GSpartan: a Geospatio-Temporal Multi-task Learning Framework for Multi-location Prediction

Jianpeng Xu* Pang-Ning Tan* Lifeng Luo† Jiayu Zhou*

Abstract

This paper presents a novel geospatio-temporal prediction framework called GSpartan to simultaneously build local regression models at multiple locations. The framework assumes that the local models share a common, low-rank representation, which makes them amenable to multi-task learning. GSpartan learns a set of base models to capture the spatio-temporal variabilities of the data and represents each local model as a linear combination of the base models. A graph Laplacian regularization is used to enforce constraints on the local models based on their spatial autocorrelation. We also introduce sparsity-inducing norms to perform feature selection for the base models and model selection for the local models. Experimental results using historical climate data from 37 weather stations showed that, on average, GSpartan outperforms single-task learning and other existing multi-task learning methods in more than 65% of the stations, which increases to 81% when there are fewer training examples.

1 Introduction

Recent years have witnessed an explosive growth of geospatio-temporal data generated by a wide array of sensing technology and through large-scale scientific simulations. Such data are prevalent across many disciplines, including geophysical and environmental sciences [15, 3, 1], medical informatics [12], and computational fluid dynamics [29]. A geospatio-temporal prediction task typically requires making predictions for a response variable at multiple locations. For example, climate scientists are interested to obtain future climate projections at multiple sites for a geographical region of interest. Similarly, in disease surveillance, researchers are interested to estimate the rate of disease incidence in multiple endemic areas [21]. A simple way to make the predictions would be to fit a local model at each location based on its historical data. This approach may not be effective especially when there are limited training examples available at each location. Furthermore, it does not utilize data from nearby locations, which are often useful due to Tobler's first law of geography: "Everything is related to everything else, but near things are more related than distant things." [24].

To overcome this limitation, this paper presents a multi-task learning (MTL) framework [4] for multi-location prediction in geospatio-temporal data. MTL is a widely-used approach for solving multiple, related learning tasks by exploiting the common structure of the problem. The two key properties of MTL [28] are that: (i) the prediction tasks are not identical, so fitting a single model for all tasks using their combined data may not be effective; and (ii) the prediction tasks are related, so their prediction models could benefit by sharing information across the tasks. However, identifying the relevant information to be shared among the tasks and integrating them into a unified learning framework for multi-location prediction are two key challenges that must be addressed.

To address this problem, we present a novel multi-location prediction framework called GSpartan, which stands for **G**eo**SPA**tio-tempo**R**al mul**T**i-t**A**sk lear**N**ing. GSpartan is developed based on the assumption that the prediction models for all locations share a common set of low-rank base models, where each base model may represent a macroscale phenomenon that potentially explains the variability of the data. The local model at each location is constructed as a linear combination of these base models. A graph Laplacian regularization is introduced to capture the spatial autocorrelation of the data, thus providing a natural way to specify the relationships among the prediction tasks. To ensure interpretability of the models, we also add sparsity and non-negativity constraints into the GSpartan formulation. We evaluated the performance of GSpartan against several baseline methods on climate data from 37 randomly chosen weather stations in Canada. Our experimental results for predicting monthly precipitation showed that, on average, GSpartan outperformed single-task learning (STL) and two other existing multi-task learning (MTL) methods in at least 65% of the stations. The improvement of GSpartan increases to more

*Department of Computer Science and Engineering, Michigan State University

†Department of Geograpy, Michigan State University

than 81% of the stations if the training data available at each station is limited to only 2 years.

2 Related Works

Multi-task learning (MTL) is a well-known machine learning paradigm for solving multiple, related prediction tasks simultaneously by considering their shared information [4]. The rationale for using MTL is that the information propagated between related tasks may enhance the overall accuracy if the models are trained jointly instead of independently. The MTL framework has been successfully applied to various learning problems including classification[28], regression [25][34], and clustering[17]. In general, the MTL methods can be classified into the following categories:

MTL based on Low-rank Representation. This category of methods assume that the underlying models share a low-rank representation, which is commonly used in different machine learning methods[5]. For example, Chen et al. [8][6] and Gong et al. [16] assumed that the models have a common low-rank structure as well as a group sparseness constraint that identifies the outlying tasks. Argyriou et al. [2] and Kang et al. [18] projected the features into a low dimensional space in which the predictive models for different tasks are inferred and the discriminant features are selected using an $\ell_{2,1}$ norm. Kumar et al. [19] assumed that each task model is a linear combination of base models, where the coefficients of the linear combination along with the base models are coupled in a multiplicative way. Chen et al. [7] incorporated both additive and multiplicative coupling into their formulation using a low-dimensional feature map shared across the different tasks. Although these methods incorporate low-rank representation, unlike GSpartan, they do not consider the task relationships provided by the domain (e.g., spatial proximity information in geospatio-temporal modeling).

MTL based on Explicit Task Relationship. This category of methods explicitly incorporates the task relationships of the domain into their MTL formulations. The task relationships are typically encoded in a matrix, whose elements represent the similarity between a pair of tasks[13]. For example, Zhou et al. [34] and Xu et al. [25] employed a task relationship matrix to ensure smoothness in time series prediction. Zhang et al. [31] and Saha et al. [23] proposed MTL formulations that simultaneously learn the task relationship and task-specific models. This category of methods is useful when the task relationships are clearly defined or provided by the domain.

MTL based on Shared Common Parameters. There are several ways for sharing model parameters across the tasks. For example, Evgeniou et al. [14] and Xu et al. [25] assumed the task models are composed of a common term and a task specific term, where the common term is shared across different tasks. Yu et al. [30] proposed a multi-task Gaussian process where the prior parameters are shared across different generative processes. Lee et al. [20] and Daume et al. [11] provided MTL formulations that allow for sharing of hyperparameters between distributions of different task models.

MTL based on Hybrid Information Sharing. More recently, there have been some efforts to combine some of the MTL methods described above. For example, Xu et al. [25] incorporated both common parameters and explicit task relationships into their MTL formulation. In another work, Xu et al. [26] combined low-rank representation with explicit task relations into their formulation. Their framework was designed to model the heterogeneity between patients in healthcare data. The GSpartan approach proposed in this paper extends the previous work in [26] to incorporate both task relation and low-rank representation into an MTL formulation for geospatio-temporal data.

3 Preliminaries

Let $S \subset \Re^2$ be a set of geo-referenced locations, where each location $s \in S$ is associated with a set of temporal fields. One of the fields is designated as the response variable we are interested in predicting, while the rest are considered predictor variables. For instance, in climate modeling, the response variable may correspond to monthly precipitation values recorded at a weather station whereas the predictor variables correspond to outputs generated from a global or regional climate model [9]. An example of the multi-location prediction task here is to infer future monthly values of precipitation for all the locations.

Formally, consider a geospatio-temporal data set $\mathcal{D} = \{(\mathbf{X}_1, \mathbf{y}_1), (\mathbf{X}_2, \mathbf{y}_2), \cdots, (\mathbf{X}_{|S|}, \mathbf{y}_{|S|})\}$, where each tuple, $(\mathbf{X}_s, \mathbf{y}_s)$, denote the temporal fields at location s. Let $\mathbf{X}_s \in \mathcal{R}^{n_s \times d} = [\mathbf{x}_{s,1}^T, \cdots, \mathbf{x}_{s,n_s}^T]$ be the matrix of predictor variables and $\mathbf{y}_s \in \mathcal{R}^{n_s}$ be the time series for the response variable observed at the discrete time points $1, 2, \cdots, n_s$.

For single-task learning (STL), each location s is treated as a separate learning task. Let n_s be the number of training examples available for task s and d be the number of predictor variables. STL seeks to learn a (local) task model $f_s(\mathbf{x}; \mathbf{w}_s)$ for each location in such a way that the following loss function is minimized:

$$\min_{\mathbf{W}} \sum_{s=1}^{|S|} \sum_{i=1}^{n_s} \ell_s \Big[f_s(\mathbf{x}_{s,i}; \mathbf{w}_s), y_{s,i} \Big]$$

where $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_{|S|}] \in \mathcal{R}^{d \times |S|}$ denote the model parameters and $\ell_s(\cdot)$ represents the loss function for task s. For brevity, we consider only task models of the form $f_s(\mathbf{x}_s; \mathbf{w}_s) = \mathbf{x}_s^T \mathbf{w}_s$ with a squared loss function.

4 Proposed GSpartan Framework

This section presents our proposed GSpartan framework for multi-location prediction. The framework was designed to satisfy the following three requirements:

1. It should learn a low-rank representation of the task models. The low-rank representation, defined by a set of base models, represents the possible macroscale phenomena that could help explain the temporal variability of the data.
2. It should incorporate domain knowledge about the spatial autocorrelation of the response variable among the various locations.
3. To ensure interpretability, each base model should depend only on a small subset of the predictor variables. In addition, each local model should be comprised of a small number of base models.

The following objective function is used in GSpartan to train the local models jointly:

$$\min_{\mathbf{W}} \sum_{s=1}^{|S|} \sum_{i=1}^{n_s} \ell_s \Big[f_s(\mathbf{x}_{s,i}; \mathbf{w}_s), y_{s,i} \Big] + \Omega(\mathbf{W})$$

where $\Omega(\mathbf{W})$ is a regularization term. In the following, we will discuss how to formulate the objective function to meet the requirements stated above.

Low-rank Representation: We assume that the local models can be expressed as a product of two low-rank matrices, i.e., $\mathbf{W} = \mathbf{UV}$, where $\mathbf{U} \in \mathcal{R}^{d \times k}$ and $\mathbf{V} \in \mathcal{R}^{k \times |S|}$. The matrix $\mathbf{U}$ is a feature representation of the base models while $\mathbf{V}$ expresses the weighted combination of the base models that form the local model at each location. Specifically, $\mathbf{u}_i$ is a column vector in $\mathbf{U}$ that represents the feature vector for the i-th base model, while $\mathbf{v}_j$ is a column vector in V that represents the weights of the base models defining the j-th local model.

Task Relation Matrix: We employ a graph Laplacian regularization to incorporate information about the spatial autocorrelation between locations. Let $\mathbf{A}$ be the task relation matrix, where $\mathbf{A}_{i,j}$ measures the spatial autocorrelation between locations i and j. The graph Laplacian regularizer can be written as follows:

$$\Omega_r(\mathbf{W}) = \sum_{i,j=1}^{|S|} \mathbf{A}_{i,j} \|\mathbf{w}_i - \mathbf{w}_j\|_2^2 = \mathrm{Tr}\Big[\mathbf{W}(\mathbf{D} - \mathbf{A})\mathbf{W}^T\Big]$$

where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{ij}$. Intuitively, if $\mathbf{A}_{i,j}$ is large, the graph Laplacian regularizer term will also be large unless $\mathbf{w}_i$ is similar to $\mathbf{w}_j$. Thus, the graph Laplacian is simply a re-statement of Tobler's first law of geography.

Model Interpretability: Sparsity constraints can be imposed to ensure that each base model depends only on a small subset of the predictor variables and each task model is a linear combination of a few base models. To improve interpretability, the coefficients of the weighted linear combination should also be non-negative. To satisfy these requirements, the following L_1 regularization penalty is added to the objective function:

$$\begin{aligned} & \Omega_s(\mathbf{W}) = \lambda_1 \|\mathbf{V}\|_1 + \lambda_2 \|\mathbf{U}\|_1 \\ \text{s.t.} \quad & \mathbf{W} = \mathbf{UV}, \quad \mathbf{V} \succeq 0 \end{aligned}$$

where λ_1 and λ_2 are the regularization parameters. The notation $\mathbf{V} \succeq 0$ implies all elements of $\mathbf{V}$ must be non-negative.

Putting everything together, we can now express our objective function for GSpartan (assuming a squared loss function) as follows:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{U},\mathbf{V}} \quad & \frac{1}{2} \sum_{s=1}^{|S|} \|\mathbf{X}_s \mathbf{w}_s - \mathbf{y}_s\|_2^2 + \lambda_1 \|\mathbf{V}\|_1 + \lambda_2 \|\mathbf{U}\|_1 \\ & + \frac{\lambda_3}{2} \mathrm{Tr}(\mathbf{W}(\mathbf{D} - \mathbf{A})\mathbf{W}^T) \\ \text{s.t.} \quad & \mathbf{V} \succeq 0, \ \mathbf{W} = \mathbf{UV} \end{aligned}$$

Since $\mathbf{W} = \mathbf{UV}$, the objective function reduces to the following simplified expression:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V}} \quad & \frac{1}{2} \sum_{i=1}^{|S|} \|\mathbf{X}_s \mathbf{U} \mathbf{v}_i - \mathbf{y}_s\|_2^2 + \lambda_1 \|\mathbf{V}\|_1 \\ & + \lambda_2 \|\mathbf{U}\|_1 + \frac{\lambda_3}{2} \mathrm{Tr}(\mathbf{UV}(\mathbf{D} - \mathbf{A})\mathbf{V}^T\mathbf{U}^T) \qquad (4.1) \\ \text{s.t.} \quad & \mathbf{V} \succeq 0 \end{aligned}$$

The preceding constrained optimization problem can be solved using a block coordinate descent approach, by alternately solving for $\mathbf{U}$ and $\mathbf{V}$. Details for solving each step efficiently is given below.

Solve U, given V:

When $\mathbf{V}$ is fixed, the objective function can be simplified as follows:

$$\begin{aligned} (4.2) \quad \min_{\mathbf{U}} \quad & \frac{1}{2} \sum_{i=1}^{|S|} \|\mathbf{X}_s \mathbf{U} \mathbf{v}_i - \mathbf{y}_s\|_2^2 + \lambda_2 \|\mathbf{U}\|_1 \\ & + \frac{\lambda_3}{2} \mathrm{Tr}(\mathbf{UV}(\mathbf{D} - \mathbf{A})\mathbf{V}^T\mathbf{U}^T) \end{aligned}$$

This optimization problem can be efficiently solved using the proximal gradient descent method. Proximal gradient descent is commonly used to solve optimization problems containing non-differentiable components. The algorithm also has faster convergence compared to other methods such as subgradient descent. The basic idea here is to minimize a corresponding upper bound function of the original objective function [22]. Based on standard assumptions such as Lipschitz continuity on the partial gradient of the differentiable part of the objective function, we can use a tighter upper bound to approximate the original objective function. Here, we will use *Prox-linear* [27] to solve our optimization problem, with the following update formula:

$$\begin{aligned}(4.3)\quad \mathbf{U}^k &= \underset{\mathbf{U}}{\operatorname{argmin}}(\mathbf{U}-\hat{\mathbf{U}}^{k-1})^T\hat{\mathbf{g}}_U^k \\ &\quad +\frac{\tau_U^{k-1}}{2}\|\mathbf{U}-\hat{\mathbf{U}}^{k-1}\|_F^2+\lambda_2\|\mathbf{U}\|_1,\end{aligned}$$

where

$$\begin{aligned}\hat{\mathbf{g}}_U^k &= \sum_{s=1}^{|S|}\Big(-\mathbf{X}_s^T\mathbf{y}_s\mathbf{v}_s^T+\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s\mathbf{v}_s^T\Big) \\ &\quad +\lambda_3\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\end{aligned}$$

and

$$\hat{\mathbf{U}}^{k-1}=\mathbf{U}^{k-1}+\omega_U^{k-1}(\mathbf{U}^{k-1}-\mathbf{U}^{k-2})$$

The solution for problem (4.3) is given by

$$(4.4)\qquad \mathbf{U}^k=\mathcal{S}_{\tau_U^{k-1}/\lambda_2}(\hat{\mathbf{U}}^{k-1}-\frac{\hat{\mathbf{g}}^{k-1}}{\tau_{\mathbf{U}}^{k-1}})$$

where $\mathcal{S}_\alpha(t)=\text{sign}(t)(\max(|t|-\alpha,0)$ is a component-wise soft-thresholding function.

Solve V, given U

Similarly, when $\mathbf{U}$ is fixed, the objective function becomes:

$$\begin{aligned}(4.5)\quad \min_{\mathbf{V}}\quad & \frac{1}{2}\sum_{i=1}^{|S|}\|\mathbf{X}_s\mathbf{U}\mathbf{v}_i-\mathbf{y}_s\|_2^2+\lambda_1\|\mathbf{V}\|_1 \\ &+\frac{\lambda_3}{2}\,\text{Tr}(\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\mathbf{U}^T)\end{aligned}$$

The update formula for $\mathbf{V}$ using proximal gradient descent approach is:

$$\begin{aligned}(4.6)\quad \mathbf{V}^k &= \underset{\mathbf{V}}{\operatorname{argmin}}(\mathbf{V}-\hat{\mathbf{V}}^{k-1})^T\hat{\mathbf{g}}_V^k \\ &\quad +\frac{\tau_V^{k-1}}{2}\|\mathbf{V}-\hat{\mathbf{V}}^{k-1}\|_F^2+\lambda_1\|\mathbf{V}\|_1\end{aligned}$$

where

$$\hat{\mathbf{g}}_V^k = \mathbf{P}+\lambda_3\mathbf{U}^T\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A})$$

The s-th column of matrix $\mathbf{P}$ is given by

$$\mathbf{p}_s=-\mathbf{U}^T\mathbf{X}_s^T\mathbf{y}_s+\mathbf{U}^T\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s$$

and

$$\hat{\mathbf{V}}^{k-1}=\mathbf{V}^{k-1}+\tau_V^{k-1}(\mathbf{V}^{k-1}-\mathbf{V}^{k-2})$$

The solution for problem (4.6) is given by

$$(4.7)\qquad \mathbf{V}^k=\mathcal{S}_{\omega_V^{k-1}/\lambda_2}(\hat{\mathbf{V}}^{k-1}-\frac{\hat{\mathbf{g}}^{k-1}}{\tau_V^{k-1}})$$

A further projection step is needed to ensure that the elements of $\mathbf{V}$ are non-negative.

Note that the learning rates τ_U^{k-1} and τ_V^{k-1} are provided by the Lipschitz continuous constant of the partial gradient $\hat{\mathbf{g}}_U^k$ and $\hat{\mathbf{g}}_V^k$, respectively. The learning rates are given in Theorems 1 and 2 below. The extrapolation term ω is selected to be $0\le\omega^k\le\delta_\omega\sqrt{\frac{\tau^{k-2}}{\tau^{k-1}}}$ for $\delta_\omega<1$.

THEOREM 1. *The partial gradient $\hat{\mathbf{g}}_U$ is Lipschitz continuous with the constant*

$$\tau_U = \sum_{s=1}^{|S|}\|\mathbf{X}_s^T\mathbf{X}_s\|\|\mathbf{v_s}\mathbf{v_s^T}\|+\lambda_3\|\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\|$$

Proof. For any U and $U^*\in\mathcal{R}^{d\times k}$,

$$\begin{aligned}&\|\hat{\mathbf{g}}_U-\hat{\mathbf{g}}_{U^*}\| \\ =\;&\Big\|\sum_{s=1}^{|S|}\big(\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s\mathbf{v}_s^T-\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}^*\mathbf{v}_s\mathbf{v}_s^T\big) \\ &+\lambda_3(\mathbf{U}-\mathbf{U}^*)\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\Big\| \\ \le\;&\sum_{s=1}^{|S|}\|\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\mathbf{v}_s\mathbf{v}_s^T-\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}^*\mathbf{v}_s\mathbf{v}_s^T\| \\ &+\lambda_3\|\mathbf{U}-\mathbf{U}^*\|\|\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\| \\ \le\;&\sum_{s=1}^{|S|}\|\mathbf{X}_s^T\mathbf{X}_s\|\|\mathbf{v_s}\mathbf{v_s^T}\|\|\mathbf{U}-\mathbf{U}^*\| \\ &+\lambda_3\|\mathbf{U}-\mathbf{U}^*\|\|\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\| \\ =\;&\|\mathbf{U}-\mathbf{U}^*\|\left(\sum_{s=1}^{|S|}\|\mathbf{X}_s^T\mathbf{X}_s\|\|\mathbf{v_s}\mathbf{v_s^T}\|+\lambda_3\|\mathbf{V}(\mathbf{D}-\mathbf{A})\mathbf{V}^T\|\right)\end{aligned}$$

THEOREM 2. *Assuming $\|\mathbf{X}_s\|\le R$, the partial gradient $\hat{\mathbf{g}}_V$ is Lipschitz continuous with the constant*

$$\tau_V = \|\mathbf{U}\|^2R^2+\lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|$$

Proof. For any $\mathbf{V}$ and $\mathbf{V}^* \in \mathcal{R}^{k\times S}$,

$$\begin{aligned}
& \|\hat{\mathbf{g}}_V - \hat{\mathbf{g}}_{V^*}\| \\
= & \Big\|\mathbf{P} + \lambda_3\mathbf{U}^T\mathbf{U}\mathbf{V}(\mathbf{D}-\mathbf{A}) - \mathbf{P}^* \\
& \qquad - \lambda_3\mathbf{U}^T\mathbf{U}\mathbf{V}^*(\mathbf{D}-\mathbf{A})\Big\| \\
\leq & \|\mathbf{P}-\mathbf{P}^*\| + \lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|\|\mathbf{V}-\mathbf{V}^*\| \\
= & \sum_{s=1}^{|S|}\|\mathbf{U}^T\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}(\mathbf{v}_s-\mathbf{v}_s^*)\| \\
& +\lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|\|\mathbf{V}-\mathbf{V}^*\| \\
\leq & \sum_{s=1}^{|S|}\|\mathbf{U}^T\mathbf{X}_s^T\mathbf{X}_s\mathbf{U}\|\|\mathbf{v}_s-\mathbf{v}_s^*\| \\
& +\lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|\|\mathbf{V}-\mathbf{V}^*\| \\
\leq & \|\mathbf{U}\|^2R^2\sum_{s=1}^{|S|}\|\mathbf{v}_s-\mathbf{v}_s^*\| \\
& +\lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|\|\mathbf{V}-\mathbf{V}^*\| \\
= & \|\mathbf{V}-\mathbf{V}^*\|\left(\|\mathbf{U}\|^2R^2+\lambda_3\|\mathbf{U}^T\mathbf{U}\|\|\mathbf{D}-\mathbf{A}\|\right)
\end{aligned}$$

A summary of the GSpartan framework is shown in Algorithm 1 below.

Input: Dataset $\mathcal{D} = \{(\mathbf{X}_1, \mathbf{y}_1), ..., (\mathbf{X}_S, \mathbf{y}_S)\}$, Task relation matrix A, parameters λ_1, λ_2, λ_3;
Initialize: Randomly generate $\mathbf{U}$ and $\mathbf{V}$ and set $k = 1$
Block coordinate descent:
while *not converge* **do**
  Solve U given V:
  Compute τ_U^k using Theorem 1
  Update $\mathbf{U}^k$ using Equation (4.4)
  Solve V given U:
  Compute τ_V^k using Theorem 2
  Update $\mathbf{V}^k$ using Equation (4.7)
  $k = k + 1$
end
return $\{U^k, V^k\}$

Algorithm 1: Pseudocode for GSpartan framework

THEOREM 3. *Let* $\{\mathbf{U}^k, \mathbf{V}^k\}$ *be the sequence generated by Algorithm 1 with* $0 \leq \omega^k \leq \delta_\omega\sqrt{\frac{\tau^{k-2}}{\tau^{k-1}}}$ *for* $\delta_\omega < 1$. *Then the sequence of* $\{\mathbf{U}^k, \mathbf{V}^k\}$ *will converge.*

The proof of convergence given by Theorem 3 can be found in Lemma 2.2 of [27].

5 Experimental Evaluation

This section presents the experiments performed to evaluate the effectiveness of the proposed GSpartan framework.

5.1 Dataset Description We evaluated the performance of GSpartan on climate data from 37 randomly chosen weather stations in Canada[1]. We use monthly precipitation data from the weather stations as the response variable. The precipitation data spans a 40-year period from January, 1961 to December, 2000. The predictor variables for building the local models were obtained from NCEP-reanalysis[2] data, which is a coarse-scale global environmental data that integrates observations with output from a numerical weather prediction model. There are 26 predictor variables, including mean temperature at 2 meters, mean sea level pressure, 500 hPa geopotential height, and near surface relative humidity[3]. We deseasonalize the precipitation time series by subtracting each monthly values with the average value for that month over the entire 40 year period. We then create 10 versions of the training set for each location by varying the length of the training period from 2 to 11 years. For example, the first version uses monthly precipitation data from 1961 and 1962 for training and the remaining 38 years for testing while the last version uses the first 11 years of monthly precipitation for training and the remaining 29 years for testing.

5.2 Baseline Methods We compared the performance of GSpartan against the following baseline.

- **LASSO**: We applied LASSO regression to the data set at each location independently. The Lasso results serve as a baseline for single-task learning.

- **MRMTL**: The mean regularized MTL (MRMTL) is an algorithm developed in [14] based on the assumption of shared common parameters among task models. Specifically, the objective function of MRMTL is given by

$$\min_W \sum\nolimits_{s=1}^{S}\|X_s\mathbf{w}_s - \mathbf{y}_s\|_2^2 + \rho_1\|W\|_1 + \rho_2\sum\nolimits_{s=1}^{S}\left\|\mathbf{w}_s - \frac{1}{S}\sum\nolimits_{i=1}^{S}\mathbf{w}_i\right\|_2^2$$

  We use the MRMTL implementation given in the MALSAR software package [33]. Note that instead of using ℓ_2 norm on W as the original paper in [14], we use ℓ_1 norm on W for a fair comparison.

[1] http://climate.weather.gc.ca/
[2] http://www.cccsn.ec.gc.ca/?page=pred-hadcm3
[3] A complete list of the features and their description is available at http://www.cccsn.ec.gc.ca/?page=pred-help

- **SLMTL**: This is an MTL algorithm proposed in [6], which assumes that the tasks are related using an incoherent rank-sparsity structure. Unlike GSpartan, SLMTL does not explicitly consider the relationships among tasks (e.g., spatial autocorrelation between locations). The objective function for SLMTL is given by [6],

$$\min_{W} \sum_{s=1}^{S} \|X_s \mathbf{w}_s - \mathbf{y}_s\|_2^2 + \gamma \|P\|_1$$
$$\text{s.t. } W = P + Q, \|Q\|_* \leq \tau$$

  where $W \in \mathcal{R}^{d \times S}$ and $W = [\mathbf{w}_1, ..., \mathbf{w}_S]$. We use the SLMTL implementation provided by the MALSAR software package [33] for our experiments.

In addition to the three baseline algorithms, we also investigate the following two variants of GSpartan.

- **GSpartan-NTR**: In this variant, we remove the graph Laplacian regularizer from the objective function given in Equation (4.1). This allows us to evaluate the importance of incorporating spatial autocorrelation into the framework.

$$\min_{W,U,V} \quad \tfrac{1}{2} \sum_{s=1}^{S} \|X_s \mathbf{w}_s - \mathbf{y}_s\|_2^2 + \lambda_1 \|V\|_1 + \lambda_2 \|U\|_1$$
$$s.t. \qquad V \succeq 0, W = UV$$

- **GSpartan-norm**: In [32] a normalized graph Laplacian regularizer was used to facilitate transfer of information:

$$\sum_{i,j=1}^{S} A_{i,j} \|\mathbf{w}_i / \sqrt{D_{i,i}} - \mathbf{w}_j / \sqrt{D_{j,j}}\|_2^2.$$

  We will compare the normalized graph Laplacian against the unnormalized one used in GSpartan.

5.3 Task relationship matrix We use the inverse of a modified variogram measure to estimate the spatial autocorrelation between locations. Variogram is a measure developed in spatial statistics to determine the spatial dependence between a pair of locations [10]. The measure is computed based on the variance of the difference in field values for two locations:

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \frac{1}{var(\mathbf{y}_i - \mathbf{y}_j)} & \text{otherwise} \end{cases}$$

where $var(z)$ denote variance of z. Since we have time series data at each location, we compute $var(\mathbf{y}_i - \mathbf{y}_j)$ using monthly precipitation from the first year (1961).

5.4 Experimental Results We evaluate the performance of various methods on the test set in terms of their root-mean-square-error (RMSE):

$$RMSE = \sqrt{\sum_{i}^{n} (y_i - \hat{y}_i)^2 / n}$$

where n is the number of test points. We set $\lambda_1 = 0.001$, $\lambda_2 = 0.001$, $\lambda_3 = 0.1$ and $k = 5$ as the default parameters for GSpartan on all datasets. Figure 1 compares the RMSE for different methods when applied to the first version of the data set (which has 2 years of data for training and 38 years of data for testing). The horizontal axis of the plot corresponds to indices for the 37 weather stations. The results suggest that GSpartan outperforms other baselines for most of the stations. In fact, looking at Table 1, which summarizes the number of wins achieved by each method compared to others, GSpartan outperforms other baselines in at least 24 (65%) out of 37 stations. Furthermore, by comparing LASSO against other methods, we observe that MTL is generally better than STL especially when there are limited training data available.

To investigate the strength of GSpartan, Figure 2 compares its RMSE against other variants of GSpartan. Observe that GSpartan performs no worse than its variants for most of the stations.Furthermore, the results in Table 1 suggest that GSpartan outperforms GSpartan -NTR in 36 out of 37 stations, which demonstrates the importance of incorporating spatial autocorrelation into the geospatio-temporal MTL framework. In addition, since GSpartan-NTR outperforms other MTL methods in at least 23 stations, this shows the importance of using low-rank representation for modeling the data.

The previous results were obtained using a data set with limited training examples (2 years for training and 38 years for testing). We next investigate the relative performance of GSpartan against other methods as the training set size increases from 2 to 11 years. Specifically, we compare the percentage of stations in which

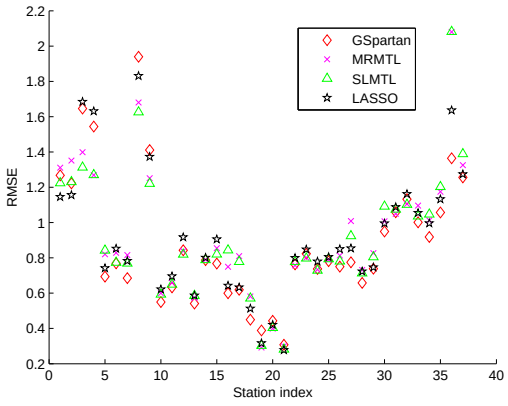


Figure 1: Comparison of GSpartan against three baseline methods

Table 1: Win-loss table comparing performance of various methods when applied to the data set with limited training examples (2 years of training data and 38 years of test data).

	GSpartan	GSpartan-norm	GSpartan-NTR	MRMTL	SLMTL	LASSO
GSpartan	-	36	35	24	25	30
GSpartan-norm	1	-	8	23	24	30
GSpartan-NTR	2	29	-	23	24	30
MRMTL	13	14	14	-	14	20
SLMTL	12	13	13	23	-	23
LASSO	7	7	7	17	14	-

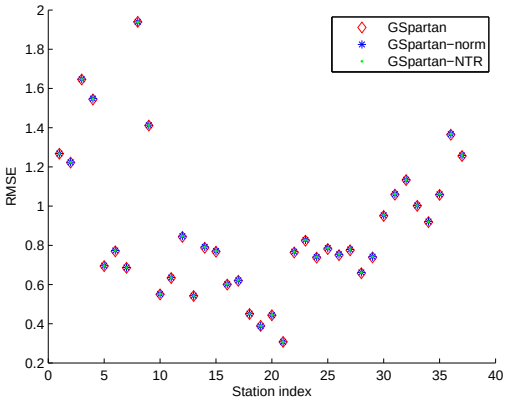


Figure 2: Comparison of GSpartan against its variants

the RMSE for GSpartan is lower than that for other methods. For example, if GSpartan outperforms another method in 32 out of 37 stations, the ratio is 0.865. Figure 3 shows the results when the training set size increases. The horizontal axis of the plot corresponds to the index of the data set (which is equivalent to number of training years), while the vertical axis corresponds to the GSpartan outperform ratio. The result shows that, on average, GSpartan outperforms the baseline methods for 65% of the stations, and increases to 81% when there are fewer training examples. This confirms our hypothesis that GSpartan can effectively train local models when there are limited training examples.

In addition, by comparing GSpartan with GSpartan-NTR, we can see that incorporating spatial autocorrelation enhances the performance of GSpartan irrespective of the training set size. By comparing GSpartan against GSpartan-norm, we also see that a normalized graph Laplacian regularizer indeed degrades the performance of GSpartan . This is because the normalization attenuates the spatial autocorrelation among the tasks, which causes the task relationship to be ineffective. Finally, comparing the results for GSpartan against the two baseline MTL algorithms, it appears that when the training set is small, GSpartan outperforms both MRMTL and SLMTL. However, with increasing training set size, both MRMTL and SLMTL perform better than GSpartan. One possible explanation is that, when there are enough labeled examples available at each station, incorporating the spatial autocorrelation information (which was computed using the first year training data only) might adversely affect the local models.

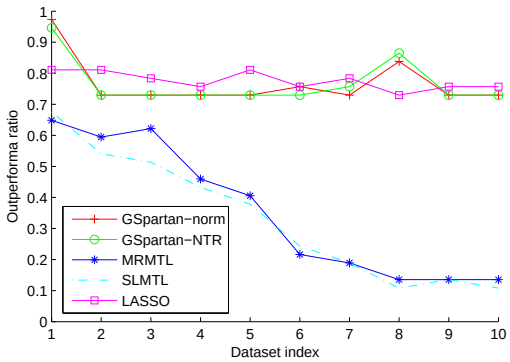
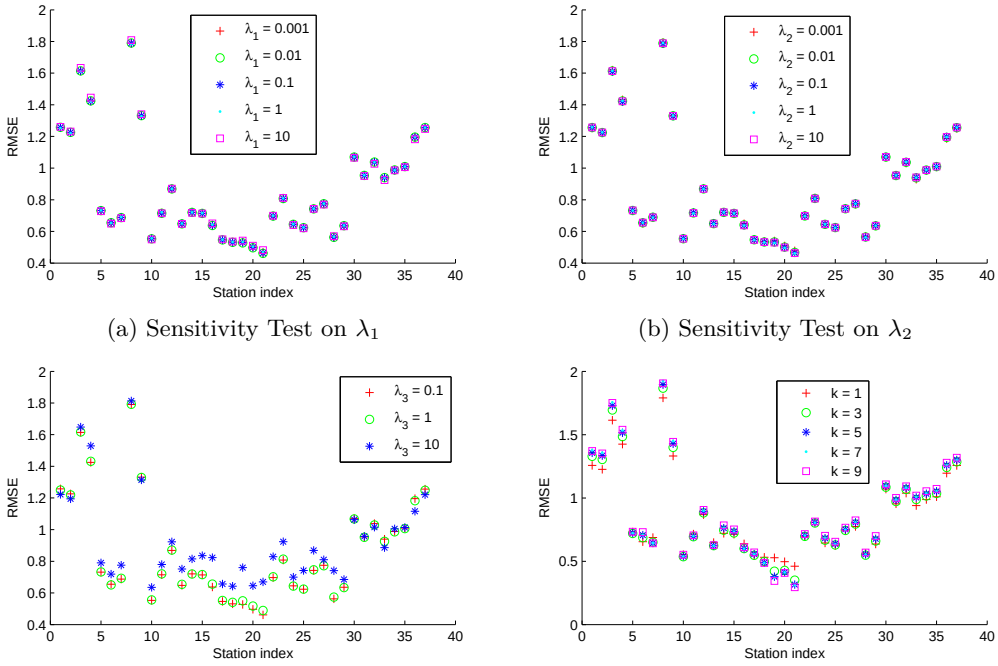


Figure 3: Performance comparison between GSpartan against baseline methods as training set size increases.

5.5 Sensitivity Analysis Since there are four parameters (λ_1, λ_2, λ_3 and k) that must be tuned in GSpartan, this subsection analyzes the performance of the framework as each parameter is varied. For this experiment, we use create a data set with 30 years of training and 10 years of testing. The results using other data sets are quite similar, so we omit them due to lack of space. Figure 4 shows the results of our experiment. The horizontal axis corresponds to each index location while the vertical axis represents RMSE values. The results from the figure show that GSpartan is not sensitive to changes in λ_1 and λ_2 for all 37 locations (see Figures 4a and 4b). Furthermore, Figure 4c showed that smaller values of λ_3 should be preferred. Figure 4d showed that GSpartan is not that sensitive to the number of base models, k, for many locations. However, for those locations where k is sensitive, a small value of k tends to produce lower RMSE.



(a) Sensitivity Test on λ_1

(b) Sensitivity Test on λ_2

(c) Sensitivity Test on λ_3

(d) Sensitivity Test on k

Figure 4: Results of sensitivity analysis on λ_1, λ_2, λ_3, and k for GSpartan. The horizontal axis represents the index of a station and the vertical axis corresponds to the RMSE value.

6 Conclusion

This paper presents a novel geospatio-temporal multi-task learning framework called GSpartan for multi-location prediction. GSpartan assumes that the local models share a common low-rank representation. The framework also enables domain-specific constraints such as spatial autocorrelation to be integrated into its formulation. Experimental results on a real world climate data set showed that the proposed framework outperformed other baseline algorithms especially when there are limited training examples available at each location.

For future work, we plan to correlate the discovered base models against some of the known climate phenomena. In addition, to improve its scalability, we will extend the methodology to an online multi-task learning setting. Finally, we will also investigate techniques to extend the approach to multivariate response predictions, in which the response variables at a location are potentially correlated (e.g., maximum or minimum temperature, humidity, and precipitation).

Acknowledgment

This research is partially supported by NOAA Climate Program office through grant #NA12OAR4310081, NASA Terrestrial Hydrology Program through grant #NNX13AI44G. and Office of Naval Research through grant N00014-14-1-0631.

References

[1] Z. Abraham, M. Liszewska, Perdinan, P.-N. Tan, J. Winkler, and S. Zhong. Distribution regularized regression framework for climate modeling. In *Proceedings of SIAM International Conference on Data Mining*, SDM '13, pages 333–341. SIAM, 2013.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, Dec. 2008.

[3] N. H. Augustin, M. Musio, K. von Wilpert, E. Kublin, S. N. Wood, and M. Schumacher. Modeling spatiotemporal forest health monitoring data. *Journal of the American Statistical Association*, 104:899–911, 2009.

[4] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997.

[5] S. Chang, G.-J. Qi, C. Aggarwal, J. Zhou, M. Wang, and T. Huang. Factorized similarity learning in networks. In *Proceedings of the 14th IEEE International Conference on Data Mining*, ICDM '14, pages 917–926, 2014.

[6] J. Chen, J. Liu, and J. Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data*, 5(4):22:1–22:31, 2012.

[7] J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning a shared predictive structure from multiple tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1025–1038, May 2013.

[8] J. Chen, J. Zhou, and J. Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *Proceedings of the 17th ACM SIGKDD Int'l Conf on Knowledge discovery and data mining*, pages 42–50. ACM, 2011.

[9] H. Cheng and P.-N. Tan. Semi-supervised learning with data calibration for long-term time series forecasting. In *Proc of ACM SIGKDD Int'l Conf on Knowledge Discovery and Data Mining*, pages 133–141, 2008.

[10] N. Cressie. *Statistics for spatial data*. Wiley, New York, 1993.

[11] H. Daumé, III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 135–142, 2009.

[12] G. Davis, N. Sevdalis, and L. Drumright. Spatial and temporal analyses to investigate infectious disease transmission within healthcare settings. pages 227–243, 2014.

[13] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, Dec. 2005.

[14] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 109–117, 2004.

[15] J. Faghmous and V. Kumar. Spatio-temporal data mining for climate data: Advances, challenges, and opportunities. In *Data Mining and Knowledge Discovery for Big Data*, volume 1 of *Studies in Big Data*, pages 83–116. Springer Berlin Heidelberg, 2014.

[16] P. Gong, J. Ye, and C. Zhang. Robust multi-task feature learning. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 895–903, 2012.

[17] Q. Gu, Z. Li, and J. Han. Learning a kernel for multi-task clustering. In *Proceedings of the 25th Conference on Artificial Intelligence (AAAI)*, 2011.

[18] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the 28th International Conference on Machine Learning*, ICML '11, pages 521–528. Omnipress, 2011.

[19] A. Kumar and H. D. III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Conference on Machine Learning*, ICML '12. icml.cc / Omnipress, 2012.

[20] S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 489–496, 2007.

[21] D. Neill. New directions in artificial intelligence for public health surveillance. *Intelligent Systems, IEEE*, 27(1):56–59, 2012.

[22] N. Parikh and S. Boyd. Proximal algorithms. *Found. Trends Optim.*, 1(3):127–239, Jan. 2014.

[23] A. Saha, P. Rai, H. D. III, and S. Venkatasubramanian. Online learning of multiple tasks and their relationships. In *Proceeding of the 14th international conference on Artificial Intelligence and Statistics*, volume 15 of *AISTATS '11*, pages 643–651. JMLR.org, 2011.

[24] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2):234–240, 1970.

[25] J. Xu, P.-N. Tan, and L. Luo. ORION: Online Regularized multI-task regressiON and its application to ensemble forecasting. In *Proceedings of the 14th IEEE International Conference on Data Mining*, ICDM '14.

[26] J. Xu, J. Zhou, and P.-N. Tan. Formula: Factorized multi-task learning for task discovery in personalized medical models. In *Proceedings of the 15th SIAM International Conference on Data Mining*, SDM '15, pages 496–504, 2015.

[27] Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal of Imaging Sciences*, 6:1758–1789, 2013.

[28] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, May 2007.

[29] H. Yang, S. Parthasarathy, and S. Mehta. A generalized framework for mining spatio-temporal patterns in scientific data. In *Proc of ACM SIGKDD Int'l Conf on Knowledge Discovery and Data Mining*, pages 716–721, 2005.

[30] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 1012–1019, 2005.

[31] Y. Zhang and D.-Y. Yeung. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, UAI '10.

[32] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Proceedings of the 16th Advances in Neural Information Processing Systems*, pages 321–328. MIT Press, 2004.

[33] J. Zhou, J. Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University, 2011.

[34] J. Zhou, L. Yuan, J. Liu, and J. Ye. A multi-task learning formulation for predicting disease progression. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 814–822, 2011.