Rep-MTL: Unleashing the Power of Representation-level Task Saliency for Multi-Task Learning

Supplementary Material

This supplementary material offers additional empirical analyses, experimental results, and further discussions of our work. The appendix sections are organized as follows:

- In Appendix A, we provide experimental setups and implementation specifications across all four benchmarks in this paper, including NYUv2 [59], Cityscapes [11], Office-Home [62], and Office-31 [52]. This includes comprehensive information on employed network architectures, optimization algorithms, training protocols, loss functions, and hyper-parameter configurations.
- In Appendix B, we provide complete experimental results on the Office-31 dataset, which were omitted from the main manuscript due to space constraints. We also discuss the proposed Rep-MTL method combined with all experimental results from four benchmarks.
- In Appendix C, we present additional ablation studies through the lens of PL exponent alpha analysis [44, 46]. These studies further demonstrate how each mechanism of Rep-MTL contributes to facilitating cross-task positive transfer while preserving task-specific learning patterns, thereby mitigating the negative transfer in MTL.
- In Appendix D, we conduct experiments to validate Rep-MTL's robustness and practical applicability, with particular emphasis on the sensitivity of hyper-parameters $\lambda_{tsr}, \lambda_{csa}$, learning rates, and its optimization speed.

A. Implementation Details

This appendix section provides an expansion of the experimental configurations and implementation specifications of the experiments from the main manuscript. We detail the network architectures, optimizers, and training recipes for each included benchmark to ensure reproducibility.

NYUv2 Dataset Following the implementations in previous studies [28, 30], we employ the DeepLabV3+ [5] network architecture, containing a dilated ResNet 50 [17] backbone pre-trained on ImageNet and the Atrous Spatial Pyramid Pooling (ASPP) as task-specific decoders. The MTL model is trained for 200 epochs using the Adam optimizer with an initial learning rate of 10^{-4} and weight decay of 10^{-5}. Consistent with prior works [28, 30], we implement a learning rate schedule where the rate is halved to 5×10^{-5} after 100 training epochs. For the three tasks on NYUv2 [59], we utilize cross-entropy loss for semantic segmentation, L_1 loss for depth estimation, and cosine loss for surface normal prediction. We adopted the same logarithmic transformation in previous studies [30, 34, 49]. During training, all input images are resized to 288×384, and we set the batch size to 8. The experiments are implemented with PyTorch and executed on NVIDIA A100-80G GPUs.

Cityscapes Dataset The implementations for Cityscapes benchmark demonstrate substantial alignment with the one on NYUv2 [28, 30]. Specifically, we adopt the identical DeepLabV3+ [5] architecture, leveraging an ImageNet-pretained dilated ResNet 50 network as the backbone, while the ASPP module serves as task-specific decoders. For model optimization, we establish a 200-epoch training regime utilizing Adam optimizer, with the initial learning rate of 10^{-4} and weight decay of 10^{-5}. The learning rate undergoes a scheduled reduction to 5×10^{-5} upon reaching the 100-epoch milestone. We maintain consistency of loss functions with NYUv2: cross-entropy loss and L_1 loss are employed for semantic segmentation and depth estimation, respectively. We also adopted the logarithmic transformation as in previous studies [30, 34, 49]. Throughout the training process, all input images are resized to 128×256, and we utilize a batch size of 64. The experiments are implemented with PyTorch on NVIDIA A100-80G GPUs.

Office-Home Dataset Building upon established protocols from prior works [28, 30], we implement an ImageNet-pretrained ResNet-18 network architecture as the shared backbone, complemented by a linear layer serving as task-specific decoders. In pre-processing, all input images are resized to 224×224. The batch size and the training epoch are set to 64 and 100, respectively. The optimization process employs the Adam optimizer with the learning rate of 10^{-4} and the weight decay of 10^{-5}. We utilize cross-entropy loss for all classification tasks, with classification accuracy serving as the evaluation metric. We also adopted the logarithmic transformation as in previous studies [30, 34, 49]. The "Avg." reported in the main manuscript represents the mean performance gains across three independent tasks, which is notably excluded from the calculation of overall task-level performance gains. The experiments are implemented with PyTorch and executed on NVIDIA A100-80G GPUs.

Office-31 Dataset The configurations on Office-31 [52] dataset exhibit notable parallels with the ones on Office-Home [28, 30]. Concretely, we deploy a ResNet-18 network architecture pre-trained on the ImageNet dataset as the shared backbone, complemented by task-specific linear layers for classification outputs. The data processing pipeline

Table 5. Performance on Office-31 dataset with 3 diverse image classification tasks. ↑ indicates the higher the metric values, the better the methods' performance. The best and second-best results of each metric are highlighted in **bold** and underline, respectively.

Method	Amazon	DSLR	Webcam	Avg.↑	Δp_{task}↑
Single-Task Baseline	**86.61**	95.63	96.85	93.03	0.00
EW	83.53	97.27	96.85	$92.55_{\pm 0.62}$	$-0.61_{\pm 0.67}$
GLS [10]	82.84	95.62	96.29	$91.59_{\pm 0.58}$	$-1.63_{\pm 0.61}$
RLW [28]	83.82	96.99	96.85	$92.55_{\pm 0.89}$	$-0.59_{\pm 0.95}$
UW [22]	83.82	97.27	96.67	$92.58_{\pm 0.84}$	$-0.56_{\pm 0.90}$
DWA [35]	83.87	96.99	96.48	$92.45_{\pm 0.56}$	$-0.70_{\pm 0.62}$
IMTL-L [34]	84.04	96.99	96.48	$92.50_{\pm 0.52}$	$-0.63_{\pm 0.58}$
IGBv2 [12]	84.52	98.36	98.05	$93.64_{\pm 0.26}$	$+0.56_{\pm 0.25}$
MGDA [13]	85.47	95.90	97.03	$92.80_{\pm 0.14}$	$-0.27_{\pm 0.15}$
GradNorm [8]	83.58	97.26	96.85	$92.56_{\pm 0.87}$	$-0.59_{\pm 0.94}$
PCGrad [66]	83.59	96.99	96.85	$92.48_{\pm 0.53}$	$-0.68_{\pm 0.57}$
GradDrop [9]	84.33	96.99	96.30	$92.54_{\pm 0.42}$	$-0.59_{\pm 0.46}$
GradVac [63]	83.76	97.27	96.67	$92.57_{\pm 0.73}$	$-0.58_{\pm 0.78}$
IMTL-G [34]	83.41	96.72	96.48	$92.20_{\pm 0.89}$	$-0.97_{\pm 0.95}$
CAGrad [32]	83.65	95.63	96.85	$92.04_{\pm 0.79}$	$-1.14_{\pm 0.85}$
MTAdam [42]	85.52	95.62	96.29	$92.48_{\pm 0.87}$	$-0.60_{\pm 0.93}$
Nash-MTL [49]	85.01	97.54	97.41	$93.32_{\pm 0.82}$	$+0.24_{\pm 0.89}$
MetaBalance [19]	84.21	95.90	97.40	$92.50_{\pm 0.28}$	$-0.63_{\pm 0.30}$
MoCo [15]	84.33	97.54	98.33	93.39	-
Aligned-MTL [55]	83.36	96.45	97.04	$92.28_{\pm 0.46}$	$-0.90_{\pm 0.48}$
IMTL [34]	83.70	96.44	96.29	$92.14_{\pm 0.85}$	$-1.02_{\pm 0.92}$
DB-MTL [30]	85.12	**98.63**	98.51	$94.09_{\pm 0.19}$	$+1.05_{\pm 0.20}$
Rep-MTL (EW)	85.93	98.54	**98.67**	**94.38**$_{\pm 0.53}$	**+1.31**$_{\pm 0.58}$

standardizes input images to 224×224, while the training protocol extends across 100 epochs with a fixed batch size of 64. The Adam optimizer configured with a learning rate of 10^{-4} and the weight decay of 10^{-5} is used. The cross-entropy loss is used for all the tasks and classification accuracy is used as the evaluation metric. We adopted logarithmic transformation as in previous studies [30, 34, 49]. The "Avg." reported in the main manuscript represents the mean performance gains across three independent tasks, which is notably excluded from the calculation of overall task-level performance gains. The experiments are implemented with PyTorch and executed on NVIDIA A100-80G GPUs.

B. Office-31 Image Classification Results

This appendix section provides a thorough discussion of our experimental results on Office-31 [52] dataset, presenting detailed observations of performance that were omitted from the main text due to space limitations.

As shown in Table 5, Rep-MTL achieves the highest overall performance among all compared MTO methods. It obtains an average accuracy (**Avg.**↑) of 94.38% and a total performance gain (Δp_{task} ↑) of $+1.31\%$ over the single-task learning (STL) baseline. This result surpasses the next-best method, DB-MTL, which achieves a gain of $+1.05\%$, and stands in stark contrast to the Equal Weighting (EW) baseline that suffers from negative transfer among tasks ($\Delta = -0.61\%$). This demonstrates Rep-MTL's superior ability to effectively manage multi-domain learning on Office-31.

In addition, task-specific performance reveals several notable findings. First, on both the **Webcam** and **Amazon** domains, Rep-MTL achieves competitive accuracies of 98.67% and 85.93%, respectively. Its performance on the challenging **Amazon** domain is particularly noteworthy, outperforming the strong DB-MTL [30] baseline by a significant margin of $+0.81\%$. This improvement is particularly significant due to the varying lighting conditions and image quality. Second, on **DSLR** domain, Rep-MTL delivers a competitive accuracy of 98.54%, narrowly trailing DB-MTL [30] (98.63%) in a tightly contested result.

These results offer key insights into the strengths and limitations of Rep-MTL. On one hand, Rep-MTL demonstrates capabilities to handle multiple tasks effectively, consistently achieving balanced and top-tier performance gains across different tasks. The substantial gains on the **Amazon** and **Webcam** tasks more than compensate for the marginal difference on **DSLR**, leading to the best overall average. On the other hand, however, this balanced approach comes with a trade-off: while Rep-MTL avoids significant performance degradation in task-specific performance compared to existing methods, it may not consistently achieve significant gains across all sub-tasks simultaneously. This observation is particularly evident in the results of the **DSLR** task on Office-31 [52] dataset, where Rep-MTL achieves strong but not leading performance.

Overall, the experimental results suggest that while Rep-MTL has successfully advanced the state-of-the-art in challenging multi-task dense prediction benchmarks, there remains scope for further enhancement. Future research directions could focus on developing mechanisms to maintain the current balanced performance with explicit information sharing while pushing the boundaries of task-specific excellence. This could potentially involve exploring more complex cross-task interactions or adaptive optimization strategies that can better leverage task-specific characteristics.

C. Ablations with PL Exponent Alpha Metrics

While our analysis in Section 4.3 demonstrates Rep-MTL's overall effectiveness in achieving effective multi-task learning—facilitating positive cross-task information sharing while preserving task-specific patterns for negative transfer mitigation—it does not isolate the contributions of individual components. This appendix section thus presents an additional empirical evaluation of Rep-MTL's two key mechanisms: Cross-Task Saliency Alignment (CSA) and Task-specific Saliency Regularization (TSR). We first introduce the practical implications of this metric, followed by ablation studies examining each component's effectiveness and distinct contribution to Rep-MTL's overall performance.

C.1. Power Law (PL) Exponent Alpha Analysis

To rigorously evaluate the effectiveness of Rep-MTL's components beyond commonly-used performance metrics,

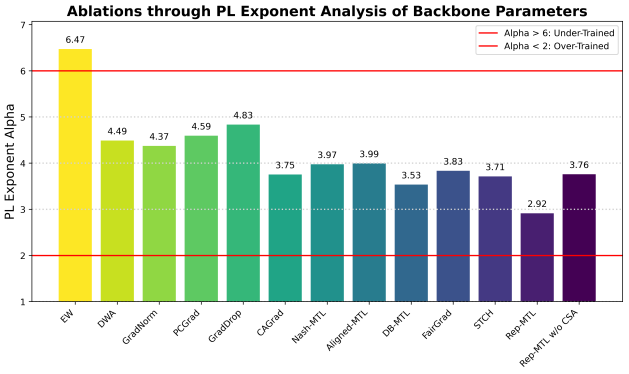

Figure 4. Ablation studies through PL exponent metrics [46] for shared parameters in backbones trained with or without cross-task saliency alignment (notated as "Rep-MTL w/o CA") on NYUv2 [59]. The PL exponent alpha quantifies how well the backbone adapts to the overall MTL objectives, where lower values indicate more effective training. Values outside the range $[2, 6]$ suggest potential over- or under-training due to the insufficient cross-task positive transfer. We leverage this measurement to validate the effectiveness of the cross-task saliency alignment mechanism in our proposed Rep-MTL, as well-trained backbones suggest beneficial information sharing to the overall MTL objectives.

we employ Power Law (PL) exponent alpha [44, 46], a theoretically grounded measure from Heavy-Tailed Self-Regularization (HT-SR) theory [41, 45]. It provides a systematic framework for analyzing the representation capacity and overall learning quality of deep neural networks. In particular, PL exponent alpha is computed for each layer's weight matrix W by fitting the Empirical Spectral Density (ESD) of its correlation matrix $X = W^T W$ to a truncated Power Law distribution: $\rho(\lambda) \sim \lambda^{-\alpha}$, where $\rho(\lambda)$ denotes the ESD, and λ represents eigenvalues of correlation matrix.

Empirical studies have established that well-trained neural networks typically exhibit PL exponent values within the range $\alpha \in [2, 4]$. Values outside this range often indicate suboptimal training dynamics: specifically, $\alpha < 2$ suggests insufficient learning, while $\alpha > 6$ indicates potential over-parameterization or training instabilities. This characteristic makes the PL exponent particularly valuable for assessing training effectiveness across different network architectures and optimization strategies.

In the context of multi-task learning, this metric offers unique insights into both cross-task knowledge transfer and task-specific learning patterns. In particular, for shared backbone parameters, lower alpha values (within the optimal range) typically indicate effective cross-task information sharing, suggesting successful optimization toward the overall MTL objectives. For task-specific heads, balanced and moderately low alpha values across different tasks suggest the preservation of task-specific patterns while minimizing negative transfer effects. Built upon this view, we can systematically evaluate how each component in Rep-MTL contributes to achieving optimal MTL dynamics.

C.2. Effects of Cross-Task Saliency Alignment

Similar to the empirical analysis in Section 4.3, we analyze the effectiveness of Cross-Task Saliency Alignment by examining PL exponent alpha of the DeepLabV3+ backbone parameters on NYUv2 [59] dataset.

As shown in Figure 4, models trained with our cross-task alignment mechanism exhibit alpha values within the optimal range of $[2, 4]$, indicating well-learned and generalizable model parameters in the shared backbone, comparing models trained with and without this alignment mechanism. This demonstrates the effectiveness of our Cross-Task Saliency Alignment for positive information sharing.

C.3. Effects of Task-specific Saliency Regulation

To evaluate the impact of Task-specific Saliency Regulation, we examine the PL exponent alpha of parameters in the DeepLabV3+ task decoder parameters on NYUv2 [59].

As illustrated in Figure 5, the result reveals that models employing our regulation mechanism demonstrate alpha values consistently within the optimal range and exhibit more balanced values across all task-specific heads. This balanced distribution suggests the successful preservation of task-specific features while avoiding over-specialization or interference between tasks $(2.60, 2.63, 2.45)$. In contrast, models trained with Rep-MTL without this regulation mechanism exhibit poor and more dispersed PL exponent alpha across decoders $(2.89, 2.74, 2.59)$. This wider variation indicates potential negative transfer and suboptimal task-specific learning. The consistency of alpha values across different task heads in regulated models provides strong evidence that the Task-specific Saliency Regulation in Rep-MTL effectively maintains task-specific patterns.

D. Additional Empirical Analysis

This appendix section presents an empirical investigation designed to further validate the effectiveness and robustness of Rep-MTL. We conduct empirical analyses of hyper-parameter sensitivity and computational efficiency to provide insights into the practical deployment considerations.

D.1. Analysis of Hyper-parameter Sensitivity

We systematically evaluate Rep-MTL's sensitivity to its two primary hyper-parameters, λ_{tsr} and λ_{csa}, on the NYUv2 [59] dataset. Figure 6 illustrates the task-level performance gains relative to STL baselines ($\Delta \mathrm{p}_{\mathrm{task}}$) across various hyper-parameter configurations. Our analysis involves fixing one hyper-parameter at 0.9 while varying the other one across a comprehensive range: $\{0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9\}$. For example, when evaluating the sensitivity of hyper-parameter λ_{tsr},

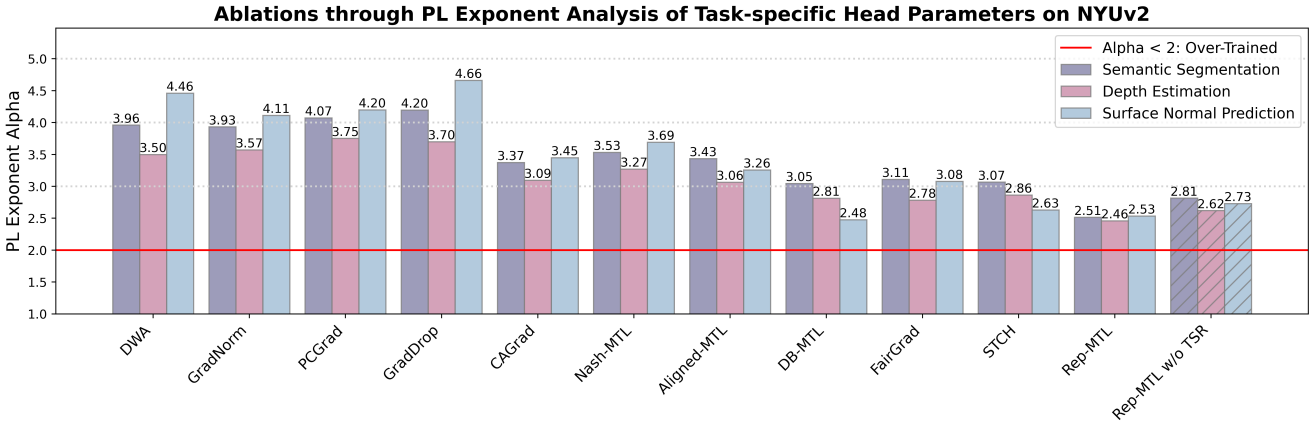

Figure 5. Ablation studies through PL exponent metrics [44, 46] for parameters in diverse decoders trained with or without task-specific saliency regulation in Rep-MTL (notated as "Rep-MTL w/o TR") on NYUv2 [59]. The PL exponent alpha quantifies how well each decoder adapts to its task-specific objective, where lower values indicate more effective training. Values outside the range $[2, 6]$ suggest potential over- or under-training due to task conflicts. The variation across different heads of each method indicates training imbalance. We leverage this measurement to validate the effectiveness of task-specific saliency regulation in Rep-MTL, as well-trained decoders should exhibit both *low and balanced* metric values, indicating successful negative transfer mitigation while preserving task-specific information. The results show that task-specific saliency regulation effectively helps task-specific learning and yields superior and more balanced metrics.

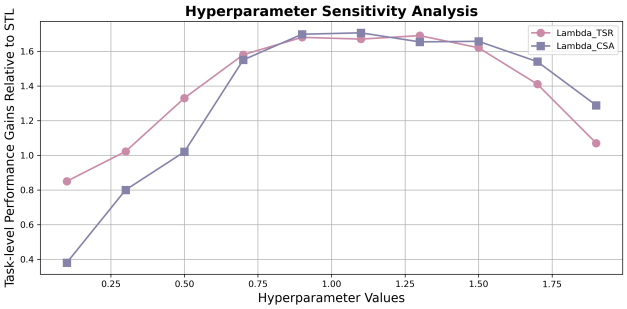

Figure 6. Hyper-parameter sensitivity analysis of our Rep-MTL on NYUv2 [59] dataset. We empirically evaluate the impact of two critical hyper-parameters, λ_{tsr} and λ_{csa}, by fixing one as $\lambda = 0.9$ while varying the other one across a comprehensive range as $\{0.1, 0.3, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 1.7, 1.9\}$. The results demonstrate that Rep-MTL maintains stable and competitive performance Δp_{task} over a substantial range $(0.7, 0.9, 1.1, 1.3, 1.5)$, indicating its robust insensitivity to hyper-parameter variations.

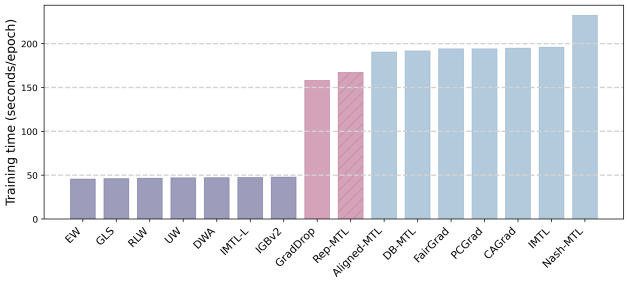

Figure 7. Training time per epoch comparison across different MTL optimization methods on NYUv2 [59]. Methods are categorized into three training efficiency tiers (indicated by different colors), highlighting the inherent trade-off between computational speed and optimization effectiveness in MTL scenarios.

when fixing the $\lambda_{csa} = 0.9$ then conduct a series of experiments. All experiments are conducted on NVIDIA A100-80G GPUs to ensure consistent evaluation conditions.

The results reveal several key findings: First, Rep-MTL demonstrates great stability across a wide range of hyper-parameter combinations, particularly within the range of $\{0.7, 0.9, 1.1, 1.3, 1.5\}$ for both λ_{tsr} and λ_{csa}. Second, the method consistently achieves positive performance gains ($\Delta p_{\text{task}} > 0$) across most hyper-parameter settings, indicating robust improvement over STL baselines. Third, Cross-task Saliency Alignment (CSA) in Rep-MTL, controlled by λ_{csa}, acts as a crucial component. While small values of λ_{csa} lead to suboptimal performance, increasing it beyond a certain threshold demonstrates a significant impact on the overall MTL performance. Based on these observations, we conducted grid search over $\{0.7, 0.9, 1.1, 1.3, 1.5\}$ for both λ_{tsr} and λ_{csa} to determine optimal configurations for all datasets in this paper.

D.2. Analysis of Training Time

To further evaluate the efficiency of Rep-MTL, we conduct a runtime empirical analysis on NYUv2 [59] dataset. Figure 7 presents the average per-epoch training time across different MTL optimization methods, with all experiments conducted over 100 epochs on NVIDIA A100-80G GPUs. Our analysis reveals that Rep-MTL achieves a comparatively favorable balance between training speed and optimization effectiveness. While it requires more training resources than loss scaling methods due to the computation

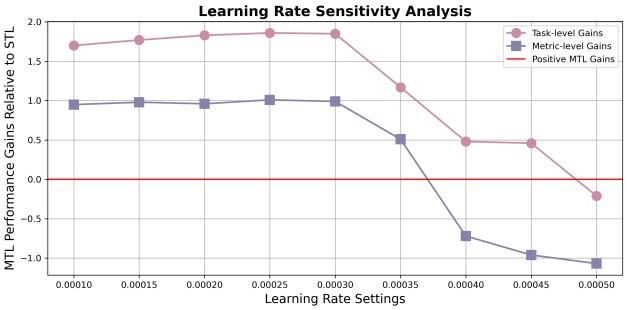

Figure 8. Learning rate sensitivity analysis of our proposed Rep-MTL on NYUv2 [59] dataset. To evaluate the impact of learning rate variations, we systematically scale the learning rate from the default benchmark setting of $1e-4$ to $5e-4$, using a step size of $5e-5$. For each setting, we report the task-level ($\Delta \mathrm{p}_{\text{task}}$) and metric-level ($\Delta \mathrm{p}_{\text{metric}}$) performance gains. Each experiment is repeated three times. The results show that Rep-MTL maintains stable and competitive $\Delta \mathrm{p}_{\text{task}}$ and $\Delta \mathrm{p}_{\text{metric}}$ over a substantial range, indicating its favorable robustness to learning rate variations.

of task saliencies as task-specific gradients in the representation space, it demonstrates superior efficiency compared to most gradient manipulation methods. This increased cost is inherent to approaches requiring second-order (gradient) information, representing a fundamental trade-off and room for further improvement in MTL optimization.

D.3. Analysis of Learning Rate Scaling

Recent studies [64] suggest that different choice of learning rate may impose a strong impact on MTO methods performance. To further demonstate Rep-MTL's robustness, we conduct experiment of learning rate sensitivity on NYUv2 [59] with diverse learning rate settings, as illustrated in Figure 8. Specifically, we scale the learning rate from the default benchmark setting of $1e-4$ to $5e-4$ with a step size of $5e-5$. For each setting, we measure the task-level ($\Delta \mathrm{p}_{\text{task}}$) and metric-level ($\Delta \mathrm{p}_{\text{metric}}$) performance gains. The results show that Rep-MTL maintains stable and competitive $\Delta \mathrm{p}_{\text{task}}$ and $\Delta \mathrm{p}_{\text{metric}}$ over a substantial range, indicating its favorable robustness to learning rate variations.