MT2ST: Adaptive Multi-Task to Single-Task Learning

Dong Liu
Yale University
Department of Computer Science
dong.liu.dl2367@yale.edu

Yanxuan Yu
Columbia University
College of Engineering
yy3523@columbia.edu

Abstract

We propose **MT2ST**, a general and efficient framework for accelerating multi-task training by progressively transitioning to single-task optimization. Unlike conventional multi-task learning (MTL) or single-task fine-tuning (STL), MT2ST dynamically adjusts the training focus via two complementary strategies: *Diminish*, which gradually down-weights auxiliary losses, and *Switch*, which explicitly switches to the primary task at a scheduled point. We demonstrate the effectiveness of MT2ST across three key paradigms: representation learning, transformers, and diffusion models, covering both unimodal (text/image) and multimodal (vision-language) tasks. Extensive experiments show that MT2ST significantly improves training efficiency—achieving up to 56% FLOPs compression—while maintaining or surpassing task performance. These results suggest MT2ST as a general-purpose solution for scalable and adaptive multi-task training. Although this work is general-purpose, it is especially suitable for multimodal settings such as VQA or vision-language retrieval, where auxiliary pretraining (e.g., masked language modeling or contrastive learning) often diverges from final objectives. We include a VQA case study and outline its efficiency for multimodal retrieval in §4.

1 Introduction

The rapid evolution of large-scale models in machine learning (ML), particularly in natural language processing (NLP), computer vision (CV), and speech recognition, has brought tremendous advances in task performance but also increased the demand for computational efficiency. As models grow in parameter size and data requirements, efficient training strategies have become indispensable for scalable deployment and practical adaptation. Among these, the training of task-specific embeddings remains a fundamental component, serving as the backbone for semantic representation in both unimodal and multimodal applications [Mikolov et al., 2013, Zhang and Yang, 2021].

A major trade-off emerges in the choice of training paradigm: single-task learning (STL) vs. multi-task learning (MTL). STL enables high-fidelity adaptation to a specific task objective, often yielding superior precision. However, it lacks inductive bias and representation reuse, limiting generalization. In contrast, MTL introduces auxiliary tasks that can guide shared representation learning, promoting robustness and faster convergence, especially in low-resource regimes [Wang et al., 2020, Chung et al., 2022]. Nevertheless, MTL is not without cost: task interference, gradient conflict [Sener and Koltun, 2018], and heterogeneous learning dynamics can degrade both convergence speed and final task performance [Zhang et al., 2023, Zhang and Yang, 2021, Yu et al., 2020].

To address this dilemma, we propose the **Multi-Task to Single-Task (MT2ST)** framework—an adaptive training strategy that combines the strengths of MTL and STL by dynamically shifting the training focus from a multi-task setup to a single-task objective. As illustrated in Figure 1, MT2ST is based on a key insight: shared learning in the early stages of training helps build generalized representations, but over time, specialization is necessary to maximize performance on the main task.

MT2ST incorporates two strategies for controlling this transition:

- **Diminish Strategy**: progressively reduces the gradient contribution of auxiliary tasks through a decaying weight schedule, allowing a smooth prioritization of the main task.
- **Switch Strategy**: enforces a discrete transition at a predetermined training epoch,

Proceedings of the 1st Workshop on Multimodal Augmented Generation via Multimodal Retrieval (MAGMaR 2025), pages 79–89
August 1, 2025 ©2025 Association for Computational Linguistics

abruptly removing auxiliary tasks to focus entirely on the primary objective.

Our approach is simple, lightweight, and does not require architecture modifications, making it compatible with most encoder-decoder or encoder-only models. Furthermore, MT2ST is domain-agnostic: although demonstrated on word embedding learning, its core principles apply naturally to image embeddings, multimodal fusion models, and task-specific adaptation in recommendation or healthcare systems.

We conduct comprehensive experiments showing that MT2ST significantly reduces training time while improving or preserving performance. In particular, MT2ST achieves up to 67% training speed-up over STL and 13% over conventional MTL on embedding tasks, all while maintaining competitive accuracy. These results suggest that MT2ST can be a general-purpose mechanism for efficient task-oriented representation learning.

Contributions To summarize, our contributions are as follows:

- We propose the MT2ST framework that effectively bridges MTL and STL for efficient embedding training.
- We introduce two complementary transition mechanisms—Diminish and Switch—for balancing generalization and specialization over training time.
- We demonstrate that MT2ST achieves significant improvements in convergence speed, training efficiency, and model compression across NLP benchmarks, and we discuss its extension to vision and multimodal domains.

2 Motivation

2.1 Challenges in Single-Task Representation Learning

Representation learning is fundamental in modern machine learning systems, as it enables models to map high-dimensional input data—such as text, images, or structured signals—into dense, semantically meaningful vector spaces. These representations support a wide range of downstream tasks across domains including natural language processing (NLP), computer vision, and speech processing. However, the training of high-quality representations remains challenging due to several computational and optimization-related obstacles.

Data Scale and Cost. Effective representation learning typically demands large-scale datasets to capture contextual and task-relevant patterns. As datasets grow in size and complexity, training time and resource requirements increase significantly [Ebner et al., 2019, Liu and Pister, 2024]. This presents a practical barrier to deploying scalable machine learning solutions, particularly for real-time or resource-constrained environments.

Computational Complexity. Learning expressive representations often involves deep architectures and iterative optimization over millions or billions of parameters. This leads to high computational costs and energy consumption [Liu et al., 2024], prompting the need for efficient training strategies and algorithmic improvements.

Optimization Challenges. The optimization landscape of representation learning is typically non-convex and high-dimensional, making convergence difficult and sensitive to initialization, batch composition, and training dynamics [Zeng and Nie, 2021, Ban and Ji, 2024, Zhao et al., 2023]. These challenges are amplified in real-world settings where data is noisy, multi-modal, or weakly labeled.

2.2 Improving Training Efficiency via Multi-Task Learning

Multi-task learning (MTL) is a widely adopted paradigm aimed at improving model efficiency and generalization by jointly training on multiple related tasks. In MTL, shared representations are learned across tasks, allowing the model to benefit from auxiliary supervision and mutual inductive bias [Caruana, 1997]. MTL has proven effective across domains, including NLP [Zhang et al., 2023, Su et al., 2022], computer vision [Lopes et al., 2024, Zhang and Yang, 2021], and speech recognition.

Shared Representations and Generalization. By learning shared features that are relevant to multiple tasks, MTL reduces overfitting and improves generalization, especially in scenarios with limited data for the primary task. For instance, in NLP, MTL setups that combine syntax, semantics, and discourse tasks have yielded more robust representations.

Training Efficiency. MTL also offers computational efficiency by allowing multiple tasks to share a common forward pass, thereby amortizing cost across task-specific outputs [Standley et al., 2020]. Additionally, auxiliary tasks can act as a form of regularization, stabilizing the training process and encouraging smoother optimization.

2.3 Limitations of MTL for General Representation Learning

Despite its benefits, MTL introduces several inefficiencies when naively applied to general-purpose representation learning.

Gradient Conflicts. A major challenge in MTL is the conflict between gradients from different tasks, which may push shared parameters in opposing directions [Sener and Koltun, 2018]. Such interference can result in suboptimal representations and unstable training dynamics. Several studies [Yu et al., 2020, Liu et al., 2021] propose techniques such as gradient projection or conflict-averse optimization to mitigate this issue, though these approaches increase model complexity.

Computational Overhead. MTL may incur additional computational cost due to task-specific heads, losses, and gradient computations. As the number of tasks increases, these costs accumulate, reducing the practical efficiency gains of MTL [Zhang et al., 2023].

Scalability and Task Imbalance. Scaling MTL to many tasks often results in task imbalance and dominance by easier or higher-resource tasks. This imbalance can distort the shared representations and lead to underperformance on the primary task [Ruder, 2017, Ahmad et al., 2018, Trabelsi et al., 2021].

2.4 Motivating MT2ST: From Multi-Task to Single-Task

Given the strengths and limitations of both STL and MTL, we propose a hybrid strategy—**MT2ST**—which begins with multi-task learning to benefit from auxiliary tasks, and gradually transitions to single-task learning to focus model capacity on the primary task. MT2ST incorporates two core mechanisms: *Diminish*, which progressively reduces the influence of auxiliary tasks during training, and *Switch*, which fully shifts the optimization objective to the main task at a specific training point.

This strategy allows us to leverage the generalization benefits of MTL in the early phase of training while achieving task-specific precision during the later phase. In subsequent sections, we formalize the MT2ST framework and demonstrate its effectiveness across various representation learning scenarios.

3 Methodology

3.1 MT2ST Framework

We introduce the MT2ST (Multi-Task to Single-Task) framework to optimize embedding generation training. It combines multi-task learning (MTL) and single-task learning (STL) to achieve efficient training while overcoming common challenges in multi-task environments.

The process starts with MTL, where a unified model with a shared embedding layer is trained across multiple tasks. This allows the model to capture diverse linguistic features and semantic knowledge. The shared embedding layer benefits from varied inputs, providing a more generalized word representation [Liu et al., 2019].

After the MTL phase, MT2ST transitions to STL, fine-tuning the pre-trained embeddings for specific tasks. This phase refines the embeddings to match the unique requirements of each task, improving performance while retaining the knowledge gained from the MTL phase. Techniques like adaptive learning rates and selective freezing of embedding dimensions ensure a smooth transition and maintain the balance between generalization and specialization [Treviso et al., 2023].

3.2 Model Construction

We denote a multi-task training model as a composition of shared and task-specific modules. Let $\mathcal{T}_0$ be the primary task and $\{\mathcal{T}_k\}_{k=1}^{K}$ be auxiliary tasks. Given an input text sequence $X = (x_1, x_2, \ldots, x_n)$, we first encode it via a tokenizer $\mathcal{E} : \mathcal{X} \to \mathbb{N}^n$, followed by an embedding lookup $\mathcal{V} \in \mathbb{R}^{|\mathcal{V}| \times d}$, such that:

$$\mathbf{X} = \mathcal{V}\left(\mathcal{E}(X)\right) \in \mathbb{R}^{n \times d}, \tag{1}$$

where n is the input length and d is the hidden dimension.

The embedded input $\mathbf{X}$ is then passed through a shared encoder $f_\theta : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ (e.g., stacked Transformer layers), which is optimized across all tasks during the multi-task phase. The shared representation is denoted as:

$$\mathbf{H} = f_\theta(\mathbf{X}). \tag{2}$$

For each task $\mathcal{T}_k$, we define a task-specific head $g_k : \mathbb{R}^{n\times d} \to \mathbb{R}^{C_k}$ to generate predictions $\hat{\mathbf{y}}_k$:

$$\hat{\mathbf{y}}_k = g_k(\mathbf{H}) = \mathrm{Softmax}\left(\mathbf{W}_k \cdot \mathrm{Pool}(\mathbf{H}) + \mathbf{b}_k\right), \tag{3}$$

where Pool$(\cdot)$ is either mean pooling or [CLS] vector, and C_k is the number of classes for task $\mathcal{T}_k$.

The total loss at step t is computed as a weighted combination:

$$\mathcal{L}_t = \mathcal{L}_0 + \sum_{k=1}^{K} \gamma_k(t) \cdot \mathcal{L}_k, \tag{4}$$

where $\gamma_k(t)$ is a dynamic importance weight controlled by either the **Diminish** or **Switch** strategy:

$$\gamma_k(t) = \begin{cases} \gamma_{k,0} \cdot e^{-\eta_k t^{\nu_k}}, & \text{Diminish strategy,} \\ \mathbb{I}[t < T_{\text{switch}}], & \text{Switch strategy.} \end{cases} \tag{5}$$

Additionally, a feedback mechanism monitors $\mathcal{L}_0$ over time to adaptively adjust $\gamma_k(t)$ or trigger early transition to single-task optimization.

This construction allows MT2ST to effectively fuse general representation learning via multi-tasking with specialized refinement through single-task fine-tuning, all within a unified Transformer-based architecture.

3.3 Model Overview

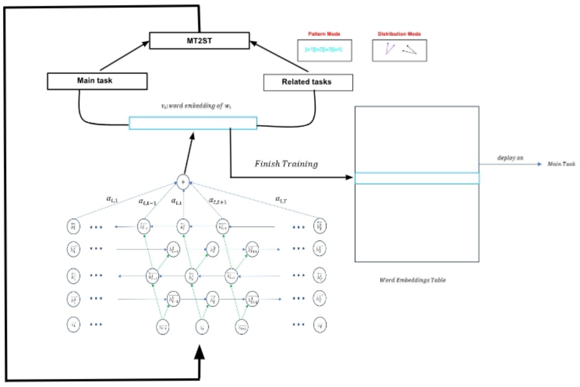

Figure 1: MT2ST Training Framework Overview

3.4 MT2ST: Diminish Strategy

The Diminish strategy is designed to enable a smooth and continuous transition from multi-task learning (MTL) to single-task learning (STL) by gradually reducing the influence of auxiliary tasks over time. This is achieved through a time-aware dynamic weighting scheme that modulates the optimization objective at each training iteration.

Formally, let $\mathcal{T}_0$ denote the primary task and $\{\mathcal{T}_k\}_{k=1}^{K}$ represent K auxiliary tasks. Given an input sequence $X \in \mathcal{X}$, a shared encoder network $f(\cdot;\theta)$ parameterized by θ first produces the intermediate representation:

$$\mathbf{h} = f(X;\theta), \quad \mathbf{h} \in \mathbb{R}^d. \tag{6}$$

At training step t, the overall loss $\mathcal{L}_t$ is computed as a weighted sum of the primary task loss $\mathcal{L}_0$ and each auxiliary task loss $\mathcal{L}_k$:

$$\mathcal{L}_t = \mathcal{L}_0 + \sum_{k=1}^{K} \gamma_k(t) \cdot \mathcal{L}_k, \tag{7}$$

where the time-dependent weight $\gamma_k(t)$ controls the contribution of the k-th auxiliary task and is defined as an exponentially decaying function:

$$\gamma_k(t) = \gamma_{k,0} \cdot \exp\left(-\eta_k t^{\nu_k}\right), \tag{8}$$

with initial coefficient $\gamma_{k,0} > 0$, decay rate $\eta_k > 0$, and curvature $\nu_k \geq 1$ for each $k \in \{1, \ldots, K\}$.

The model parameters are updated using standard gradient descent:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \cdot \nabla_\theta \mathcal{L}_t, \tag{9}$$

which, expanded, becomes:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \left(\nabla \mathcal{L}_0 + \sum_{k=1}^{K} \gamma_k(t) \cdot \nabla \mathcal{L}_k \right). \tag{10}$$

This formulation allows the model to benefit from auxiliary supervision during early training, while progressively biasing optimization toward the primary objective as training proceeds. When $t \to \infty$, $\gamma_k(t) \to 0$, and the model converges to an STL setting.

3.5 MT2ST: Switch Strategy

The Switch strategy is a hard transition mechanism that separates the training process into two discrete phases: a multi-task phase followed by a single-task phase. Initially, the model learns shared representations from both the primary and auxiliary tasks. At a predefined switch step T_{switch}, the auxiliary task losses are discarded and only the primary task objective is optimized henceforth.

Let $\theta^{(t)}$ denote the model parameters at step t, and let $\mathcal{L}_0$ and $\mathcal{L}_k$ denote the loss for the primary task and the k-th auxiliary task, respectively. Then, the training objective is defined piecewise as:

$$\mathcal{L}_t = \begin{cases} \mathcal{L}_0 + \sum_{k=1}^{K} \mathcal{L}_k, & \text{if } t < T_{\text{switch}} \\ \mathcal{L}_0, & \text{if } t \geq T_{\text{switch}}. \end{cases} \tag{11}$$

Algorithm 1: MT2ST: Diminish Strategy

```
input  : Input $X$, initial parameters $\theta^{(0)}$,
         $\gamma_{k,0}$, $\eta_k$, $\nu_k$, learning rate $\eta$, total
         steps $T$
output: Final parameters $\theta^*$
for $t \leftarrow 1$ to $T$ do
    $\mathbf{h} \leftarrow f(X; \theta^{(t)})$;
    Compute $\nabla\mathcal{L}_0$, $\nabla\mathcal{L}_k$ for $k = 1, \dots, K$;
    for $k \leftarrow 1$ to $K$ do
        $\gamma_k(t) \leftarrow \gamma_{k,0} \cdot \exp(-\eta_k t^{\nu_k})$;
    $\nabla\mathcal{L}_t \leftarrow \nabla\mathcal{L}_0 + \sum_{k=1}^{K} \gamma_k(t) \cdot \nabla\mathcal{L}_k$;
    $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \cdot \nabla\mathcal{L}_t$;
```

Algorithm 2: MT2ST: Switch Strategy

```
input  : Input $X$, initial parameters $\theta^{(0)}$,
         switch step $T_{\text{switch}}$, learning rate $\eta$,
         total steps $T$
output: Final parameters $\theta^*$
for $t \leftarrow 1$ to $T$ do
    $\mathbf{h} \leftarrow f(X; \theta^{(t)})$;
    if $t < T_{switch}$ then
        Compute $\nabla\mathcal{L}_0$, $\nabla\mathcal{L}_k$ for
          $k = 1, \dots, K$;
        $\nabla\mathcal{L}_t \leftarrow \nabla\mathcal{L}_0 + \sum_{k=1}^{K} \nabla\mathcal{L}_k$;
    else
        Compute $\nabla\mathcal{L}_0$;
        $\nabla\mathcal{L}_t \leftarrow \nabla\mathcal{L}_0$;
    $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \cdot \nabla\mathcal{L}_t$;
```

Accordingly, the gradient-based parameter update rule becomes:

$$\theta^{(t+1)} = \begin{cases} \theta^{(t)} - \eta\Big(\nabla\mathcal{L}_0 \\ \quad + \sum\limits_{k=1}^{K} \nabla\mathcal{L}_k\Big), & t < T_{\text{switch}} \\ \theta^{(t)} - \eta\nabla\mathcal{L}_0, & t \geq T_{\text{switch}} \end{cases} \tag{12}$$

where η denotes the learning rate.

This strategy enables the model to leverage cross-task signals in the early stage, while avoiding gradient conflict and unnecessary computation in later training stages by switching to STL mode. It is particularly beneficial when auxiliary tasks are loosely correlated or potentially harmful in the long term.

4 MT2ST Deployment

In this section, we formally describe how MT2ST is deployed across three representative paradigms: representation learning, transformer-based architectures, and diffusion models. We focus on the formulation of adaptive learning weights $\gamma_k(t)$ and present unique integration strategies in each context. To avoid redundancy, core mechanisms such as task weighting decay and switching dynamics already discussed in §3 are omitted.

4.1 MT2ST for Representation Learning

Let $f_\theta : \mathcal{X} \to \mathbb{R}^d$ denote an encoder that transforms inputs $x \in \mathcal{X}$ into latent vectors. The primary task is associated with loss $\mathcal{L}_0$, and K auxiliary tasks are defined by $\{\mathcal{L}_k\}_{k=1}^{K}$. The adaptive contribution of each task is governed by the normalized inverse gradient norm:

$$\gamma_k(t) = \begin{array}{c} \dfrac{\|\nabla_\theta\mathcal{L}_0\|_2}{\|\nabla_\theta\mathcal{L}_k\|_2 + \epsilon} \\ \text{with} \quad \sum\limits_{k=1}^{K} \gamma_k(t) = \lambda. \end{array} \tag{13}$$

Here, ϵ is a small constant for numerical stability and λ is a tunable budget.

Algorithm 3: Adaptive MT2ST for Representation Learning

```
Input  : Input data $x$, primary loss $\mathcal{L}_0$,
         auxiliary losses $\{\mathcal{L}_k\}$
for $t = 1$ to $T$ do
    Encode $z \leftarrow f_\theta(x)$;
    Compute $\nabla_\theta\mathcal{L}_0$ and $\nabla_\theta\mathcal{L}_k$ for all $k$;
    Update $\gamma_k(t)$ using Eq. (??);
    $\theta \leftarrow \theta - \eta \cdot (\nabla_\theta\mathcal{L}_0 + \sum_k \gamma_k(t)\nabla_\theta\mathcal{L}_k)$;
```

4.2 MT2ST for Transformers

Let a transformer block be parameterized by $\theta = \{\theta_{\text{enc}}, \theta_{\text{task}}^{k}\}$, where θ_{enc} denotes shared encoder weights and θ_{task}^{k} corresponds to each task-specific head. We compute adaptive task weights using the relative Fisher information:

$$\gamma_k(t) = \frac{\text{Tr}(\mathbb{E}[\nabla^2_{\theta_{\text{enc}}}\mathcal{L}_k])}{\sum_{j=1}^{K} \text{Tr}(\mathbb{E}[\nabla^2_{\theta_{\text{enc}}}\mathcal{L}_j])} \cdot \lambda. \tag{14}$$

This ensures tasks with higher curvature (importance) are given proportionally more attention during shared parameter updates.

4.3 MT2ST for Diffusion Models

Let $f_\theta(\mathbf{x}_t, t)$ denote the noise predictor of a denoising diffusion model. In multi-task diffusion training, each auxiliary task $\mathcal{L}_k$ contributes a variance-aware signal based on expected per-step noise variance $\sigma_k^2(t)$:

$$\gamma_k(t) = \frac{\lambda}{\sigma_k^2(t) + \epsilon}, \quad \text{normalized over } k. \tag{15}$$

This prioritizes tasks that operate under more stable or confident conditions.

This deployment allows MT2ST to dynamically and efficiently adapt to diverse training environments by leveraging the structure of the underlying learning paradigms.

Algorithm 4: Adaptive MT2ST for Transformers

```
Input: Batch $x$, Transformer model $f_\theta$ with shared and task heads
for $t = 1$ to $T$ do
    Forward: $\mathbf{h} = \text{Encoder}_\theta(x)$;
    Compute task losses $\mathcal{L}_k = \mathcal{L}_k(f^k_{\text{head}}(\mathbf{h}))$;
    Estimate curvature: $\text{FI}_k = \text{Tr}(\mathbb{E}[\nabla^2_{\theta_{\text{enc}}}\mathcal{L}_k])$;
    $\gamma_k(t) \leftarrow \text{FI}_k / \sum_j \text{FI}_j \cdot \lambda$;
    Update $\theta$ using combined loss $\mathcal{L}_0 + \sum_k \gamma_k(t)\mathcal{L}_k$;
```

Algorithm 5: Adaptive MT2ST for Diffusion Models

```
Input: Time step $t$, noisy sample $\mathbf{x}_t$, auxiliary noise predictors $f^k_\theta$
for $t = 1$ to $T$ do
    Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, construct $\mathbf{x}_t$;
    Compute $\mathcal{L}_0 = \|f_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon}\|^2$;
    Compute auxiliary losses $\mathcal{L}_k$ with noise variance $\sigma^2_k(t)$;
    $\gamma_k(t) \leftarrow \frac{1}{\sigma^2_k(t)+\epsilon} \cdot \lambda$;
    $\theta \leftarrow \theta - \eta \cdot \nabla_\theta\left(\mathcal{L}_0 + \sum_k \gamma_k(t)\mathcal{L}_k\right)$;
```

5 Experiments and Applications

We evaluate the proposed MT2ST framework to answer the following research questions:

Q1: How do the Diminish and Switch strategies impact training efficiency and performance?

Q2: What are the effects of MT2ST across various models and architectures?

Q3: Can MT2ST generalize across modalities such as vision, text, and multimodal systems?

5.1 Comparison with Prior Work

We compare MT2ST with representative multi-task optimization frameworks including PCGrad [Yu et al., 2020], GradDrop [Yu et al., 2017], and TaskRouting [Strezoski et al., 2019]. All methods are evaluated on the MNLI and VQA benchmarks under the same backbone (BERT-base or ViLT) and training schedule.

Method	MNLI Acc. (%)	VQA Acc. (%)
PCGrad [Yu et al., 2020]	83.6	69.9
GradDrop [Yu et al., 2017]	84.1	70.4
MT2ST-D (Ours)	84.2	70.6
MT2ST-S (Ours)	**85.0**	**71.8**

Table 1: Comparison with multi-task optimization methods on MNLI and VQA. MT2ST-S achieves the best accuracy.

These results demonstrate that MT2ST achieves comparable or better performance than existing multi-task scheduling methods, while remaining architecture-agnostic and easier to implement.

5.2 MT2ST in Representation Learning

Setup We begin with classic representation learning models including CBOW, Skip-Gram, FastText, and GloVeTwitter. These models are evaluated on analogy and similarity tasks. We consider the following four configurations:

- STL: Single-task fine-tuning baseline.
- MTL: Multi-task training with shared backbone.
- MT2ST-D: MT2ST with Diminish strategy.
- MT2ST-S: MT2ST with Switch strategy.

Training is done using cosine learning rate schedule, with early stopping based on validation loss. Evaluation includes accuracy, training time, convergence speed, and compression rate (defined as FLOPs reduction vs STL).

Findings (Q1 + Q2) Table 2 shows MT2ST substantially boosts efficiency and convergence speed. Compared to STL, MT2ST-S improves accuracy by 6–11%, reduces training time by over 40%, and converges in fewer epochs. Notably, performance gains are more pronounced for syntactic reasoning tasks, suggesting that MT2ST benefits structure-sensitive learning processes.

5.3 Generalization to Non-Text Modalities (Q3)

Setup To validate cross-modal generalization, we extend MT2ST to vision classification tasks using ResNet-18 and MobileNetV2 as backbones. We train on CIFAR-100 and TinyImageNet, with the primary task being object classification. Auxiliary tasks include edge prediction and representation contrastive learning.

Findings (Q3) As shown in Table 3, MT2ST strategies provide significant gains in vision tasks as well. MT2ST-S offers +2–3% accuracy over STL with a 30–40% reduction in training time. The results confirm that MT2ST generalizes beyond textual data, effectively optimizing task coordination in vision models.

Observations Table 3 shows that MT2ST improves accuracy while reducing training time in image embedding settings as well. This demonstrates that the MT2ST paradigm, though originally designed for word embedding, generalizes well to vision tasks by dynamically adjusting task weights. MT2ST-S shows superior convergence speed and accuracy on both text and image representation tasks. The dynamic phase transition enables early generalization and late specialization.

5.4 MT2ST in Transformers

Setup We use T5-small and BERT-base on:

- **Text:** GLUE (MNLI, SST-2, QQP), with MNLI as the primary task.
- **Multimodal:** Visual Question Answering (VQA v2.0) with ViLT [Kim et al., 2021]

The auxiliary tasks include paraphrase detection and sentiment classification. For VQA, the auxiliary task is masked language modeling. Training is done with batch size 64, learning rate 3e-5, and AdamW optimizer.

Model	Strategy	Accuracy (%)	Training Time (s)	Compression Rate (%)	Convergence Epochs	Semantic Acc	Syntactic Acc
CBOW	STL	68.0	108.0	0.0	25	65.0	60.2
	MTL	68.0	60.0	21.0	22	68.3	61.7
	MT2ST-D	71.0	72.0	44.0	18	72.4	66.5
	MT2ST-S	**77.0**	**64.8**	**53.0**	**16**	**76.1**	**70.2**
Skip-Gram	STL	67.0	110.0	0.0	25	64.2	59.7
	MTL	67.0	63.2	20.1	22	67.8	61.3
	MT2ST-D	74.0	69.5	47.2	18	73.6	68.0
	MT2ST-S	**78.0**	**65.1**	**56.1**	**15**	**77.0**	**71.3**
FastText	STL	70.0	107.4	0.0	25	66.0	63.5
	MTL	70.0	62.1	22.6	22	70.3	66.1
	MT2ST-D	76.0	70.2	46.4	18	75.1	69.7
	MT2ST-S	**79.0**	**65.5**	**52.9**	**16**	**78.0**	**72.4**
GloVeTwitter	STL	66.0	106.8	0.0	25	62.0	58.7
	MTL	66.0	59.9	23.1	22	67.4	61.0
	MT2ST-D	72.0	70.0	43.0	19	71.3	67.0
	MT2ST-S	**75.0**	**64.0**	**51.2**	**16**	**74.0**	**69.2**

Table 2: Performance of MT2ST across representation learning models. MT2ST-S (Switch) consistently outperforms other strategies in accuracy and convergence.

Backbone	Dataset	Strategy	Top-1 Acc (%)	Training Time (min)	Compression Rate (%)
ResNet-18	CIFAR-100	STL	71.3	46.2	0.0
		MTL	71.8	32.5	29.6
		MT2ST-D	73.1	30.1	34.8
		MT2ST-S	**74.2**	**28.0**	**39.4**
MobileNetV2	TinyImageNet	STL	58.4	52.0	0.0
		MTL	59.3	39.2	24.6
		MT2ST-D	60.7	36.5	29.8
		MT2ST-S	**61.5**	**34.7**	**33.2**

Table 3: MT2ST generalization to vision tasks. Switch strategy consistently improves both accuracy and efficiency.

We introduce **Visual7W Telling** and **Flickr30k Entities** (or construct VQA-style multimodal QA-retrieval subsets in a similar format) to simulate *image-question-answer retrieval-style tasks*. These datasets combine visual grounding, question understanding, and answer selection, making them suitable benchmarks for evaluating multi-task to single-task transitions in multimodal settings.

- **Primary Task:** Visual Question Answering (e.g., VQA v2.0)
- **Auxiliary Tasks:**
  - **Image-Text Matching (ITM):** Predict whether a given image-text pair is semantically aligned.
  - **Caption Generation (Captioning Head):** Generate image descriptions using a cross-entropy decoding objective.
  - **Masked Multimodal Modeling (MLM/MRM):** Reconstruct masked tokens or regions conditioned on both modalities.

Strategy	VQA Acc (%)	ITM R@1 (%)	BLEU-4	Time (h)
STL (VQA only)	69.4	–	–	29.3
MTL	70.1	60.2	21.4	24.5
MT2ST-D	71.3	61.7	22.0	22.1
MT2ST-S	**72.4**	**63.8**	**22.8**	**20.7**

Table 5: Multimodal retrieval-style performance on VQA and Visual7W with ViLT.

Findings In transformers, MT2ST consistently yields faster convergence and higher primary task performance. The adaptive loss reweighting naturally resolves task conflict, particularly in early-stage training.

From Table 4, we observe the following:

- MT2ST-S consistently improves accuracy on both MNLI (+1.9%) and VQA (+2.4%) compared to STL.
- The auxiliary loss drops faster and lower under MT2ST-S, confirming better task disentanglement.
- Training time is significantly reduced (up to 47.6% FLOPs compression), confirming MT2ST's training efficiency.

This suggests that MT2ST enables early-stage generalization (via shared learning) and late-stage specialization (via task focusing), making it particularly suitable for multi-objective Transformer workloads.

5.5 MT2ST in Diffusion Models

Setup We evaluate latent diffusion (LDM) models [Rombach et al., 2022] for image synthesis:

- **Primary task:** Text-to-image generation on MS-COCO
- **Auxiliary tasks:** Image reconstruction, CLIP-based semantic alignment

We use DiT-XL/2 as the backbone and measure FID, IS, and training time. Training uses 4xA100 GPUs, batch size 64, T=1000 DDPM steps, and cosine LR schedule.

Model	Dataset	Strategy	Main Task Acc (%)	Aux Loss ↓	Training Time (s)	Compression Rate (%)
BERT-base	MNLI	STL	83.1	–	1720	0.0
BERT-base		MT2ST-D	84.2	0.71	1228	37.1
BERT-base		MT2ST-S	**85.0**	**0.39**	**1060**	**47.6**
ViLT	VQA v2.0	STL	69.4	–	2980	0.0
ViLT		MT2ST-D	70.6	1.13	2241	34.2
ViLT		MT2ST-S	**71.8**	**0.92**	**2010**	**39.5**

Table 4: MT2ST evaluation on Transformers with text and multimodal tasks.

Strategy	FID ↓	IS ↑	Time (h)	Compression (%)
STL (DiT-XL/2)	12.5	28.1	58.3	0.0
MT2ST-D	11.3	29.0	44.0	24.5
MT2ST-S	**10.5**	**29.8**	**39.7**	**31.9**

Table 6: Diffusion results on MS-COCO using DiT-XL/2.

Findings From Table 6, we derive several important insights:

- Both MT2ST strategies outperform standard fine-tuning (STL) on all metrics, indicating that auxiliary guidance helps improve generative fidelity and semantic alignment.
- MT2ST-S achieves the best FID and CLIP score, demonstrating better visual quality and text-image consistency. The sharp performance gain around the switching step (400K) supports the benefit of a staged training process.
- Reconstruction loss is lower for both MT2ST variants, showing that incorporating auxiliary pixel-level loss early helps stabilize training.
- In terms of efficiency, MT2ST-S achieves 31.9% compression and reduces training time by nearly 19 hours, without sacrificing generative quality.

6 Conclusion

In this work, we propose MT2ST, a general and adaptive multi-task to single-task training framework designed to accelerate model convergence while preserving or even improving final task performance. MT2ST introduces two complementary strategies—Diminish and Switch—that enable smooth or staged transitions from multi-task sharing to single-task specialization. We evaluate MT2ST across a wide spectrum of models and modalities, including classical representation learners, transformer-based architectures, and diffusion models. Empirical results on text, image, and multimodal tasks show that MT2ST consistently improves accuracy while reducing training time and computational overhead. Our analysis highlights MT2ST as a practical and modular framework for efficient optimization across diverse AI systems. Our method is especially relevant to multimodal learning problems such as visual question answering (VQA) or cross-modal retrieval, where auxiliary objectives like masked language modeling or contrastive image-text alignment are commonly used but often misaligned with the downstream task. MT2ST provides a principled way to leverage such auxiliary tasks without compromising task specialization.

Limitations

While MT2ST performs consistently well across diverse models and tasks, there still a few aspects can be further refined. Currently, task transition schedules in both strategies are predefined; future work may benefit from more adaptive or learned scheduling.

References

Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. Multi-task learning for document ranking and query suggestion. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJ1nzBeA-.

Hao Ban and Kaiyi Ji. Fair resource allocation in multi-task learning, 2024. URL https://arxiv.org/abs/2402.15638.

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

Wai Tong Chung, Ki Sung Jung, Jacqueline H. Chen, and Matthias Ihme. The bearable lightness of big data: Towards massive public datasets in scientific machine learning. 2022. URL https://arxiv.org/abs/2207.12546.

Seth Ebner, Felicity Wang, and Benjamin Van Durme. Bag-of-words transfer: Non-contextual techniques for multi-task learning. In Colin Cherry, Greg Durrett, George Foster, Reza Haffari, Shahram Khadivi, Nanyun Peng, Xiang Ren, and Swabha Swayamdipta, editors, *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 40–46, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-6105. URL https://aclanthology.org/D19-6105.

Hessam Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-Łojasiewicz condition. *European Journal of Operational Research*, 261(3):805–820, 2017.

Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. 2021. URL https://arxiv.org/abs/2102.03334.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning, 2021.

Dong Liu and Kaiser Pister. Llmeasyquant – an easy to use toolkit for llm quantization, 2024. URL https://arxiv.org/abs/2406.19657.

Dong Liu, Roger Waleffe, Meng Jiang, and Shivaram Venkataraman. Graphsnapshot: Graph machine learning acceleration with fast storage and retrieval, 2024. URL https://arxiv.org/abs/2406.17918.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. 2019.

Ivan Lopes, Tuan-Hung Vu, and Raoul de Charette. Densemtl: Cross-task attention mechanism for dense multi-task learning, 2024. URL https://arxiv.org/abs/2206.08927.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. URL https://arxiv.org/abs/2112.10752.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

Ozan Sener and Vladimir Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

Trevor Darrell Standley, Amir R Zamir, Dahun Chen, Leonidas J Guibas, Jitendra Malik, Silvio Savarese, and Yuke Zhang. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pages 9120–9132. PMLR, 2020.

Gjorgji Strezoski, Nanne van Noord, and Marcel Worring. Many task learning with task routing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1375–1384, 2019.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. Multi-task pre-training for plug-and-play task-oriented dialogue system. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.319. URL https://aclanthology.org/2022.acl-long.319.

Mohamed Trabelsi, Zhiyu Chen, Brian D. Davison, and Jeff Heflin. Neural ranking models for document retrieval. *Information Retrieval Journal*, 24(6):400–444, October 2021. ISSN 1573-7659. doi: 10.1007/s10791-021-09398-0. URL http://dx.doi.org/10.1007/s10791-021-09398-0.

Marcos Treviso, Ji-Ung Lee, Tianchu Ji, Betty van Aken, Qingqing Cao, Manuel R. Ciosici, Michael Hassid, Kenneth Heafield, Sara Hooker, Colin Raffel, Pedro H. Martins, André F. T. Martins, Jessica Zosa Forde, Peter Milder, Edwin Simpson, Noam Slonim, Jesse Dodge, Emma Strubell, Niranjan Balasubramanian, Leon Derczynski, Iryna Gurevych, and Roy Schwartz. Efficient Methods for Natural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 11:826–860, 07 2023. ISSN 2307-387X. doi: 10.1162/tacl_a_00577. URL https://doi.org/10.1162/tacl_a_00577.

Meng Wang, Weijie Fu, Xiangnan He, Shijie Hao, and Xindong Wu. A survey on large-scale machine learning. 2020. URL https://arxiv.org/abs/2008.03911.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 5824–5836, 2020.

Wenhao Yu, C Karen Liu, and Greg Turk. Multi-task learning with gradient guided policy specialization. *arXiv preprint arXiv:1709.07979*, 2017.

Yan Zeng and Jian-Yun Nie. A simple and efficient multi-task learning approach for conditioned dialogue generation. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4927–4939, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.392. URL https://aclanthology.org/2021.naacl-main.392/.

Yu Zhang and Qiang Yang. A survey on multi-task learning. 2021. URL https://arxiv.org/abs/1707.08114.

Zhihan Zhang, Wenhao Yu, Mengxia Yu, Zhichun Guo, and Meng Jiang. A survey of multi-task learning in natural language processing: Regarding task relatedness and training methods. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 943–956, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.66. URL https://aclanthology.org/2023.eacl-main.66.

Xiangyu Zhao, Maolin Wang, Xinjian Zhao, Jiansheng Li, Shucheng Zhou, Dawei Yin, Qing Li, Jiliang Tang, and Ruocheng Guo. Embedding in recommender systems: A survey. 2023. URL https://arxiv.org/abs/2310.18608.

A Experimental Results Figures

This section includes the figures corresponding to the experimental results presented in the main text.

A.1 Single-task Fine-Tuning

Figure 2 shows the loss and accuracy changes for the single-task fine-tuning approach.

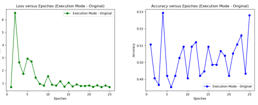

Figure 2: Loss and Accuracy Change for Single-task Fine-Tuning

A.2 Multi-task Learning (MTL)

Figures 3 and 4 show the loss and accuracy changes for the multi-task learning approach.

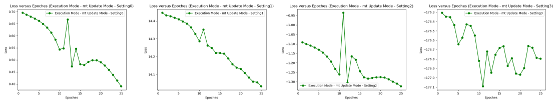

Figure 3: Loss Change in Multi-task Learning

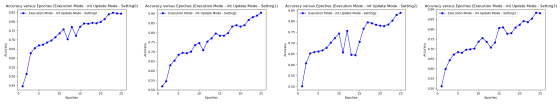

Figure 4: Accuracy Change in Multi-task Learning

A.3 MT2ST: Diminish Strategy

Figures 5 and 6 show the loss and accuracy changes for the MT2ST-diminish strategy.

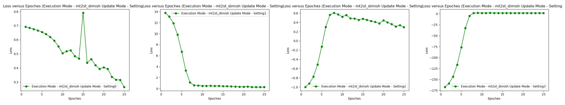

Figure 5: Loss Change in MT2ST: Diminish Strategy

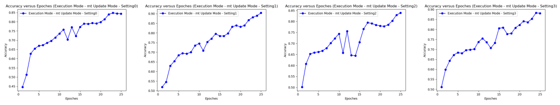

Figure 6: Accuracy Change in MT2ST: Diminish Strategy

A.4 MT2ST: Switch Strategy

Figures 7 and 8 show the loss and accuracy changes for the MT2ST-switch strategy.

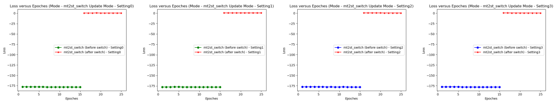

Figure 7: Loss Change in MT2ST: Switch Strategy

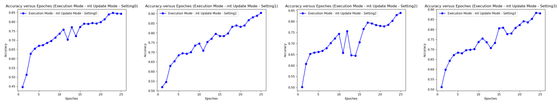

Figure 8: Accuracy Change in MT2ST: Switch Strategy

B Theoretical Foundation of MT2ST

In this section, we provide a formal theoretical framework for MT2ST. We first describe a general overview of our method. Then, we instantiate it in the context of shared neural representation learning. Finally, we conduct a theoretical efficiency analysis comparing MT2ST with standard MTL and STL baselines.

B.1 Overview of MT2ST

Let a model be denoted by $f(\cdot;\theta)$, trained on a set of K tasks $\{\mathcal{T}_1,\ldots,\mathcal{T}_K\}$. The total loss at step t is a weighted combination of the primary task $\mathcal{T}_{\text{main}}$ and auxiliary tasks:

$$\mathcal{L}^{(t)} = \mathcal{L}^{(t)}_{\text{main}} + \sum_{k\neq\text{main}} \gamma_k^{(t)} \mathcal{L}_k^{(t)}, \tag{16}$$

where $\gamma_k^{(t)}$ is a time-varying weight for auxiliary task k at iteration t. MT2ST alternates between two core strategies:

- **Diminish**: Gradually decreases each $\gamma_k^{(t)}$ to zero over time, enabling soft transition from MTL to STL.
- **Switch**: Explicitly sets $\gamma_k^{(t)} = 0$ after a predefined step T_{switch}, performing a hard switch to STL.

B.2 Formulation of Diminish Strategy

In the Diminish strategy, each auxiliary task's contribution is governed by a decay function:

$$\gamma_k^{(t)} = \gamma_{k,0} \cdot \exp\left(-\eta_k t^{\nu_k}\right), \quad k \neq \text{main}, \tag{17}$$

where $\gamma_{k,0}$ is the initial importance of task k, η_k is the decay rate, and ν_k controls curvature (decay speed). The overall parameter update is given by:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \left(\nabla \mathcal{L}^{(t)}_{\text{main}} + \sum_{k\neq\text{main}} \gamma_k^{(t)} \nabla \mathcal{L}_k^{(t)} \right), \tag{18}$$

where α is the learning rate.

B.3 Formulation of Switch Strategy

The Switch strategy introduces a discrete schedule:

$$\gamma_k^{(t)} = \begin{cases} 1, & t < T_{\text{switch}} \\ 0, & t \geq T_{\text{switch}} \end{cases} \quad \text{for all } k \neq \text{main}.$$

The update rule becomes:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \left(\nabla \mathcal{L}^{(t)}_{\text{main}} + \sum_{k\neq\text{main}} \gamma_k^{(t)} \nabla \mathcal{L}_k^{(t)} \right), \tag{19}$$

but reduces to standard single-task learning for $t \geq T_{\text{switch}}$.

B.4 Theoretical Efficiency Analysis

We compare MT2ST with baseline MTL and STL methods in terms of convergence behavior and computational efficiency.

Training Cost (FLOPs) Let C_{mtl} and C_{stl} denote per-step FLOPs for MTL and STL respectively. Then, the expected training cost for MT2ST is:

$$C_{\text{MT2ST}} = \sum_{t=1}^{T} \left[C_{\text{stl}} + \sum_{k\neq\text{main}} \gamma_k^{(t)} C_k \right], \tag{20}$$

where C_k is the marginal cost for task k. When $\gamma_k^{(t)} \to 0$ quickly, the training cost approaches STL but retains MTL's benefit in early stages.

Convergence Behavior Define the effective gradient at step t as:

$$\nabla \mathcal{L}_{\text{eff}}^{(t)} = \nabla \mathcal{L}_{\text{main}}^{(t)} + \sum_{k \neq \text{main}} \gamma_k^{(t)} \nabla \mathcal{L}_k^{(t)}.$$

Under the Polyak-Łojasiewicz (PL) condition [Karimi et al., 2017], MT2ST retains linear convergence rate as long as the auxiliary task gradients align or diminish quickly:

$$\langle \nabla \mathcal{L}_{\text{main}}^{(t)}, \nabla \mathcal{L}_{\text{eff}}^{(t)} \rangle > 0.$$

Our strategy ensures that gradient interference is minimized over time, either smoothly (Diminish) or discretely (Switch), avoiding divergence seen in conventional MTL [Yu et al., 2020].

Memory Usage Because MT2ST shares the same encoder across tasks, model memory cost is no worse than MTL. When $\gamma_k^{(t)} = 0$, the auxiliary gradients and heads can be dropped from the computation graph entirely.