Election-Based Task Pruning in Mixture-of-Experts for Scalable Multi-Task Learning

Andrei Bachinin, Shrideep Pallickara, and Sangmi Lee Pallickara

Department of Computer Science
Colorado State University
Fort Collins, Colorado, 80523
Email: {Andrei.Bachinin, Shrideep.Pallickara, Sangmi.Pallickara}@colostate.edu

***Abstract*— Scientific analyses often involve multiple tasks that are closely related to each other. These tasks inherently share dependencies that can be effectively modeled through multi-task learning (MTL), which enables shared representation learning and cross-task generalization. However, applying MTL to complex scientific datasets with numerous, weakly related tasks remains challenging due to the computational burden of exploring extensive task combinations and the lack of automated task selection strategies. To address these challenges, we propose Self Expert Exploration for Mixture of Experts models (SEEM), a novel MTL pruning framework that adaptively identifies and retains only the most relevant tasks during training. SEEM introduces an importance-based pruning algorithm and a two-tier grouping mechanism distinguishing primary and secondary tasks. It quantifies inter-task relationships via gating patterns, graph-based centrality, and consensus voting to drive pruning decisions. By dynamically pruning tasks, SEEM produces a compact, computationally efficient model while preserving shared learning performance and enhancing computing efficiency. We evaluate SEEM using advanced Mixture of Experts architectures across multiple scientific and real-world datasets, spanning both regression and classification tasks. Our results demonstrate that SEEM achieves superior scalability and reduced computational cost, while improving interpretability by revealing key inter-task relationships and mitigating negative transfer from weakly related tasks.**

I. Introduction

Scientific observations are typically followed by analyses that attempt to extract patterns and identify meaningful interactions among attributes. Multi-spectral satellite imagery, for example, can be leveraged for analyses targeting vegetation monitoring, flood mapping, urban expansion, and deforestation assessment; many of which are inherently interrelated. Likewise, soil analysis data describing hundreds of chemical properties supports detailed investigations of interactions among those properties and their spatiotemporal trends. Effectively capturing the relationships between observations and target characteristics, along with the interactions among multiple attributes, is therefore essential for accelerating scientific discovery. These requirements motivate learning frameworks that can jointly exploit shared information across related tasks.

Multi-task learning (MTL) offers such framework for addressing these interconnected problems by jointly training models on multiple tasks to enhance computational efficiency, improve accuracy through cross-task regularization, and promote knowledge sharing across task-specific sub-networks. Mixture-of-Experts (MoE) models [1] further extend this idea by employing specialized gating networks to dynamically combine shared sub-modules for flexible knowledge sharing. Despite these advantages, the inter-task relationships captured by these methods remain indirect and difficult to interpret. These approaches also depend on manually selected task groups. Consequently, the final trained models often retain all gating modules, including those that contribute minimally, leading to unnecessary computational overhead during both training and inference.

These limitations become especially acute in large scientific analyses, where a model must handle a large set of tasks with unknown inter-task relationships, introducing numerous combinatorial configurations to explore. Expecting a model to learn effectively from an excessively large set of tasks without careful task selection not only increases computational demands but also limits the efficacy of shared representation learning. At the same time, a well-trained MTL model can enhance interpretability by revealing shared structures and inter-task relationships; this is an especially valuable property in scientific applications, where such insights often guide subsequent analyses. However, existing MTL approaches lack principled strategies for identifying effective task combinations, which limits their adaptability and scalability [2], [3].

In this study, we propose Self Expert Exploration for MoE models (SEEM), a novel learnable task pruning framework for MTL. SEEM introduces an importance-based task pruning algorithm that quantifies the effectiveness of shared learning and thereby enables a learnable task selection mechanism during training. Starting from a large mixture of tasks, the framework iteratively retains only the most important tasks through a two-tier grouping mechanism consisting of primary and secondary tasks. Pruning is guided to enhance the performance of primary tasks while preserving only essential secondary tasks, resulting in faster inference, reduced computational cost, and improved scientific interpretability across the retained tasks.

We evaluate SEEM using advanced MoE architectures (Figure 1): ML-MMoE [4] and AdaTT [5]. Our evaluation encompasses diverse real-world and scientific datasets to assess the framework's adaptability to varying task characteristics and underlying model structures. To examine scalability, we test SEEM across different numbers of tasks within each

architecture. This work has three main contributions:

- We propose a novel model pruning algorithm that performs learnable expert and task selection throughout training, producing a compact and computationally efficient final model for both training and inference while preserving the benefits of shared learning.
- The proposed pruning mechanism enhances scientific interpretability by explicitly identifying key related tasks during training.
- We evaluate the scalability and effectiveness of SEEM against non-pruning baseline configurations across two distinct MoE architectures and demonstrate its superior computational efficiency and inference performance.

II. Related Work

Multi-Task Learning and Mixture-of-Experts. Multi-task learning (MTL) improves generalization by leveraging shared representations across related tasks [6]–[8]. To address negative transfer in hard parameter sharing, Mixture-of-Experts (MoE) approaches use gating networks to dynamically allocate expert capacity [1], [9]. Recent advances include Multi-gate Mixture-of-Experts (MMoE) [4], Progressive Layered Extraction (PLE) [10], and Adaptive Task-to-Task Fusion Network (AdaTT) [5], which introduce task-specific gating and progressive knowledge extraction. However, these methods lack automated mechanisms for task selection and expert pruning, limiting scalability to problems with large numbers of tasks.

Neural Network Pruning. Pruning reduces model complexity while maintaining performance through structured or unstructured weight removal [2], [11]–[13]. Unstructured pruning targets individual weights [14], while structured pruning removes entire components for hardware-agnostic speedups [15]. The Lottery Ticket Hypothesis [16] demonstrated that sparse subnetworks can match full network performance when trained from appropriate initializations, inspiring iterative magnitude pruning methods [17]. Despite extensive research on single-task models, pruning multi-expert architectures remains underexplored, with unique challenges in preserving task relationships and expert diversity.

Knowledge Guided Machine Learning Several efforts have explored the use of knowledge guided machine learning (KGML) methods to model spatiotemporally evolving phenomena. These approaches are often coupled with mechanistic or process-based domain models and typically incorporate customized, multi-part loss functions that encode scientific knowledge. Examples include enforcing physical constraints from soil hydrology (such as the van Genuchten water-retention relationships Richards'-equation–based models of hydraulic conductivity [18], [19] as well as incorporating vegetation indices [20]; evapotranspiration [21]; and preserving graph properties such as betweenness centrality [22]. Other efforts account for human perceptual limits during visualization [23] and masking cloud occlusions in satellite imagery [24]. Related applications also include soil salinity modeling [25], hyperspectral imagery [26], and accounting for correlations between soil spectroscopic properties [27].

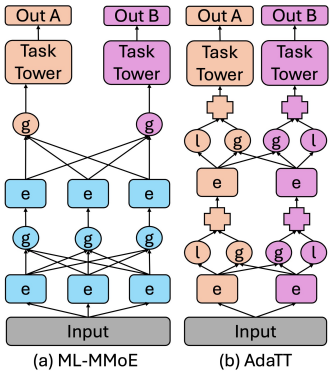

Fig. 1. Different types of MTL MoE architectures, such as ML-MMoE and AdaTT, exhibit distinct interaction patterns among experts. g: gate, l: linear fusion, e: expert, OutA: Output of task A, OutB: Output of task B.

III. Methodology

We propose Self Expert Exploration for MoE Models (SEEM), a progressive pruning framework that automatically identifies and retains relevant auxiliary tasks for designated target tasks during training. Throughout, we refer to tasks whose performance we primarily care about as *target* tasks and all remaining tasks as *auxiliary* tasks. Conceptually, SEEM behaves like a closed-loop controller around an MoE backbone: (1) observe routing behavior (gates), (2) estimate inter-task structure, and (3) actuate by disconnecting low-utility tasks. This control-loop framing matters because it clarifies why pruning is triggered by training dynamics (plateau/gradients) rather than a fixed schedule: the system waits until it has enough signal to distinguish "helpful sharing" from "harmful interference," then reduces the active task set to stabilize and accelerate convergence. SEEM dynamically prunes less beneficial tasks based on learned inter-task relationships, producing computationally efficient models with improved target task performance and interpretable task relationships. To make such pruning decisions principled rather than ad hoc, task contributions are quantified through three signals: (1) direct importance from gating weights, (2) global importance from graph centrality, and (3) consensus voting among tasks.

A. Compatibility with MoE Architectures

SEEM is designed to be model-agnostic. It operates on MoE architectures that use gating networks to route information through expert modules. Formally, consider a general MoE model with T tasks and L fusion layers. At layer ℓ with E_ℓ experts, task i's gating network computes expert selection weights $g_i^\ell = \text{softmax}(W_i^\ell h_i^{\ell-1})$, where $W_i^\ell \in \mathbb{R}^{E_\ell \times d_{\ell-1}}$ are the learnable gate weights for task i at layer ℓ; $h_i^{\ell-1} \in \mathbb{R}^{d_{\ell-1}}$ is the input representation for task i from the previous layer (with $d_{\ell-1}$ being the input dimension); and $g_i^\ell \in \mathbb{R}^{E_\ell}$ represents the normalized importance weights over all E_ℓ experts. The task output at this layer is computed as a weighted combination of expert outputs $o_i^\ell = \sum_{e=1}^{E_\ell} g_{i,e}^\ell \cdot f_e^\ell(h_i^{\ell-1})$, where $f_e^\ell(\cdot)$ is the e-th expert function and $g_{i,e}^\ell$ is the weight assigned by task i to

expert e. Within this general MoE setup, SEEM maintains an evolving active task set $\mathcal{A} \subseteq \{1, \ldots, T\}$, initially containing all tasks. Less relevant tasks are gradually moved to pruned set $\mathcal{P} = \{1, \ldots, T\} \setminus \mathcal{A}$ by setting $g^{\ell}_{i,e} = 0$ for all experts and layers, effectively disconnecting those tasks from the shared expert pool and eliminating their computational footprint.

B. Task Importance Quantification

The foundation of SEEM is extracting meaningful inter-task relationships from the model. To do so, we construct importance matrix $M \in \mathbb{R}^{T \times T}$ where M_{ij} indicates how much task i depends on task j. The computation method depends on the MoE architecture. This separation allows us to use a single pruning framework across structurally different MoE backbones. Interpreting M as an empirical dependency graph is useful: edges summarize how much one task's learned routing relies on another's representational capacity. In task-specific-expert models, gate magnitudes provide a direct proxy for "how often" the target borrows from a task's experts (Eq. 1). In shared-expert models, cosine similarity of gate-induced expert preferences (Eq. 2) instead captures structural affinity—tasks that repeatedly select similar experts are likely to share features even when no explicit task-expert ownership exists.

Models with task-specific experts (e.g., AdaTT) organize experts by task affiliation. Each task j has dedicated experts, allowing direct importance extraction from gate weights at final layer L:

$$M_{ij} = \frac{1}{E_j \cdot d_L} \sum_{e \in \mathcal{E}_j} \sum_{k=1}^{d_L} |W^L_{i,e,k}| \tag{1}$$

where $\mathcal{E}_j$ denotes experts affiliated with task j; $E_j = |\mathcal{E}_j|$ is the number of experts for task j; d_L is the input dimension at layer L; and $W^L_{i,e,k}$ are gate weight matrix elements. The absolute value operation and averaging across both experts and input dimensions give us a scalar measure of how strongly task i's gating network attends to task j's experts, and hence how much task i benefits from task j's specialized capacity.

By contrast, **the models with shared experts** (e.g., ML-MMoE) share a common expert pool with no task-expert affiliation. We measure importance through *expert preference similarity*. We first compute each task's expert importance vector $v_i = \frac{1}{d_L} \sum_{k=1}^{d_L} |W^L_{i,:,k}|$, where $W^L_{i,:,k}$ is the k-th column of gate weight matrix for task i; and $v_i \in \mathbb{R}^{E_L}$ captures expert importance for task i. The importance between tasks is then computed as:

$$M_{ij} = \frac{1 + \cos(v_i, v_j)}{2} \tag{2}$$

where $\cos(v_i, v_j) = \frac{v_i \cdot v_j}{\|v_i\| \|v_j\|}$ is cosine similarity, normalized to $[0, 1]$. Higher values indicate stronger task relationships, since tasks that consistently favor similar experts are likely to share representational structure.

C. Estimating the Task Importance

SEEM evaluates the importance of auxiliary tasks to inform pruning decisions. For a designated target task t^* (never pruned), each candidate task gets an importance score based on three components. Each component captures a distinct but complementary aspect of the inter-task relationships, and together they determine the overall pruning score $s_{\text{importance}}(j)$.

1. Direct Importance. The direct contribution to the target task is quantified using $s_{\text{direct}}(j) = M_{t^*,j}$. A higher value indicates that the target task's gating network places greater attention on candidate task j.

2. Graph centrality-based Importance. Some tasks are more closely related to one another, and these multivariate relationships provide key insights for identifying groups of tasks that should be analyzed together. To detect such groups, we apply weighted PageRank centrality analysis to the importance matrix and consider tasks with high PageRank scores as closely related (or tightly connected) tasks. Let $P \in \mathbb{R}^{T \times T}$ be a set of normalized weights between all possible pairs of two task experts. First, we compute transition matrix P from M using temperature-based softmax:

$$P_{ij} = \frac{\exp(M_{ij}/\tau)}{\sum_{k=1}^{T} \exp(M_{ik}/\tau)} \tag{3}$$

where $\tau > 0$ is a temperature parameter controlling the sharpness of transitions (lower values emphasize high-importance connections). The PageRank scores are computed iteratively $r^{(n+1)} = \alpha P^T r^{(n)} + (1 - \alpha)\frac{1}{T}$, where $r^{(n)} \in \mathbb{R}^T$ are importance scores at iteration n, $\alpha \in (0, 1)$ is a damping factor (typically 0.85), and $1 \in \mathbb{R}^T$ is a vector of ones. The term $(1 - \alpha)\frac{1}{T}$ represents the teleportation probability in PageRank, ensuring that the random walk does not get stuck and assigns some baseline importance to all tasks. The iteration converges to a stationary distribution r^*, giving us $s_{\text{graph}}(j) = r^*_j$.

3. Voting Mechanism. Whereas the first two components measure attraction among tasks, this component implements a consensus-based approach where other candidate tasks vote on which candidates should be pruned. Each active auxiliary task $i \in \mathcal{A} \setminus \{t^*, j\}$ can vote to prune candidate j. For a voter task i, empirically selected threshold (bottom 25%) is used to identify least important tasks. If candidate j falls into this bottom quartile for voter i, then voter i contributes a vote with strength $v_{ij} = M_{t^*,i} \cdot (1 - M_{i,j})$.

The intuition is that if task i is important to the target ($M_{t^*,i}$ is high) and task j is unimportant to task i ($M_{i,j}$ is low, making $1 - M_{i,j}$ high), then i's vote to prune j carries more weight. The total vote score for candidate j is $s_{\text{vote}}(j) = \sum_{i \in \mathcal{V}_j} v_{ij}$, where $\mathcal{V}_j \subseteq \mathcal{A}$ is the set of voters that identified j in their bottom 25%. This raw vote score is then normalized to ensure all three components have comparable scales:

$$\bar{s}_{\text{vote}}(j) = \frac{s_{\text{vote}}(j)}{\max_{k \in \mathcal{A} \setminus \{t^*\}} s_{\text{vote}}(k)} \tag{4}$$

Finally, to obtain a single scalar measure of task utility, three components are combined into a pruning score using a weighted linear combination:

$$s_{\text{importance}}(j) = \beta_1 \cdot s_{\text{direct}}(j) + \beta_2 \cdot s_{\text{graph}}(j) - \beta_3 \cdot \bar{s}_{\text{vote}}(j) \tag{5}$$

where the tunable weights, β_1, β_2, $\beta_3 > 0$ are combination coefficients satisfying $\beta_1 + \beta_2 + \beta_3 = 1$. The negative coefficient on the voting score ensures that tasks receiving many prune-votes are ranked lower. Tasks assigned lower importance scores are more likely to be pruned during the current epoch, while tasks with consistently high scores are retained as key contributors.

Intuitive Summary. Task importance scoring can be understood as follows. Direct importance measures how much the target task explicitly attends to each auxiliary task through gating weights. Graph centrality identifies tasks that are central hubs in the overall task-relationship network. Voting captures collective agreement among tasks about which auxiliaries are least useful. A task is more likely to survive pruning when it scores well on the first two signals and receives few prune-votes from the third, ensuring that different forms of utility are preserved.

D. Pruning Decision and Training Procedure

SEEM monitors the training progress to identify epochs where the model's learning for the target task has plateaued. We track three indicators over a sliding window of w most recent epochs. Together, these indicators summarize progress in terms of performance, optimization dynamics, and temporal stability.

Improvement Rate. We compute the average change in the target task's validation metric (R^2 for regression, AUC for classification) over the window $r_{\text{imp}} = \frac{1}{w-1}\sum_{k=1}^{w-1}(m_{k+1} - m_k)$, where m_k is the validation metric at epoch k. A value near zero indicates performance has plateaued, signaling diminishing returns from continued training with the current task set.

Gradient Norm. For each epoch, we compute the L2 norm of gradients for the target task's parameters $g_{\text{norm}}^{(k)} = \frac{1}{|\theta_{t^*}|}\sqrt{\sum_{p\in\theta_{t^*}} \|\nabla_p \mathcal{L}_{t^*}\|_2^2}$, where $\mathcal{L}_{t^*}$ is the loss for the target task, θ_{t^*} represents parameters specific to the target task (its tower and gate network), and $|\theta_{t^*}|$ is the number of parameter tensors. We then track the windowed average $\bar{g}_{\text{norm}} = \frac{1}{w}\sum_{k=1}^{w} g_{\text{norm}}^{(k)}$ over the most recent w epochs. Small gradient norms indicate the model has converged for the target task.

Epochs Without Improvement: Counter c tracking consecutive epochs without validation improvement.

Once these signals are available, the three indicators are combined into a pruning readiness score:

$$p_{\text{score}} = \begin{cases} \frac{\epsilon_r - r_{\text{imp}}}{\epsilon_r} & \text{if } r_{\text{imp}} < \epsilon_r \\ 0 & \text{otherwise} \end{cases} + \begin{cases} \frac{\epsilon_g - g_{\text{norm}}}{\epsilon_g} & \text{if } g_{\text{norm}} < \epsilon_g \\ 0 & \text{otherwise} \end{cases} + \begin{cases} 0.5 & \text{if } c > c_{\text{patience}} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where ϵ_r and ϵ_g are threshold hyperparameters for improvement rate and gradient norm respectively, and c_{patience} is the patience threshold for epochs without improvement. Each term contributes to the score only when its corresponding indicator signals convergence. The score ranges from 0 (no indicators suggest pruning) to approximately 2.0 (all indicators agree), providing a smooth, interpretable trigger for pruning events.

Pruning occurs when $p_{\text{score}} > \theta_p$ (typically 0.7 to 0.8) and at least Δ_e epochs (typically 15-25) have passed since the last pruning event. When these conditions are met, tasks are ranked by their final scores from Equation (5), and approximately $p\%$ of the lowest-scoring tasks (typically 15-25%) are removed, subject to maintaining at least T_{min} active tasks. This gradual schedule avoids overly aggressive pruning early in training while still yielding a compact task set by convergence.

A practical caveat is that pruning is a commitment: removing tasks changes the optimization landscape, which can amplify asymmetries in multi-target settings when targets compete for incompatible features. The union-based strategy is conservative, but it may still retain "good for A, bad for B" auxiliaries, producing the observed trade-offs. SEEM's gradual schedule and minimum-active-task constraint act as safeguards against over-pruning early, but future variants could add *cool-down periods* (partial gate suppression) to reduce oscillations.

The complete training procedure integrates SEEM with standard multi-task optimizations. For example, we employ uncertainty weighting to balance task losses:

$$\mathcal{L}_{\text{total}} = \sum_{i\in\mathcal{A}} \left(\frac{e^{-s_i}}{2}\mathcal{L}_i + \frac{s_i}{2} \right) \tag{7}$$

where s_i are learnable task-specific log-variance parameters (initialized to zero), and $\mathcal{L}_i$ is the loss for task i. This equation is equivalent to $\frac{1}{2\sigma_i^2}\mathcal{L}_i + \log\sigma_i$ where $s_i = \log\sigma_i^2$, with $e^{-s_i} = 1/\sigma_i^2$ acting as precision weights. The target task loss can be further weighted by a multiplier $\lambda_{t^*} > 1$ to emphasize its optimization. After each pruning event, we update the active task set $\mathcal{A}$ and zero out computations for pruned experts, reducing computational cost for subsequent training epochs and inference.

Extension to Multiple Target Tasks. The framework described above assumes a single designated target task. In many scientific applications, however, community might be interested to jointly optimize multiple target properties of interest. For instance, in soil spectroscopy applications, both soil organic carbon (SOC) and iron (Fe) concentrations may be equally important target properties. SEEM naturally extends to this scenario through a union-based pruning strategy. In this multi-target setting, when multiple target tasks $\mathcal{T}^* = \{t_1^*, t_2^*, \ldots, t_k^*\}$ are designated, we compute pruning scores separately for each target task following the procedure in Equation (5). For each candidate auxiliary task j, we then aggregate scores across targets by taking the maximum: $s_{\text{final}}(j) = \max_{t^*\in\mathcal{T}^*} s_{\text{pruning}}^{t^*}(j)$, where $s_{\text{pruning}}^{t^*}(j)$ denotes the pruning score computed with respect to target t^*. This conservative aggregation ensures that a task is pruned only if it exhibits low importance to all target tasks simultaneously. The target tasks themselves are never pruned and all receive

elevated weights in the loss function, preserving their central role in guiding the pruning process.

IV. Experimental Setup

Benchmark Datasets We evaluate SEEM on three diverse datasets, each paired with a distinct underlying model task type. For evaluating classification-style tasks, we use the *CelebA Facial Attributes dataset [28]*, which contains 40 classes over 202,599 facial images with train/validation/test splits. We also employ multi-spectral satellite images from the Harmonized Landsat Sentinel-2 (HLS) [29] using RGB bands from 3 summer months, combined with rasterized land cover type data from *NLCD* [30]. Based on these satellite images, we construct 16 binary classification tasks for different land cover types.

To evaluate SEEM on regression tasks, we use a scientific dataset that combines spectral information with soil analysis results from the *National Cooperative Soil Survey (NCSS) database [31]*. This database provides soil spectral data along with corresponding concentration of various soil chemical properties. We use mid-infrared spectral data for 13 soil properties, with each spectral record containing 1,765 channels. Unless otherwise noted, all datasets use an 80/10/10 train/validation/test split. Tables I and II summarize the impact of SEEM across these datasets, contrasting runs with and without pruning for single and multiple target tasks, respectively.

Model Architectures and Implementation Details We implement SEEM on two MTL-MoE architectures: AdaTT (task-specific experts) and ML-MMoE (shared expert pool). Both share identical input processing, task towers, and optimization, but differ in expert allocation. This contrast assesses SEEM's transferability between disjoint versus shared expert capacity. We tailor backbone designs to each dataset while keeping MoE and pruning components comparable. Model capacity varies substantially (NCSS: 24M, CelebA: 6.7M, NLCD: 227K), reflecting input complexity rather than task difficulty. Notably, while NLCD is the most challenging dataset, it requires minimal capacity to avoid overfitting given severe class imbalance and limited samples.

CelebA: ViT feature extractor (embedding 256, 4 layers, 4 heads, 16×16 patches) feeds three fusion layers [128, 64, 32] and task towers (dim 32). AdaTT: 2 experts/task; ML-MMoE: 80 shared experts, 40 gates/layer. Dropout 0.15.

NLCD: Lightweight ViT (embedding 64, 2 layers, 2 heads, 32×32 patches, 9-channel input) feeds two fusion layers [32, 16] and task towers (dim 8). AdaTT: 2 experts/task (32 total/layer); ML-MMoE: 32 shared experts, 16 gates/layer. Dropout 0.2 (AdaTT), 0.25 (ML-MMoE).

NCSS: Three fusion layers [512, 256, 128], task towers (dim 32), 512-dim input projection. AdaTT: 4 experts/task (52 total/layer); ML-MMoE: 52 shared experts, 13 gates/layer.

All models use ReLU activation and layer normalization. NCSS and CelebA use uncertainty-weighted loss; NLCD uses focal loss that focuses on hard examples and down-weights easy ones. Progressive pruning is applied during training for joint task selection and representation learning.

Training and Evaluation. We train all models using the AdamW optimizer with cosine annealing learning rate schedules. Batch sizes are 64. Initial learning rates are 0.003 (NCSS), 0.0003 (CelebA), and 0.0002 (NLCD). For SEEM variants, we set temperature $\tau = 1.0$, PageRank damping factor, $\alpha = 0.85$, and prune approximately 15-25% of lowest-scoring tasks when triggered. We measure task-specific performance (R^2 or AUC), inference throughput (samples/minute with batch size 64), and total parameter count. All experiments run on NVIDIA A100 GPUs (80GB) using PyTorch 2.0 with CUDA 11.8.

Training-Time Overhead. The computational cost of SEEM's pruning components is negligible compared to the forward and backward passes. Computing the importance matrix from gate weights is $O(T^2 \cdot d)$ where T is the number of tasks and d is the gate input dimension. PageRank iteration converges in approximately 50 steps with $O(T^2)$ per iteration. The voting mechanism requires $O(T^2)$ comparisons. These operations are performed only at pruning checkpoints (every 15-25 epochs) and involve only small matrices ($T \times T$, typically 13×13 to 40×40).

V. Results & Discussion

A. Evaluation with Single Target Tasks

For single target tasks (Table I), SEEM consistently outperforms in both predictive accuracy and computational efficiency across all datasets and architectures. On NCSS with nitrogen as target property, AdaTT with SEEM achieves $R^2 = 0.9927$ (vs. 0.9910 baseline) while reducing the parameters by 30% (24M to 16.6M) and improving inference throughput by 41%. Although improvements also occur with ML-MMoE, parameter reduction and inference throughput gains are limited due to the shared expert architecture where experts cannot be removed. With AdaTT, the parameter reduction occurs when removing entire task branches (experts, gates, and towers). However, with ML-MMoE, we can only remove final-layer gates and task-specific prediction towers since the shared experts are still needed to perform all remaining tasks.

On CelebA, targeting the "Wearing Earring" attribute, SEEM delivers substantial gains: AdaTT improves AUC from 0.8849 to 0.9072 while achieving 17% faster inference. The framework successfully identifies and prunes 6 of 40 tasks, retaining only those attributes most relevant to earring detection (e.g., hair style, gender-related features). For the challenging NLCD dataset with the task identifying *cultivated crops*, AdaTT with SEEM achieves the largest relative improvement (AUC 0.6379 to 0.6951), demonstrating that pruning is particularly effective when the baseline performance is potentially limited by negative transfer from unrelated tasks. In this setting, SEEM acts as a mechanism for automatically removing sources of harmful sharing while preserving beneficial ones.

Across all experiments, the task-specific expert architecture of AdaTT enables greater parameter reduction (12-31%) compared to ML-MMoE (less than 5%), demonstrating that architectural design impacts pruning efficiency. However, both

TABLE I

PERFORMANCE COMPARISON ACROSS DATASETS WITH AND WITHOUT SEEM, FOR SINGLE TARGET TASK. **BOLD**: BEST; BLUE : SEEM

Dataset, Target	Model	Pruning	Metric	Pruned Tasks	Inference (Samples/Min)	Number of Parameters
NCSS, Nitrogen	AdaTT	w/o	$R^2\uparrow$, 0.9910	–	346,882	24M
		w	$R^2\uparrow$, **0.9927**	4/13	**488,220**	**16.6M**
	ML-MMoE	w/o	$R^2\uparrow$,0.9872	–	341,588	24M
		w	$R^2\uparrow$, **0.9896**	4/13	**358,604**	**23.9M***
CelebA, Wearing Earrings	AdaTT	w/o	$AUC\uparrow$, 0.8849	–	189,202	6.7M
		w	$AUC\uparrow$, **0.9072**	6/40	**220,729**	**6M**
	ML-MMoE	w/o	$AUC\uparrow$, 0.8872	–	167,983	6.6M
		w	$AUC\uparrow$, **0.9060**	6/40	**182,779**	**6.5M***
NLCD, Cultivated Crops	AdaTT	w/o	$AUC\uparrow$, 0.6379	–	584,697	227K
		w	$AUC\uparrow$, **0.6951**	4/16	**744,734**	**199K**
	ML-MMoE	w/o	$AUC\uparrow$, 0.6235	–	535,303	227K
		w	$AUC\uparrow$, **0.6261**	7/16	**709,065**	**211K***

TABLE II

PERFORMANCE COMPARISON ACROSS DATASETS WITH AND WITHOUT SEEM, FOR MULTIPLE TARGET TASK.

Dataset, Targets	Model	Pruning	Metric	Pruned Tasks	Inference (Samples/Min)	Number of Parameters
NCSS, Fe, Nitrogen	AdaTT	w/o	Fe $R^2\uparrow$, 0.9820; N $R^2\uparrow$, 0.9910	–	346,882	24M
		w	Fe $R^2\uparrow$, **0.9859**; N $R^2\uparrow$, **0.9921**	4/13	**485,643**	**16.6M**
	ML-MMoE	w/o	Fe $R^2\uparrow$, 0.9819; N $R^2\uparrow$, 0.9872	–	341,588	24M
		w	Fe $R^2\uparrow$, **0.9821**; N $R^2\uparrow$, **0.9912**	6/13	**369,399**	**23.9M***
CelebA, Wearing Earrings (WE), No Beard (NB)	AdaTT	w/o	WE $AUC\uparrow$, 0.8849; NB $AUC\uparrow$, 0.9804	–	189,202	6.7M
		w	WE $AUC\uparrow$, **0.9152**; NB $AUC\uparrow$, **0.9808**	6/40	**216,106**	**6M**
	ML-MMoE	w/o	WE $AUC\uparrow$, 0.8872; NB $AUC\uparrow$, **0.9814**	–	167,983	6.6M
		w	WE $AUC\uparrow$, **0.9114**; NB $AUC\uparrow$, 0.9795	6/40	**178,951**	**6.5M***
NLCD, Cultivated Crops (CC), Woody Wetlands (WW)	AdaTT	w/o	CC $AUC\uparrow$, 0.6379; WW $AUC\uparrow$, **0.6413**	–	584,697	227K
		w	CC $AUC\uparrow$, **0.7108**; WW $AUC\uparrow$, 0.6152	2/16	**649,910**	**212K**
	ML-MMoE	w/o	CC $AUC\uparrow$, 0.6235; WW $AUC\uparrow$, **0.6298**	–	535,303	227K
		w	CC $AUC\uparrow$, **0.6619**; WW $AUC\uparrow$, 0.6213	4/16	**577,320**	**222K***

*The limited parameter reduction in ML-MMoE is architectural: with AdaTT, pruning removes entire task branches (experts, gates, and towers), while with ML-MMoE, only final-layer gates and task-specific towers can be removed since shared experts remain necessary for all active tasks.

architectures benefit substantially from improved inference speed (5-41% gains) and enhanced target task performance, indicating that task-importance signals used by SEEM are valuable even when structural pruning opportunities are limited.

B. Evaluation with Multiple Target Tasks

When optimizing for multiple target tasks simultaneously (Table II), SEEM employs a conservative union-based pruning strategy that retains any task important to at least one target. On NCSS with Fe and N as joint targets, both targets improve while maintaining similar computational gains to the single-target scenario. For CelebA with "Wearing Earrings" and "No Beard" as targets, the earrings target shows strong improvement while the latter maintains baseline performance. These results suggest that union-based pruning can enhance at least one target without sacrificing the others when their underlying feature requirements overlap substantially.

However, on the most challenging NLCD dataset with joint targets (cultivated crops and woody wetlands), the results reveal the limitations of the multi-target approach. While the accuracy for the task identifying cultivated crops improves substantially (0.6379 to 0.7108 for AdaTT), woody wetlands task shows slight degradation (0.6413 to 0.6152). This trade-off emerges because the union-based aggregation strategy conservatively retains tasks important to either target, potentially including tasks that benefit one target while interfering with the other. We observed that task combinations involving highly distinct land cover types, those with different spatial patterns and spectral signatures, struggle during optimization due to continuous competition among tasks, particularly when the dataset exhibits a high degree of class imbalance. In such cases, the shared representation space must accommodate conflicting gradients, making it harder for a single model to serve all targets equally well. This suggests that for extremely difficult multi-target scenarios, alternative aggregation strategies (e.g., intersection-based or weighted combinations) should be considered. Despite these challenges, SEEM consistently delivers computational efficiency gains across all multi-target experiments, with inference improvements ranging from 6% to 40% depending on dataset and architecture, highlighting that task pruning remains beneficial even when accuracy trade-offs must be carefully managed.

VI. Conclusion & Future Work

We presented SEEM, a progressive pruning framework for multi-task learning with mixture-of-experts architectures that automates task selection by quantifying inter-task relationships through gating weights, graph-based centrality, and consensus voting. Our results demonstrate multi-faceted benefits: (1) improved predictive accuracy with up to 9% gains in AUC or R^2; (2) enhanced interpretability by explicitly identifying relevant auxiliary tasks; (3) parameter reduction of up to 31% for task-specific architectures; and (4) inference speedup of 5-41%. Taken together, these findings show that principled task pruning can improve both model quality and efficiency rather than forcing a trade-off between them. They also indicate that routing capacity toward a smaller, better-chosen subset of tasks can mitigate negative transfer while preserving the advantages of shared representation learning. Viewed more broadly, SEEM turns task selection in large-scale MTL into a data-driven, training-time decision rather than a manual design choice, making MoE-based systems more deployable when the task set is large and task relatedness is unknown a priori.

Future work will address limitations in multi-target scenarios by exploring alternative aggregation strategies such as weighted combinations or adaptive mechanisms based on task similarity metrics. In particular, dynamically adjusting pruning decisions as inter-task relationships evolve during training may help balance competing objectives across targets. Extensions could include partial pruning for finer-grained control, application to other MoE variants, and evaluation in continual learning settings where tasks arrive sequentially. Such directions would further test the robustness of SEEM and clarify its role as a general mechanism for scalable, interpretable multi-task learning in scientific and high-dimensional application domains.

VII. Acknowledgment

This research was supported by the National Science Foundation (1931363, 2312319), the National Institute of Food Agriculture (COL014021223, 2025-77039-45531), and an NSF/NIFA Artificial Intelligence Institutes AI-LEAF Award [2023-03616].

References

[1] R. A. Jacobs *et al.*, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[2] Y. LeCun *et al.*, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, 1989.

[3] S. Han *et al.*, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015.

[4] J. Ma *et al.*, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1930–1939.

[5] D. Li *et al.*, "Adatt: Adaptive task-to-task fusion network for multitask learning in recommendations," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 4370–4379.

[6] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[7] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017.

[8] I. Misra *et al.*, "Cross-stitch networks for multi-task learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3994–4003.

[9] N. Shazeer *et al.*, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017.

[10] H. Tang *et al.*, "Progressive layered extraction (PLE): A novel multi-task learning (MTL) model for personalized recommendations," in *Proceedings of the 14th ACM conference on recommender systems*, 2020, pp. 269–278.

[11] H. Li *et al.*, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016.

[12] H. Cheng *et al.*, "A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[13] Z. Liu *et al.*, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2736–2744.

[14] S.-K. Yeom *et al.*, "Pruning by explaining: A novel criterion for deep neural network pruning," *Pattern Recognition*, vol. 115, p. 107899, 2021.

[15] P. Molchanov *et al.*, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016.

[16] J. Frankle *et al.*, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.

[17] T. J. Saleem *et al.*, "Insights into the lottery ticket hypothesis and iterative magnitude pruning," *arXiv preprint arXiv:2403.15022*, 2024.

[18] P. Khandelwal *et al.*, "Deepsoil: A science-guided framework for generating high precision soil moisture maps by reconciling measurement profiles across in-situ and remote sensing data," in *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*, 2024, pp. 233–246.

[19] ——, "Subterra: Estimating soil moisture at root zone depths using science-guided learning," in *2025 IEEE Conference on Artificial Intelligence (CAI)*. IEEE, 2025, pp. 328–335.

[20] K. Bruhwiler *et al.*, "Lightweight, embeddings based storage and model construction over satellite data collections," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 246–255.

[21] S. Armstrong *et al.*, "Attention-based convolutional capsules for evapotranspiration estimation at scale," *Environmental Modelling & Software*, vol. 152, p. 105366, 2022.

[22] A. Matin *et al.*, "Rapid betweenness centrality estimates for transportation networks using capsule networks," in *2022 Fourth International Conference on Transdisciplinary AI (TransAI)*. IEEE, 2022, pp. 89–96.

[23] S. Mitra *et al.*, "Glance: A generative approach to interactive visualization of voluminous satellite imagery," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 359–367.

[24] P. Khandelwal *et al.*, "Cloudnet: A deep learning approach for mitigating occlusions in landsat-8 imagery using data coalescence," in *2022 IEEE 18th International Conference on e-Science (e-Science)*. IEEE, 2022, pp. 117–127.

[25] R. Dey *et al.*, "Deepsalt: Bridging laboratory and satellite spectra through domain adaptation and knowledge distillation for large-scale soil salinity estimation," 2025.

[26] A. Matin *et al.*, "Knowledge-guided masked autoencoder with linear spectral mixing and spectral-angle–aware reconstruction," 2026.

[27] A. Bachinin *et al.*, "Science-informed multitask transformer for soil property prediction from ftir spectroscopy," in *2025 IEEE International Conference on eScience (eScience)*. IEEE, 2025, pp. 48–57.

[28] Z. Liu *et al.*, "Deep learning face attributes in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3730–3738.

[29] M. Claverie *et al.*, "The harmonized landsat and sentinel-2 surface reflectance data set," *Remote sensing of environment*, vol. 219, pp. 145–161, 2018.

[30] J. Dewitz, "National land cover database (NLCD) 2021 products," *US Geological Survey (USGS) Data Release*, p. 828, 2023.

[31] USDA Natural Resources Conservation Service. (2024) NCSS Lab Data Mart - Kellogg Soil Survey Laboratory (KSSL) Soil Characterization Database. Accessed: Oct. 20, 2025. [Online]. Available: https://ncsslabdatamart.sc.egov.usda.gov/