# Selective Task Group Updates for Multi-Task Optimization

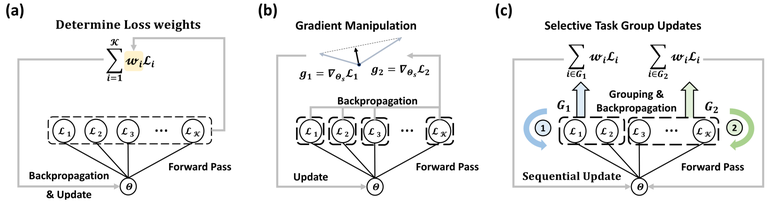
Wooseong Jeong, and Kuk-Jin Yoon
{stk14570, kjyoon}@kaist.ac.kr ; Visual Intelligence Lab, KAIST, Korea

## Background

- **Multi-task learning (MTL)** enables shared learning across tasks to improve generalization and efficiency, but often suffers from negative transfer due to conflicting task objectives.
- Prior methods mitigate this by balancing gradients or loss weights in shared parameters, yet overlook the learning dynamics of task-specific parameters.
- This work proposes a new optimization approach that dynamically groups tasks based on their affinity and updates them sequentially, effectively addressing both shared and task-specific parameter learning.

## Motivation

- Existing MTL methods adjust losses or gradients but overlook task-specific parameter interactions (see Figure 1).
- We find that grouping and updating tasks sequentially improves task specialization and reduces negative transfer.




- Inter-task affinity measures how the gradient update of one task affects the loss of another, but is impractical to track during optimization due to its computational cost.

**Definition 1** (Inter-Task Affinity). *Consider a multi-task network shared by tasks $i$ and $k$. For a data sample $z^t$ and a learning rate $\eta$, the task-specific gradients from $\mathcal{L}_i$ are applied to update the shared parameters of the network as follows: $\Theta_{s|i}^{t+1} = \Theta_s^t - \eta \nabla_{\Theta_s^t} \mathcal{L}_i(z^t, \Theta_s^t, \Theta_i^t)$. The inter-task affinity from task $i$ to task $k$ at time step $t$ is then defined as:*

$$\mathcal{A}_{i \to k}^t = 1 - \frac{\mathcal{L}_k(z^t, \Theta_{s|i}^{t+1}, \Theta_k^t)}{\mathcal{L}_k(z^t, \Theta_s^t, \Theta_k^t)} \tag{1}$$

## Main Results

- Proximal inter-task affinity approximates inter-task relations by measuring loss changes after joint updates of task groups, enabling efficient tracking during optimization.

**Definition 2** (**Proximal Inter-Task Affinity**). *Consider a multi-task network shared by the task set $G$, with their respective losses defined as $\mathcal{L}_G$. For a data sample $z^t$ and a learning rate $\eta$, the gradients of task set $G$ are updated to the parameters of the network as follows: $\Theta_{s|G}^{t+1} = \Theta_s^t - \eta \nabla_{\Theta_s^t} \mathcal{L}_G(z^t, \Theta_s^t, \Theta_G^t)$ and $\Theta_k^{t+1} = \Theta_k^t - \eta \nabla_{\Theta_k^t} \mathcal{L}_k(z^t, \Theta_s^t, \Theta_k^t)$ for $k \in G$. Then, the proximal inter-task affinity from the task set $G$ to $\tau_k$ at time step $t$ is defined as:*

$$\mathcal{B}_{G \to k}^t = 1 - \frac{\mathcal{L}_k(z^t, \Theta_{s|G}^{t+1}, \Theta_k^{t+1})}{\mathcal{L}_k(z^t, \Theta_s^t, \Theta_k^t)}$$

- Higher inter-task affinity between tasks leads to better gradient alignment during optimization.

**Theorem 1.** *Let $g_k$ denote the task-specific gradients backpropagated from the loss function $\mathcal{L}_k$ with respect to the parameters $\Theta_s^t$. At a given time step $t$, if the inter-task affinity from task group $\{i,k\}$ to task $k$ is greater than or equal to the inter-task affinity from group $\{j,k\}$ to task $k$, denoted as $A_{i,k \to k}^t \geq A_{j,k \to k}^t$. Then for a sufficiently small learning rate $\eta \ll 1$, it follows that $g_i \cdot g_k \geq g_j \cdot g_k$.*

- Better gradient alignment between tasks results in greater loss reduction on the reference task.

**Theorem 2.** *Let $g_k$ denote the task-specific gradients backpropagated from the loss function $\mathcal{L}_k$ with respect to the parameters $\Theta_s^t$. Let $\Theta_{s|i,k}^{t+1}$ represent the updated parameters after applying the gradients. Assume that for tasks $i$, $j$, and $k$, the inequality $g_i \cdot g_k \geq g_j \cdot g_k$ holds. Then, for a sufficiently small learning rate $\eta \ll 1$, the inequality $\mathcal{L}_k(z^t, \Theta_{s|i,k}^{t+1}, \Theta_k^t) \leq \mathcal{L}_k(z^t, \Theta_{s|j,k}^{t+1}, \Theta_k^t)$ holds.*

- Proximal inter-task affinity provides a reliable approximation of true inter-task affinity for guiding task grouping.

**Theorem 3.** *The affinity between $\{i,k\} \to k$ and $i \to k$ satisfies $\mathcal{A}_{i,k \to k}^t \geq \mathcal{A}_{i \to k}^t$.*

- Sequentially updating high-affinity task groups reduces multi-task loss more effectively than joint updates.

**Theorem 5.** *Consider three tasks $\{i,j,k\}$, where task groups are formed with positive inter-task affinity as $\{i,k\}$ and $\{j\}$. Assume all losses are convex and differentiable, and the change in affinity during a single step from $t+(m-1)/\mathcal{M}$ to $t+m/\mathcal{M}$ is negligible. The affinity, which learns all tasks jointly, is denoted as $\mathcal{B}_{i,j,k \to k}^{t+(m-1)/\mathcal{M}}$. The affinity for the updating sequence $(\{i,k\},\{j\})$ is represented as $\mathcal{B}_{i,k;j \to k}^{t+(m-1)/\mathcal{M}}$, while for the sequence $(\{j\},\{i,k\})$, it is represented as $\mathcal{B}_{j;i,k \to k}^{t+(m-1)/\mathcal{M}}$. Then, for a sufficiently small learning rate $\eta \ll 1$, the following holds:*

$$\mathcal{B}_{i,j,k \to k}^{t+(m-1)/\mathcal{M}} \simeq \mathcal{B}_{i,k;j \to k}^{t+(m-1)/\mathcal{M}} \qquad \mathcal{B}_{i,j,k \to k}^{t+(m-1)/\mathcal{M}} \leq \mathcal{B}_{j;i,k \to k}^{t+(m-1)/\mathcal{M}}$$

## Experimental Results

- Our group-wise update strategy avoids costly gradient manipulation while achieving efficient optimization through sequential task grouping.

Table 1. Comparison of time complexity and memory consumption between our optimization methods and other multi-task optimization approaches, including loss-based and gradient-based methods.

| Method | Forward Pass | Backpropagation | Gradient Manipulation | Optimizer Step | Affinity Update | Memory |
|---|---|---|---|---|---|---|
| Loss-based Methods | $\mathcal{O}(1)$ | $\mathcal{O}(1)$ | ✗ | $\mathcal{O}(1)$ | ✗ | $\mathcal{O}(1)$ |
| Gradient-based Methods | $\mathcal{O}(1)$ | $\mathcal{O}(\mathcal{K})$ | ✓ | $\mathcal{O}(1)$ | ✗ | $\mathcal{O}(\mathcal{K})$ |
| Ours | $\mathcal{O}(\mathcal{M})$ | $\mathcal{O}(\mathcal{M})$ | ✗ | $\mathcal{O}(\mathcal{M})$ | ✓ | $\mathcal{O}(1)$ |

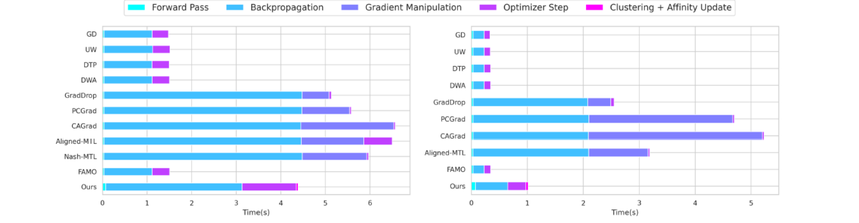
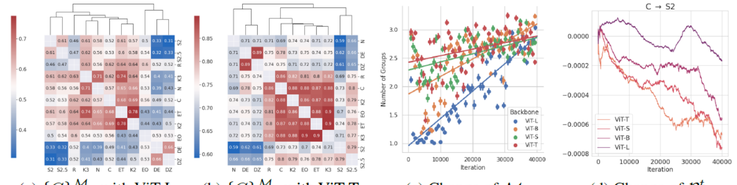



Figure 1. Comparison of the average time required by each optimization process to handle a single batch for 5tasks on PASCAL-Context (left) and 11 tasks on Taskonomy (right).

Table 2. Experimental results on the Taskonomy dataset using ViT-L.

| Task | DE | DZ | EO | ET | K2 | K3 | N | C | R | S2 | S2.5 | $\triangle_m$ (↑) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | RMSE ↓ | L1 Dist. ↓ | L1 Dist. ↓ | L1 Dist. ↓ | |
| Single Task | 0.0155 | 0.0160 | 0.1012 | 0.1713 | 0.1620 | 0.082 | 0.2169 | 0.7103 | 0.1357 | 0.1700 | 0.1435 | - |
| GD | 0.0163 | 0.0167 | 0.1211 | 0.1742 | 0.1715 | 0.093 | 0.2333 | 0.7527 | 0.1625 | 0.1837 | 0.1487 | -8.65 ±0.229 |
| GradDrop | 0.0168 | 0.0172 | 0.1229 | 0.1744 | 0.1727 | 0.091 | 0.2562 | 0.7615 | 0.1656 | 0.1862 | 0.1511 | -10.81 ±0.377 |
| MGDA | - | - | - | - | - | - | - | - | - | - | - | - |
| UW | 0.0167 | 0.0151 | 0.1212 | 0.1728 | 0.1712 | 0.089 | 0.2360 | 0.7471 | 0.1607 | 0.1829 | 0.1538 | -7.65 ±0.087 |
| DTP | 0.0169 | 0.0153 | 0.1213 | **0.1720** | 0.1707 | 0.089 | 0.2517 | 0.7481 | 0.1603 | 0.1814 | 0.1503 | -8.16 ±0.081 |
| DWA | 0.0147 | 0.0155 | 0.1209 | 0.1725 | 0.1711 | 0.089 | 0.2619 | 0.7486 | 0.1613 | 0.1845 | 0.1543 | -7.92 ±0.077 |
| PCGrad | 0.0162 | 0.0165 | 0.1206 | 0.1735 | 0.1696 | 0.090 | 0.2301 | 0.7540 | 0.1625 | 0.1830 | 0.1483 | -7.72 ±0.206 |
| CAGrad | 0.0162 | 0.0166 | 0.1202 | 0.1769 | 0.1651 | 0.091 | 0.2565 | 0.7653 | 0.1661 | 0.1861 | 0.1571 | -10.05 ±0.346 |
| IMTL | 0.0162 | 0.0165 | 0.1206 | 0.1741 | 0.1710 | 0.090 | 0.2268 | 0.7497 | 0.1617 | 0.1832 | 0.1543 | -8.03 ±0.179 |
| Aligned-MTL | 0.0150 | 0.0155 | **0.1135** | 0.1725 | **0.1630** | **0.086** | 0.2513 | 0.8039 | 0.1646 | 0.1800 | **0.1438** | -6.22 ±0.285 |
| Nash-MTL | - | - | - | - | - | - | - | - | - | - | - | - |
| FAMO | 0.0157 | 0.0155 | 0.1211 | 0.1730 | 0.1702 | 0.090 | 0.2433 | 0.7479 | 0.1610 | 0.1823 | 0.1527 | -7.58 ±0.211 |
| Ours | **0.0140** | **0.0145** | 0.1136 | 0.1735 | 0.1679 | 0.087 | **0.2029** | **0.7166** | **0.1500** | **0.1769** | 0.1469 | **-1.42** ±0.208 |




(a) $\{G\}_{i=1}^{\mathcal{M}}$ with ViT-L (b) $\{G\}_{i=1}^{\mathcal{M}}$ with ViT-T (c) Change of $\mathcal{M}$ (d) Change of $\mathcal{B}^t$

Figure 2. The averaged grouping results on the Taskonomy benchmark are shown for ViT-L in (a) and for ViT-T in (b). (c) illustrates how the number of task groups changes during optimization. (d) shows the change in proximal inter-task affinity from DE to C.