DIRECTIONAL CURVATURE GOVERNS TASK-VECTOR TRANSFER: THEORY AND CURVATURE COMPENSATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Amid rapid iteration of pretrained models, task vectors—the parameter difference between a base model and its fine-tuned counterpart—offer a training-free transfer rule: adding the task vector learned on one model to another to impart similar task capabilities. However, prior empirical studies are mixed: task-vector transfer can succeed in some settings but fail dramatically in others. In this paper, under suitable regularity conditions, we theoretically characterize both when vanilla transfer is exact and how its suboptimality is controlled. Specifically, we prove that vanilla transfer is exact if and only if the model displacement $\theta_2 - \theta_1$ lies in the null space of the path-averaged Hessian $\bar{H}$, i.e., the directional curvature condition $\bar{H}(\theta_2 - \theta_1) = 0$. We further show that the violation magnitude $\|\bar{H}(\theta_2 - \theta_1)\|^2$ controls the suboptimality of transfer and provide proxy-based empirical support for this relationship in neural networks. To overcome the intrinsic limitations of vanilla transfer, we propose the Curvature-compensated Task Vector (CurvTV), which enhances task-vector transfer by compensating for parameter deviations induced by violations of the directional curvature condition. Theoretically, CurvTV retains a quadratic optimality-gap envelope while admitting a sharper quartic refinement in the local regime, and this local gain persists under controlled Hessian estimation error. Extensive experiments across RoBERTa and Llama-family models demonstrate that CurvTV yields consistent improvements in task-averaged performance over vanilla task-vector transfer. Code is available at https://anonymous.4open.science/r/CurvTV.

1 INTRODUCTION

Modern pretrained models have made the "pretrain-then-fine-tune" paradigm the standard pipeline for adapting models to specific tasks (Han et al., 2024). In this context, the notion of a task vector has attracted growing interest (Li et al., 2025). A task vector is defined as the parameter difference between a model and its fine-tuned counterpart, which is typically viewed as a compact parameterization of task-specific capability. Task vectors enable efficient, training-free model editing—for example, adding a task vector to impart its associated capability (Ortiz-Jimenez et al., 2023), subtracting it to achieve targeted forgetting (Ilharco et al., 2022), or linearly combining multiple vectors to synthesize multi-task behavior (Yang et al., 2024).

With rapid turnover of pretrained models, recent work has explored transferring task vectors to endow a target model with similar task capabilities without re-running fine-tuning. Let θ_1 be an anchor model and $\theta_1^\star$ its fine-tuned counterpart on a downstream task. Given a target model θ_2, task-vector transfer constructs the candidate model $\theta_{\mathrm{tv}} := \theta_2 + (\theta_1^\star - \theta_1)$. One line of work (Shahroz et al., 2025; Cao et al., 2025) reports that θ_{tv} "equips the target model with post-training capabilities, achieving performance comparable to direct fine-tuning." In contrast, another line of work (Rinaldi et al., 2025a;b) reaches the opposite conclusion, observing that "the knowledge acquired during fine-tuning on θ_1 may not transfer to θ_2 with a mere addition of the original task vector." These conflicting findings motivate the central question:

When can a task vector be successfully transferred?

We answer this question by analyzing the deviation between the transferred solution $\theta_{\mathrm{tv}} = \theta_2 + (\theta_1^\star - \theta_1)$ and the solution $\theta_2^\star$ obtained by directly fine-tuning θ_2.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

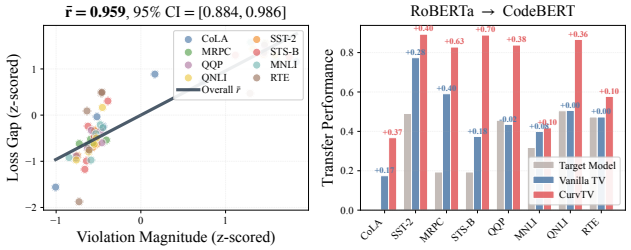


Figure 1: (a) Point-Hessian proxy for the violation magnitude of the directional curvature condition versus loss gap; a random-effects meta-analysis yields an overall correlation of $\bar{r} = 0.959$. (b) CurvTV augments vanilla task-vector transfer with a curvature-based compensation term and consistently outperforms the vanilla baseline.

For a general smooth task loss, we show that, whenever $\alpha I + \bar{H}$ is non-singular, θ_{tv} exactly matches $\theta_2^\star$ if and only if the anchor displacement $\theta_2 - \theta_1$ lies in the null space of the path-averaged Hessian $\bar{H}$. We term this requirement the **directional curvature condition**, formalized as $\bar{H}(\theta_2 - \theta_1) = 0$ (Section 3.1). Under local curvature bounds, the violation magnitude $\|\bar{H}(\theta_2 - \theta_1)\|^2$ controls the loss gap between θ_{tv} and $\theta_2^\star$. Because $\bar{H}$ is unavailable in practice, we evaluate a point-Hessian proxy estimated by K-FAC (Martens & Grosse, 2015) and observe a strong positive association between this proxy and the loss gap. A random-effects meta-analysis across tasks yields an overall correlation of $\bar{r} = 0.959$ (Section 3.2). Fig. 1(a) visualizes the pooled linear relationship after z-scoring within each task.

Unfortunately, the directional curvature condition rarely holds, and as a result, vanilla task-vector transfer can incur a non-negligible performance gap relative to direct fine-tuning. This mismatch prompts the second question:

Can we improve upon task-vector transfer to close the gap?

In response, we propose the **Curv**ature-compensated **T**ask **V**ector (CurvTV), which improves task-vector transfer by compensating for parameter deviations caused by violations of the directional curvature condition (Section 4). Specifically, we derive an exact relation between the transferred solution θ_{tv} and the target fine-tuning optimum $\theta_2^\star$ in terms of a path-averaged Hessian $\bar{H}$ and the displacement $\theta_2 - \theta_1$. Since $\bar{H}$ is not directly available, CurvTV substitutes it with the Hessian evaluated at a known point, $\theta_1^\star$, and plugs this proxy into the derived relation to construct a closed-form compensation term. As illustrated in Fig. 1(b), CurvTV simply applies this compensation term to θ_{tv} to re-align it toward $\theta_2^\star$, thereby yielding substantial performance improvements over vanilla transfer (Section 5). Our main contributions are summarized as follows:

- Under suitable regularity conditions, we establish a necessary and sufficient criterion for exact task-vector transfer—the directional curvature condition. We further show that the magnitude of its violation controls transfer suboptimality, with neural-network experiments providing proxy-based empirical support for this relationship.
- We propose CurvTV, a curvature-compensated refinement of task-vector transfer when directional curvature is violated. Unlike heuristic methods, CurvTV leverages a closed-form compensation term derived from the exact deviation relation to correct θ_{tv} toward $\theta_2^\star$.
- We show that CurvTV provides an $O(\|\theta_2 - \theta_1\|^2)$ optimality-gap envelope, together with a tighter $O(\|\theta_2 - \theta_1\|^4)$ bound for local anchor displacements. This local improvement persists under controlled Hessian estimation error.
- We evaluate CurvTV on the RoBERTa and Llama model families, demonstrating consistent improvements in task-averaged performance over vanilla task-vector transfer across target models.

2 Preliminaries

We consider two given models $\theta_1, \theta_2 \in \mathbb{R}^d$ that share the same network architecture. Fix a task τ with a smooth loss $\ell_\tau(\theta)$. For $i \in \{1, 2\}$, we consider the regularized fine-tuning objective

$$\mathcal{L}_i(\theta) := \ell_\tau(\theta) + \frac{\alpha}{2}\|\theta - \theta_i\|^2, \tag{1}$$

where $\alpha > 0$ is the regularization coefficient. We assume that $\mathcal{L}_i$ admits a minimizer and use $\theta_i^\star$ to denote one such minimizer obtained by fine-tuning θ_i on task τ.

Following prior work (Ilharco et al., 2022), we define the task vector obtained by fine-tuning the anchor model θ_i on task τ as $\Delta_i := \theta_i^\star - \theta_i$, for $i \in \{1, 2\}$. The core idea of task-vector transfer is that if Δ_1 captures the task-specific capability, then the same parameter update can be ported to another model θ_2, endowing θ_2 with similar task capability without re-running fine-tuning. Mathematically, the **vanilla task-vector transfer** rule produces the candidate model

$$\theta_{\mathrm{tv}} := \theta_2 + \Delta_1 = \theta_2 + (\theta_1^\star - \theta_1).$$

However, prior studies on this rule are mixed: in some settings, θ_{tv} performs comparably to directly fine-tuning θ_2, while in others it suffers severe degradation, sometimes even collapsing. In Section 3, we develop a theoretical characterization of the mechanisms behind this discrepancy; in Section 4, we propose an improved task-vector transfer method that addresses regimes where θ_{tv} fails.

3 A Condition for Task-vector Transfer

Under suitable regularity conditions, we derive the directional curvature condition as a necessary and sufficient criterion for exact task-vector transfer in Section 3.1 and show that its violation magnitude controls transfer suboptimality in Section 3.2.

3.1 Directional Curvature Condition

To characterize exact transfer for a general smooth task loss ℓ_τ, we relate the displacement between the anchor models to that between their fine-tuned solutions. The key quantity is the path-averaged Hessian along the segment connecting $\theta_1^\star$ and $\theta_2^\star$:

$$\bar{H} := \int_0^1 \nabla^2 \ell_\tau\big(\theta_1^\star + t(\theta_2^\star - \theta_1^\star)\big)\, dt.$$

The fundamental theorem of calculus gives the exact gradient relation

$$\nabla \ell_\tau(\theta_2^\star) - \nabla \ell_\tau(\theta_1^\star) = \bar{H}(\theta_2^\star - \theta_1^\star).$$

Combining this identity with the stationarity conditions for $\theta_1^\star$ and $\theta_2^\star$ yields

$$\alpha(\theta_2 - \theta_1) = \alpha(\theta_2^\star - \theta_1^\star) + \big(\nabla \ell_\tau(\theta_2^\star) - \nabla \ell_\tau(\theta_1^\star)\big) = (\alpha I + \bar{H})(\theta_2^\star - \theta_1^\star).$$

This relation exposes the error introduced by vanilla task-vector transfer. Since $\theta_{\mathrm{tv}} = \theta_2 + (\theta_1^\star - \theta_1)$, its deviation from the target fine-tuned solution satisfies

$$\begin{aligned}(\alpha I + \bar{H})(\theta_{\mathrm{tv}} - \theta_2^\star) &= (\alpha I + \bar{H})(\theta_2 - \theta_1) - (\alpha I + \bar{H})(\theta_2^\star - \theta_1^\star) \\ &= (\alpha I + \bar{H})(\theta_2 - \theta_1) - \alpha(\theta_2 - \theta_1) \\ &= \bar{H}(\theta_2 - \theta_1).\end{aligned} \tag{2}$$

Thus, $\bar{H}(\theta_2 - \theta_1)$ isolates the curvature-induced residual arising from task-vector transfer between the two anchor models. If $\alpha I + \bar{H}$ is nonsingular, this residual vanishes exactly when the transferred solution equals the target fine-tuned solution:

$$\theta_{\mathrm{tv}} = \theta_2^\star \quad \Longleftrightarrow \quad \bar{H}(\theta_2 - \theta_1) = 0. \tag{3}$$

We refer to Eq. (3) as the **directional curvature condition**. It requires the path-averaged Hessian to annihilate the anchor displacement $\theta_2 - \theta_1$. When the condition holds, the task vector learned from θ_1 remains an exact update at θ_2; when it fails, the residual $\bar{H}(\theta_2 - \theta_1)$ captures the resulting mismatch, whose effect on transfer suboptimality is quantified in the next subsection.

Remark 3.1 (Gradient-field interpretation). In the quadratic case, $\bar{H}$ reduces to a constant Hessian H, and the directional curvature condition $H(\theta_2 - \theta_1) = 0$ has a direct geometric interpretation: the gradient field of $\mathcal{L}_2$ is exactly the translation of that of $\mathcal{L}_1$ by $\theta_2 - \theta_1$. Appendix B.2 derives this identity and illustrates both the exact and violated cases in two dimensions.

Remark 3.2 (Beyond exact proximal stationarity). We discuss how nonzero proximal stationarity residuals modify this condition in Appendix B.4.

3.2 Violation Magnitude Controls the Suboptimality

In the previous subsection, we established that task-vector transfer is exact if and only if the directional curvature condition holds. We now show that the magnitude of its violation controls the suboptimality of the transferred solution and provide proxy-based empirical support for this relationship in neural networks. The following theorem characterizes the loss gap of the transferred solution θ_{tv} relative to the target fine-tuned solution $\theta_2^\star$.

Theorem 3.3. *Suppose that $\alpha I + \bar{H}$ is nonsingular. Then there exists a point $\widetilde{\theta}$ on the line segment between $\theta_2^\star$ and θ_{tv} such that*

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) = \frac{1}{2}\big(\bar{H}(\theta_2-\theta_1)\big)^\top \big(\alpha I + \bar{H}\big)^{-1}\big(\alpha I + \nabla^2\ell_\tau(\widetilde{\theta})\big)\big(\alpha I + \bar{H}\big)^{-1}\bar{H}(\theta_2-\theta_1).$$

Moreover, if $0 \preceq \nabla^2\ell_\tau(\theta) \preceq \beta I$ for all $\theta \in [\theta_1^\star, \theta_2^\star] \cup [\theta_2^\star, \theta_{\mathrm{tv}}]$, then

$$\frac{\alpha\left\|\bar{H}(\theta_2-\theta_1)\right\|^2}{2(\alpha+\beta)^2} \le \mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{(\alpha+\beta)\left\|\bar{H}(\theta_2-\theta_1)\right\|^2}{2\alpha^2}.$$

Hence, $\|\bar{H}(\theta_2-\theta_1)\|^2$ quantifies the violation magnitude of the directional curvature condition.

The proof is deferred to Appendix B.1. The first identity gives an exact expression for the suboptimality of θ_{tv} as a curvature-weighted quadratic form of the residual $\bar{H}(\theta_2-\theta_1)$. The subsequent bounds further show that the loss gap is controlled, from both above and below, by the violation magnitude $\left\|\bar{H}(\theta_2-\theta_1)\right\|^2$. Thus, up to the multiplicative factors determined by α and β, larger violations of the directional curvature condition tend to correspond to larger loss gaps.

We next empirically examine whether the relationship predicted by Theorem 3.3 is reflected in neural networks. Specifically, we fine-tune RoBERTa (Liu et al., 2019) on each of the eight GLUE (Wang et al., 2019) tasks to obtain $\theta_1^\star$, and transfer the resulting task vectors to seven target models θ_2, yielding $8 \times 7 = 56$ transfer instances. Because directly evaluating $\bar{H}$ is computationally prohibitive, we use the point Hessian $H_1^\star := \nabla^2\ell_\tau(\theta_1^\star)$ as a proxy and construct a K-FAC estimate $\widehat{H}_1^\star$ using $N = 5000$ training samples. We compute $X := \|\widehat{H}_1^\star(\theta_2-\theta_1)\|^2$ and define the response as the task-loss gap $Y := \ell_\tau(\theta_{\mathrm{tv}}) - \ell_\tau(\theta_2^\star)$. We omit the explicit proximal regularization term from the loss gap in Theorem 3.3 because the standard fine-tuning protocol we follow (Yu et al., 2024) uses weight decay instead. To account for differences in loss scales across tasks, we perform a separate correlation analysis for each task. See Appendix B.3 for details.

Fig. 4 presents the per-task linear trends and reports **Pearson correlations** and p**-values**. To summarize across tasks, we aggregate the eight Pearson correlations using a random-effects meta-analysis with Fisher's z-transformation, obtaining an overall correlation of $\bar{r} = 0.959$ with a 95% confidence interval of $[0.884, 0.986]$. Fig. 1(a) additionally visualizes the pooled linear trend after z-scoring X and Y within each task. Overall, the strong positive association indicates that larger violations of the directional curvature condition tend to correspond to larger loss gaps, providing proxy-based empirical support for the relationship predicted by our analysis.

4 Curvature-compensated Task Vector

Since the directional curvature condition is seldom satisfied in realistic settings, vanilla task-vector transfer can leave a nontrivial performance gap. To address this issue, Section 4.1 derives an explicit compensation term that corrects θ_{tv} toward $\theta_2^\star$, leading to our CurvTV method. Section 4.2 then establishes optimality-gap guarantees for vanilla transfer and CurvTV.

4.1 Curvature-compensated Task Vector

To translate the preceding curvature analysis into a practical task-vector reuse method, we must account for both violations of the directional curvature condition and possible differences in regularization strength between source and target fine-tuning. Accordingly, we extend the shared-coefficient formulation in Eq. (1) by assigning separate coefficients α_1 and α_2 to the source and target objectives, respectively. If $\bar{H} + \alpha_2 I$ is nonsingular, the deviation identity in Eq. (2) generalizes to

$$\theta_{\mathrm{tv}} - \theta_2^\star = (\bar{H} + \alpha_2 I)^{-1}\left[\bar{H}(\theta_2-\theta_1) + (\alpha_2-\alpha_1)(\theta_1^\star-\theta_1)\right]. \tag{4}$$

To obtain a practical procedure, we approximate the path-averaged Hessian $\bar{H}$ in Eq. (4) with the Hessian at the known source optimum, $H_1^\star = \nabla^2 \ell_\tau(\theta_1^\star)$, because computing $\bar{H}$ exactly is prohibitively expensive. Assuming that $H_1^\star + \alpha_2 I$ is nonsingular, we define the **Curv**ature-compensated **T**ask **V**ector (CurvTV) method, which augments vanilla task-vector transfer with a curvature-based compensation:

$$\theta_{\mathrm{CurvTV}} = \underbrace{\theta_{\mathrm{tv}}}_{\text{vanilla transfer}} \underbrace{-(H_1^\star + \alpha_2 I)^{-1}\left[H_1^\star(\theta_2 - \theta_1) + (\alpha_2 - \alpha_1)(\theta_1^\star - \theta_1)\right]}_{\text{our compensation}}.$$

Conceptually, CurvTV corrects the vanilla transferred solution θ_{tv} toward the target optimum $\theta_2^\star$ by approximating the exact deviation in Eq. (4). Here, we treat α_1 and α_2 as hyperparameters, since the corresponding source and target regularization strengths may be unknown or unavailable in practice.

Hessian implementation and cost. For standard negative-log-likelihood objectives, such as cross-entropy and squared error, the Fisher coincides with the model-expected Hessian under regularity conditions (Martens, 2020); in practice, the empirical Fisher is widely used as a tractable curvature surrogate (Kunstner et al., 2019). Following this practice, we construct a surrogate for $H_1^\star$ using K-FAC, a widely used structured Fisher approximation (Dangel et al., 2025) whose efficiently invertible Kronecker form has been observed to capture the Fisher's coarse structure well (Martens & Grosse, 2015). For each layer l, the resulting block is $\widehat{H}_1^{\star(l)} := A^{(l)} \otimes G^{(l)}$, where $A^{(l)}$ and $G^{(l)}$ are empirical second moments of input activations and output gradients, respectively. Collecting these statistics on N calibration samples incurs an $O(NC)$ forward–backward cost, where C is the cost of a forward–backward pass per sample. Storing the Kronecker factors requires $O\big(\sum_l[(d_{\mathrm{in}}^{(l)})^2 + (d_{\mathrm{out}}^{(l)})^2]\big)$ memory, where $d_{\mathrm{in}}^{(l)}$ and $d_{\mathrm{out}}^{(l)}$ are the input and output dimensions of layer l. See Appendix C.1 for implementation details and Section 5.5 for empirical costs.

4.2 Theoretical Guarantees

Approximating $\bar{H}$ with $H_1^\star$ does not by itself guarantee that CurvTV improves upon vanilla task-vector transfer, motivating a theoretical analysis of its performance. For analytical simplicity, we focus on the shared-regularization setting $\alpha_1 = \alpha_2 = \alpha$, under which the regularization-mismatch term vanishes and CurvTV reduces to the single-α form. In this setting, we establish upper bounds on the optimality gap w.r.t. the target fine-tuning loss, $\mathcal{L}_2(\theta) - \mathcal{L}_2(\theta_2^\star)$, for both vanilla task-vector transfer and CurvTV. Our main results are summarized below, with proofs provided in Appendix C.3.

1. **Vanilla TV.** The optimality-gap upper bound for Vanilla TV scales as $O(\|\theta_2 - \theta_1\|^2)$.
2. **CurvTV.** Our curvature compensation cancels the dominant curvature-induced parameter bias, yielding a quadratic optimality-gap envelope together with a sharper $O(\|\theta_2 - \theta_1\|^4)$ refinement in the local regime.
3. **Robust CurvTV.** This local fourth-order upper-bound guarantee persists when $H_1^\star$ is approximated by $H_1^\star + \Delta$, provided that $\|\Delta\| \le \epsilon\|\theta_2 - \theta_1\|$.

We now formalize the analysis under local regularity conditions. Let $\mathcal{C} := \{\theta_{\mathrm{tv}}, \theta_{\mathrm{CurvTV}}, \tilde{\theta}_{\mathrm{CurvTV}}\}$ denote the candidate solutions analyzed below, where $\tilde{\theta}_{\mathrm{CurvTV}}$ is the CurvTV solution obtained using a fixed approximate Hessian. We define $\mathcal{R} := [\theta_1^\star, \theta_2^\star] \cup \bigcup_{\theta_{\mathrm{cand}} \in \mathcal{C}} [\theta_2^\star, \theta_{\mathrm{cand}}]$ as the union of the line segments relevant to the analysis.

Assumption 4.1 (Regularized Curvature)**.** There exists $\beta > 0$ such that, for every $\theta \in \mathcal{R}$,

$$0 \prec \nabla^2 \ell_\tau(\theta) + \alpha I \preceq (\alpha + \beta) I.$$

Assumption 4.2 (Hessian Lipschitzness)**.** There exists $\rho > 0$ such that, for all $\theta, \theta' \in [\theta_1^\star, \theta_2^\star]$,

$$\|\nabla^2 \ell_\tau(\theta) - \nabla^2 \ell_\tau(\theta')\| \le \rho\|\theta - \theta'\|.$$

Assumption 4.1 bounds the curvature of the regularized objective on $\mathcal{R}$; its upper bound is a local form of the standard smoothness condition in optimization (Nesterov, 2013), while its strict lower bound ensures that the inverses involving $\bar{H}$ and $H_1^\star$ are well defined. Assumption 4.2 controls the variation of the Hessian along $[\theta_1^\star, \theta_2^\star]$ and is commonly used in nonconvex analysis to bound

higher-order Taylor remainders (Nesterov & Polyak, 2006). Both assumptions are restricted to the line segments used in the analysis rather than imposed globally.

To state the ensuing bounds compactly, for $\alpha > 0$ and matrix A such that $\alpha I + A$ is invertible, define

$$\psi_\alpha(A) := (\alpha I + A)^{-1} A. \tag{5}$$

The following theorem provides an upper bound on the optimality gap of the vanilla transferred solution θ_{tv}.

Theorem 4.3 (Vanilla TV). *Under Assumption 4.1, the vanilla transferred solution θ_{tv} satisfies*

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \|\psi_\alpha(\bar{H})\|^2 \|\theta_2 - \theta_1\|^2,$$

where $\psi_\alpha(\bar{H})$ is defined in Eq. (5).

Theorem 4.3 gives an optimality-gap upper bound for vanilla transfer in which the anchor displacement enters through the quadratic factor $\|\theta_2 - \theta_1\|^2$. This bound suggests that a smaller local upper-curvature parameter β may yield a tighter upper bound. The role of α is more subtle: for fixed $\bar{H}$, increasing α shrinks $\|\psi_\alpha(\bar{H})\|$ as the eigenvalue magnitudes $\left|\frac{\lambda_i}{\alpha+\lambda_i}\right|$ move toward zero, but simultaneously enlarges the local curvature factor $\alpha + \beta$. Moreover, α changes the fine-tuned solutions, the path defining $\bar{H}$, and the regularized objective itself, so tuning α alone is unlikely to resolve transfer failures.

Next, we establish a two-regime guarantee for CurvTV, combining a quadratic envelope under the stated regularity conditions with a sharper quartic refinement when the anchor displacement is local.

Theorem 4.4 (CurvTV). *Under Assumptions 4.1 and 4.2, the CurvTV solution satisfies*

$$\mathcal{L}_2(\theta_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{(\alpha+\beta)\Upsilon_\alpha^2}{2} \|\theta_2 - \theta_1\|^2 \min\left\{ \frac{\rho^2}{4}\|\theta_2 - \theta_1\|^2,\ \frac{(\alpha+\beta)^2}{\|I - \psi_\alpha(\bar{H})\|^2} \right\},$$

where $\Upsilon_\alpha := \frac{\|I-\psi_\alpha(\bar{H})\|^2 \|I-\psi_\alpha(H_1^\star)\|}{\alpha}$.

The two branches of Theorem 4.4 characterize CurvTV's optimality-gap upper bound across complementary displacement regimes. The quartic branch captures CurvTV's advantage over Vanilla TV: it improves the $O(\|\theta_2 - \theta_1\|^2)$ optimality-gap bound in Theorem 4.3 to $O(\|\theta_2 - \theta_1\|^4)$ and is tighter than the quadratic branch whenever $\|\theta_2 - \theta_1\| \le \frac{2(\alpha+\beta)}{\rho\|I-\psi_\alpha(\bar{H})\|}$. For larger anchor displacements, the quadratic branch gives the tighter bound. Together, the two branches provide a quadratic envelope with a sharper quartic refinement in the local regime.

Finally, we quantify the effect of Hessian approximation on CurvTV's optimality gap.

Corollary 4.5 (Robustness). *Let $\tilde{H}_1^\star := H_1^\star + \Delta$ denote the fixed approximate Hessian used to obtain $\tilde{\theta}_{\mathrm{CurvTV}} \in \mathcal{C}$. Under Assumptions 4.1 and 4.2, suppose additionally that $\alpha I + \tilde{H}_1^\star$ is invertible. Then the perturbed solution $\tilde{\theta}_{\mathrm{CurvTV}}$ satisfies*

$$\mathcal{L}_2(\tilde{\theta}_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \left(\frac{\rho\Upsilon_\alpha}{2}\|\theta_2 - \theta_1\|^2 + \Gamma_\alpha \|\Delta\| \|\theta_2 - \theta_1\| \right)^2,$$

where $\Gamma_\alpha := \frac{\|I-\psi_\alpha(\tilde{H}_1^\star)\| \|I-\psi_\alpha(H_1^\star)\|}{\alpha}$. Moreover, if the Hessian estimation error scales as $\|\Delta\| \le \epsilon\|\theta_2 - \theta_1\|$ for some $\epsilon \ge 0$, then

$$\mathcal{L}_2(\tilde{\theta}_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \left(\frac{\rho\Upsilon_\alpha + 2\epsilon\Gamma_\alpha}{2} \right)^2 \|\theta_2 - \theta_1\|^4.$$

Corollary 4.5 quantifies how Hessian approximation error affects CurvTV's optimality-gap bound. The first bound separates two contributions to the optimality-gap upper bound: (i) the intrinsic error of exact CurvTV, corresponding to the quartic branch of Theorem 4.4; and (ii) an additional term induced by Hessian approximation. More importantly, if the Hessian error is of the same order as the anchor displacement, $\|\Delta\| \le \epsilon\|\theta_2 - \theta_1\|$, both residuals scale quadratically with the anchor displacement, so the resulting optimality-gap upper bound remains $O(\|\theta_2 - \theta_1\|^4)$. When $\Delta = 0$, the approximation residual vanishes and the bound reduces exactly to the quartic branch of the exact-Hessian guarantee in Theorem 4.4. The general bound explicitly accounts for Hessian approximation error, while the local quartic refinement follows under the additional scaling assumption.

Table 1: Transfer results for the RoBERTa model family across eight GLUE tasks. In the *Reference* row, the Gain and Gap columns report the performance of θ_2 and $\theta_2^\star$, respectively.

	CoLA		SST-2		MRPC		STS-B		QQP		MNLI		QNLI		RTE	
Metrics	Gain	Gap	Gain	Gap	Gain	Gap	Gain	Gap	Gain	Gap	Gain	Gap	Gain	Gap	Gain	Gap
RoBERTa → TimeLM-22																
Reference	0.00	0.63	0.53	0.94	0.16	0.91	0.09	0.91	0.45	0.90	0.32	0.87	0.51	0.92	0.47	0.76
Vanilla TV	+0.56	0.07	+0.41	0.01	+0.74	0.01	+0.77	0.05	+0.44	0.01	+0.54	0.01	+0.40	0.01	+0.25	0.05
Re-Basin	+0.56	0.07	+0.40	0.01	+0.74	0.01	+0.77	0.05	+0.44	0.01	+0.54	0.01	+0.40	0.01	+0.25	0.05
GradFix	+0.50	0.13	+0.39	0.02	+0.62	0.14	+0.80	0.02	−0.13	0.58	+0.13	0.42	+0.34	0.07	+0.24	0.05
CurvTV	+0.53	0.10	+0.41	0.01	+0.75	0.01	+0.82	0.00	+0.44	0.01	+0.51	0.04	+0.41	0.00	+0.30	−0.01
RoBERTa → BioMed																
Reference	0.00	0.59	0.55	0.92	0.29	0.90	0.20	0.90	0.45	0.90	0.32	0.86	0.51	0.92	0.47	0.71
Vanilla TV	+0.50	0.09	+0.35	0.02	+0.61	0.00	+0.49	0.21	+0.42	0.03	+0.51	0.03	+0.36	0.06	+0.03	0.22
Re-Basin	+0.50	0.09	+0.35	0.02	+0.61	0.00	+0.49	0.21	+0.42	0.03	+0.51	0.03	+0.36	0.06	+0.03	0.22
GradFix	+0.44	0.16	+0.32	0.05	+0.47	0.14	+0.68	0.02	+0.00	0.44	+0.22	0.33	+0.33	0.08	+0.22	0.02
CurvTV	+0.59	0.00	+0.38	−0.01	+0.60	0.01	+0.70	0.00	+0.43	0.02	+0.51	0.03	+0.42	0.00	+0.31	−0.06
RoBERTa → CodeBERT																
Reference	0.00	0.59	0.49	0.92	0.19	0.87	0.19	0.87	0.46	0.89	0.32	0.82	0.50	0.89	0.47	0.73
Vanilla TV	+0.17	0.41	+0.28	0.14	+0.40	0.28	+0.18	0.50	−0.02	0.46	+0.08	0.42	+0.00	0.38	+0.00	0.26
Re-Basin	+0.17	0.41	+0.28	0.14	+0.40	0.28	+0.18	0.50	−0.02	0.46	+0.08	0.42	+0.00	0.38	+0.00	0.26
GradFix	+0.15	0.44	+0.28	0.15	+0.56	0.11	+0.63	0.05	−0.14	0.58	+0.17	0.33	+0.27	0.12	+0.14	0.12
CurvTV	+0.37	0.22	+0.40	0.02	+0.63	0.04	+0.70	−0.01	+0.38	0.05	+0.10	0.40	+0.36	0.02	+0.10	0.15

5 Experiments

5.1 Transfer on RoBERTa (0.12B)

We use RoBERTa (Liu et al., 2019) as the anchor model θ_1. Following Yu et al. (2024), we fine-tune θ_1 on each of the eight tasks from the GLUE benchmark (Wang et al., 2019), obtaining task-specific models $\theta_1^\star$. We select publicly available models obtained by continued pretraining from RoBERTa as target models θ_2, including: 1) **TimeLM-22** (Loureiro et al., 2022), continued pretraining on 154 million tweets; 2) **BioMed** (Gururangan et al., 2020), continued pretraining on 2.68 million biomedical scientific articles; and 3) **CodeBERT** (Feng et al., 2020), continued pretraining on 2 million paired (comment, code) examples.

For evaluation, we follow Huang et al. (2024) and treat the validation set as the test set. We use the standard GLUE metric $m(\cdot)$ for each task, e.g., Accuracy, F1, Matthews correlation. For more details, please refer to Appendix D.1. To answer (i) **does transfer improve the target model?** and (ii) **what performance gap remains?**, we report two metrics:

1. Transfer **Gain**[↑] over the target model: $\text{Gain} := m(\theta_{\text{transfer}}) - m(\theta_2)$.
2. Transfer **Gap**[↓] to the fine-tuned reference $\theta_2^\star$: $\text{Gap} := m(\theta_2^\star) - m(\theta_{\text{transfer}})$.

Here, **Vanilla TV** (Ilharco et al., 2022), **Re-Basin** (Rinaldi et al., 2025a), and **GradFix** (Rinaldi et al., 2025b) are the baselines, while **CurvTV** is our proposed method. We grid-search hyperparameters on BioMed to maximize transfer gain, with an additional native capability constraint for CurvTV, and keep them fixed for the other RoBERTa targets; see Appendix D.3 for details. Results in Table 1 support three key observations.

Obs 1: Vanilla TV is effective on some targets but fails on others. On TimeLM-22 and BioMed, Vanilla TV yields positive gain and relatively small gaps, with average gaps of approximately 0.027 and 0.081, respectively. These results indicate that, for some target models, direct addition can indeed inject task-specific capability into θ_2. In contrast, on CodeBERT, the gap increases sharply to an average of about 0.356, with pronounced task-level failures. These CodeBERT results illustrate that direct task-vector addition can remain substantially suboptimal in some target settings, consistent with the failure mode analyzed in Theorem 3.3.

Obs 2: Existing corrections provide only partial and setting-dependent improvements. Re-Basin closely tracks Vanilla TV across all three settings, with both methods achieving only limited gains on CodeBERT. One plausible reason is that the parameter mismatch between the anchor models is small, so parameter alignment produces only a limited correction and leaves the Re-Basin solution close to vanilla transfer. In terms of task-averaged gain, GradFix is less consistent across settings: it outperforms both Vanilla TV and Re-Basin on CodeBERT but falls behind them on TimeLM-22 and

BioMed. This variation may arise from its conservative masking strategy, which can preserve stable performance in challenging settings while suppressing useful task-vector components in easier cases.

Obs 3: CurvTV achieves the strongest task-averaged transfer performance across targets. Across all three RoBERTa transfer settings, CurvTV achieves the highest task-averaged gain. Its advantage is most pronounced on CodeBERT, where its average gain of 0.381 exceeds the 0.257 achieved by the strongest competing correction, GradFix. At the same time, CurvTV reduces the average reference gaps to approximately 0.020, -0.004, and 0.112 on TimeLM-22, BioMed, and CodeBERT, respectively. Overall, these results demonstrate that CurvTV remains effective where vanilla transfer and existing corrections fall short, reducing the task-averaged performance gap to directly fine-tuned target models.

5.2 Transfer on Llama-3.1 (8B)

Our anchor model θ_1 is Llama-3.1-8B-Instruct (Grattafiori et al., 2024). For task-specific fine-tuned models $\theta_1^\star$, we consider supervised fine-tuning (SFT) models provided by He et al. (2025), corresponding to three capability domains: multilingual understanding, mathematical reasoning, and code generation. As target models θ_2, we select three publicly available variants: (i) **Llama-VARCO**, continued pretraining on mixed Korean and English data; (ii) **Meditron-3**, continued pretraining on clinical-medicine data; and (iii) **Llama-3.1-abliterated**, a significantly enhanced variant created with abliteration. We evaluate transfer performance on six benchmarks spanning the above domains: Belebele (Bandarkar et al., 2024), M-MMLU (Hendrycks et al., 2020; Lai et al., 2023), MATH-500 (4-shot) (Hendrycks et al., 2021; Lewkowycz et al., 2022), GSM8K (8-shot) (Cobbe et al., 2021), HumanEval+ (Chen et al., 2021; Liu et al., 2023), and MBPP+ (Austin et al., 2021; Liu et al., 2023). Following the definition in Section 5.1, we report transfer **Gain**[↑] relative to each target model θ_2. Further details are provided in Appendix D.2.

Table 2: Transfer gains for the Llama-3.1 model family across six tasks.

	Belebele	M-MMLU	MATH-500	GSM8K	HumanEval+	MBPP+
Llama-3.1 → Llama-VARCO						
Target θ_2	0.80	0.51	0.29	0.54	0.47	0.51
Vanilla TV	+0.01	+0.02	+0.11	+0.19	+0.09	+0.04
CurvTV	+0.05	+0.06	+0.12	+0.29	+0.19	+0.08
Llama-3.1 → Meditron-3						
Target θ_2	0.71	0.54	0.37	0.72	0.30	0.54
Vanilla TV	+0.03	+0.00	+0.00	+0.08	+0.19	+0.01
CurvTV	+0.09	+0.02	+0.03	+0.11	+0.35	+0.04
Llama-3.1 → Llama-3.1-abliterated						
Target θ_2	0.79	0.54	0.24	0.50	0.49	0.56
Vanilla TV	−0.03	−0.04	−0.07	−0.47	−0.23	−0.19
CurvTV	+0.02	+0.00	+0.13	+0.30	+0.09	+0.00

Table 2 summarizes the LLM results. Beyond the trends already in Section 5.1, we draw two additional observations.

Obs 1: Vanilla TV can exhibit negative transfer on LLMs, while CurvTV largely avoids this failure mode. Across target LLMs, the average gain of Vanilla TV ranges from 0.08 down to -0.17. The largest failure occurs on Llama-3.1-abliterated, where Vanilla TV reduces the target's performance by 0.47 on GSM8K. The transferred task vector may deviate from a descent direction, resulting in performance degradation. CurvTV, which aims to reduce the deviation from the target optimum through curvature compensation, achieves positive task-averaged gains across all three target LLMs in our experiments.

Obs 2: CurvTV remains effective at the 8B scale. CurvTV achieves positive average gains of approximately 0.13, 0.11, and 0.09 on Llama-VARCO, Meditron-3, and Llama-3.1-abliterated, respectively. Notably, these gains are obtained using a K-FAC curvature surrogate estimated from $N = 1000$ samples, suggesting that CurvTV remains practical at this scale.

5.3 Impact of Hyperparameters

We study CurvTV's sensitivity to the regularization coefficients $\alpha_1, \alpha_2 \in \{0.05, 0.1, 0.5, 1, 5\}$ and to the number of Hessian samples $N \in \{50, 100, 500, 1000, 2000\}$. We fix $N = 1000$ for the coefficient sweep and $(\alpha_1, \alpha_2) = (0.1, 0.5)$ for the N sweep; target-specific coefficient results appear in Fig. 5. We report transfer gain as defined in Section 5.1 and native gap, where native gap is $m_{\text{native}}(\theta_2) - m_{\text{native}}(\theta_{\text{transfer}})$ and m_{native} denotes masked-token top-1 accuracy. A smaller native gap indicates better preservation of the target's native capability.

Fig. 2(a) reveals a clear trade-off between transfer gain and native gap in the joint (α_1, α_2) sweep. With the other coefficient fixed, the native gap consistently increases with α_1 and generally decreases with α_2. Transfer gain is non-monotonic in both coefficients and is generally strongest when they are of comparable scale. In the global average over all 24 target–task pairs, the pair selected on BioMed lies on the Pareto frontier and strictly dominates all three transfer baselines in both metrics.

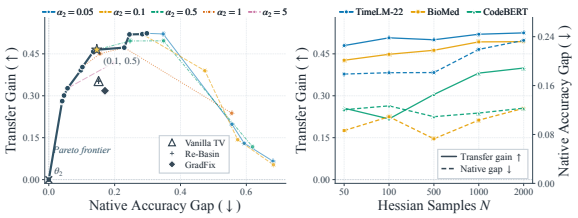


Figure 2: CurvTV's hyperparameter sensitivity. (a) Joint (α_1, α_2) sweep at $N = 1000$, averaged over all 24 target–task pairs. (b) N sweep at $(\alpha_1, \alpha_2) = (0.1, 0.5)$, with each target curve averaged over eight GLUE tasks.

Fig. 2(b) shows that transfer gain generally improves with N, with diminishing returns beyond $N = 1000$. Native gap remains relatively stable across the sweep. We use $N = 1000$ to balance transfer gain and the computational cost of Hessian estimation.

5.4 Role of Curvature

We examine the roles of curvature and regularization coefficients through the two ablations in Table 3. Identity replaces the K-FAC curvature surrogate with the identity matrix while retaining $(\alpha_1, \alpha_2) = (0.1, 0.5)$. This replacement reduces average transfer gain from 0.465 to 0.448 and increases average native gap from 0.146 to 0.160. Task-averaged gain decreases on all three targets, although Identity achieves a smaller native gap on TimeLM-22. Shared-α CurvTV retains the curvature surrogate and sets $\alpha_1 = \alpha_2 = 0.5$, removing the regularization-mismatch term. Relative to CurvTV, this variant achieves higher transfer gain at the cost of a larger native gap on each target. These results support the role of curvature and show how distinct coefficients balance transfer gain and native preservation.

Table 3: Curvature and coefficient ablations, averaged over eight GLUE tasks.

Target	Metric	CurvTV	Identity	Shared-α
TimeLM-22	Gain ↑	0.520	0.515	0.537
	Native gap ↓	0.219	0.184	0.347
BioMed	Gain ↑	0.493	0.456	0.504
	Native gap ↓	0.103	0.127	0.214
CodeBERT	Gain ↑	0.381	0.374	0.448
	Native gap ↓	0.115	0.170	0.179

5.5 Transfer-stage Runtime and Memory

To assess practical efficiency, we average transfer-stage runtime and peak allocated GPU memory over all $3 \times 8 \times 5 = 120$ runs, covering three target models, eight tasks, and five runs per model–task pair on a single NVIDIA A6000. As shown in Table 4, Vanilla TV is the most efficient, while Re-Basin incurs additional runtime from solving for weight permutations. GradFix and CurvTV require gradient computations, increasing both runtime and memory; CurvTV is nevertheless slightly faster and uses marginally less memory than GradFix. Together with the transfer results, these measurements characterize the empirical performance–efficiency trade-off.

Table 4: Transfer-stage runtime and peak allocated GPU memory over 120 runs on an NVIDIA A6000 (mean ± SD).

Method	Time (s)	Peak memory (MiB)
Vanilla TV	1.251 ± 0.740	2399.78 ± 0.00
Re-Basin	10.331 ± 4.831	2507.34 ± 0.05
GradFix	64.320 ± 1.472	3552.03 ± 0.45
CurvTV	59.711 ± 1.410	3547.15 ± 0.11

6 Conclusion

This work clarifies why task-vector transfer can be effective in some settings yet fail catastrophically in others. Under suitable regularity conditions, we establish the directional curvature condition as an if-and-only-if criterion for exact task-vector transfer and show that its violation magnitude controls transfer suboptimality. CurvTV compensates for curvature-induced parameter deviations and admits a quadratic optimality-gap envelope with a sharper local quartic refinement that persists under controlled Hessian estimation error. Experiments on RoBERTa and Llama models demonstrate consistent improvements in task-averaged performance over vanilla transfer.

AI USE STATEMENT

In this work, we used generative AI tools to polish the wording and exposition of proofs and to assist with implementing parts of the method in code. We did not use generative AI tools for any other tasks requiring disclosure. Additionally, we used generative AI tools to improve the language and readability of the rest of the manuscript. All AI-assisted manuscript text and proof exposition were manually verified, and the AI-assisted code was reviewed and successfully executed to obtain the reported results. All authors have reviewed the AI-assisted content and take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We open-source our implementation and experimental code at https://anonymous.4open.science/r/CurvTV to support independent reproduction of the reported results. Experimental settings, evaluation protocols, hyperparameter selection, and implementation details are documented in Section 5 and Appendices B.3, C.1 and D.1 to D.3. The definitions, assumptions, and theoretical results underlying CurvTV are stated in Sections 3 and 4, with complete proofs provided in Appendices B.1 and C.3.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. 8, 25

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 749–775, 2024. 8, 25

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1644–1650, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.148. URL https://aclanthology.org/2020.findings-emnlp.148/. 25

Sheng Cao, Mingrui Wu, Karthik Prasad, Yuandong Tian, and Zechun Liu. ParamΔ for direct mixing: Post-train large language model at zero cost. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=vqbd2OQnGp. 1, 14

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 8, 25

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 8, 25

Xiang Dai, Ilias Chalkidis, Sune Darkner, and Desmond Elliott. Revisiting transformer-based models for long document classification. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 7212–7230, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.534. URL https://aclanthology.org/2022.findings-emnlp.534/. 24

Felix Dangel, Bálint Mucsányi, Tobias Weber, and Runa Eschenhagen. Kronecker-factored approximate curvature (kfac) from scratch. *arXiv preprint arXiv:2507.05127*, 2025. 5

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/. 24

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. In *Findings of the association for computational linguistics: EMNLP 2020*, pp. 1536–1547, 2020. 7, 24

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 8, 25

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020. 7, 24

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024. 1

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *arXiv preprint arXiv:2408.13656*, 2024. 14

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms. *arXiv preprint arXiv:2505.10833*, 2025. 8

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020. 8, 25

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021. 8, 25

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: Tuning-free high-performance model merging. *Advances in Neural Information Processing Systems*, 37:122741–122769, 2024. 7, 24

Youwei Huang, Jianwen Li, Sen Fang, Yao Li, Peng Yang, and Bin Hu. Smart contract intent detection with pre-trained programming language model. *arXiv preprint arXiv:2508.20086*, 2025. 24

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. CodeSearchNet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*, 2019. URL https://arxiv.org/abs/1909.09436. 25

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022. 1, 3, 7, 14

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A dataset for biomedical research question answering. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2567–2577, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1259. URL https://aclanthology.org/D19-1259/. 25

Josué Kpodo, Parisa Kordjamshidi, and A. Pouyan Nejadhashemi. Agxqa: A benchmark for advanced agricultural extension question answering. *Computers and Electronics in Agriculture*, 225:109349, 2024. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2024.109349. URL https://www.sciencedirect.com/science/article/pii/S0168169924007403. 24

Frederik Kunstner, Philipp Hennig, and Lukas Balles. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems*, 32, 2019. 5

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In Yansong Feng and Els Lefever (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 318–327, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-demo.28. URL https://aclanthology.org/2023.emnlp-demo.28/. 8, 25

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022. 8, 25

Hongkang Li, Yihua Zhang, Shuai Zhang, Meng Wang, Sijia Liu, and Pin-Yu Chen. When is task vector provably effective for model editing? a generalization analysis of nonlinear transformers. *arXiv preprint arXiv:2504.10957*, 2025. 1, 14

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36:21558–21572, 2023. 8, 25

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 4, 7, 24

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S Weld. S2orc: The semantic scholar open research corpus. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 4969–4983, 2020. 24

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. Timelms: Diachronic language models from twitter. *arXiv preprint arXiv:2202.03829*, 2022. 7, 24

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020. URL http://jmlr.org/papers/v21/17-678.html. 5

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015. 2, 5, 18, 19

Michael S Matena and Colin Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022. 20

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe. 24

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013. 5

Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical programming*, 108(1):177–205, 2006. 6

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 66727–66754. Curran Associates, Inc., 2023. 1, 14

Filippo Rinaldi, Giacomo Capitani, Lorenzo Bonicelli, Donato Crisostomi, Federico Bolelli, Elisa Ficarra, Emanuele Rodola, Simone Calderara, and Angelo Porrello. Update your transformer to the latest release: Re-basin of task vectors. *arXiv preprint arXiv:2505.22697*, 2025a. 1, 7, 14

Filippo Rinaldi, Aniello Panariello, Giacomo Salici, Fengyuan Liu, Marco Ciccone, Angelo Porrello, and Simone Calderara. Gradient-sign masking for task vector transport across pre-trained models. *arXiv preprint arXiv:2510.09658*, 2025b. 1, 7

Rana Shahroz, Pingzhi Li, Sukwon Yun, Zhenyu Wang, Shahriar Nirjon, Chau-Wai Wong, and Tianlong Chen. PortLLM: Personalizing evolving large language models with training-free and portable model patches. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=gyHoR6uFhU. 1

Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura O'Mahony, Mike Zhang, Ramith Hettiarachchi, Joseph Wilson, Marina Machado, Luisa Moura, Dominik Krzemiński, Hakimeh Fadaei, Irem Ergun, Ifeoma Okoh, Aisha Alaagib, Oshan Mudannayake, Zaid Alyafeai, Vu Chien, Sebastian Ruder, Surya Guthikonda, Emad Alghamdi, Sebastian Gehrmann, Niklas Muennighoff, Max Bartolo, Julia Kreutzer, Ahmet Üstün, Marzieh Fadaee, and Sara Hooker. Aya dataset: An open-access collection for multilingual instruction tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11521–11567, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.620. URL https://aclanthology.org/2024.acl-long.620/. 26

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *Advances in Neural Information Processing Systems*, 37:7821–7846, 2024. 26

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJ4km2R5t7. 4, 7, 24

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jimenez, François Fleuret, and Pascal Frossard. Localizing task information for improved model merging and compression. *arXiv preprint arXiv:2405.07813*, 2024. 14

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with OSS-instruct. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 52632–52657. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wei24h.html. 26

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023. 14

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. *arXiv preprint arXiv:2310.02575*, 2023. 14

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024. 1

Kotaro Yoshida, Yuji Naraki, Takafumi Horie, Ryosuke Yamaki, Ryotaro Shimizu, Yuki Saito, Julian McAuley, and Hiroki Naganuma. Mastering task arithmetic: τJp as a key indicator for weight disentanglement. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=1VwWi6zbxs. 14

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024. 4, 7, 17, 18

Frederic Z Zhang, Paul Albert, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. Knowledge composition using task vectors with learned anisotropic scaling. *Advances in Neural Information Processing Systems*, 37:67319–67354, 2024. 14

Luca Zhou, Daniele Solombrino, Donato Crisostomi, Maria Sofia Bucarelli, Giuseppe Alessio D'Inverno, Fabrizio Silvestri, and Emanuele Rodolà. On task vectors and gradients. *arXiv preprint arXiv:2508.16082*, 2025. 14

A Additional Discussions

A.1 Related Works

Ilharco et al. (2022) define a task vector as the weight difference between a pretrained model and its fine-tuned counterpart on a downstream task. They systematically demonstrate the arithmetic manipulability of this representation—adding task vectors can improve multi-task behavior without additional training, while negating a task vector enables targeted forgetting. Indeed, task vectors have become a common primitive in model merging (Yang et al., 2023; Yadav et al., 2023; He et al., 2024). Here, we focus on three aspects most relevant to our study: task-vector understanding, task-vector interference, and task-vector transfer.

Task-vector understanding. Ortiz-Jimenez et al. (2023) argue that weight disentanglement is central to why such task arithmetic can work, and show that disentanglement can emerge during pretraining and be amplified during fine-tuning by linearizing models in their tangent space. More recently, Zhou et al. (2025) provide an optimization-based interpretation: under standard gradient descent, the task vector produced by one fine-tuning epoch is equivalent to a learning-rate–scaled negative gradient, with this connection holding approximately across multiple epochs. From a different perspective, Li et al. (2025) study when task vectors can be provably effective through a generalization analysis of nonlinear Transformers. In a conceptual learning setting, they characterize conditions under which task-vector addition/negation is effective across different task relations (aligned, unrelated, and conflicting), together with generalization guarantees that depend on coefficient choices. Orthogonal to these works, we study the transferability of task vectors across base models and provide conditions and corrections that explain when a task vector learned on one model can be ported to another.

Task-vector interference. A parallel line of work focuses on mitigating interference in task-vector arithmetic. Zhang et al. (2024) propose aTLAS, which learns block-wise scaling coefficients to form anisotropic weighted combinations, improving disentanglement and reducing interference. Yoshida et al. (2025) introduce a new metric τ-Jp, establish its causal connection to arithmetic interference, and suggest mitigating interference by regularizing to reduce τ-Jp. Wang et al. (2024) proposes TALL-masks, which identify task support to retrieve substructures that approach single-task performance, enabling a finer-grained use of task vectors. In contrast, we study interference induced by cross-model transfer of a single task vector.

Task-vector transfer. Recent works have explored task-vector transfer, which directly applies a task vector learned from one model to another in order to impart similar task capabilities without re-running fine-tuning. For example, Cao et al. (2025) propose ParamΔ, showing that adding the task vector to a newer base model can restore much of the post-training capability at zero additional training cost. Rinaldi et al. (2025a) propose to re-basin task vectors via weight permutations so that updates can better align the target model. The mixed empirical outcomes in these works motivate our

study. Under suitable regularity conditions, we provide a theoretical account of when transfer works. We also propose CurvTV to correct transfer when vanilla task-vector transfer breaks down.

A.2 Limitations

Our study has two main limitations. First, in this work, we restrict both our theoretical analysis and CurvTV to source and target models that share the same architecture and vocabulary. This regime narrows the scope of our analysis and limits CurvTV's applicability to settings with aligned parameterizations and compatible tokenizers, where direct parameter addition and curvature-based compensation are well defined. Extending either the analysis or the method beyond this regime to settings with heterogeneous architectures or vocabularies is left to future work. Second, our formulation uses proximal fine-tuning objectives. This differs from common fine-tuning practice, which rarely uses explicit proximal regularization and instead relies on mechanisms such as early stopping, learning-rate schedules, and weight decay. An important open direction is therefore to connect α_1, α_2 more precisely to the effective regularization induced by these practical training procedures, rather than treating them purely as tunable hyperparameters.

B Further details on Section 3

B.1 Proof of Theorem 3.3

Theorem 3.3*. *Suppose that $\alpha I + \bar{H}$ is nonsingular. Then there exists a point $\widetilde{\theta}$ on the line segment between $\theta_2^\star$ and θ_{tv} such that*

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) = \frac{1}{2}\big(\bar{H}(\theta_2-\theta_1)\big)^\top \big(\alpha I + \bar{H}\big)^{-1}\big(\alpha I + \nabla^2\ell_\tau(\widetilde{\theta})\big)\big(\alpha I + \bar{H}\big)^{-1}\bar{H}(\theta_2-\theta_1).$$

Moreover, if $0 \preceq \nabla^2\ell_\tau(\theta) \preceq \beta I$ along the line segments connecting $\theta_1^\star$ to $\theta_2^\star$ and $\theta_2^\star$ to θ_{tv}, then

$$\frac{\alpha\left\|\bar{H}(\theta_2-\theta_1)\right\|^2}{2(\alpha+\beta)^2} \le \mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{(\alpha+\beta)\left\|\bar{H}(\theta_2-\theta_1)\right\|^2}{2\alpha^2}.$$

Hence, $\|\bar{H}(\theta_2-\theta_1)\|^2$ quantifies the violation magnitude of the directional curvature condition.

Proof. Let

$$d := \theta_2 - \theta_1, \qquad r := \bar{H}d, \qquad A := \alpha I + \bar{H}.$$

By Eq. (2), the deviation of the transferred solution satisfies

$$A(\theta_{\mathrm{tv}} - \theta_2^\star) = r.$$

Since $\bar{H}$ is an average of symmetric Hessians, A is symmetric. Its nonsingularity gives

$$\theta_{\mathrm{tv}} - \theta_2^\star = A^{-1}r. \tag{6}$$

Because $\theta_2^\star$ minimizes $\mathcal{L}_2$, it satisfies $\nabla\mathcal{L}_2(\theta_2^\star) = 0$. Applying Taylor's theorem to $\mathcal{L}_2$ along the line segment from $\theta_2^\star$ to θ_{tv}, there exists a point $\widetilde{\theta}$ on this segment such that

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) = \frac{1}{2}(\theta_{\mathrm{tv}} - \theta_2^\star)^\top \nabla^2\mathcal{L}_2(\widetilde{\theta})(\theta_{\mathrm{tv}} - \theta_2^\star).$$

Using

$$\nabla^2\mathcal{L}_2(\widetilde{\theta}) = \alpha I + \nabla^2\ell_\tau(\widetilde{\theta})$$

and substituting Eq. (6) yields

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) = \frac{1}{2}r^\top A^{-1}\big(\alpha I + \nabla^2\ell_\tau(\widetilde{\theta})\big)A^{-1}r,$$

which proves the first identity.

It remains to bound this quadratic form. Averaging the assumed curvature bounds along the segment from $\theta_1^\star$ to $\theta_2^\star$ gives

$$0 \preceq \bar{H} \preceq \beta I, \qquad \alpha I \preceq A \preceq (\alpha+\beta)I.$$

The same bounds along the segment from $\theta_2^\star$ to θ_{tv} imply

$$\alpha I \preceq \alpha I + \nabla^2 \ell_\tau(\widetilde{\theta}) \preceq (\alpha + \beta) I.$$

Consequently,

$$\frac{\alpha}{2}\|A^{-1}r\|^2 \le \mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2}\|A^{-1}r\|^2.$$

Finally, the spectral bounds on A give

$$\frac{1}{\alpha+\beta}\|r\| \le \|A^{-1}r\| \le \frac{1}{\alpha}\|r\|.$$

Combining the last two displays and recalling $r = \bar{H}(\theta_2 - \theta_1)$ proves the stated lower and upper bounds. □

B.2 Gradient-Field Translation in the Quadratic Case

For a quadratic task loss, the directional curvature condition has a direct interpretation in terms of gradient-field translation. Intuitively, when the directional curvature condition holds, the gradient fields of $\mathcal{L}_1(\theta)$ and $\mathcal{L}_2(\theta)$ coincide up to translation. Suppose

$$\ell_\tau(\theta) = \frac{1}{2}\theta^\top H \theta + b^\top \theta + c, \qquad H \succeq 0,$$

so that $\nabla \ell_\tau(\theta) = H\theta + b$ and the path-averaged Hessian reduces to the constant matrix $\bar{H} = H$. The gradient of the regularized objective decomposes as

$$\nabla \mathcal{L}_i(\theta) = \nabla \ell_\tau(\theta) + \alpha(\theta - \theta_i).$$

When the directional curvature condition $H(\theta_2 - \theta_1) = 0$ holds, the task gradient is translation-invariant along $\theta_2 - \theta_1$:

$$\begin{aligned} \nabla \ell_\tau\big(\theta - (\theta_2 - \theta_1)\big) &= \nabla \ell_\tau(\theta) - H(\theta_2 - \theta_1) \\ &= \nabla \ell_\tau(\theta). \end{aligned} \tag{7}$$

Meanwhile, the regularization gradient is translation-equivariant:

$$\alpha\big(\theta - (\theta_2 - \theta_1) - \theta_1\big) = \alpha(\theta - \theta_2). \tag{8}$$

Combining Eqs. (7) and (8) gives

$$\begin{aligned} \nabla \mathcal{L}_1\big(\theta - (\theta_2 - \theta_1)\big) &= \nabla \ell_\tau\big(\theta - (\theta_2 - \theta_1)\big) \\ &\quad + \alpha\big(\theta - (\theta_2 - \theta_1) - \theta_1\big) \\ &= \nabla \ell_\tau(\theta) + \alpha(\theta - \theta_2) \\ &= \nabla \mathcal{L}_2(\theta). \end{aligned}$$

Thus, the gradient field of $\mathcal{L}_2$ is exactly the translation of that of $\mathcal{L}_1$ by $\theta_2 - \theta_1$.

When $H(\theta_2 - \theta_1) \neq 0$, the same calculation instead yields

$$\nabla \mathcal{L}_1\big(\theta - (\theta_2 - \theta_1)\big) = \nabla \mathcal{L}_2(\theta) - H(\theta_2 - \theta_1).$$

Using $\theta_{\mathrm{tv}} - (\theta_2 - \theta_1) = \theta_1^\star$, evaluating this identity at $\theta = \theta_{\mathrm{tv}}$ gives

$$\nabla \mathcal{L}_2(\theta_{\mathrm{tv}}) = \nabla \mathcal{L}_1(\theta_1^\star) + H(\theta_2 - \theta_1) = H(\theta_2 - \theta_1).$$

Hence, $H(\theta_2 - \theta_1)$ is exactly the gradient residual at θ_{tv}: it vanishes when the directional curvature condition holds and otherwise quantifies the failure of gradient-field translation in this quadratic setting.

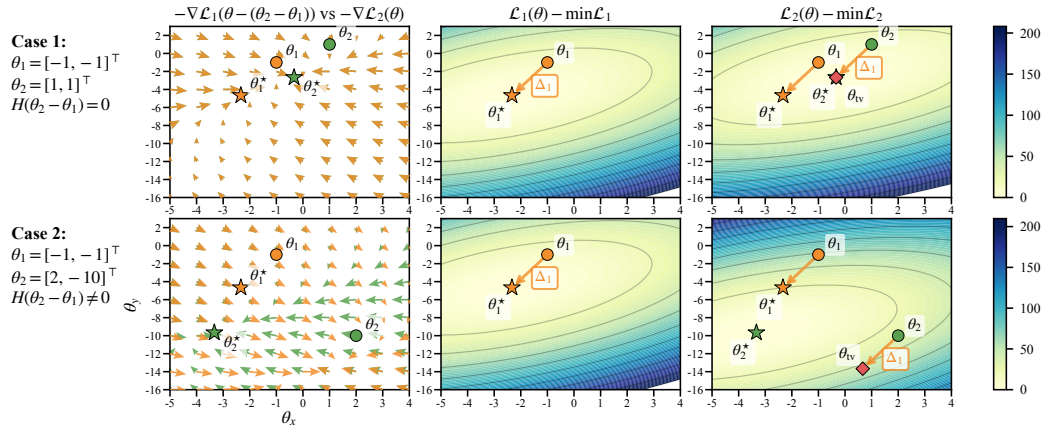


Figure 3: Gradient-field translation in a two-dimensional quadratic example. The left column compares $-\nabla\mathcal{L}_1\big(\theta - (\theta_2 - \theta_1)\big)$ with $-\nabla\mathcal{L}_2(\theta)$, while the middle and right columns show the landscapes of $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. The top and bottom rows correspond to $H(\theta_2 - \theta_1) = 0$ and $H(\theta_2 - \theta_1) \neq 0$, respectively.

Two-dimensional illustration. Fig. 3 illustrates this mechanism using a two-dimensional quadratic example. Consider the task loss

$$\ell_\tau(\theta_x, \theta_y) = \frac{1}{2}\left(\begin{bmatrix}\theta_x\\ \theta_y\end{bmatrix} - \begin{bmatrix}1\\ 0\end{bmatrix}\right)^\top \begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\left(\begin{bmatrix}\theta_x\\ \theta_y\end{bmatrix} - \begin{bmatrix}1\\ 0\end{bmatrix}\right) + [0 \quad 5]\begin{bmatrix}\theta_x\\ \theta_y\end{bmatrix}.$$

Its Hessian has null space $\mathrm{Null}(H) = \mathrm{span}\{(1, 1)^\top\}$. We set $\alpha = 1$ in both cases below.

Case 1 (top row). We set $\theta_1 = [-1, -1]^\top$ and $\theta_2 = [1, 1]^\top$, so the directional curvature condition holds. The translated gradient field of $\mathcal{L}_1$ exactly matches that of $\mathcal{L}_2$ (*left*), and the task vector learned at θ_1 (*middle*) reaches $\theta_2^\star$ when applied to θ_2 (*right*).

Case 2 (bottom row). Keeping $\theta_1 = [-1, -1]^\top$, we set $\theta_2 = [2, -10]^\top$, so the directional curvature condition is violated. The translated gradient field of $\mathcal{L}_1$ no longer aligns with that of $\mathcal{L}_2$ (*left*), so the same task vector (*middle*) fails to reach $\theta_2^\star$ and may even move the transferred model into a higher-loss region (*right*).

B.3 Correlation Analysis

This appendix provides additional experimental details for the correlation analysis in Section 3.2.

Construction of transfer instances. We use RoBERTa as the anchor model θ_1. Following Yu et al. (2024), we adopt a standard fine-tuning protocol with weight decay for both the source and target models. For each of the eight GLUE tasks (see Appendix D.1.2 for task details), we fine-tune θ_1 to obtain $\theta_1^\star$ and define the corresponding task vector

$$\Delta_1 := \theta_1^\star - \theta_1.$$

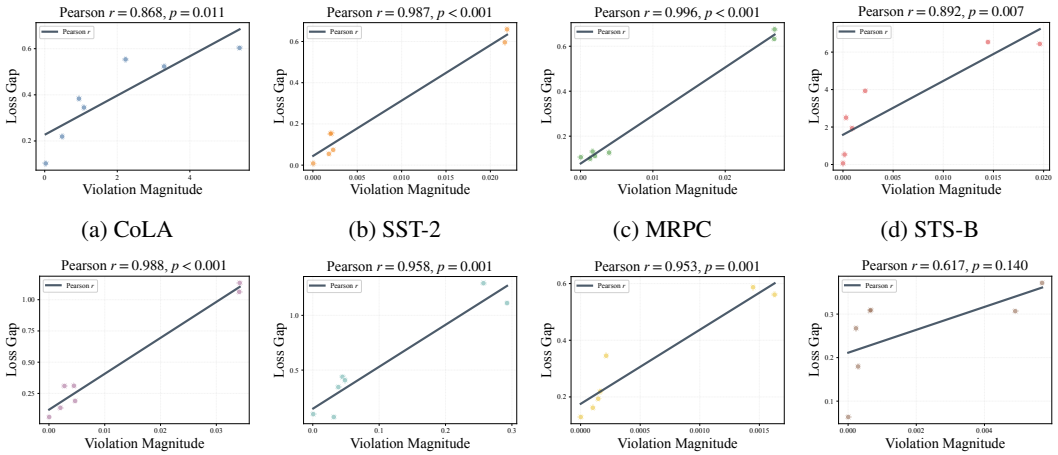
We then select seven publicly available RoBERTa-based continued-pretraining models as transfer targets θ_2, including TimeLM-22, BioMed, CodeBERT, SmartBERT, AEC-RoBERTa, MIMIC-RoBERTa, and Wiki-RoBERTa. Appendix D.1.1 describes these models and their Hugging Face links. For each task τ and each target model θ_2, we construct the transferred model

$$\theta_{\mathrm{tv}} := \theta_2 + \Delta_1,$$

resulting in $8 \times 7 = 56$ transfer instances. For each instance, we also directly fine-tune the target model θ_2 on the same task to obtain the reference model $\theta_2^\star$ for loss-gap evaluation.

Correlation analysis. For each transfer instance, the theoretical violation magnitude is

$$X_{\mathrm{theory}} := \|\bar{H}(\theta_2 - \theta_1)\|^2.$$



(a) CoLA (b) SST-2 (c) MRPC (d) STS-B

(e) QQP (f) MNLI (g) QNLI (h) RTE

Figure 4: Point-Hessian proxy for the violation magnitude of the directional curvature condition versus task-loss gap Y, excluding regularization terms, across the eight GLUE tasks.

Directly computing $\bar{H}$ requires integrating Hessians along the path between the two fine-tuned solutions and is computationally prohibitive. We therefore use the point Hessian $H_1^\star := \nabla^2 \ell_\tau(\theta_1^\star)$ as a proxy for $\bar{H}$ and estimate it using K-FAC (Martens & Grosse, 2015) with $N = 5000$ training samples. Denoting this estimate by $\widehat{H}_1^\star$, we compute the practical proxy

$$X_{\text{proxy}} := \|\widehat{H}_1^\star(\theta_2 - \theta_1)\|^2.$$

For each transfer instance, we define the response as the task-loss gap

$$Y := \ell_\tau(\theta_{\text{tv}}) - \ell_\tau(\theta_2^\star).$$

Our fine-tuning procedure uses weight decay for regularization (Yu et al., 2024), so Y is computed without the explicit proximal penalty in the theoretical loss gap of Theorem 3.3. To account for differences in loss scales across tasks, we perform correlation analyses within each task. Specifically, for a fixed task τ, we collect the seven transfer instances $\{(X_{\text{proxy},j}, Y_j)\}_{j=1}^7$ across target models and compute the Pearson correlation coefficient r_τ and its p-value.

Results. Fig. 4 visualizes the per-task linear trends and reports Pearson correlations and p-values. To summarize correlations across the eight tasks, we aggregate $\{r_\tau\}_{\tau=1}^8$ using a random-effects meta-analysis with Fisher's z-transformation, obtaining an overall correlation $\bar{r} = 0.959$ with a 95% confidence interval of $[0.884, 0.986]$. In addition, we z-score X_{proxy} and Y within each task and pool the normalized points to visualize the overall linear trend in Fig. 1(a). Overall, these results show a strong positive association between the point-Hessian proxy for the violation magnitude and the loss gap, providing empirical support for our theoretical analysis.

B.4 Beyond Exact Proximal Stationarity

To account for departures from exact proximal stationarity, we interpret $\theta_i^\star$ in this discussion as the endpoints produced by fine-tuning and define, for reference coefficients $\alpha_i > 0$,

$$r_i := \nabla \ell_\tau(\theta_i^\star) + \alpha_i(\theta_i^\star - \theta_i), \qquad i \in \{1, 2\}.$$

For shared $\alpha_1 = \alpha_2 = \alpha$, the gradient-difference identity gives

$$(\bar{H} + \alpha I)(\theta_{\text{tv}} - \theta_2^\star) = \bar{H}(\theta_2 - \theta_1) + r_1 - r_2.$$

If $\bar{H} + \alpha I$ is nonsingular, exact task-vector transfer therefore satisfies

$$\theta_{\text{tv}} = \theta_2^\star \quad \Longleftrightarrow \quad \bar{H}(\theta_2 - \theta_1) = r_2 - r_1.$$

The original directional curvature condition is thus modified by the difference between the source and target stationarity residuals. In particular, equal residuals recover the original condition even when neither residual vanishes.

Allowing distinct α_1 and α_2 for CurvTV, replacing $\bar{H}$ by $H_1^\star$ gives the following residual-corrected expression whenever $H_1^\star + \alpha_2 I$ is nonsingular:

$$\theta_{\mathrm{tv}} - (H_1^\star + \alpha_2 I)^{-1}\left[H_1^\star(\theta_2 - \theta_1) + (\alpha_2 - \alpha_1)(\theta_1^\star - \theta_1) + r_1 - r_2\right].$$

The additional term incorporates the difference between the stationarity residuals into the compensation. Omitting this term recovers the CurvTV update used in our experiments. Because r_2 depends on the unavailable target fine-tuning endpoint, this expression is an analytical extension whose practical use requires access to, or an estimate of, $r_1 - r_2$.

C Further Details on Section 4

C.1 Hessian Implementation

K-FAC approximation. Consider a linear layer l with input activation vector $a^{(l)} \in \mathbb{R}^{d_{\mathrm{in}}^{(l)}}$ and output gradient $g^{(l)} \in \mathbb{R}^{d_{\mathrm{out}}^{(l)}}$. The per-example gradient with respect to the weight matrix $W^{(l)} \in \mathbb{R}^{d_{\mathrm{out}}^{(l)} \times d_{\mathrm{in}}^{(l)}}$ takes the outer-product form

$$\nabla_{W^{(l)}} \ell_\tau = g^{(l)} {a^{(l)}}^\top.$$

Vectorizing gives $\mathrm{vec}(\nabla_{W^{(l)}} \ell_\tau) = a^{(l)} \otimes g^{(l)}$. Under the standard K-FAC independence approximation (Martens & Grosse, 2015), we use the following curvature surrogate for the layer-wise Hessian block:

$$H^{(l)} \approx \mathbb{E}\left[\mathrm{vec}(\nabla_{W^{(l)}} \ell_\tau)\mathrm{vec}(\nabla_{W^{(l)}} \ell_\tau)^\top\right] \approx \underbrace{\mathbb{E}\left[a^{(l)} {a^{(l)}}^\top\right]}_{A^{(l)}} \otimes \underbrace{\mathbb{E}\left[g^{(l)} {g^{(l)}}^\top\right]}_{G^{(l)}}.$$

Here, $A^{(l)}$ and $G^{(l)}$ are second-order statistics of input activations and output gradients, respectively. We retain both Kronecker factors in the CurvTV implementation. Given N calibration samples, we estimate these factors using empirical second moments:

$$A^{(l)} = \frac{1}{N}\sum_{i=1}^{N} a_i^{(l)} {a_i^{(l)}}^\top, \qquad G^{(l)} = \frac{1}{N}\sum_{i=1}^{N} g_i^{(l)} {g_i^{(l)}}^\top.$$

Remark C.1 (Numerical stabilization). We add a small diagonal damping term to each K-FAC block, scaled by its largest eigenvalue, then normalize the overall curvature approximation to unit mean eigenvalue. Reported regularization coefficients use this normalized scale.

Computational and storage costs. Collecting the activations and output gradients on N calibration samples incurs an $O(NC)$ forward–backward cost, where C is the cost of a forward–backward pass per sample. Computing the empirical moments and applying the compensation require additional matrix operations. Since $A^{(l)}$ and $G^{(l)}$ are square matrices of sizes $d_{\mathrm{in}}^{(l)}$ and $d_{\mathrm{out}}^{(l)}$, respectively, storing all Kronecker factors requires

$$O\left(\sum_l \left[(d_{\mathrm{in}}^{(l)})^2 + (d_{\mathrm{out}}^{(l)})^2\right]\right)$$

memory. This bound covers factor storage only and excludes model parameters and temporary activation, gradient, and solver memory.

C.2 A Newton Step Followed by Merging

Under the proximal formulation, source stationarity gives the following Newton step on the task loss when $H_1^\star$ is nonsingular:

$$\theta_{\mathrm{N}} = \theta_1^\star - (H_1^\star)^{-1}\nabla \ell_\tau(\theta_1^\star) = \theta_1^\star + \alpha_1 (H_1^\star)^{-1}\Delta_1.$$

Rearranging the CurvTV update yields

$$\begin{aligned}\theta_{\mathrm{CurvTV}} &= (H_1^\star + \alpha_2 I)^{-1}\left(H_1^\star \theta_1^\star + \alpha_1 \Delta_1 + \alpha_2 \theta_2\right) \\ &= (H_1^\star + \alpha_2 I)^{-1}\left(H_1^\star \theta_{\mathrm{N}} + \alpha_2 \theta_2\right).\end{aligned}$$

Thus, CurvTV admits an interpretation as a Newton step followed by a curvature-weighted merge with θ_2.

The stationarity identity motivates using $-\alpha_1\Delta_1$ as a task-gradient proxy. This makes the two coefficients' roles explicit in the local quadratic approximation to the proximal objectives: the α_1 term removes the source penalty's contribution, while α_2 introduces a new penalty centered at θ_2. In practice, CurvTV combines this gradient proxy with a K-FAC curvature surrogate.

Relation to Fisher merging. Fisher merging directly combines trained model parameters using Fisher-based precision weights (Matena & Raffel, 2022). In the decomposition above, CurvTV includes an additional Newton step before merging. Its target weight $\alpha_2 I$ comes from the proximal penalty rather than an estimated target Fisher matrix.

C.3 Proofs of Theorems in Section 4.2

Lemma C.2. *Under Assumption 4.1, every $\theta_{\text{cand}} \in \mathcal{C}$ satisfies*

$$\mathcal{L}_2(\theta_{\text{cand}}) - \mathcal{L}_2(\theta_2^\star) \;\le\; \frac{\alpha+\beta}{2}\|\theta_{\text{cand}} - \theta_2^\star\|^2.$$

Proof. Fix $\theta_{\text{cand}} \in \mathcal{C}$ and let $d := \theta_{\text{cand}} - \theta_2^\star$. For every $t \in [0,1]$, the point $\theta_2^\star + td$ lies in $[\theta_2^\star, \theta_{\text{cand}}] \subseteq \mathcal{R}$. Hence, Assumption 4.1 gives

$$\nabla^2\mathcal{L}_2(\theta_2^\star + td) = \nabla^2\ell_\tau(\theta_2^\star + td) + \alpha I \preceq (\alpha+\beta)I.$$

Using the integral form of Taylor's theorem and the stationarity condition $\nabla\mathcal{L}_2(\theta_2^\star) = 0$, we obtain

$$\mathcal{L}_2(\theta_{\text{cand}}) - \mathcal{L}_2(\theta_2^\star) = \int_0^1 (1-t)d^\top\nabla^2\mathcal{L}_2(\theta_2^\star + td)d\,dt \le \frac{\alpha+\beta}{2}\|d\|^2,$$

as claimed. □

Theorem 4.3* (Vanilla TV). *Under Assumption 4.1, the vanilla transferred solution θ_{tv} satisfies*

$$\mathcal{L}_2(\theta_{\text{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2}\|\psi_\alpha(\bar{H})\|^2\|\theta_2 - \theta_1\|^2,$$

where $\psi_\alpha(\bar{H})$ is defined in Eq. (5).

Proof. We start from the first-order optimality conditions of $\theta_1^\star$ and $\theta_2^\star$:

$$\nabla\ell_\tau(\theta_1^\star) + \alpha(\theta_1^\star - \theta_1) = 0, \qquad \nabla\ell_\tau(\theta_2^\star) + \alpha(\theta_2^\star - \theta_2) = 0. \tag{9}$$

Subtracting the first identity from the second yields

$$\nabla\ell_\tau(\theta_2^\star) - \nabla\ell_\tau(\theta_1^\star) + \alpha(\theta_2^\star - \theta_1^\star) = \alpha(\theta_2 - \theta_1). \tag{10}$$

To make progress, we express the gradient difference along the line segment between $\theta_1^\star$ and $\theta_2^\star$. The fundamental theorem of calculus gives

$$\nabla\ell_\tau(\theta_2^\star) - \nabla\ell_\tau(\theta_1^\star) = \left(\int_0^1 \nabla^2\ell_\tau\big(\theta_1^\star + t(\theta_2^\star - \theta_1^\star)\big)\,dt\right)(\theta_2^\star - \theta_1^\star) = \bar{H}(\theta_2^\star - \theta_1^\star), \tag{11}$$

where

$$\bar{H} := \int_0^1 \nabla^2\ell_\tau\big(\theta_1^\star + t(\theta_2^\star - \theta_1^\star)\big)\,dt. \tag{12}$$

Plugging Eq. (11) into Eq. (10), we obtain the linear system

$$(\alpha I + \bar{H})(\theta_2^\star - \theta_1^\star) = \alpha(\theta_2 - \theta_1). \tag{13}$$

Under Assumption 4.1, $\alpha I + \bar{H}$ is invertible, and hence

$$\theta_2^\star - \theta_1^\star = \alpha(\alpha I + \bar{H})^{-1}(\theta_2 - \theta_1). \tag{14}$$

Recalling $\theta_{\mathrm{tv}} = \theta_2 + (\theta_1^\star - \theta_1)$, we can relate θ_{tv} to $\theta_2^\star$:

$$\begin{aligned}\theta_{\mathrm{tv}} - \theta_2^\star &= (\theta_2 - \theta_1) + (\theta_1^\star - \theta_2^\star) \\ &= \Big(I - \alpha(\alpha I + \bar{H})^{-1}\Big)(\theta_2 - \theta_1) \\ &= (\alpha I + \bar{H})^{-1}\bar{H}(\theta_2 - \theta_1) = \psi_\alpha(\bar{H})(\theta_2 - \theta_1),\end{aligned}$$

where the last equality follows from the definition of $\psi_\alpha(\cdot)$ in Eq. (5). Taking norms yields

$$\|\theta_{\mathrm{tv}} - \theta_2^\star\| \le \|\psi_\alpha(\bar{H})\|\,\|\theta_2 - \theta_1\|. \tag{15}$$

Finally, applying Lemma C.2 with $\theta_{\mathrm{cand}} = \theta_{\mathrm{tv}}$, we obtain

$$\mathcal{L}_2(\theta_{\mathrm{tv}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2}\|\theta_{\mathrm{tv}} - \theta_2^\star\|^2.$$

Combining this with Eq. (15) yields the desired bound. □

Theorem 4.4* (CurvTV). *Under Assumptions 4.1 and 4.2, the CurvTV solution satisfies*

$$\mathcal{L}_2(\theta_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{(\alpha+\beta)\Upsilon_\alpha^2}{2}\|\theta_2 - \theta_1\|^2 \min\left\{\frac{\rho^2}{4}\|\theta_2 - \theta_1\|^2,\ \frac{(\alpha+\beta)^2}{\|I - \psi_\alpha(\bar{H})\|^2}\right\},$$

where $\Upsilon_\alpha := \frac{\|I-\psi_\alpha(\bar{H})\|^2\|I-\psi_\alpha(H_1^\star)\|}{\alpha}$.

Proof. As shown in Eq. (4), vanilla task-vector transfer satisfies

$$\theta_{\mathrm{tv}} - \theta_2^\star = \psi_\alpha(\bar{H})(\theta_2 - \theta_1). \tag{16}$$

By definition of CurvTV,

$$\theta_{\mathrm{CurvTV}} = \theta_{\mathrm{tv}} - \psi_\alpha(H_1^\star)(\theta_2 - \theta_1),$$

where $H_1^\star := \nabla^2\ell_\tau(\theta_1^\star)$. Combining this definition with Eq. (16) yields the exact decomposition

$$\theta_{\mathrm{CurvTV}} - \theta_2^\star = \big(\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)\big)(\theta_2 - \theta_1). \tag{17}$$

We next control the operator difference $\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)$. Using $\psi_\alpha(A) = I - \alpha(\alpha I + A)^{-1}$, we can rewrite

$$\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star) = \alpha\big[(\alpha I + H_1^\star)^{-1} - (\alpha I + \bar{H})^{-1}\big].$$

By the resolvent identity, we have

$$(\alpha I + H_1^\star)^{-1} - (\alpha I + \bar{H})^{-1} = (\alpha I + H_1^\star)^{-1}(\bar{H} - H_1^\star)(\alpha I + \bar{H})^{-1},$$

and therefore

$$\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star) = \alpha(\alpha I + H_1^\star)^{-1}(\bar{H} - H_1^\star)(\alpha I + \bar{H})^{-1}. \tag{18}$$

Under Assumption 4.1, both $\alpha I + H_1^\star$ and $\alpha I + \bar{H}$ are positive definite, so the inverses exist. Moreover, using $\alpha(\alpha I + H_1^\star)^{-1} = I - \psi_\alpha(H_1^\star)$ and $\alpha(\alpha I + \bar{H})^{-1} = I - \psi_\alpha(\bar{H})$, we have

$$\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star) = \frac{1}{\alpha}\big(I - \psi_\alpha(H_1^\star)\big)(\bar{H} - H_1^\star)\big(I - \psi_\alpha(\bar{H})\big).$$

Taking operator norms gives

$$\|\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)\| \le \frac{\|I - \psi_\alpha(H_1^\star)\|\,\|I - \psi_\alpha(\bar{H})\|}{\alpha}\,\|\bar{H} - H_1^\star\|. \tag{19}$$

We first derive a displacement-independent bound. Averaging Assumption 4.1 along the path defining $\bar{H}$ yields

$$0 \prec \alpha I + \bar{H} \preceq (\alpha+\beta)I.$$

The same assumption at $\theta_1^\star$ gives

$$0 \prec \alpha I + H_1^\star \preceq (\alpha+\beta)I.$$

Consequently,

$$-(\alpha+\beta)I \prec \bar{H} - H_1^\star \prec (\alpha+\beta)I,$$

and hence

$$\|\bar{H} - H_1^\star\| \le \alpha + \beta. \tag{20}$$

Combining Eqs. (19) and (20) gives

$$\|\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)\| \le \frac{(\alpha+\beta)\Upsilon_\alpha}{\|I - \psi_\alpha(\bar{H})\|}. \tag{21}$$

The denominator is nonzero because $I - \psi_\alpha(\bar{H}) = \alpha(\alpha I + \bar{H})^{-1} \succ 0$.

We next derive a sharper displacement-dependent bound. By definition of $\bar{H}$,

$$\bar{H} - H_1^\star = \int_0^1 \Big(\nabla^2 \ell_\tau(\theta_1^\star + t(\theta_2^\star - \theta_1^\star)) - \nabla^2 \ell_\tau(\theta_1^\star)\Big) dt.$$

Assumption 4.2 implies that for each $t \in [0,1]$,

$$\big\|\nabla^2 \ell_\tau(\theta_1^\star + t(\theta_2^\star - \theta_1^\star)) - \nabla^2 \ell_\tau(\theta_1^\star)\big\| \le \rho\, t\, \|\theta_2^\star - \theta_1^\star\|.$$

Integrating over t yields

$$\|\bar{H} - H_1^\star\| \le \frac{\rho}{2}\, \|\theta_2^\star - \theta_1^\star\|. \tag{22}$$

To express $\|\theta_2^\star - \theta_1^\star\|$ in terms of $\|\theta_2 - \theta_1\|$, we reuse Eq. (13) from the baseline argument:

$$(\alpha I + \bar{H})(\theta_2^\star - \theta_1^\star) = \alpha(\theta_2 - \theta_1),$$

which implies

$$\theta_2^\star - \theta_1^\star = \alpha(\alpha I + \bar{H})^{-1}(\theta_2 - \theta_1) = \big(I - \psi_\alpha(\bar{H})\big)(\theta_2 - \theta_1).$$

Consequently,

$$\|\theta_2^\star - \theta_1^\star\| \le \|I - \psi_\alpha(\bar{H})\|\, \|\theta_2 - \theta_1\|. \tag{23}$$

Combining Eqs. (19), (22) and (23) yields

$$\|\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)\| \le \frac{\rho}{2} \cdot \frac{\|I - \psi_\alpha(\bar{H})\|^2 \|I - \psi_\alpha(H_1^\star)\|}{\alpha} \|\theta_2 - \theta_1\| = \frac{\rho \Upsilon_\alpha}{2} \|\theta_2 - \theta_1\|.$$

Combining this bound with Eq. (21) gives

$$\|\psi_\alpha(\bar{H}) - \psi_\alpha(H_1^\star)\| \le \Upsilon_\alpha \min\left\{ \frac{\rho}{2}\|\theta_2 - \theta_1\|,\ \frac{\alpha+\beta}{\|I - \psi_\alpha(\bar{H})\|} \right\}.$$

Plugging this bound into Eq. (17) gives

$$\|\theta_{\mathrm{CurvTV}} - \theta_2^\star\| \le \Upsilon_\alpha \|\theta_2 - \theta_1\| \min\left\{ \frac{\rho}{2}\|\theta_2 - \theta_1\|,\ \frac{\alpha+\beta}{\|I - \psi_\alpha(\bar{H})\|} \right\}. \tag{24}$$

Finally, applying Lemma C.2 with $\theta_{\mathrm{cand}} = \theta_{\mathrm{CurvTV}}$ converts this distance bound into a function-value bound:

$$\mathcal{L}_2(\theta_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \|\theta_{\mathrm{CurvTV}} - \theta_2^\star\|^2.$$

Substituting the above distance bound and squaring the minimum yields

$$\mathcal{L}_2(\theta_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{(\alpha+\beta)\Upsilon_\alpha^2}{2} \|\theta_2 - \theta_1\|^2 \min\left\{ \frac{\rho^2}{4}\|\theta_2 - \theta_1\|^2,\ \frac{(\alpha+\beta)^2}{\|I - \psi_\alpha(\bar{H})\|^2} \right\},$$

which is the desired two-regime optimality-gap bound. □

Corollary 4.5* (Robustness). *Let $\tilde{H}_1^\star := H_1^\star + \Delta$ denote the fixed approximate Hessian used to obtain $\tilde{\theta}_{\mathrm{CurvTV}} \in \mathcal{C}$. Under Assumptions 4.1 and 4.2, suppose additionally that $\alpha I + \tilde{H}_1^\star$ is invertible. Then the perturbed solution $\tilde{\theta}_{\mathrm{CurvTV}}$ satisfies*

$$\mathcal{L}_2(\tilde{\theta}_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \left(\frac{\rho \Upsilon_\alpha}{2} \|\theta_2 - \theta_1\|^2 + \Gamma_\alpha \|\Delta\| \|\theta_2 - \theta_1\| \right)^2,$$

where $\Gamma_\alpha := \frac{\|I - \psi_\alpha(\tilde{H}_1^\star)\| \|I - \psi_\alpha(H_1^\star)\|}{\alpha}$. Moreover, if the Hessian estimation error scales as $\|\Delta\| \le \epsilon \|\theta_2 - \theta_1\|$ for some $\epsilon \ge 0$, then

$$\mathcal{L}_2(\tilde{\theta}_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha+\beta}{2} \left(\frac{\rho \Upsilon_\alpha + 2\epsilon \Gamma_\alpha}{2} \right)^2 \|\theta_2 - \theta_1\|^4.$$

Proof. By definition,

$$\tilde{\theta}_{\mathrm{CurvTV}} = \theta_{\mathrm{tv}} - \psi_\alpha(\tilde{H}_1^\star)(\theta_2 - \theta_1), \qquad \theta_{\mathrm{CurvTV}} = \theta_{\mathrm{tv}} - \psi_\alpha(H_1^\star)(\theta_2 - \theta_1).$$

We first compare $\tilde{\theta}_{\mathrm{CurvTV}}$ with $\theta_2^\star$. The triangle inequality gives

$$\|\tilde{\theta}_{\mathrm{CurvTV}} - \theta_2^\star\| \le \|\theta_{\mathrm{CurvTV}} - \theta_2^\star\| + \|\tilde{\theta}_{\mathrm{CurvTV}} - \theta_{\mathrm{CurvTV}}\|. \tag{25}$$

Taking the local branch of Eq. (24) gives

$$\|\theta_{\mathrm{CurvTV}} - \theta_2^\star\| \le \frac{\rho \Upsilon_\alpha}{2} \|\theta_2 - \theta_1\|^2. \tag{26}$$

For the perturbation term, the two definitions above imply

$$\tilde{\theta}_{\mathrm{CurvTV}} - \theta_{\mathrm{CurvTV}} = \left(\psi_\alpha(H_1^\star) - \psi_\alpha(\tilde{H}_1^\star)\right)(\theta_2 - \theta_1),$$

and thus

$$\|\tilde{\theta}_{\mathrm{CurvTV}} - \theta_{\mathrm{CurvTV}}\| \le \|\psi_\alpha(\tilde{H}_1^\star) - \psi_\alpha(H_1^\star)\| \, \|\theta_2 - \theta_1\|. \tag{27}$$

It remains to bound $\|\psi_\alpha(H_1^\star) - \psi_\alpha(\tilde{H}_1^\star)\|$. Using the same resolvent factorization as in the proof of Theorem 4.4, we have

$$\psi_\alpha(\tilde{H}_1^\star) - \psi_\alpha(H_1^\star) = \alpha(\alpha I + H_1^\star)^{-1}(\tilde{H}_1^\star - H_1^\star)(\alpha I + \tilde{H}_1^\star)^{-1}. \tag{28}$$

Since $\tilde{H}_1^\star - H_1^\star = \Delta$ and $\alpha(\alpha I + A)^{-1} = I - \psi_\alpha(A)$, this becomes

$$\psi_\alpha(\tilde{H}_1^\star) - \psi_\alpha(H_1^\star) = \frac{1}{\alpha}\left(I - \psi_\alpha(H_1^\star)\right) \Delta \left(I - \psi_\alpha(\tilde{H}_1^\star)\right).$$

Hence, taking operator norms yields

$$\|\psi_\alpha(\tilde{H}_1^\star) - \psi_\alpha(H_1^\star)\| \le \frac{\|I - \psi_\alpha(\tilde{H}_1^\star)\| \, \|I - \psi_\alpha(H_1^\star)\|}{\alpha} \|\Delta\| = \Gamma_\alpha \|\Delta\|. \tag{29}$$

Combining Eq. (29) and Eq. (27) yields

$$\|\tilde{\theta}_{\mathrm{CurvTV}} - \theta_{\mathrm{CurvTV}}\| \le \Gamma_\alpha \|\Delta\| \, \|\theta_2 - \theta_1\|. \tag{30}$$

Combining Eq. (25), Eq. (26), and Eq. (30), we arrive at

$$\|\tilde{\theta}_{\mathrm{CurvTV}} - \theta_2^\star\| \le \frac{\rho \Upsilon_\alpha}{2} \|\theta_2 - \theta_1\|^2 + \Gamma_\alpha \|\Delta\| \, \|\theta_2 - \theta_1\|.$$

Finally, applying Lemma C.2 with $\theta_{\mathrm{cand}} = \tilde{\theta}_{\mathrm{CurvTV}}$ to the preceding distance bound proves the first claim.

For the second claim, the scaling assumption $\|\Delta\| \le \epsilon \|\theta_2 - \theta_1\|$ implies

$$\frac{\rho \Upsilon_\alpha}{2} \|\theta_2 - \theta_1\|^2 + \Gamma_\alpha \|\Delta\| \|\theta_2 - \theta_1\| \le \left(\frac{\rho \Upsilon_\alpha}{2} + \epsilon \Gamma_\alpha\right) \|\theta_2 - \theta_1\|^2.$$

Substituting this bound into the first inequality yields

$$\mathcal{L}_2(\tilde{\theta}_{\mathrm{CurvTV}}) - \mathcal{L}_2(\theta_2^\star) \le \frac{\alpha + \beta}{2} \left(\frac{\rho \Upsilon_\alpha + 2\epsilon \Gamma_\alpha}{2}\right)^2 \|\theta_2 - \theta_1\|^4,$$

which concludes the proof. □

D More Experimental Details and Results

D.1 Transfer on RoBERTa

D.1.1 Models

Here, we describe the models used in our correlation analysis (Section 3.2) and RoBERTa experiments (Section 5.1).

- RoBERTa (Liu et al., 2019)
  - A 12-layer Transformer encoder pretrained on large-scale English corpora with masked language modeling.
  - RoBERTa checkpoint
- TimeLM-22 (Loureiro et al., 2022)
  - A continued-pretraining RoBERTa model trained on approximately 154 million tweets collected up to December 2022.
  - TimeLM-22 checkpoint
- BioMed (Gururangan et al., 2020)
  - A continued-pretraining RoBERTa model trained on roughly 2.68 million biomedical scientific articles from the S2ORC (Lo et al., 2020).
  - BioMed checkpoint
- CodeBERT (Feng et al., 2020)
  - A continued-pretraining RoBERTa trained on about 2 million paired (comment, code) examples from open source libraries.
  - CodeBERT checkpoint
- SmartBERT (Huang et al., 2025)
  - A continued-pretraining RoBERTa trained on over 40,000 smart contracts.
  - SmartBERT checkpoint
- AEC-RoBERTa
  - A continued-pretraining RoBERTa trained on the AEC v1 dataset (Kpodo et al., 2024).
  - AEC-RoBERTa checkpoint
- MIMIC-RoBERTa (Dai et al., 2022)
  - A continued-pretraining RoBERTa trained on MIMIC-III datasets.
  - MIMIC-RoBERTa checkpoint
- Wiki-RoBERTa
  - A continued-pretraining RoBERTa trained on wikitext-2-raw-v1 dataset (Merity et al., 2017).
  - Wiki-RoBERTa checkpoint

D.1.2 GLUE benchmark

GLUE (Wang et al., 2019) is a popular benchmark for evaluating natural language understanding across a diverse set of sentence- and sentence-pair classification and regression tasks. We use eight tasks[1] from the GLUE benchmark, including CoLA, SST-2, MRPC, STS-B, QQP, MNLI, QNLI, and RTE. For the transfer-performance experiments in Section 5.1, we follow Huang et al. (2024) by treating the official validation set as the test set and reporting the task-specific evaluation metric $m(\cdot)$ for each dataset, summarized in Table 5. For Table 1, all entries are rounded to two decimal places, whereas averages and rankings in the main-text discussion are computed from unrounded results.

[1] It is standard practice to exclude the problematic WNLI task; see Devlin et al. (2019).

Table 5: Evaluation metrics used for GLUE tasks.

GLUE task	**Metric** $m(\cdot)$
CoLA	Matthews correlation
SST-2	Accuracy
MRPC	Average of Accuracy and F1
STS-B	Average of Pearson and Spearman correlations
QQP	Average of Accuracy and F1
MNLI	Accuracy
QNLI	Accuracy
RTE	Accuracy

D.1.3 Native capability evaluation

We evaluate native capability using TweetEval sentiment (Barbieri et al., 2020) for TimeLM-22, PubMedQA (`pqa_artificial`) (Jin et al., 2019) for BioMed, and a 10,000-example Python subset of CodeSearchNet (Husain et al., 2019) for CodeBERT. We measure masked-token top-1 accuracy on these datasets for each target model and its transferred variants.

D.2 Transfer on Llama-3.1

D.2.1 Models

Here, we describe the models used in Llama-3.1 (8B) experiments (Section 5.2).

- **Llama-3.1-8B-Instruct** (Grattafiori et al., 2024)
  - The instruction-tuned version of Llama-3.1-8B, a decoder-only Transformer language model.
  - Llama-3.1-8B-Instruct checkpoint
- **Llama-VARCO**
  - A continued-pretraining Llama-3.1 model trained on mixed Korean and English data.
  - Llama-VARCO checkpoint
- **Meditron-3**
  - A continued-pretraining Llama-3.1 model trained on clinical medicine.
  - Meditron-3 checkpoint
- **Llama-3.1-abliterated**
  - An enhanced variant produced via abliteration, yielding substantially different behavior from the base Llama-3.1 model.
  - Llama-3.1-abliterated checkpoint

D.2.2 Datasets

We evaluate transfer performance on six benchmarks spanning multilingual understanding, mathematical reasoning, and code generation. Belebele (Bandarkar et al., 2024) is a multilingual reading-comprehension benchmark covering 122 languages via multiple-choice questions. M-MMLU (Lai et al., 2023) extends MMLU (Hendrycks et al., 2020) to multilingual evaluation over 57 subjects. For both Belebele and M-MMLU, we evaluate on four languages: French, Spanish, German, and Russian. MATH-500 (Lewkowycz et al., 2022) is a 500-problem subset of MATH (Hendrycks et al., 2021) and is evaluated under a 4-shot prompting setup. GSM8K (Cobbe et al., 2021) contains 8500 grade-school math word problems and is evaluated under an 8-shot prompting setup. For code generation, we use HumanEval+ and MBPP+ (Liu et al., 2023), which are augmented versions of HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) with additional test cases for more stringent evaluation. Table 6 summarizes the task-specific evaluation metric $m(\cdot)$ for each benchmark.

Table 6: Evaluation metrics used in the Llama-3.1 (8B) experiments.

Benchmark	**Metric**
Belebele	Accuracy
M-MMLU	Accuracy
MATH-500 (4-shot)	Math-Verify
GSM8K (8-shot)	Exact Match
HumanEval+	pass@1
MBPP+	pass@1

Remark D.1. We estimate the Hessian using training samples drawn from the SFT data used to obtain $\theta_1^\star$. Concretely, for each domain, we compute the Hessian using $N = 1000$ randomly sampled training instances from Aya (Singh et al., 2024) (multilingual understanding), DartMath (Tong et al., 2024) (mathematical reasoning), and MagiCoder (Wei et al., 2024) (code generation), respectively. This design matches the Hessian-estimation data to the domain of the downstream capability being transferred.

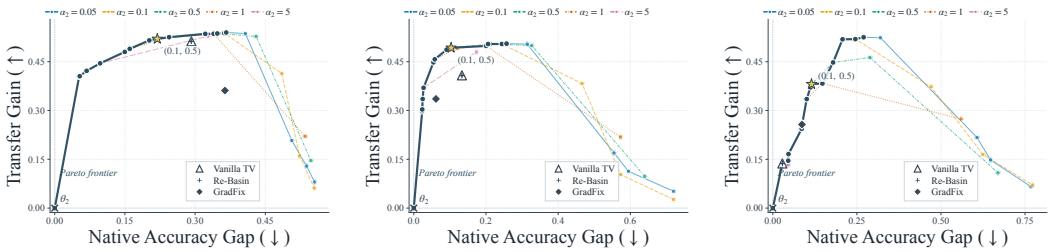
D.3 Impact of Hyperparameters

Hyperparameter selection. For the RoBERTa experiments, all transfer hyperparameters are selected on BioMed. Using random seed 42, we select 10% of each GLUE task's training set as validation data for tuning transfer gain. For the native-gap criterion on BioMed, we use 10% of the PubMedQA (`pqa_artificial`) training set, sampled with the same seed, as validation data.

For Vanilla TV, we grid-search the scaling coefficient in $\theta_2 + \lambda(\theta_1^\star - \theta_1)$ over $\lambda \in \{0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$ to maximize transfer gain. For Re-Basin and GradFix, we tune the scaling coefficients applied to their respective task vectors over the same six candidate values, also maximizing transfer gain without imposing a native capability constraint. The selected scaling coefficients are 1 for Vanilla TV and Re-Basin and 0.1 for GradFix. We keep these values fixed for transfer to TimeLM-22 and CodeBERT.

For CurvTV, we grid-search $\alpha_1, \alpha_2 \in \{0.05, 0.1, 0.5, 1, 5\}$ with $N = 1000$ Hessian samples. We apply a stricter selection criterion by first retaining only configurations whose native accuracy gap, averaged over the eight GLUE tasks, is smaller than that of Vanilla TV with its selected scaling coefficient. Among these configurations, we choose the one with the highest transfer gain averaged over the same tasks. With these coefficients fixed, we evaluate $N \in \{50, 100, 500, 1000, 2000\}$. We manually select $N = 1000$ from the BioMed curve, as increasing N further yields only marginal transfer gains while increasing Hessian-estimation cost. The final CurvTV configuration is $(\alpha_1, \alpha_2, N) = (0.1, 0.5, 1000)$, which we keep fixed for transfer to TimeLM-22 and CodeBERT.

Fig. 5 disaggregates the (α_1, α_2) sensitivity analysis by target model, averaging over the eight GLUE tasks within each target. The sweeps on TimeLM-22 and CodeBERT characterize sensitivity to the coefficients; their main transfer results use the pair selected on BioMed.



(a) TimeLM-22 (b) BioMed (c) CodeBERT

Figure 5: Target-specific (α_1, α_2) sweeps, averaged over the eight GLUE tasks for each target model.