Task-Adaptive Meta-Learning Framework for Advancing Spatial Generalizability

Zhexiong Liu[1], Licheng Liu[2], Yiqun Xie[3], Zhenong Jin[2], Xiaowei Jia[1]

[1]Department of Computer Science, University of Pittsburgh, Pennsylvania 15260 USA
[2] The University of Minnesota, Twin City, Minnesota 55108 USA
[3] The University of Maryland, College Park, Maryland 20742 USA
{zhexiong.liu, xiaowei}@pitt.edu, {lichengl, jinzn}@umn.edu, xie@umd.edu

Abstract

Spatio-temporal machine learning is critically needed for a variety of societal applications, such as agricultural monitoring, hydrological forecast, and traffic management. These applications greatly rely on regional features that characterize spatial and temporal differences. However, spatio-temporal data often exhibit complex patterns and significant data variability across different locations. The labels in many real-world applications can also be limited, which makes it difficult to separately train independent models for different locations. Although meta learning has shown promise in model adaptation with small samples, existing meta-learning methods remain limited in handling a large number of heterogeneous tasks, e.g., a large number of locations with varying data patterns. To bridge the gap, we propose task-adaptive formulations and a model-agnostic meta-learning framework that transforms regionally heterogeneous data into location-sensitive meta tasks. We conduct task adaptation following an easy-to-hard task hierarchy in which different meta models are adapted to tasks of different difficulty levels. One major advantage of our proposed method is that it improves the model adaptation to a large number of heterogeneous tasks. It also enhances the model generalization by automatically adapting the meta model of the corresponding difficulty level to any new tasks. We demonstrate the superiority of our proposed framework over a diverse set of baselines and state-of-the-art meta-learning frameworks. Our extensive experiments on real crop yield data show the effectiveness of the proposed method in handling spatial-related heterogeneous tasks in real societal applications.

Introduction

The explosive growth of spatio-temporal data emphasizes the needs for automatically discovering spatial-related knowledge (Shekhar et al. 2003). Spatio-temporal data are complex due to inherent data characteristics such as implicit spatial relationships between variables and the data variability across locations (Huang et al. 2018; Zheng et al. 2020; Huang et al. 2020). For example, Figure 1 shows the normalized average corn yield for every county in the Midwestern United States. The yield data exhibit a strong spatial variability due to the variation in weather, soils, and management practices across different counties. Hence, a global model trained over large regions may not perform well for every individual county (Karpatne et al. 2018). In addition, it is often expensive to collect a large number of labeled samples in real-world societal applications, which makes it challenging to train advanced deep neural network models separately for each location. Therefore, the development of effective machine learning techniques for spatial-related tasks with strong spatial variability is urgently needed.

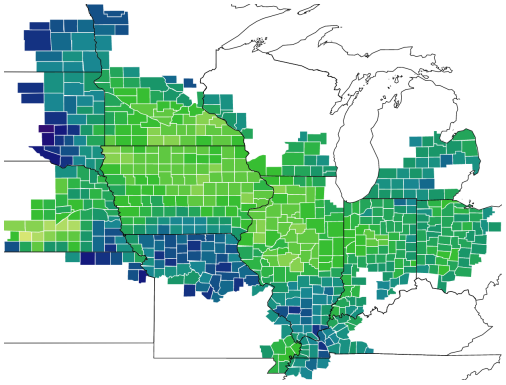


Figure 1: Normalized average corn yields in 21 consecutive years from 2000 to 2020 across 630 counties in Illinois, Wisconsin, Minnesota, Iowa, Missouri, Indiana, Ohio, Michigan, North Dakota, South Dakota, Nebraska, Kansas, Kentucky, and Tennessee in the United States. Dark blue means low-yield regions while light green represents high-yield regions. Geographically, corn yields are highly correlated to spatial locations that are complex to model with contemporary machine learning methods.

Transfer learning methods have been widely explored for adapting machine learning models across space. For example, previous works use a Long-Short Term Memory (LSTM) structure with the attention mechanism to transfer spatial-related information (Nigam et al. 2019; Sharma, Rai, and Krishnan 2020; Jiang et al. 2022). However, these methods directly learn on global data but do not consider regional discrepancy across space. Recent deep learning-based domain adaptation approaches (Nevavuori, Narra, and Lipping 2019; Elavarasan and Vincent 2020) have demonstrated success on several tasks when trained with sufficiently labeled

data; however, their performance can be degraded given limited labeled data in regression tasks.

Few-shot learning has shown promise in reducing the need for large labeled samples. Nevertheless, standard few-shot learning methods often perform worse if data are from heterogeneous distributions, which is a common issue in real spatial datasets. Meta learning addresses this issue through the idea of task-adaptive learning. Specifically, meta learning aims to extract meta knowledge from multiple training tasks, which can then be used to facilitate task-adaptive learning for a single task using a small number of data samples. Meta learning has shown encouraging results in many important societal problems, such as agricultural monitoring and traffic management (Pan et al. 2020; Li, Zhang, and Huang 2020; Tseng et al. 2021). Existing meta-learning methods can be categorized based on how they leverage meta knowledge in new tasks, e.g., the optimization-based methods (Finn, Abbeel, and Levine 2017; Li et al. 2017; Antoniou, Edwards, and Storkey 2019), the feed-forward model-based methods (Mishra et al. 2018; Qiao et al. 2018), and metric-learning-based methods (Sung et al. 2018; Willard et al. 2021). For example, the Model-Agnostic Meta Learning (MAML) algorithm (Finn, Abbeel, and Levine 2017) aims to learn an initial model (i.e., meta-model) that can be quickly adapted to new tasks. However, most existing meta-learning methods have limits in handling a large number of heterogeneous tasks, e.g., modeling data from a large number of locations with non-stationary relationships between input and output variables. This can be a common issue in many societal applications. For example, the variation of weather and soils over space interact with the complex carbon, nitrogen, and water cycles during crop growth, which ultimately leads to a strong variability in crop yield patterns.

In this paper, we develop a task-adaptive meta-learning framework by adapting the predictive model gradually over space via a "spatialized" easy-to-hard task hierarchy. In particular, we first train a standard MAML model by considering each location as a separate task. Then we iteratively split the set of tasks to create new branches of harder tasks. Moreover, we synchronously transform the meta-learning model following the obtained easy-to-hard task hierarchy. Given a new task, we can first identify its difficulty level and then adapt the meta-model from the corresponding layer of the hierarchy to the new task. Our contributions can be summarized as follows:

- We create the first meta-learning method that uses spatial-related tasks in crop yield prediction, which is critical for ensuring food supply and estimating farmers' insurance and subsidies;
- We propose a new meta-learning strategy to learn different difficulty levels of tasks in an easy-to-hard hierarchy that can be quickly adapted to new tasks;
- We extend existing meta-learning methods to handle a large number of heterogeneous tasks;
- Our evaluation on real crop yield data over large regions show the superiority of our proposed approach over standard machine learning and meta-learning baselines.

Related Work

Few-shot Meta Learning

Few-shot learning has been widely adopted for addressing real-world small data problems due to its great diversity and feasibility (Thrun and Pratt 1998; Finn, Abbeel, and Levine 2017; Wang et al. 2020). Typically, few-shot learning has gained attention in three fields: (1) metric learning-based methods that learn a similarity space, which helps build the connections between new few-shot examples with existing data (Vinyals et al. 2016; Snell, Swersky, and Zemel 2017; Jiang et al. 2020; Matsumi and Yamada 2021); (2) memory network-based methods that learn to gain experience in training, and generalize learned knowledge to unseen tasks (Santoro et al. 2016; Munkhdalai and Yu 2017; Zhao et al. 2021); (3) Gradient descent-based meta-learning methods that learn to adapt a specific base-learner to few-shot examples from different tasks. For example, MAML (Finn, Abbeel, and Levine 2017) uses a meta learner to find the optimal initialization for a base learner and adapts it to new learning tasks with a few training samples. However, existing MAML-based methods have degraded performance on a large number of heterogeneous tasks such as spatial-related tasks with high data variability across different locations.

Multi-task Learning

Multi-task learning (MTL) aims to learn shared representations jointly from multiple training tasks (Caruana 1997). It assumes the shared information across different tasks can be leveraged to improve the overall performance in all tasks (Zhang and Yang 2018; Ma, Du, and Matusik 2020). These approaches assume that such shared representations could transfer to other tasks, such as object detection (Zhang et al. 2014; Li et al. 2016), image segmentation (Kendall, Gal, and Cipolla 2018), multi-lingual machine translation (Dong et al. 2015; Zhou et al. 2019) and understanding (Liu et al. 2019; Wu, Zhang, and Ré 2020). However, the spatial-related tasks cannot be directly learned with the multi-task objectives. This is because location-based data (e.g., crop yields across the United States) have significantly different distributions based on their geographic features. In addition, tasks are relatively independent in real scenario problems thus unable to jointly learn an overall model that benefits every task. A potential solution is to explore task-based feature relations (Xie et al. 2021; Zhao et al. 2020); however, it requires sufficient labels thus has limitations in many real applications.

Methods

Preliminaries

In this section, we introduce the notation and definition used in the crop yield prediction problem. This is essentially a regression task that involves real spatio-temporal data collected in the Midwestern United States. The inputs are an array of daily-collected time-series features, including weather and soil conditions and plant property. The target variable is the county-level crop yield (for corns) at a yearly scale. Specifically, let $x_i \in \mathcal{R}^{s \times t}$ denote input features and $y_i \in \mathcal{R}$ denote crop yield label of the U.S. county

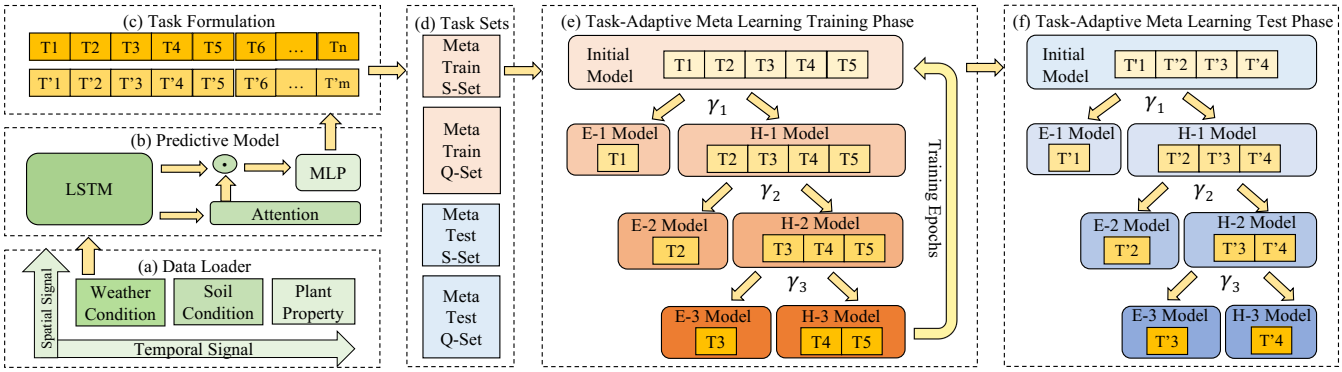


Figure 2: The framework of task-adaptive MAML with an easy-to-hard hierarchy. (a) The weather and soil condition and plant property data that has both spatial and temporal distributions are loaded to (b) train a predictive model. (c) Meta-learning tasks are formulated as county-level crop yield predictions, of which every meta task is trained with the predictive model. (d) The tasks are split into meta training and meta test sets based on their spatial (county) locations, of which the meta training and test sets are split into a support set (S-Set) and a query set (Q-Set) based on their temporal (year) information, respectively. (e) In the adaptive training phase, the Train S-Set is used to train an initial model (that is either a pretrained predictive model or a hard model in the bottom task layer obtained from the last epoch training) and the trained meta model is evaluated on Train Q-Set using metric R^2. An array of task-specific R^2 is used as the input of Algorithm 2 that returns an easy-hard task splitting threshold (γ). In the first task layer ($r = 0$), the task T_1 has R^2 greater than the threshold ($\gamma = \gamma_1$) thus is used to train an easy (E-1) model, otherwise, the tasks (T_2 to T_5) are used to train a hard (H-1) model. The adaptive training will run multiple splitting iterations until the maximum split number is reached (e.g., 3 splits). The splitting thresholds (γ) will update in each epoch training as shown in Table 3. (f) In the adaptive test phase, input tasks (T'_1 to T'_4) are split using the thresholds (γ) updated in the training, and apply the same splitting strategies as training to obtain its best adaptive meta model.

i, where $i \in \{1, 2 \ldots n\}$, s is the spatial dimension (i.e., the number of locations sampled from each county), and t is the temporal dimension. Let $T_i = (X_i, Y_i)$ denote the crop yield prediction task of county i with the input data $(X_i, Y_i) = \left\{\left(x_p^{(i)}, y_p^{(i)}\right); \left(x_q^{(i)}, y_q^{(i)}\right)\right\}$, $p \in \{1, 2, \ldots, k\}$, and $q \in \{1, 2 \ldots, l\}$. Here $\left(x_p^{(i)}, y_p^{(i)}\right)$ is a task-specific training sample from the Train Support Set (Train S-Set) of size k, and $\left(x_q^{(i)}, y_q^{(i)}\right)$ is a validation sample from the Train Query Set (Train Q-Set) of size l, which is reserved for evaluating the training task performance.

MAML (Finn, Abbeel, and Levine 2017) aims to solve meta-learning problems by optimizing the adaptability of the meta model $\mathcal{F}_\theta$. It learns parameter θ over a set of training tasks $\mathcal{T}_{train}$ where $\mathcal{T}_{train} = \{T_1, T_2, T_3, ..., T_n\}$, such that the learned meta model $\mathcal{F}_\theta$ is able to quickly solve new tasks $T'_j \in \mathcal{T}_{test}$ by slightly fine-tuning $\mathcal{F}_\theta$ with a small amount of task-specific samples (X'_j, Y'_j). Here $\mathcal{T}_{test} = \{T'_1, T'_2, T'_3, ..., T'_m\}$, and $(X'_j, Y'_j) = \left\{\left(x'^{(j)}_{p'}, y'^{(j)}_{p'}\right); \left(x'^{(j)}_{q'}, y'^{(j)}_{q'}\right)\right\}$, $p' = 1, 2, \ldots, k', q' = 1, 2 \ldots, l'$. $\left(x'^{(j)}_{p'}, y'^{(j)}_{p'}\right)$ is a sample used to fine-tune the learned meta model from the Test Support Set (Test S-Set), and $\left(x'^{(j)}_{q'}, y'^{(j)}_{q'}\right)$ is the evaluation sample for the performance in the Test Query Set (Test Q-Set).

Specifically, we train $\mathcal{F}_\theta$ on the task $T_i \in \mathcal{T}_{train}$ with gradient descent optimization

$$\theta_i \leftarrow \theta - \alpha \nabla \mathcal{L}_{T_i}(\mathcal{F}_\theta), \tag{1}$$

$$\mathcal{L}_{T_i}(\mathcal{F}_\theta) = \frac{1}{k} \sum_{p=1}^{k} \ell\left(\mathcal{F}_\theta\left(x_p^{(i)}\right), y_p^{(i)}\right), \tag{2}$$

where $\mathcal{L}_{T_i}(\mathcal{F}_\theta)$ is the task-related training (outer) loss, α is a meta learning rate, and ℓ is the associated (inner) loss (e.g., mean squared loss, MSE). In the adaptation stage, MAML optimizes θ such that the following meta loss is minimized using the task-wise fine-tuned parameter θ_i over validation samples of each training task $\left(x_q^{(i)}, y_q^{(i)}\right)$:

$$\min_\theta \mathcal{L}_{\text{MAML}}(\mathcal{F}_\theta) = \frac{1}{l} \sum_{i=1}^{l} \ell\left(\mathcal{F}_{\theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(\mathcal{F}_\theta)}\left(x_q^{(i)}\right), y_q^{(i)}\right). \tag{3}$$

The meta parameter θ is then updated by gradient descent $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}_{\text{MAML}}(\mathcal{F}_\theta)$. The learned meta model $\mathcal{F}_\theta$ can be used to fine-tune a new task $T'_j \in \mathcal{T}_{test}$ through Eq. 1.

Predictive Model

In this section, we introduce the LSTM-Attention network (Xu et al. 2020) for the corp yield prediction, as shown in Figure 2 (b). The inputs are fed to an LSTM layer to learn hidden states and their attentions. The outputs are model predictions learned by a multi-layer perception. In particular, the LSTM module trains on the spatial and temporal input features to learn its weights by computing multiple gates (input gate i, forget gate f, and output gate o) that determine whether the incoming data stream should retain or forget:

$$i_t = \sigma(W_{xi} x_t + W_{hi} h_{t-1} + W_{ci} c_{t-1} + b_i) \tag{4}$$

$$f_t = \sigma(W_{xf} x_t + W_{hf} h_{t-1} + W_{cf} c_{t-1} + b_f) \tag{5}$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc} x_t + W_{hc} h_{t-1} + b_c\right) \quad (6)$$

$$o_t = \sigma\left(W_{xo} x_t + W_{ho} h_{t-1} + W_{co} c_t + b_o\right) \quad (7)$$

$$h_t = o_t \tanh\left(c_t\right) \quad (8)$$

where σ denotes the sigmoid function, c is the cell state, W is the weight matrix, b is the bias term, h is the hidden state, and t is the time step. In addition, we use an attention module that contains several dense layers and a softmax layer to learn the attention α for each hidden state h_t at time step t

$$\alpha_t = \text{Softmax}\left(W_{att} \cdot h_t + b_{att}\right), \quad (9)$$

where W_{att} and b_{att} are attention weight and bias, respectively. Afterward, the aggregated attention α and hidden state h over all the time steps are fed to a multi-layer perception that returns predicted corp yield $\hat{y}$, as

$$\hat{y} = \text{MLP}(\alpha \cdot h). \quad (10)$$

Task-adaptive MAML

Existing meta-learning approaches require new testing task $T' \in \mathcal{T}_{test}$ to be from the same distribution as the training tasks $\mathcal{T}_{train}$. The adaptation performance to the new task T' can often be degraded when the training task distribution $p(\mathcal{T}_{train})$ is highly heterogeneous due to a large number of training tasks $\mathcal{T}_{train}$. To address this issue, we consider decomposing the training task distribution $p(\mathcal{T}_{train})$ based on the task difficulty level and have the model be adapted gradually following an easy-to-hard task hierarchy, as shown in Figure 2 (e). In particular, we start with building an initial predictive model using all the task samples in the Train S-Set. Different predictive models can be used in the proposed framework thus we adopt the LSTM-Attention network (introduced from Eq. 4 to 10) to train a meta model on the Train S-Set and optimize its MSE loss using Eq. 1.

The performance of the learned meta model on the validation data of each task T_i from the Train Q-Set can serve as a proxy measure for the task difficulty level. When the validation loss for a specific task is higher, it indicates that this task has different patterns compared to the majority of tasks in $\mathcal{T}_{train}$ that dominates the training of the initial model. Hence, we can split the current set of training tasks $\mathcal{T}_{train}$ into Easy Task (E-r) and Hard Task (H-r), where $r = 0, 1, 2, \ldots, u$ indicates the task layer (difficulty level of the easy-to-hard task hierarchy shown in Figure 2 (c)), by using a threshold γ on the validation performance. Specifically, each task T_i on the task layer r is categorized as

$$D(T_i) = \begin{cases} \text{Hard Task (H-}r\text{)} & \text{if } \mathrm{R}^2[\mathcal{F}_{\theta_i}(x_q^{(i)}), y_q^{(i)}] < \gamma \\ \text{Easy Task (E-}r\text{)} & \text{others} \end{cases} \quad (11)$$

where R^2 is the performance metric measured on the validation data of task T_i from the Train Q-Set. We repeat this process for every task and gather an array of R^2 for all the tasks. The threshold γ is selected based on a statistical test over the obtained R^2 array, which will be discussed later.

Training phase to build the hierarchy. In the training phase, we iteratively bi-partition the set of hard tasks obtained from the previous task layer (H-r-1), where $r \geq 1$. The underlying intuition is to identify the set of tasks that cannot be well captured by the current model (H-r Model). Starting from the second task layer, we build a meta initial model to be fine-tuned to the tasks in the current task set (i.e., all the hard tasks from the previous task layer) via only a few gradient descent steps, following the standard MAML method. Again, we use the validation performance (measured by R^2) of each task-specific fine-tuned model to split the current task set into Easy Task (E-r) and Hard Task (H-r) sets via the threshold-based method (Eq. 11). Here a higher validation loss for a specific task indicates that the meta model cannot generalize well on the task with a small refinement. To accelerate the training, we initialize the meta model with the predictive model (if $r = 1$) or the meta model (if $r \geq 2$) learned from the previous task layer. The process is summarized in Algorithm 1.

Selection of split threshold. We discuss the selection of the threshold γ for splitting the task set at each layer r into Easy (E-r) and Hard (H-r) tasks. Given the obtained validation performance metrics e_i (e.g., R^2) for each task i, we aim to identify a subset of tasks that have significantly $\{e_i\}$

Algorithm 1: Task-adaptive Meta Learning

Output: Optimized meta model weight θ^*, Easy Task E-r, Hard Task H-r in the task layer r
Input: Tasks $\mathcal{T}$, meta model $\mathcal{F}_\theta$
Initialization: learning rate α, β, task layer $r = 0$
while *not done* **do**
  if *task layer r ≥ 1* **then**
    Current task $\mathcal{T} \leftarrow$ H-r
  end
  while *not done* **do**
    for *support tasks* T_i *in* $\mathcal{T}$ **do**
      Adapt meta model θ on task T_i by Eq. 1: $\theta_i \leftarrow \theta - \alpha \nabla \mathcal{L}_{T_i}(\mathcal{F}_\theta)$
    end
    Compute query loss $\mathcal{L}_{MAML}(\mathcal{F}_\theta)$ by Eq. 3
    Update $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}_{TMAML}(\mathcal{F}_\theta)$
  end
  Compute R^2, γ using Alg 2
  Split tasks $\mathcal{T}$ into E-r and H-r using γ
  $r \leftarrow r + 1$
end

Algorithm 2: Threshold γ Selection Algorithm

Output: Threshold γ
Input: R^2 array; Lower/upper bounds ratio a, b
Initialization: Rank R^2 in ascending order; Set array $V = \emptyset$, N the length of R^2 array, index $k = 0$
while k *is less than* N **do**
  $U \leftarrow \mathrm{R}^2[:k]$
  $U' \leftarrow \mathrm{R}^2[k:]$
  $V_k \leftarrow$ Var(U) + Var(U')
  $k \leftarrow k + 1$
end
$\gamma = \mathrm{R}^2[\text{ArgMin}(V[\lfloor aN \rfloor : \lfloor bN \rfloor])]$

values compared to the remaining tasks. Hence, we adopt a statistical test, where the null hypothesis H_0 states that e_i for all the tasks follow a single normal distribution while the alternative hypothesis H_1 states that there exists a subset of tasks U, and they follow a different normal distribution from the remaining tasks U'. Here U can be either the hard or easy tasks. The optimal set U can be obtained by solving the following optimization problem:

$$U^* = \text{argmax}_S \log \frac{\text{Likelihood } (H_1|U)}{\text{Likelihood } (H_0)}. \quad (12)$$

According to the prior work (Xie et al. 2021), this can be solved by minimizing the sum of the variance of U and U'. Hence we can select the threshold γ that leads to the smallest value of the sum of Var(U) and Var(U'). This process is summarized in Algorithm 2.

Testing phase using the hierarchy. In the testing phase, given any new task $T' \in \mathcal{T}_{test}$, we need to identify its difficulty level so that the learned meta models (E-r Model and H-r Model) on the corresponding task layer r can be adapted to the new task. Specifically, starting from the initial model in the easy-to-hard task hierarchy shown in Figure 2 (f), we adapt the corresponding meta model to the new task and measure its validation performance in the Test Q-Set. The obtained R^2 will be compared against the threshold γ on the task layer r. Then we move to the next task layer based on the comparison outcome. This process is repeated until reaching a leaf node of the easy-to-hard task hierarchy. Then we will adapt the final selected model (either the E-r Model where $r = 1, 2, \ldots, u$ or the H-r Model w.r.t. $r = u$) to the new task T'_i for testing.

Data and Experiments

Dataset

The crop yield data are provided by USDA - National Agricultural Statistics Service (NASS) across 630 counties in the United States, and each county has 300 sampled datapoints. Each sample collects 19 daily features including weather and soil conditions, and plant properties, such as temperature, sand content, silt content, and crop yields in 21 consecutive years from 2000 to 2020 (Liu et al. 2021). Inspired by prior work (Jia et al. 2021a,b), we generate the simulated data using a physics-based Ecosys model (Zhou et al. 2021) over Illinois, Indiana and Iowa for pretraining the predictive model, which yields 10K county-level simulation with the same 19 daily features as NASS in 18 consecutive years. We define the meta-learning tasks as county-level crop yield prediction thus we have 630 tasks in total. We construct the meta-learning dataset using the NASS data of which we sample 80% counties in the Midwest U.S. states, including Illinois, Wisconsin, Minnesota, Iowa, Missouri, Ohio, Michigan, North Dakota, South Dakota, Nebraska, Kansas, Kentucky, and Tennessee, as a training set, and 20% counties that are mostly in Indiana as a test set. In both training and testing sets, we use the first 5-year data from 2000 to 2004 as a support set, and the rest 16-year data from 2005 to 2020 as a query set. Next, we select 25 samples for every county in the support set as the Train Support Set (Train S-Set), and 75 samples for every county in the query set as the Train Query Set (Train Q-Set). We apply the same sampling strategy to the test set to obtain Test Support Set (Test S-Set) and Test Query Set (Test Q-Set) sets. For the simulation data, we randomly sample 60%, 20%, and 20% counties as synthetic training, validation, and test set, respectively.

Candidate Methods

We implement a diverse set of baselines and meta-learning-based models for model comparison.

Baseline-A. The predictive model trains on the Train S-Set and Train Q-Set, and tests on the Test Q-Set.

Baseline-B. The predictive model trains on the Train S-Set, Train Q-Set and Test S-Set, and tests on the Test Q-Set.

Baseline-C. The predictive model trains on the Train S-Set and Train Q-Set. Afterward, it fine-tunes on Test S-Set before conducting test on the Test Q-Set.

Origin-MAML. The original MAML (Finn, Abbeel, and Levine 2017) of which the meta model is trained on the Train S-Set and adapted on the Train Q-Set while training. In the test, the learned meta model is quickly fine-tuned on the Test S-Set before testing on the Test Q-Set.

Transfer-MAML. The transfer meta-learning model (Soh, Cho, and Cho 2020) learns a global model on the Train S-Set and transfers the weights to learn the MAML.

Condition-MAML. The conditional meta-learning model (Denevi, Pontil, and Ciliberto 2020), which first trains several clusters (i.e., 4 clusters) using Train S-Set and learns the meta model to each cluster. In the test phase, it fine-tunes on the corresponding meta model based on its clustering prediction before testing.

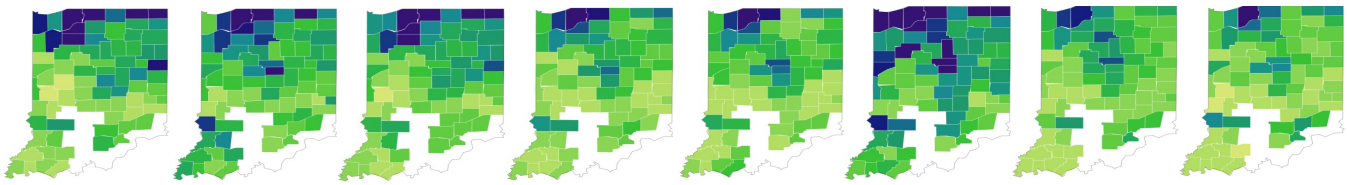
Adaptive-MAML-A. The proposed adaptive meta-learning model implemented using Algorithm 1. In this version, each Easy (E-r) and Hard (H-r) Model is trained with multiple inner epochs (i.e., 3 inner epochs), where $r = 1, 2, \ldots, u$.

Adaptive-MAML-B. The proposed adaptive meta-learning model that trains only 1 inner epoch on the Easy (E-r) and Hard (H-r) Model where $r = 1, 2, \ldots, u$. In the outer epoch iteration shown in Figure 2 (e), it uses the H-r w.r.t. $r = u$ (the hard model in the last task layer) learned in the previous outer epoch as the initial model for the current epoch training, except the first outer epoch that initializes with the pretrained predictive model.

Implementation Details

We implement the proposed task-adaptive meta-learning framework based on the learn2learn backbone (Arnold et al. 2020) with Pytorch[1]. We pretrain the predictive model with the simulated dataset that achieves 0.9898 R^2. Afterward, the pretrained predictive model is used to initialize the initial meta model weights in the first task layer of the easy-to-hard hierarchy. We use Adam optimizer with 0.001 learning rate. In the training phase, we use 32 tasks (out of 630 tasks) as a batch to learn the meta model. In the test phase, we quickly fine-tune the learned meta model for every separate task on its support set and report the performance on its query set. We use the Mean Squared Error (MSE) as the loss function

[1]The code is available at https://github.com/ZhexiongLiu/Task-Adaptive-Meta-Learning.



(a) Baseline-A (b) Baseline-B (c) Baseline-C (d) Origin-MAML (e) Transfer-MAML (f) Condition-MAML (g) Adapt-MAML-A (h) Adapt-MAML-B

Figure 3: Spatially (county-level) visualized predicted R^2 performance in Indiana, where dark blue means low R^2 regions while light green represents high R^2 regions. The best model is the Adaptive-MAML-B.

Models	R^2 (%)		
	Whole Yield	Low Yield	High Yield
Baseline-A	71.34	37.96	62.83
Baseline-B	69.23	55.33	58.28
Baseline-C	70.62	36.76	63.34
Origin-MAML	78.98	60.62	73.07
Condition-MAML	62.84	35.81	54.35
Transfer-MAML	79.01	60.42	72.32
Adaptive-MAML-A	79.77	63.81	73.08
Adaptive-MAML-B	**80.58**	**64.39**	**74.68**

Table 1: The MAML and baseline performance in terms of whole-yield, low-yield and high-yield predictions.

and R^2 as the evaluation metric. We set hyper-parameter a and b in Algorithm 2 as 0.35 and 0.65, respectively, which are experimental values that would guide the model to learn a good threshold (γ) through a subset of the tasks that are between these two bounds. The default adaption number is 1, the (inner) epoch number is 1, and the maximum splitting number is 3 if not specified. We run 30 (outer) epochs with an Nvidia Titan X GPU and report the best performance.

Analysis

Crop Yield Prediction

In Table 1, we compared the R^2 performance of different methods for the crop (corn) yield predictions. We evaluate the performance under three different sets of counties: (1) all the counties, (2) low-yield counties that exclude the top 1/3 high-yield counties, and (3) high-yield counties that exclude the bottom 1/3 low-yield counties. As observed, our proposed adaptive-MAMLs outperform baselines and state-of-the-art meta-learning frameworks. In particular, the Adaptive-MAML-B achieves the best performance in all three sets. In the low-yield counties, the adaptive-MAMLs strongly dominate the baselines and state-of-the-art MAMLs because the low-yield counties are usually along the state boundaries (as shown in Figure 1) and are more difficult to model with existing machine learning methods. Hence, adaptive training on hard tasks will help learn more on these poorly-performed counties. In terms of high-yield counties, they usually have lower data heterogeneity due to the technical improvement in crop cultivation thus the performance is relatively higher than the low-yield. In addition, Table 2 shows the MAML performance with 2 adaptations on trained meta models. As exhibited, the Adaptive-MAML-B improves the performance on the low-yield counties, which

Models	R^2 (%)		
	Whole Yield	Low Yield	High Yield
Origin-MAML	78.53	61.90	71.19
Condition-MAML	65.28	41.80	57.93
Transfer-MAML	78.95	58.46	72.69
Adaptive-MAML-A	77.98	62.48	71.09
Adaptive-MAML-B	**80.57**	**68.52**	**73.89**

Table 2: The MAML performance with 2 adaptations in terms of whole-yield, low-yield and high-yield predictions.

suggests that properly fine-tuning adaptation iterations will benefit low-yield counties. Moreover, the Adaptive-MAML-A performance is degraded (compared to Table 1) because it is overfitting due to a large number of iterations (i.e., 2 adaptations in 3 inner epochs).

Spatial-dimension Performance

Figure 3 shows county-level predicted performance in Indiana. The Adaptive-MAML-B has mostly higher R^2 (light green in the figures) than the other models. In the baseline models, a region of the difficult tasks (counties) in the upper left boundary shows poor performance; however, our proposed adaptive-MAML greatly reduces the poorly-performed area, which indicates that our model is able to adaptively learn difficult tasks regardless of the spatial-related data heterogeneity. This is because our proposed framework learns a set of meta models that can be quickly adapted to different difficulty levels of tasks in the easy-to-hard hierarchy.

Temporal-dimension Performance

Figure 4 shows predicted R^2 performance in 16 consecutive years from 2005 to 2020. The Adaptive-MAML-A and Adaptive-MAML-B perform better than the other models in most of the year, but they perform relatively worse in the year 2012. This is because the national-wide low crop yield occurs due to extreme weather and nature conditions. As for the years 2019 and 2020, Adaptive-MAMLs have relatively poor performance because they are only designed to learn spatial heterogeneity and thus may have limitations on capturing temporal data variability, which will be future work.

Parameter Sensitivity

In this section, we discuss the parameter sensitivity in adaptive-MAMLs. As shown in Figure 2 (e) and (f), the splitting threshold γ is updated each time either starting a

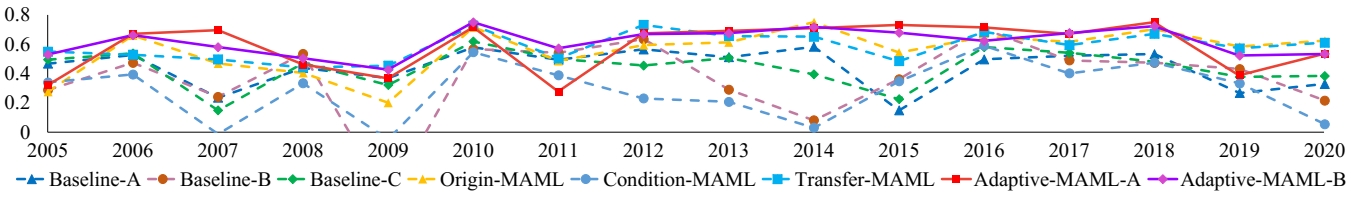
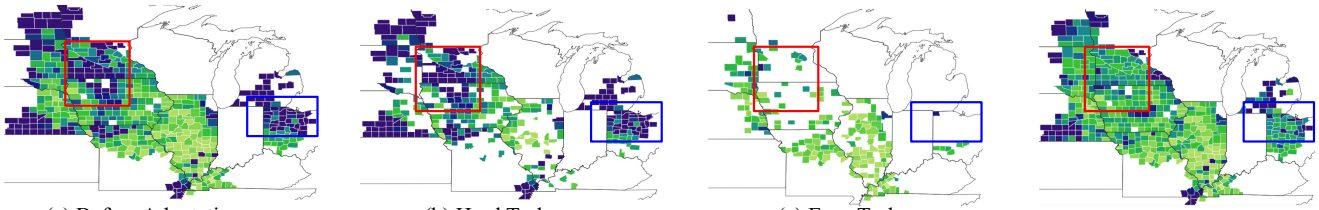


Figure 4: Temporally visualized predicted R^2 performance in 16 years from 2005 to 2020. The proposed Adaptive-MAMLs perform better than the other models in most years.



(a) Before Adaptation (b) Hard Tasks (c) Easy Tasks (d) After Adaptation

Figure 5: An example shows county-level R^2 improvement in (d) before and after adaptive training on (b) hard and (c) easy tasks, of which the areas enclosed by the red and blue boxes in (a) have low R^2.

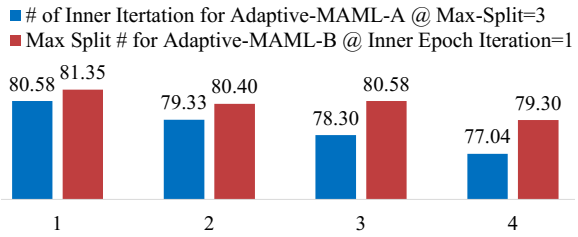


Figure 6: Parameter sensitivity (with values ranging from 1 to 4) in terms of inner epoch iterations and the maximum split towards R^2 (%) for the Adaptive-MAMLs.

Epochs	γ_1 (%)	γ_2 (%)	γ_3 (%)
Epoch1	40.56	32.68	16.47
Epoch2	51.62	34.90	29.19
Epoch3	58.35	48.63	46.12
Epoch4	68.53	62.70	54.09
Epoch5	73.48	67.41	61.05

Table 3: The splitting threshold γ dynamically updates on different task layers r and training epochs in Adaptive-MAML-B, where $r = 1, 2, 3$.

new split or training a new epoch. We show the dynamic update of the threshold γ in the first 5 epoch runs in Table 3. As exhibited, the threshold γ becomes higher as the number of epochs increases. During the training process, the adpative-MAML model is gradually updated to better fit training samples, and the γ becomes higher as the model is setting a higher standard for hard tasks. In addition, the threshold γ decreases with the splits going deeper of the easy-to-hard hierarchy, which indicates the tasks are more difficult in bottom task layers (e.g., $r = 3$), thus the γ becomes lower. In addition, we test different settings of inner epochs for Adaptive-MAML-A and the maximum split for Adaptive-MAML-B. The results (Figure 6) show that the performance decreases given the increase of the inner epoch training. This shows that over-training under a premature task hierarchy can degrade the model performance. As for the maximum split, both 1, 2, and 3 are acceptable numbers. The small numbers mean fewer easy and hard models will be trained, which is helpful when the data has low heterogeneity.

Case Study

In Figure 5, we study examples to show the effectiveness of adaptive-MAMLs. The areas (shown as counties on the map) marked with the red and blue rectangles in (a) exhibit low R^2 before adaptive learning. However, the Algorithm 2 splits current training tasks in (a) into (b) hard tasks and (c) easy tasks, and the Algorithm 1 iteratively trains on the hard tasks until a maximum splitting number is reached. As exhibited in (d), the poorly-performed areas (hard tasks) were greatly improved while well-performed areas (easy tasks) maintained excellence compared to (a).

Conclusion

Standard meta-learning methods, e.g., MAML, can have degraded performance given a large number of heterogeneous tasks because spatial data variability is one of the most common issues in many spatial datasets. To bridge the gap, we proposed a task-adaptive MAML to learn spatial-related tasks with an easy-to-hard hierarchy that helps adapt the meta model to new tasks. Extensive experiments show that our methods are superior to a diverse set of baselines and state-of-the-art models on real crop yield data in the Midwest of the United States. The proposed framework demonstrates meta-learning generalizability on a substantial number of spatial-sensitive meta models. In future work, we plan to develop robust MAMLs that adaptively learn both spatial and temporal data heterogeneity and ultimately promote models' feasibility to a wide range of societal problems.

Acknowledgements

This work was supported by NSF awards 2147195, 2105133, and 2126474, NASA award 80NSSC22K1164, the USGS awards G21AC10207, G21AC10564, and G22AC00266, Google's AI for Social Good Impact Scholars program, the DRI award at the University of Maryland, and CRC at the University of Pittsburgh.

References

Antoniou, A.; Edwards, H.; and Storkey, A. 2019. How to train your MAML. In *International Conference on Learning Representations*.

Arnold, S. M. R.; Mahajan, P.; Datta, D.; Bunner, I.; and Zarkias, K. S. 2020. learn2learn: A Library for Meta-Learning Research. *CoRR*, abs/2008.12284.

Caruana, R. 1997. Multitask learning. *Machine learning*, 28(1): 41–75.

Denevi, G.; Pontil, M.; and Ciliberto, C. 2020. The advantage of conditional meta-learning for biased regularization and fine tuning. *Advances in Neural Information Processing Systems*, 33: 964–974.

Dong, D.; Wu, H.; He, W.; Yu, D.; and Wang, H. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1723–1732.

Elavarasan, D.; and Vincent, P. D. 2020. Crop yield prediction using deep reinforcement learning model for sustainable agrarian applications. *IEEE access*, 8: 86886–86901.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, 1126–1135. PMLR.

Huang, C.; Zhang, J.; Zheng, Y.; and Chawla, N. V. 2018. DeepCrime: Attentive hierarchical recurrent networks for crime prediction. In *Proceedings of the 27th ACM international conference on information and knowledge management*, 1423–1432.

Huang, R.; Huang, C.; Liu, Y.; Dai, G.; and Kong, W. 2020. LSGCN: Long Short-Term Traffic Prediction with Graph Convolutional Networks. In *IJCAI*, 2355–2361.

Jia, X.; Willard, J.; Karpatne, A.; Read, J. S.; Zwart, J. A.; Steinbach, M.; and Kumar, V. 2021a. Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles. *ACM/IMS Transactions on Data Science*, 2(3): 1–26.

Jia, X.; Xie, Y.; Li, S.; Chen, S.; Zwart, J.; Sadler, J.; Appling, A.; Oliver, S.; and Read, J. 2021b. Physics-Guided Machine Learning from Simulation Data: An Application in Modeling Lake and River Systems. In *2021 IEEE International Conference on Data Mining (ICDM)*, 270–279. IEEE.

Jiang, T.; Huang, M.; Segovia-Dominguez, I.; Newlands, N.; and Gel, Y. R. 2022. Learning Space-Time Crop Yield Patterns with Zigzag Persistence-Based LSTM: Toward More Reliable Digital Agriculture Insurance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 12538–12544.

Jiang, W.; Huang, K.; Geng, J.; and Deng, X. 2020. Multi-scale metric learning for few-shot learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3): 1091–1102.

Karpatne, A.; Ebert-Uphoff, I.; Ravela, S.; Babaie, H. A.; and Kumar, V. 2018. Machine learning for the geosciences: Challenges and opportunities. *IEEE Transactions on Knowledge and Data Engineering*, 31(8): 1544–1554.

Kendall, A.; Gal, Y.; and Cipolla, R. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Li, D.; Zhang, J.; and Huang, K. 2020. Learning to learn cropping models for different aspect ratio requirements. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12685–12694.

Li, X.; Zhao, L.; Wei, L.; Yang, M.-H.; Wu, F.; Zhuang, Y.; Ling, H.; and Wang, J. 2016. Deepsaliency: Multi-task deep neural network model for salient object detection. *IEEE transactions on image processing*, 25(8): 3919–3930.

Li, Z.; Zhou, F.; Chen, F.; and Li, H. 2017. Meta-sgd: Learning to learn quickly for few-shot learning. *CoRR*, abs/1707.09835.

Liu, L.; Zhou, W.; Jin, Z.; Tang, J.; Jia, X.; Jiang, C.; Guan, K.; Peng, B.; Xu, S.; Yang, Y.; et al. 2021. Estimating the Autotrophic and Heterotrophic Respiration in the US Crop Fields using Knowledge Guided Machine Learning. In *AGU Fall Meeting Abstracts*, volume 2021, B25O–13.

Liu, X.; He, P.; Chen, W.; and Gao, J. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4487–4496. Florence, Italy: Association for Computational Linguistics.

Ma, P.; Du, T.; and Matusik, W. 2020. Efficient continuous pareto exploration in multi-task learning. In *International Conference on Machine Learning*, 6522–6531. PMLR.

Matsumi, S.; and Yamada, K. 2021. Few-Shot Learning Based on Metric Learning Using Class Augmentation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 196–201. IEEE.

Mishra, N.; Rohaninejad, M.; Chen, X.; and Abbeel, P. 2018. A simple neural attentive meta-learner. In *International Conference on Learning Representations*.

Munkhdalai, T.; and Yu, H. 2017. Meta networks. In *International Conference on Machine Learning*, 2554–2563. PMLR.

Nevavuori, P.; Narra, N.; and Lipping, T. 2019. Crop yield prediction with deep convolutional neural networks. *Computers and electronics in agriculture*, 163: 104859.

Nigam, A.; Garg, S.; Agrawal, A.; and Agrawal, P. 2019. Crop yield prediction using machine learning algorithms. In *2019 Fifth International Conference on Image Information Processing (ICIIP)*, 125–130. IEEE.

Pan, Z.; Zhang, W.; Liang, Y.; Zhang, W.; Yu, Y.; Zhang, J.; and Zheng, Y. 2020. Spatio-temporal meta learning for urban traffic prediction. *IEEE Transactions on Knowledge and Data Engineering*.

Qiao, S.; Liu, C.; Shen, W.; and Yuille, A. L. 2018. Few-shot image recognition by predicting parameters from activations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7229–7238.

Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; and Lillicrap, T. 2016. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, 1842–1850. PMLR.

Sharma, S.; Rai, S.; and Krishnan, N. C. 2020. Wheat Crop Yield Prediction Using Deep LSTM Model. *CoRR*, abs/2011.01498.

Shekhar, S.; Zhang, P.; Huang, Y.; and Vatsavai, R. R. 2003. Trends in spatial data mining. *Data mining: Next generation challenges and future directions*, 357–380.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Soh, J. W.; Cho, S.; and Cho, N. I. 2020. Meta-transfer learning for zero-shot super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3516–3525.

Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H.; and Hospedales, T. M. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1199–1208.

Thrun, S.; and Pratt, L. 1998. Learning to learn: Introduction and overview. In *Learning to learn*, 3–17. Springer.

Tseng, G.; Kerner, H.; Nakalembe, C.; and Becker-Reshef, I. 2021. Learning to predict crop type from heterogeneous sparse labels using meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1111–1120.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Wang, Y.; Yao, Q.; Kwok, J. T.; and Ni, L. M. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3): 1–34.

Willard, J. D.; Read, J. S.; Appling, A. P.; Oliver, S. K.; Jia, X.; and Kumar, V. 2021. Predicting Water Temperature Dynamics of Unmonitored Lakes With Meta-Transfer Learning. *Water Resources Research*, 57(7): e2021WR029579.

Wu, S.; Zhang, H. R.; and Ré, C. 2020. Understanding and Improving Information Transfer in Multi-Task Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Xie, Y.; He, E.; Jia, X.; Bao, H.; Zhou, X.; Ghosh, R.; and Ravirathinam, P. 2021. A statistically-guided deep network transformation and moderation framework for data with spatial heterogeneity. In *2021 IEEE International Conference on Data Mining (ICDM)*, 767–776. IEEE.

Xu, J.; Zhu, Y.; Zhong, R.; Lin, Z.; Xu, J.; Jiang, H.; Huang, J.; Li, H.; and Lin, T. 2020. DeepCropMapping: A multi-temporal deep learning approach with improved spatial generalizability for dynamic corn and soybean mapping. *Remote Sensing of Environment*, 247: 111946.

Zhang, Y.; and Yang, Q. 2018. An overview of multi-task learning. *National Science Review*, 5(1): 30–43.

Zhang, Z.; Luo, P.; Loy, C. C.; and Tang, X. 2014. Facial landmark detection by deep multi-task learning. In *European conference on computer vision*, 94–108. Springer.

Zhao, H.; Stretcu, O.; Smola, A. J.; and Gordon, G. J. 2020. Efficient multitask feature and relationship learning. In *Uncertainty in Artificial Intelligence*, 777–787. PMLR.

Zhao, Y.; Zhong, Z.; Yang, F.; Luo, Z.; Lin, Y.; Li, S.; and Sebe, N. 2021. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6277–6286.

Zheng, C.; Fan, X.; Wang, C.; and Qi, J. 2020. Gman: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 1234–1241.

Zhou, S.; Zeng, X.; Zhou, Y.; Anastasopoulos, A.; and Neubig, G. 2019. Improving robustness of neural machine translation with multi-task learning. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, 565–571.

Zhou, W.; Guan, K.; Peng, B.; Tang, J.; Jin, Z.; Jiang, C.; Grant, R.; and Mezbahuddin, S. 2021. Quantifying carbon budget, crop yields and their responses to environmental variability using the ecosys model for US Midwestern agroecosystems. *Agricultural and Forest Meteorology*, 307: 108521.