817(.013)**
Mirflickr	ACC	.586(.012)	.594(.009)	.603(.015)	.627(.005)	.580(.020)	.604(.008)	.545(.021)	.617(.000)	**.663(.000)**
	AUC	.639(.006)	.637(.009)	.637(.014)	.476(.017)	.571(.023)	.650(.009)	.577(.012)	.594(.000)	**.663(.000)**
Corel5k	ACC	.603(.010)	.610(.041)	.586(.052)	.786(.062)	.777(.015)	.601(.022)	.511(.013)	.818(.000)	**.829(.000)**
	AUC	.692(.026)	.670(.040)	.652(.039)	.507(.019)	**.728(.030)**	.679(.032)	.551(.021)	.577(.001)	.677(.001)

(a) Designed $\mathcal{W}^*_{::1}$ (b) $\mathcal{W}_{::1}$ by vMSTF (c) $\mathcal{W}_{::1}$ by MSTF

Figure 2. Illustration of sparse structures in weight tensor $\mathcal{W} \in \mathbb{R}^{20\times20\times5}$ recovered by vMSTF and MSTF on the synthetic dataset. Here the first frontal slice $\mathcal{W}_{::1} \in \mathbb{R}^{20\times20}$ is shown. The darker the color, the larger the values, and white means zero.



(a) Correlation modeling (b) Running time

Figure 3. Study on correlation modeling (a) and running time (b) on the synthetic datasets. In (a), the latent factor ratio α is varied from 0.05 to 0.5 by step 0.05 and three synthetic datasets are generated with different $\alpha^* \in \{0.2, 0.25, 0.3\}$. In (b), experiments are conducted on five synthetic datasets, which are generated by varying the number of views V from 2 to 6. RMSE and running time in seconds is shown in logarithm scale.

7.1.4 *Configuration*

For each task, we randomly select 60%, 20% and 20% of total samples as training set, validation set and testing set, respectively. We repeat this procedure five times and report the mean value and standard deviation of two metrics: Accuracy (ACC) and Area Under ROC-Curve (AUC). For grid search, the regularization coefficients of all the comparing methods and vMSTF are selected from $\{10^{-5}, 10^{-4}, \cdots, 10^{5}\}$. The regularization coefficients of MSTF are selected from $\{10^{-8}, 10^{-7}, \cdots, 10^{2}\}$. For VSTG and KMSV, the number K of bases is selected from $\{1, 2, 3, 4\}$, and k of k-support norm in VSTG is selected from $\{1, 2, 3\}$. For MFM, the dimension R of latent factors is fixed to be 20, as recommended in [19]. In SPLIT, the number of latent topics is selected from $\{1, 2, 3, 4\}$. In the proposed MSTF, we set a latent factor ratio $\alpha = \frac{r_v}{d_v+1}, \forall v$ and select it from $\{0.05, 0.1, 0.15, ..., 0.5\}$. For each algorithm, we set the maximum number of iterations to be 1000 and terminate it once the relative change of its objective value is less than 10^{-4}.

7.2 *Experiments on Synthetic Datasets*

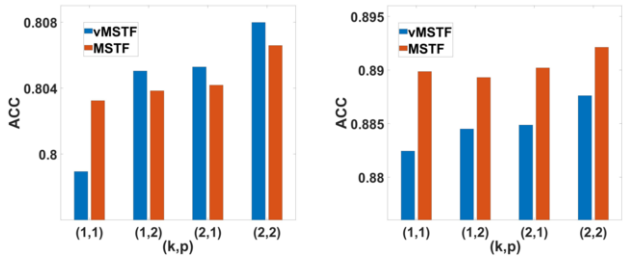
7.2.1 *Illustration of Feature Interaction Selection*

We investigate the effect of feature interaction selection by vMSTF and MSTF on the synthetic dataset. Figure 2 shows the result of the first frontal slice $\mathcal{W}_{::1} \in \mathbb{R}^{20\times20}$ of the weight tensor $\mathcal{W} \in \mathbb{R}^{20\times20\times5}$, which models zero-order, first-order and second-order feature interactions in the first task. The results of the other slices are omitted, as they exhibit similar patterns. Here $\mathcal{W}^*_{::1}$ in Figure 2(a) denotes the designed pattern, $\mathcal{W}_{::1}$ in Figure 2(b) is learned by vMSTF with $\lambda_1 = 10^3$ and $\lambda_2 = 1$, and $\mathcal{W}_1$ Figure 2(c) is learned by MSTF with $\lambda_1 = \lambda_2 = \lambda_3 = 0.1$ and $\alpha = 0.2$. As shown in Figure 2, both vMSTF and MSTF successfully recover the sparse structure in the weight tensor, by selecting the designed subset of mode-3 fibers of $\mathcal{W}$, which are shared across five tasks. Since $\mathcal{W}^*_{::1}$ is designed according to the setting of MSTF, MSTF recovers the underlying sparse pattern better than vMSTF.

7.2.2 *Analysis on Correlation Modeling and Running Time*

To show the effectiveness of tensor factorization on modeling task-view correlations, we generate three synthetic datasets with $\alpha^* \in \{0.2, 0.25, 0.3\}$ and apply MSTF by changing α from 0.05 to 0.5 by step 0.05. As the synthetic datasets belong to the regression problem, Root Mean Squared Error (RMSE) is used as the metric. We report the results of MSTF in Figure 3(a). It shows that on three datasets, RMSE drops first and then stabilizes as α increases, and MSTF always performs the best when $\alpha = \alpha^*$. Therefore, once the interactions are modeled by a small number of latent factors, MSTF has a chance to improve its performance in less model complexity.

To compare the running time of vMSTF and MSTF, we conduct an experiment on five synthetic datasets by varying the number V of views from 2 to 6. Experimental results are shown in Figure 3(b). It shows that the running time of vMSTF rapidly increases as V increases, due to its exponentially growth of the model size w.r.t. V. The running time of MSTF grows much more slowly compared to vMSTF. Therefore, we can conclude that MSTF can reduce the computational time by tensor factorization, especially when the number



(a) Yeast (b) NUS-Object

Figure 4. Comparison of different variants ($k, p \in \{1, 2\}$ in (9) and (12)) of vMSTF and MSTF in ACC on the Yeast and NUS-Object datasets.

of views is large, and achieve high performance at the same time, once the dataset indeed exhibits a small number of latent factors.

7.3 Experiments on Real-World Datasets

7.3.1 Evaluation on Comparing Methods

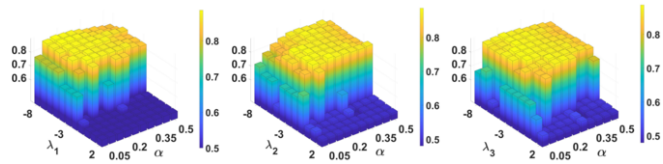
We conduct experiments on seven real-world datasets, and report the results in Table 3, where the best results are highlighted in boldface. From Table 3, we find that vMSTF and MSTF together perform the best in 12 out of total 14 cases. This performance superiority of vMSTF and MSTF on solving MVMTL problems probably results from their capability on collectively capturing full-order information in multilinear predictive models and selecting useful interactions shared accross tasks. MFM obtains the best result in AUC on the Yeast and the Corel5k dataset, which shows the necessity to capture full-order interactions in dual-heterogeneous datasets. SPLIT and racMVMT are not as competitive as MFM, vMSTF and MSTF, since SPLIT only considers first-order interactions and can not learn a sparse and interpretable model, and racMVMT only saves second-order interactions. As a matrix-based MTL method, MMTFL does not perform very well, which demonstrates that simply promoting sparsity in first-order linear model might not be effective enough to deal with complex practical problems. KMSV outperforms MMTFL and VSTG, which shows the importance of task correlation modeling by imposing low-rank constraint on the weight matrix.

7.3.2 Analysis on Different Variants of vMSTF and MSTF

To evaluate the different variants of vMSTF and MSTF, experiments are conducted by selecting $k, p \in \{1, 2\}$ in (9) and (12), and the results in ACC on the Yeast and NUS-Object datasets are shown in Figure 4. The results on other datasets are given in the supplement. As shown in Figure 4, we can find that for both vMSTF and MSTF, the best performance is achieved when $k = 2$ and $p = 2$. In contrast, the performance is the worst when $k = 1$ and $p = 1$. This indicates that promoting strong sparsity (small values of k and p) may not be suitable for some real-world datasets. In practice, setting an appropriate degree of sparsity is preferred for both vMSTF and MSTF.

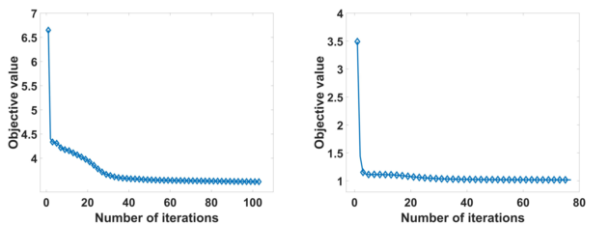
7.3.3 Hyperparameter Sensitivity Analysis

Hyperparameter sensitivity analysis of the four hyperparameters ($\lambda_1, \lambda_2, \lambda_3$ and α) of MSTF is conducted on the NUS-Object dataset. Specifically, λ_1 and λ_2 control the sparsity of task-shared $\mathcal{A}$ and task-specific $\mathcal{B}_t$, respectively, λ_3 controls the strength of regularization on the core tensor $\mathcal{G}_t$ and the factor ratio α controls the number of latent



(a) $\lambda_2 = \lambda_3 = 10^{-3}$ (b) $\lambda_1 = \lambda_3 = 10^{-3}$ (c) $\lambda_1 = \lambda_2 = 10^{-3}$

Figure 5. Sensitivity analysis of $\lambda_1, \lambda_2, \lambda_3$ and α of MSTF in AUC on the NUS-Object dataset. Values (shown in the logarithm scale) of $\lambda_1, \lambda_2, \lambda_3$ are selected from $\{10^{-8}, 10^{-7}, ..., 10^{2}\}$ while the value of α is selected from $\{0.05, 0.1, ..., 0.5\}$.



(a) Yeast (b) NUS-Object

Figure 6. Convergence analysis of MSTF on two datasets. Algorithm converges at 104th and 76th iterations on Yeast and NUS-Object, respectively.

factors. We select $\lambda_1, \lambda_2, \lambda_3$ from $\{10^{-8}, 10^{-7}, ..., 10^{2}\}$, and select α from $\{0.05, 0.1, ..., 0.5\}$. Experimental results are shown in Figure 5. The first experiment in (a) is conducted by fixing $\lambda_2 = \lambda_3 = 10^{-3}$ and similar setting is applied in (b) and (c). From Figure 5, we reach two conclusions: (1) As α increases, the performance rises first and then stabilizes. It reaches the best when α is around 0.2; (2) The best performance on the NUS-Object dataset is achieved when $\lambda_1 \leq 10^{-3}$ and $\lambda_2, \lambda_3 \leq 10^{-2}$. Typically, it is recommended to set small values to λ_1, λ_2 and λ_3, and set α between 0.1 and 0.4.

7.3.4 Convergence analysis

Convergence analysis of MSTF is conducted on the Yeast and the NUS-Object dataset, where hyperparameters are fixed by $\lambda_1 = \lambda_2 = \lambda_3 = 10^{-3}$ and $\alpha = 0.2$. We terminate the algorithm once the relative change of the objective value is below 10^{-4} and report the convergence curves in Figure 4. Figure 4 shows that the objective value usually converges after a few number of iterations, which demonstrates the effectiveness and efficiency of the proposed algorithms.

8 Conclusion

In this paper, we propose a multiplicative sparse tensor factorization (MSTF) method for MVMTL, which considers full-order interactions using a tensor structure and selects useful interactions across tasks by multiplicative decomposition. Thanks to tensor factorization, the number of model parameters is significantly reduced and a consensus latent feature representation is learned. Theoretical analysis reveals the equivalence between MSTF and a family of formulations with a general form of joint regularization, which essentially promotes group sparse structure on the weight tensor. We develop an efficient alternating algorithm to solve the optimization problems of MSTF and vMSTF, and show its effectiveness on both synthetic and real-world dual-heterogeneous datasets.

Acknowledgement

This work has been partially supported by the following fundings: MEXT KAKENHI [21H05027 and 22H03645] to H.M. and MEXT KAKENHI [22K12150] to C.H.N.

References

[1] Aviad Barzilai and Koby Crammer, 'Convex multi-task learning by clustering', in *Artificial Intelligence and Statistics*, pp. 65–73. PMLR, (2015).

[2] Wei Chang, Feiping Nie, Rong Wang, and Xuelong Li, 'New tight relaxations of rank minimization for multi-task learning', in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2910–2914, (2021).

[3] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang, 'Mv-rnn: A multi-view recurrent neural network for sequential recommendation', *IEEE Transactions on Knowledge and Data Engineering*, **32**(2), 317–331, (2018).

[4] Pinghua Gong, Jieping Ye, and Chang-shui Zhang, 'Multi-stage multi-task feature learning', *Advances in neural information processing systems*, **25**, (2012).

[5] Pinghua Gong, Jieping Ye, and Changshui Zhang, 'Robust multi-task feature learning', in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 895–903, (2012).

[6] Jingrui He and Rick Lawrence, 'A graphbased framework for multi-task multi-view learning', in *ICML*, (2011).

[7] Feiran Huang, Xiaoming Zhang, Zhonghua Zhao, Zhoujun Li, and Yueying He, 'Deep multi-view representation learning for social images', *Applied Soft Computing*, **73**, 106–118, (2018).

[8] Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep Ravikumar, 'A dirty model for multi-task learning', *Advances in neural information processing systems*, **23**, (2010).

[9] Jun-Yong Jeong and Chi-Hyuck Jun, 'Variable selection and task grouping for multi-task learning', in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1589–1598, (2018).

[10] Xin Jin, Fuzhen Zhuang, Shuhui Wang, Qing He, and Zhongzhi Shi, 'Shared structure learning for multiple tasks with multiple views', in *Joint European conference on machine learning and knowledge discovery in databases*, pp. 353–368. Springer, (2013).

[11] Xin Jin, Fuzhen Zhuang, Hui Xiong, Changying Du, Ping Luo, and Qing He, 'Multi-task multi-view learning for heterogeneous tasks', in *Proceedings of the 23rd ACM international conference on information and knowledge management*, pp. 441–450, (2014).

[12] Henk AL Kiers, 'Towards a standardized notation and terminology in multiway analysis', *Journal of Chemometrics: A Journal of the Chemometrics Society*, **14**(3), 105–122, (2000).

[13] Tamara G Kolda and Brett W Bader, 'Tensor decompositions and applications', *SIAM review*, **51**(3), 455–500, (2009).

[14] Xiaoli Li and Jun Huan, 'aptmtvl: Nailing interactions in multi-task multi-view multi-label learning using adaptive-basis multilinear factor analyzers', in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 1171–1180, (2016).

[15] Xiaoli Li and Jun Huan, 'Interactions modeling in multi-task multi-view learning with consistent task diversity', in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 853–861, (2018).

[16] Han Liu, Mark Palatucci, and Jian Zhang, 'Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery', in *Proceedings of the 26th annual international conference on machine learning*, pp. 649–656, (2009).

[17] Shengchao Liu, Meng Qu, Zuobai Zhang, Huiyu Cai, and Jian Tang, 'Structured multi-task learning for molecular property prediction', in *International Conference on Artificial Intelligence and Statistics*, pp. 8906–8920. PMLR, (2022).

[18] Aurelie C Lozano and Grzegorz Swirszcz, 'Multi-level lasso for sparse multi-task regression', in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 595–602, (2012).

[19] Chun-Ta Lu, Lifang He, Weixiang Shao, Bokai Cao, and Philip S Yu, 'Multilinear factorization machines for multi-task multi-view learning', in *Proceedings of the tenth ACM international conference on web search and data mining*, pp. 701–709, (2017).

[20] Run-kun Lu, Jian-wei Liu, Si-ming Lian, and Xin Zuo, 'Multi-view representation learning in multi-task scene', *Neural Computing and Applications*, **32**(14), 10403–10422, (2020).

[21] Yurii Nesterov, *Introductory lectures on convex optimization: A basic course*, volume 87, Springer Science & Business Media, 2013.

[22] Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye, 'Trace norm regularization: Reformulations, algorithms, and multi-task learning', *SIAM Journal on Optimization*, **20**(6), 3465–3489, (2010).

[23] Vikas Sindhwani and David S Rosenberg, 'An rkhs for multi-view learning and manifold co-regularization', in *Proceedings of the 25th international conference on Machine learning*, pp. 976–983, (2008).

[24] Lu Sun, Canh Hao Nguyen, and Hiroshi Mamitsuka, 'Multiplicative sparse feature decomposition for efficient multi-view multi-task learning.', in *IJCAI*, pp. 3506–3512, (2019).

[25] Sebastian Thrun and Joseph O'Sullivan, 'Discovering structure in multiple learning tasks: The tc algorithm', in *ICML*, volume 96, pp. 489–497, (1996).

[26] Robert Tibshirani, 'Regression shrinkage and selection via the lasso', *Journal of the Royal Statistical Society: Series B (Methodological)*, **58**(1), 267–288, (1996).

[27] Ledyard R Tucker, 'Some mathematical notes on three-mode factor analysis', *Psychometrika*, **31**(3), 279–311, (1966).

[28] Xin Wang, Jinbo Bi, Shipeng Yu, Jiangwen Sun, and Minghu Song, 'Multiplicative multitask feature learning', *Journal of Machine Learning Research*, **17**(80), 1–30, (2016).

[29] Li Xiao, Julia M Stephen, Tony W Wilson, Vince D Calhoun, and Yu-Ping Wang, 'A manifold regularized multi-task learning model for iq prediction from two fmri paradigms', *IEEE Transactions on Biomedical Engineering*, **67**(3), 796–806, (2019).

[30] Zhiyong Yang, Qianqian Xu, Yangbangyan Jiang, Xiaochun Cao, and Qingming Huang, 'Generalized block-diagonal structure pursuit: Learning soft latent task assignment against negative transfer', *Advances in Neural Information Processing Systems*, **32**, (2019).

[31] Jintao Zhang and Jun Huan, 'Inductive multi-task learning with multiple view data', in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 543–551, (2012).

[32] Yu Zhang and Qiang Yang, 'A survey on multi-task learning', *IEEE Transactions on Knowledge and Data Engineering*, (2021).

[33] Lecheng Zheng, Yu Cheng, and Jingrui He, 'Deep multimodality model for multi-task multi-view learning', in *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 10–18. SIAM, (2019).