DivMerge: A divergence-based model merging method for multi-tasking

Brahim Touayouch[1,2,*] **Loïc Fosse**[1,3,*] **Géraldine Damnati**[1] **Gwénolé Lecorvé**[1]
[1]Orange Research, Lannion, France
[2]École Polytechnique, Institut Polytechnique de Paris, Palaiseau, France
[3]CNRS, LIS, Aix Marseille Université, France
Contact: `first.last@orange.com`

Abstract

Merging fine-tuned models is a promising alternative to costly multi-task training, but task interference remains a challenge, especially as the number of tasks grows. We present DivMerge, a reference-free method that merges models trained on different tasks by minimizing Jensen-Shannon divergence between their outputs and those of the merged model, automatically balancing task importance. While the method exhibits strong theoretical properties, experiments on classification and generative tasks with autoregressive models show that DivMerge consistently outperforms prior work, and remains robust when scaling to more tasks.

1 Introduction

While demonstrating strong performance, the standard multi-task learning approach for Transformer-based language models—which involves mixing training datasets from various tasks—is expensive (Vaswani et al., 2017; Brown et al., 2020). Model merging—which combines parameters from task-specific models to produce a multi-task model—has emerged as a cost-effective alternative (Ilharco et al., 2023; Goddard et al., 2024; Yang et al., 2024b). This paradigm is particularly attractive given the large number of specialized models available on collaborative platforms such as Hugging Face (Wolf et al., 2019). However, existing methods often struggle with interference between tasks, leading to performance drops, especially when the number of tasks increases. A major difficulty in model merging is the absence of an explicit target distribution for the merged model, unlike in standard multi-task learning, where training data from various tasks are combined. To address this issue, we propose DivMerge, a novel merging method that leverages the divergence between the merged model and the original fine-tuned models to compute optimal merging coefficients for the combination of fine-tuned models' parameters. The motivation is that the output distributions of fine-tuned models can serve as practical surrogate targets for the merged model, since each fine-tuned model is optimized for its own data distribution. In practice, the merging coefficients for a set of task-specific fine-tuned models are obtained *via* an iterative optimization process. During each step, unlabelled samples specific to each task are used to estimate the divergence between the models' output probability distributions and the target distributions, guiding the process. Furthermore, all fine-tuned models are assumed to stem from a common base model. So, the coefficients are derived from task vectors—*i.e.* the parameter shifts from the base to each fine-tuned model—rather than from the models' parameters themselves. The strengths of DivMerge are that it provides:

1. **Theoretical guarantees and connections** with task interference and the standard multi-task learning approach where training data from all tasks are merged.
2. **Improved performance** compared to state-of-the-art methods, especially in standard two-task merging scenarios.
3. **Better scalability** as the number of merged tasks increases, demonstrating effective mitigation of interference.

Finally, while our method is applied on autoregressive language models, it is worth noting that the core of the method applies to other types of AI models. In the remainder, Sec. 2 positions our work in the literature before, Sec. 3 formally introduces DivMerge. Then, Sec. 4 analyses theoretical properties of the method, while Sec. 5 reports experiments on various types of tasks and models.

2 Related Work

This section first reminds the historical idea of combining tasks within a unique model, before browsing various objectives of model merging in general,

[*]Equal Contribution

and then specifically focusing on model merging for the multi-task objective.

Multi-task learning. Multi-task learning (Caruana, 1997) refers to methods that produce a model capable of solving several tasks. Standard multi-task learning generally consists in combining datasets from multiple tasks and training a model on this union, which is justified by classical results such as Stein's paradox (Stein, 1956): combining unrelated observations allows for a better estimation of their respective expectations. This philosophy is at the core of the production of most current models such as T5 (Raffel et al., 2020) and, more recently, models trained on instructions (Shengyu et al., 2023). Although the results of these models are now considered state-of-the-art in NLP, this approach has some limitations. Indeed, several studies (Baxter, 2000; Ben-David and Schuller, 2003; Maurer et al., 2016; Standley et al., 2020; Fifty et al., 2021; Jeong and Yoon, 2025) show that the choice of tasks is important in order to avoid the negative effect of some tasks to others, commonly referred to as *interference* (Yu et al., 2020). In this study, we propose a novel method grounded in information theory (Cover and Thomas, 1991) to produce a multi-task model. An originality is to establish a theoretical connection between model merging and the classical multi-task learning strategy of mixing datasets, thus unifying these perspectives within an information-theoretic framework.

Objectives of model merging. Model merging combines different models at the parameter level to create a new model with specific desired properties, whether these properties come from the original models or emerge from their combination. While such motivations for model merging are discussed by Yang et al. (2024b), the most common ones are summarized below. Based on ideas originally developed in ensemble methods (Dietterich, 2000), a historical objective of model merging is to *reduce variance* to increase generalization (Jin et al., 2023; Matena and Raffel, 2022; Ferret, 2025; Izmailov et al., 2018; Wortsman et al., 2022). The principle is to produce a more reliable model on a single target task by merging models trained on the same task with different initializations or hyperparameter settings. However the most popular objective in model merging is rather the *multi-task ability* (Ilharco et al., 2023; Yang et al., 2024c; Yu et al., 2024; Pfeiffer et al., 2021): by merging models specialized on different tasks, the resulting model should achieve good performances on all tasks. Zhou et al. (2024) and Ortiz-Jimenez et al. (2023) provide theoretical insights into why model merging can work well for this multi-task objective. Another objective is *task unlearning* (Ilharco et al., 2023; Kuo et al., 2025; Kim et al., 2024), which means that, by merging certain models (using addition and/or negation), specific behaviours or skills can be removed or "forgotten" by the resulting model. Finally, *modular learning* (Ballard, 1987; Pfeiffer et al., 2023; Chronopoulou et al., 2024) is an interesting but less explored objective, which consists of creating a model that performs well on a task by merging models trained on other (possibly unrelated) tasks, leveraging the notion of transfer between tasks.

Merging method for multi-tasking. Several model merging methods have previously been designed for the multi-task objective. The most straightforward and simple approach is *model averaging* (Wortsman et al., 2022) which simply consists in taking the uniform average of the models' parameters. As discussed in (Matena and Raffel, 2022) this uniform average is not well motivated from a theoretical point of a view and this is the reason why they proposed the *Fisher merging* method which is an average of models' parameters weighted by Fisher information of each model. This philosophy is also at the root of the methods proposed in (Jin et al., 2023; Daheim et al., 2023). Ilharco et al. (2023) introduces the notion of task vector as the shift in the parameter space from a pre-trained model to a fine-tuned one. This concept has led to the framework of *Task Arithmetic* (TA) which is now extensively used (Ortiz-Jimenez et al., 2023) and has proven its efficiency across a wide range of applications. To enhance TA, Jang et al. (2024) have proposed *SLERP* (Spherical Linear Interpolation), a merging method that preserves certain geometric properties of the task vectors in order to mitigate interference problems between tasks. Also motivated by this interference issue, a wide range of other methods follow a two-step process: first, task vectors are modified using techniques such as masking or singular value decomposition (Wang et al., 2024; Yadav et al., 2023; Stewart, 1993); second, the preprocessed task vectors are interpolated to produce the merged model. Examples of such methods are *TIES* (Yadav et al., 2023) or *DARE* (Yu et al., 2024). Finally, Yang et al. (2024c) proposed *AdaMerging*, where the novelty lies in performing model merging as a data-driven optimization process that learns the optimal way to combine each model's parameters. In practice, it is based on the TA framework, and data from the different tasks are used to maximize the entropy of the predicted tokens without requiring associated references, trying to increase the confidence of the model in its predictions (independently of the fact that these predictions are correct or not). Like AdaMerging, our method employs reference-free, data-driven optimization using task arithmetic. However, whereas AdaMerging disregards the alignment with original fine-tuned models being merged, the optimization criterion in our approach aims to minimize a divergence between these models and the merged model.

3 Proposed Method: DivMerge

A key challenge in the context of model merging is the absence of an explicit target distribution to drive the merging process. Unlike standard supervised learning, where models are trained to minimize a divergence with respect to a known target distribution derived from reference data, model merging typically lacks such direct supervision. Since the fine-tuned models for each task t have been optimized to perform well on their respective data distributions, it can be reasonably assumed that their output distributions approximate the true, but unknown, target distributions—at least in terms of divergence. While this assumption may not be perfect, it is often the most practical option available in the absence of ground-truth targets. Building upon this idea, our DivMerge method proposes to use the output distributions of the fine-tuned models as surrogate targets to build an optimal merging function. After introducing the prerequisite notions, we formally define this method in this section.

3.1 Background Notions

Language model. A language model with parameters θ is denoted as the conditional probability distribution $M(Y|X\,;\theta)$, where Y and X are random variables on the outputs and inputs, respectively. Given the focus on autoregressive models in this work, X and Y are assumed to share the same space, that is all sequences from the Kleene closure on model's vocabulary. For short, applying M on a sample sequence $\mathbf{x} \sim X$ will be denoted as $M(\mathbf{x};\theta)$ and will correspond to probability measure over sequences. To simplify equations in the remainder, the model and its distribution are sometimes denoted simply by their weights, *i.e.* $M \equiv \theta$.

Tasks. A task t is modelled here as a probability measure $\mathbb{P}_{X_t,Y_t}$ on a product space, following the formalism of Baxter (2000). A model fine-tuned on task t is denoted with its parameters θ_t, and the distribution with $M_t(\cdot|\cdot) \triangleq M(\cdot|\cdot\,;\theta_t)$. Given a task $\mathbb{P}_{X_t,Y_t}$ we will denote the support of the task as $\mathcal{S}_{X_t}$, and roughly define it as $\mathcal{S}_{X_t} \triangleq \{x \in \mathcal{X} \mid \mathbb{P}_{X_t}(x) > 0\}$.

Task vectors. Based on (Ilharco et al., 2023), a task vector τ_t for a task t is defined as the shift between the parameters of a pre-trained model $\theta_0 \in \mathbb{R}^d$ (d being the number of parameters, *i.e.* the size of the model) and those of the same model after fine-tuning on task t, $\theta_t \in \mathbb{R}^d$, that is:

$$\tau_t \triangleq \theta_t - \theta_0 \in \mathbb{R}^d\ . \tag{1}$$

Model merging. In its most commonly adopted paradigm, given a set of tasks $\mathcal{T}$, model merging consists in combining task vectors $\mathbf{T} \triangleq \{\tau_t \,|\, t \in \mathcal{T}\}$ originating from a unique pre-trained model θ_0. This is further denoted as the following merging function f_Γ:

$$f_\Gamma(\theta_0, \mathbf{T}) \in \mathbb{R}^d, \tag{2}$$

where θ_0 are the pre-trained model parameters and Γ is the set of parameters for the merging method.

Task arithmetic. Introduced in (Ilharco et al., 2023; Ortiz-Jimenez et al., 2023), Task Arithmetic (TA) defines model merging as linear interpolation on task vectors, *i.e.* formally:

$$\mathrm{TA}_\Gamma(\theta_0, \mathbf{T}) = \theta_0 + \sum_{t\in\mathcal{T}} \Gamma_t \times \tau_t, \tag{3}$$

where Γ_t are real-valued (possibly negative) parameters of the merging method, referred in the following as the merging coefficients. Estimating Γ can be challenging, computationally intensive, and strongly depends on the user-defined objective of the model merging, like multi-task learning, modularity, task unlearning, *etc*. In this work, our objective is multi-task learning: the merged model should be able to perform all tasks for which the individual component models were fine-tuned. We will see in Sec. 4 that the main property to address in order to satisfy this objective is weight disentanglement (Ortiz-Jimenez et al., 2023).

3.2 Our Method

The key idea in our method is to automatically estimate the merging coefficients Γ by using the output distributions of task-specific fine-tuned models as surrogate targets. This allows us to define a divergence-based objective function to determine the optimal value Γ^*. More formally, given a divergence D between probability measures, π a probability distribution on $\mathcal{X}$, the divergence between language models M_1 and M_2 can be defined as:

$$\mathrm{D}_\pi(M_1 \| M_2) \triangleq \mathbb{E}_{X\sim\pi}\big[\mathrm{D}(M_1(\cdot|X) \,\|\, M_2(\cdot|X))\big] .$$

Then, given a set of fine-tuned models $\{\theta_t \,|\, t \in \mathcal{T}\}$ from a pre-trained model θ_0, and a merging function f, the optimization problem to determine Γ^* is defined as:

$$\Gamma^* \triangleq \arg\min_{\Gamma} \sum_{t\in\mathcal{T}} \mathrm{D}_{\mathbb{P}_{X_t}}(\theta_t \| f_\Gamma(\theta_0, \mathbf{T})). \quad (4)$$

Hence, our approach does not require labelled data for the tasks $t \in \mathcal{T}$, but only a set of samples from the input distributions $\{\mathbb{P}_{X_t}\}_{t\in\mathcal{T}}$ is required. As originally presented in (Hinton et al., 2015; Tian et al., 2019) and recently reframed in (Formont et al., 2025), Eq. 4 can be seen as a model distillation problem: we try to distil knowledge from fine-tuned model in the merged model.

Backbone merging method. In practice, DivMerge relies on the TA merging approach. We select it because, as explained in App. F, most current model merging approaches are based on TA. However, our theoretical framework is not limited to TA and extension to other general methods could be explored in the future. Relying on Eq. 3, the effective optimization is thus:

$$\Gamma^* \triangleq \arg\min_{\Gamma} \sum_{t\in\mathcal{T}} \mathrm{D}_{\mathbb{P}_{X_t}}(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})). \quad (5)$$

In a geometrical point of view, this objective function can be thought as finding Γ such that the merged model is the **centroid** of the fine-tuned models (with respect to the divergence D) (Nielsen, 2020). The idea is simple: a multi-task model must be close to models specialized on the different tasks. This geometric point of view has first been introduced by Csiszár (1975), and is now refereed to the framework of information geometry (Amari and Nagaoka, 2000; Amari, 2006). Moreover, this information geometric point of view is also at the root of merging methods such as Fisher merging (Matena and Raffel, 2022) on which the merging operation can be seen as the solution of a second order Taylor approximation of the objective function defined in Eq. 4.

Optimization algorithm. The optimization problem described in Eq. 5 is implemented as illustrated in Figure 1. Starting from uniform values for the coefficients Γ, the procedure iterates over each task t as follows: an input token sequence $\mathbf{x}_t \sim \mathbb{P}_{X_t}$ is sampled, and a completion $\hat{\mathbf{y}}_t \sim M_t(.|\mathbf{x}_t)$ is generated, along with the vectors of probabilities $\hat{\mathbf{p}}_t$ over all tokens of the model's vocabulary for each position in $\hat{\mathbf{y}}_t$. The concatenated sequence $\mathbf{x}_t \oplus \hat{\mathbf{y}}_t$ is then fed to the merged model under construction, yielding a sequence of probability vectors $\tilde{\mathbf{p}}_t$ over $\hat{\mathbf{y}}_t$. The divergence D is computed between $\hat{\mathbf{p}}_t$ and $\tilde{\mathbf{p}}_t$, and averaged over multiple samples of $\mathbf{x}_t$ to obtain D_{X_t}. This process is repeated for all tasks, and the resulting divergences are aggregated to form a loss function. The gradient of this loss is used to update the coefficients Γ. In practice, the final coefficients Γ^* are obtained after a fixed number of update steps. The full algorithm is provided in App. A, and an analysis of the required number of updates is presented in Sec. 5.3.

Divergences. Two divergences are considered in our work: Kullback and Leibler (KL) and Jensen-Shannon (JS) (Kullback and Leibler, 1951; Wong and You, 1985). Given two sequences of probabilities $\mathbf{p}$ and $\mathbf{q}$ for a same sequence of n tokens from the vocabulary $\mathcal{V}$, we define approximations of the KL and JS between models as follows:

$$\mathrm{KL}(\mathbf{p}\|\mathbf{q}) \triangleq \frac{1}{n} \sum_{1\leqslant i\leqslant n} \sum_{w\in\mathcal{V}} p_w^i \log \frac{p_w^i}{q_w^i} , \quad (6)$$

and:

$$\mathrm{JS}(\mathbf{p},\mathbf{q}) \triangleq \tfrac{1}{2}\left(\mathrm{KL}\left(\mathbf{p} \,\middle\|\, \tfrac{\mathbf{p}+\mathbf{q}}{2}\right) + \mathrm{KL}\left(\mathbf{q} \,\middle\|\, \tfrac{\mathbf{p}+\mathbf{q}}{2}\right)\right) . \quad (7)$$

KL and JS divergences possess properties that allows for the derivation of theoretical guarantees on the objective function defined in Eq. 4, as discussed in the next section.

Task-wise and layer-wise TA. Following AdaMerging (Yadav et al., 2024), two variants can be considered for the merging coefficients. The one described in Eq. 3 is called *task-wise* since one coefficient Γ_t is given for each task t. Considering (deep neural) language models as stacks of layers, an alternative—called *layer-wise*—is to associate

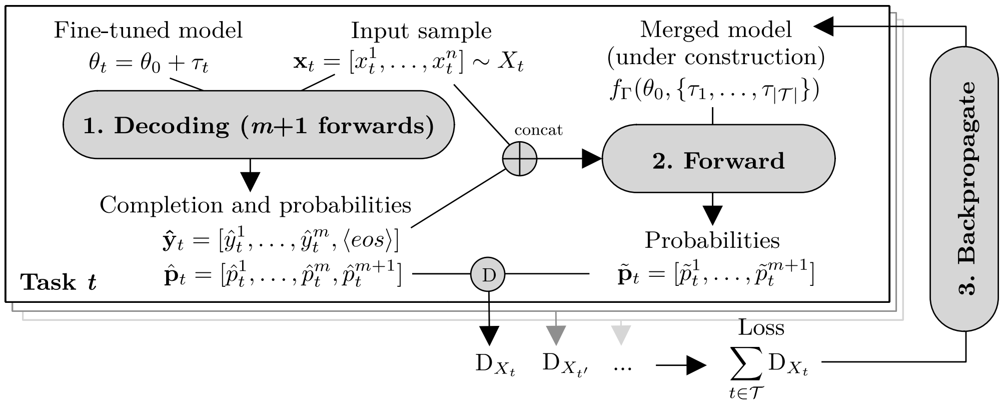

Fine-tuned model
$\theta_t = \theta_0 + \tau_t$
Input sample
$\mathbf{x}_t = [x_t^1, \ldots, x_t^n] \sim X_t$
Merged model (under construction)
$f_\Gamma(\theta_0, \{\tau_1, \ldots, \tau_{|\mathcal{T}|}\})$
1. Decoding (m+1 forwards)
concat
2. Forward
Completion and probabilities
$\hat{\mathbf{y}}_t = [\hat{y}_t^1, \ldots, \hat{y}_t^m, \langle eos \rangle]$
$\hat{\mathbf{p}}_t = [\hat{p}_t^1, \ldots, \hat{p}_t^m, \hat{p}_t^{m+1}]$
Probabilities
$\tilde{\mathbf{p}}_t = [\tilde{p}_t^1, \ldots, \tilde{p}_t^{m+1}]$
D
Task t
3. Backpropagate
D_{X_t}
$\mathrm{D}_{X_{t'}}$
...
Loss
$\sum_{t \in \mathcal{T}} \mathrm{D}_{X_t}$

Figure 1: Illustration of our divergence-based model merging method as an iterative optimization process.

a parameter Γ_l^t for each layer l for task t. This results in $|\mathcal{T}| \times L$ merging coefficients, where L is the number of layers.

4 Theoretical Analysis

This section presents a set of theoretical guarantees or interpretations that can be derived from our approach. We show that our method has direct links between weight disentanglement as defined by Ortiz-Jimenez et al. (2023), and we will show that our method directly connects with the objective of standard multi-task.

4.1 Weight Disentanglement

From (Ortiz-Jimenez et al., 2023, Property 3), weight disentanglement is defined as follows:

Definition 1 (Weight Disentanglement)**.** Let $M(.;\theta)$ be a model parametrized by θ. Consider a set of tasks, $\{(X_t, Y_t),\ t \in \mathcal{T}\}$, and their corresponding task vectors $\{\tau_t,\ t \in \mathcal{T}\}$ relatively to the model M. If tasks have non-overlapping supports, *i.e.* $\mathcal{S}_{X_i} \cap \mathcal{S}_{X_j} = \emptyset$ for all $i \neq j$, we say that a merging method f satisfies weight disentanglement iff:

$$M\left(\mathbf{x}; f_\Gamma\left(\theta_0, \{\tau_t\}_{t\in\mathcal{T}}\right)\right) = \begin{cases} M(\mathbf{x}; \theta_0 + \tau_i) & \text{if } \mathbf{x} \in \mathcal{S}_{X_i}, \\ M(\mathbf{x}; \theta_0) & \text{if } \mathbf{x} \notin \bigcup_{t=1}^{T} \mathcal{S}_{X_t}. \end{cases} \quad (8)$$

In other words, a merging method satisfies Definition 1 if adding τ_t does not affect model's output outside the corresponding task support $\mathcal{S}_{X_t}$. A merging method satisfying weight disentanglement assures that the performance on all the merged tasks will be preserved, thus assuring good behaviour in terms of multi-tasking. In Proposition 1, we propose a direct link between DivMerge's objective function (*cf.* Eq. 4) and this property of weight disentanglement, with the proof given in Sec. B.2.

Proposition 1. *For either* $\mathrm{D} = \mathrm{KL}$ *or* $\mathrm{D} = \mathrm{JS}$*, the objective function defined in Eq. 4 has a minimum value of* 0 *iff the merging method satisfies Definition 1 around* θ_0 *on the merged tasks.*

The optimization problem in Eq. 4 admits a minimum value of 0 iff there exists a choice of merging parameters Γ respecting the weight disentanglement property, giving added value to our method.

4.2 Links with Standard Multi-Task Learning

As stated earlier, the standard approach for multi-task learning mainly consists in merging datasets from different tasks, and then training a model on this mixture using Cross-Entropy (CE) loss. This method is, in most cases, the state of the art in multi-task learning and has interesting theoretical guarantees. Our optimization approach attempts to align the merged model with the original fine-tuned ones, we aim to quantify how well this serves as a proxy for directly optimizing the performance of the merged model, by quantifying how far our merged model will be from its standard multi-task counterpart. In Proposition 2, we propose a first link between this classical multi-task framework in the case of $\mathrm{D} = \mathrm{KL}$.

Proposition 2. *For* $\mathrm{D} = \mathrm{KL}$*, the optimization problem defined in Eq. 4 is a Moment projection* (M*-projection) approximation of the standard multi-task objective when tasks are merged before training.*

We give the proof of Proposition 2 in Sec. B.3. The proof is based on the classical decomposition of the CE loss function (Boudiaf et al., 2020), which is the main loss function used in multi-task learning. This decomposition gives a direct link between KL divergence and CE. Moreover, following the experimental protocol described in Sec. 5, an illustration of this theoretical result is provided in App. C by empirically showing the correlation between KL and JS divergences with the performance on the tasks. Finally, in Proposition 3 we directly quantify the distance between the model obtained with our model approximation method and the standard multi-task model.

Proposition 3. *Let M^* denote the standard multi-task model, and $M^\dagger$ the multi-task model obtained from fine-tuned models. Let the probability distribution $\mu \triangleq \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathbb{P}_{X_t}$, then,*

$$\mathrm{KL}_\mu(M^* \| M^\dagger) \leqslant \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \underbrace{\mathrm{KL}_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| M_t)}_{\textit{Loss of fine-tuned model } M_t} .$$

The proof is provided in Sec. B.4, where we also demonstrate that μ is indeed a natural probability distribution to consider when working with multi-task models. This final theoretical result guarantees that DivMerge's objective function in Eq. 4 will produce a model close to the standard multi-task model M^*, provided that the fine-tuned models are sufficiently good.

5 Experiments

This section first introduces the experimental setup, then compares the results of DivMerge with those of state-of-the-art methods, and finally analyses the stability of its optimization process.

5.1 Experimental Setup

Models and datasets. As summarized in Table 1, our experiments seek to validate our method from different scenarios varying according to different aspects: classification tasks only or mixed classification and generative tasks; decoder-only or encoder-decoder architectures; model fine-tuned by us or available off-the-shelf. Hence, for classification tasks, the base model θ_0 is `Qwen2.5-0.5B` (Yang et al., 2024a) and the considered tasks are, as is common in recent studies, 7 tasks from the GLUE benchmark (Wang et al., 2018) for which we fine-tuned the base model. Each model is produced with the same amount of data *i.e.* 6000 for the training and 500 for the validation except for the MRPC dataset for which the train dataset is limited to 3000 instances. This is due to the fact that task vectors can have properties that are highly dependent of the number of training data. Since our method targets autoregressive models, classification is performed as a completion task (see Table 2 for an example of input and output). In the mixed type scenario, the base model is `T5-Base` (Raffel et al., 2020) and the fine-tuned models are as available on Hugging Face[1] for 4 tasks: IMDB (polarity classification on movie reviews, (Maas et al., 2011), QASC (MCQA, classification, (Khot et al., 2020)), SQuAD 2.0 (MRQA, generation, (Rajpurkar et al., 2016), and CommonGen (concept-to-text generation, (Lin et al., 2020)). For both scenarios, fine-tunings are on all the model's parameters (full fine-tuning as opposed to low-rank adaptation). To estimate the divergences D_{X_t} while optimizing the Γ coefficients, 500 samples are taken from the validation set of each task t.

Evaluation metrics. To fairly compare a merged model with several fine-tuned models across tasks

[1] `https://huggingface.co/mrm8488`

Type	Tasks/Datasets	Base model	Fine-tuned models
Classif. only	CoLA, SST-2, QQP, QNLI, MNLI, RTE, MRPC	`Qwen2.5-0.5B-Instruct`	Fine-tuned by us
Mixed (Cl.+Gen.)	IMDB, SQuAD 2.0, QASC, CommonGen	`T5-base`	HuggingFace checkpoints

Table 1: Overview of the experimental setups.

Description		Data
Input	(prompt)	`Is the following sentence linguistically acceptable or unacceptable in English?`
	(data)	`Sodium is a little too peppy for me to want to try mixing and water in a teacup.`
	(labels)	`Answer with acceptable or unacceptable. Answer:`
Output (answer)		`unacceptable`$\langle eos\rangle$

Table 2: Data example for the fine-tuning on the GLUE Benchmark (CoLA dataset). Completion only Fine-tuning (SFT) after the pattern "`Answer:`".

	CoLA	SST-2	QQP	QNLI	MNLI	RTE	MRPC
Accuracy	82.2	92.8	84.6	86.4	78.0	75.8	85.0

(a) Classif. / Fine-tuned from Qwen2.5-0.5B-Instruct

	IMDB	QASC	SQuAD 2.0	CommonGen
Accuracy	96.0	95.5	–	–
ROUGE-1	–	–	0.434	0.327

(b) Mixed / Fine-tuned from T5-base

Table 3: Performances of the fine-tuned models for the classification and mixed tasks.

that use different evaluation metrics, we introduce the Average Normalized Performance (ANP). Given a performance metric $\mathrm{PERF}(M; t)$ of a model M for each task t, the performance of the merged model is divided by the performance of the corresponding fine-tuned model, and these ratios are averaged over all tasks, *i.e.*

$$\mathrm{ANP} \triangleq \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{\mathrm{PERF}(f_\Gamma(\theta_0, \mathbf{T}); t)}{\mathrm{PERF}(\theta_t; t)}, \quad (9)$$

where $f_\Gamma(\theta_0, \mathbf{T})$ represents the merged model (whatever the method) and θ_t the fine-tuned model for task t. The metric PERF corresponds to accuracy for classification tasks and ROUGE-1 (Lin, 2004) for generative tasks. When multiple merging experiments are performed, ANP is computed for each of them and the average is reported. This metric allows for multitask evaluation relative to the baselines, and reaches 1 (or 100 %) iff merging is perfectly disentangled (Definition 1). To enable this interpretation in terms of absolute performance, Table 3 reports accuracy and ROUGE-1 scores of the fine-tuned models for each task in the classification and mixed scenarios.

Baselines. Our method is compared to popular methods from the literature: model averaging (Wortsman et al., 2022), Multi-SLERP (Goddard et al., 2024), TIES (Yadav et al., 2023), AdaMerging (Yang et al., 2024c) and MetaGPT (Zhou et al., 2024)[2]. For TIES, we followed the recommended recipe from (Yadav et al., 2023). For Multi-SLERP, the weights associated to each tasks were set to $\frac{1}{n}$. For optimization-based methods (*i.e.* AdaMerging and ours), task-wise and layer-wise variants are experimented, and we used the same hyper-parameters across all experiments, with a batch size of $4 \times |\mathcal{T}|$ for each iteration. For AdaMerging we checked that hyper-parameters provided good results for a faire comparison. More training details are provided in App. E.

5.2 Results

This section reports our model merging results, first when merging pairs of task-specific models, then when considering more models.

Pairs of tasks. For the classification and mixed scenarios, all pairs of tasks are considered and

[2]MetaGPT method was only implemented on the classification set-up.

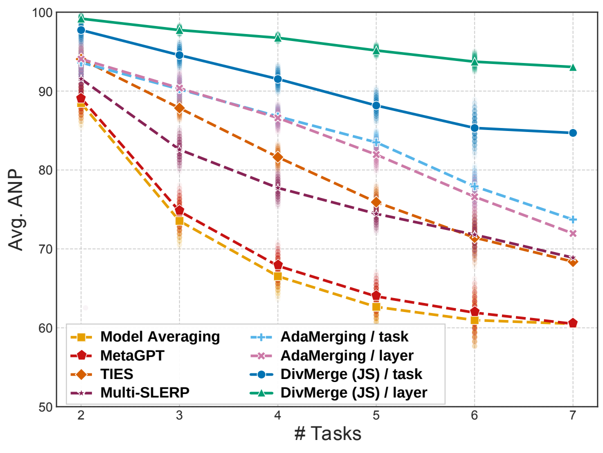
Method	Level	Classification	Mixed (Cl.+Gen.)
Model Averag.	*task*	88.48 (± 3.17)	94.38 (± 2.6)
MetaGPT	*task*	89.12 (± 2.96)	-
Multi-SLERP	*task*	91.54 (± 2.98)	76.39 (± 21.04)
TIES	*task*	94.06 (± 1.81)	95.53 (± 4.44)
AdaMerging	*task*	93.62 (± 2.53)	93.42 (± 10.08)
	layer	94.06 (± 2.95)	83.20 (± 9.94)
DivMerge (KL)	*task*	97.68 (± 1.94)	93.97 (± 3.46)
	layer	99.16 (± 0.50)	97.50 (± 1.73)
DivMerge (JS)	*task*	<u>97.73</u> (± 2.01)	<u>97.29</u> (± 1.94)
	layer	**99.18** (± 0.51)	**98.93** (± 1.05)

Table 4: Average ANP (%) for various methods on classification and mixed task pairs. **Best results**, <u>second best</u>, (standard deviations).

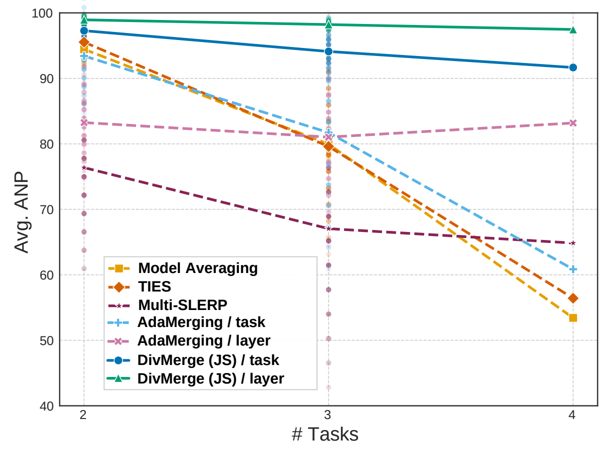
merging is performed: $\binom{7}{2} = 21$ merging experiments are run given the 7 classification tasks ; $\binom{4}{2} = 6$ for the 4 mixed tasks. Then, ANP scores from Eq. 9 are computed for all the resulting models and averaged for each method and scenario. The results are presented in Table 4. We can clearly observe that our method achieves the best average performance. Furthermore, our method is more stable than the others as indicated by the low standard deviations. Then, we can see that the layer-wise variant slightly outperforms the task-level one, which is expected since the former has a greater number of merging coefficients (one for each layer of each task), providing a higher degree of granularity. Finally, the JS divergence seems to provide better results than KL. In the further sections, only results for JS are reported but exhaustive results can be found in App. D.

More tasks. A major limitation of existing merging methods, as noted in (Yadav et al., 2023), is their lack of robustness when the number of models to be merged increases. To assess this robustness, all possible combinations of tasks are tested for each scenario: all combinations of 2 up to 7 tasks are merged together for classification ; 2 up to 4 for generation. Figure 2 shows the average ANP scores across all possible merging experiments as a function of the number of tasks, for both the classification (Figure 2a) and mixed (Figure 2b) setups[3]. First, we observe that regardless of the used method, increasing the number of tasks generally leads to a degradation in the average ANP, which is consistent with the fact that task interfer-

[3]Detailed results for each task can be found in Sec. D.2.



(a) Classification



(b) Mixed

Figure 2: Evolution of the average ANP metric as a function of the number of merged tasks. For each number of tasks k on the x-axis, several merging experiments were conducted ($\binom{n}{k}$ in total), and we report the 95% confidence interval.

Method	Level	2 Tasks	3 Tasks
Model Averaging	*task*	±2.60	±2.96
Multi-SLERP	*task*	±21.04	±27.97
TIES	*task*	±4.44	±19.01
AdaMerging	*task*	±10.08	±23.94
	layer	±9.94	±19.03
DivMerge (JS)	*task*	±1.94	±5.37
	layer	±1.05	±1.09

Table 5: Confidence interval margins for different merging methods on all combinations of 2 or 3 tasks for the mixed setup.

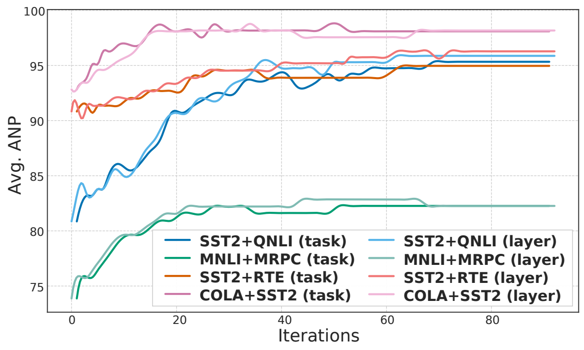


Figure 3: Evolution of the ANP metric along the optimization iteration step when merging pairs of classification tasks using DivMerge, task-wise or layer-wise.

ence becomes more apparent. We can also observe that, regardless of the number of merged tasks, our method (both task-wise and layer-wise) consistently provides better results, with curves that remain higher throughout the graph. Moreover, the drop in performance as the number of tasks increases is less pronounced for our method, illustrating its robustness with respect to the number of tasks. Figure 2 also reports confidence intervals (CI) across experiments for various tasks with a fixed number of tasks to merge. Our method demonstrates strong stability, both in classification and mixed tasks. Since some state-of-the-art methods show large CIs for the mixed tasks, exact CI margins are reported in Table 5. Notably, our method exhibits greater stability compared to AdaMerging, another optimization-based approach. However, our method is far from the standard multi-task approach which has an ANP of 0.98 on the classification set-up.

5.3 Method Behaviour Analysis

This section further investigates the training dynamics of the optimization process in DivMerge. We focus here exclusively on classification tasks.

Performance convergence. Figure 3 shows the evolution of the ANP metric across training iterations when merging pairs of tasks, either task-wise or layer-wise. In all cases, the merging process converges smoothly without signs of over-fitting (*i.e.* a sudden drop in performance), indicating that the proposed methods are effective and stable, consistently merging task-specific representations over training iterations.

Dataset size influence. Another factor in the optimization process is the number of samples derived from $\{X_t\}_{t\in\mathcal{T}}$, based on which divergences are estimated (see Sec. 3). Figure 4 plots the evolution of the ANP metric for DivMerge as a function of the amount of samples from each dataset X_t used by our method. This evolution is compared to ANP scores of the state-of-the-art methods. The merged pairs of tasks are (CoLA, SST-2), (QNLI, MNLI),

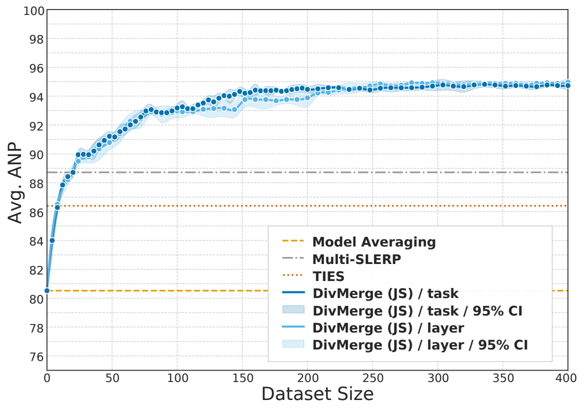


Figure 4: Impact of the number of samples from each dataset X_t on averaged ANP in DivMerge, compared to non-optimization-based baselines. Average ANP is computed across on the task pairs (CoLA, SST-2), (QNLI, MNLI), and (RTE, MRPC).

and (RTE, MRPC). ANR scores are averaged on these 3 setups. We can see that our method outperforms the others with as few as 25 samples, which corresponds to only 0.4% of the training corpus used for fine-tuning the merged models and 5% of the validation dataset.

6 Conclusion

In this work, we introduced DivMerge: a new, data-driven but reference-free, merging method which consists in finding the probabilistic centroid of fine-tuned models in order to produce a multi-task model. After analysing theoretical properties of our method, showing links with the concept of weight disentanglement and the usual multi-task learning approach, we empirically demonstrate that our method consistently outperforms the state-of-the-art methods on various experimental setups. This is verified when merging pairs of tasks and becomes even more substantial when more tasks are involved, confirming that our method better handles interference issues. Finally, we show that our method has high training stability and requires a relatively small amount of data to work.

Limitations

Despite positive and solid results, several limitations can be highlighted from our method.

Other fine-tuning methods. Our method has been extensively tested when the task-specific models were constructed using full fine-tuning. In this setup, the task vectors are sparse, and consequently, the interference problem is more limited. However, in fine-tuning based on low-rank adaptation (LoRA) (Hu et al., 2022), task vectors (i.e., LoRA matrices) influence the task arithmetic paradigm and can cause significant performance degradation when merging. A limitation of our work is that we have not experimented within this constrained setup.

Dataset influence. Our method assumes that, for each task t, we have access to a sample from the distribution $\mathbb{P}_{X_t}$ corresponding to the input data for task t. However, in some scenarios, we may not have access to such a distribution, but only to an approximation of it ($\mathbb{P}_{\tilde{X}_t}$). We have not addressed this case here. Nonetheless, we propose an initial theoretical analysis of this scenario in Sec. B.5. We believe that exploring this direction will lead to more robust results for model merging.

Ethical Considerations

While not specific to our approach, model merging raises the question of merging features or properties of models that were initially not designed to be used together (for instance, merging a model that mimics a specific person with a model specialized to generated toxic text). This kind of usage is both to the responsibility of people willing to merge the models, and usage licences from people providing elementary models that could be merged.

Acknowledgments

Generative AI tools, including OpenAI's GPT-4.1 models and Anthropic's Claude 2.5 Sonnet, were used in the writing process of this paper. These tools were solely employed to improve grammar or refine style and phrasing.

References

Shun-ichi Amari. 2006. Differential geometry of statistical inference. In *Proc. USSR-Japan Symposium*.

Shun-Ichi Amari. 2009. a-divergence is unique, belonging to both f-divergence and bregman divergence classes. *IEEE Transactions on Information Theory*, 55(11):4925–4931.

Shun-ichi Amari and Hiroshi Nagaoka. 2000. *Methods of information geometry*. American Mathematical Soc.

Dana H Ballard. 1987. Modular learning in neural networks. In *Proc. of the National conference on Artificial intelligence*.

Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. 2005. Clustering with bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749.

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198.

Shai Ben-David and Reba Schuller. 2003. Exploiting task relatedness for multiple task learning. In *Proc. Annual Conference on Learning Theory*.

Yaiza Bermudez, Gaetan Bisson, Iñaki Esnaola, and Samir Perlaza. 2025. Démonstrations des théorèmes folkloriques sur la dérivée de radon-nikodym. In *GRETSI'25-XXXe Colloque Francophone de Traitement du Signal et des Images*.

Yochai Blau and Tomer Michaeli. 2018. The perception-distortion tradeoff. In *Proc. of the IEEE conference on computer vision and pattern recognition*.

Yochai Blau and Tomer Michaeli. 2019. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *Proc. ICML*.

Malik Boudiaf, Jérôme Rony, Imtiaz Masud Ziko, Eric Granger, Marco Pedersoli, Pablo Piantanida, and Ismail Ben Ayed. 2020. A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses. In *Proc. ECCV*.

Lev M Bregman. 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. In *Proc. NEURIPS*.

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

Alexandra Chronopoulou, Jonas Pfeiffer, Joshua Maynez, Xinyi Wang, Sebastian Ruder, and Priyanka Agrawal. 2024. Language and task arithmetic with parameter-efficient layers for zero-shot summarization. In *Proc. of the Fourth Workshop on Multilingual Representation Learning*.

T. M. Cover and J. A. Thomas. 1991. *Elements of Information Theory*. John Wiley & Sons, Inc.

H. Crauel. 2002. *Random Probability Measures on Polish Spaces*. Taylor & Francis.

Imre Csiszár. 1975. I-divergence geometry of probability distributions and minimization problems. *The annals of probability*, pages 146–158.

Nico Daheim, Thomas Möllenhoff, Edoardo Maria Ponti, Iryna Gurevych, and Mohammad Emtiyaz Khan. 2023. Model merging by uncertainty-based gradient matching. *arXiv preprint arXiv:2310.12808*.

Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Multiple Classifier Systems*.

Olivier Ferret. 2025. Projeter pour mieux fusionner: une histoire de bandit et de lit. In *Proc. TALN*.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently Identifying Task Groupings for Multi-Task Learning. In *Proc. NEURIPS*.

Philippe Formont, Maxime Darrin, Banafsheh Karimian, Jackie CK Cheung, Eric Granger, Ismail Ben Ayed, Mohammadhadi Shateri, and Pablo Piantanida. 2025. Learning task-agnostic representations through multi-teacher distillation. *arXiv preprint arXiv:2510.18680*.

Loïc Fosse, Frederic Bechet, Benoit Favre, Géraldine Damnati, Gwénolé Lecorvé, Maxime Darrin, Philippe Formont, and Pablo Piantanida. 2025. Statistical deficiency for task inclusion estimation. In *Proc. ACL*.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*.

Karsten Grove and Hermann Karcher. 1973. How to conjugate c 1-close group actions. *Mathematische Zeitschrift*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. In *Proc. ICLR*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *Proc. ICLR*.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*.

Young Kyun Jang, Dat Huynh, Ashish Shah, Wen-Kai Chen, and Ser-Nam Lim. 2024. Spherical linear interpolation and text-anchoring for zero-shot composed image retrieval. In *Proc. ECCV*.

Wooseong Jeong and Kuk-Jin Yoon. 2025. Selective task group updates for multi-task optimization. *arXiv preprint arXiv:2502.11986*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models. In *Proc. ICLR*.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition. *arXiv:1910.11473v2*.

Hyoseo Kim, Dongyoon Han, and Junsuk Choe. 2024. Negmerge: Consensual weight negation for strong machine unlearning. *arXiv preprint arXiv:2410.05583*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Kevin Kuo, Amrith Setlur, Kartik Srinivas, Aditi Raghunathan, and Virginia Smith. 2025. Exact unlearning of finetuning data via model merging at scale. *arXiv preprint arXiv:2504.04626*.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Proc. Findings of EMNLP*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proc. of Text Summarization Branches Out*, Barcelona, Spain.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proc. ACL*.

Michael S Matena and Colin A Raffel. 2022. Merging models with fisher-weighted averaging. In *Proc. NEURIPS*.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. 2016. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32.

Frank Nielsen. 2020. On a generalization of the jensen–shannon divergence and the jensen–shannon centroid. *Entropy*, 22(2):221.

Frank Nielsen and Richard Nock. 2007. On the centroids of symmetrized bregman divergences. *arXiv preprint arXiv:0711.3242*.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Proc. NEURIPS*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proc. EACL*.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Maria Ponti. 2023. Modular deep learning. *arXiv preprint arXiv:2302.11529*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proc. EMNLP*.

Henry Scheffé. 1947. A Useful Convergence Theorem for Probability Distributions. *The Annals of Mathematical Statistics*.

Zhang Shengyu, Dong Linfeng, Li Xiaoya, Zhang Sen, Sun Xiaofei, Wang Shuhe, Li Jiwei, Runyi Hu, Zhang Tianwei, Fei Wu, and 1 others. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *Proc. ICLR*.

Charles Stein. 1956. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proc. of the third Berkeley symposium on mathematical statistics and probability*.

Gilbert W Stewart. 1993. On the early history of the singular value decomposition. *SIAM review*.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Proc. NEURIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proc. EMNLP*.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and 1 others. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Andrew KC Wong and Manlai You. 1985. Entropy and distance of random graphs with application to structural pattern recognition. *IEEE transactions on pattern analysis and machine intelligence*, (5):599–609.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *Proc. ICML*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. In *Proc. NEURIPS*.

Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. 2024. What matters for model merging at scale? *arXiv preprint arXiv:2410.03617*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

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

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024c. Adamerging: Adaptive model merging for multi-task learning. In *Proc. ICLR*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Proc. ICML*.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Proc. NEURIPS*.

Yuefeng Zhang. 2023. A rate-distortion-classification approach for lossy image compression. *Digital Signal Processing*, 141:104163.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. 2024. Metagpt: Merging large language models using model exclusive task arithmetic. In *Proc. EMNLP*.

A Optimization algorithm

Algorithm 1 Model Merging via Divergence-Based Optimization. In this procedure, for a sequence y and a model M, $\mathrm{Logits}(y, M)$ denotes the logits (soft probs) of the sequence y given by the model M. From a technical point of view, this is only a forward pass of y through M.

```
Require: :
  $\{X_t \mid t \in \mathcal{T}\}$ : Sets of input samples for each task
  $f_\Gamma$ : Merging method
  $\theta_0$ : Pretrained model
  $\{\theta_t \,|\, t \in \mathcal{T}\}$ : Fine-tuned models
  // Initialize coefficients
  $\Gamma_i \leftarrow \frac{1}{|\mathcal{T}|} \quad \forall i$
  // Build task vectors
  for $t \in \mathcal{T}$ do
      $\tau_t \leftarrow \theta_t - \theta_0$
  end for
  // Get logits of the data by generation
  for $t \in \mathcal{T}$ do
      for $\mathbf{x} \sim X_t$ do
          $\hat{\mathbf{y}}_t^{\mathbf{x}} \leftarrow [y^1, \ldots, y^m, \langle eos \rangle] \sim M(\mathbf{x}; \theta_t)$ // Decoding (m+1 forward steps)
          $\hat{\boldsymbol{\ell}}_t \leftarrow \mathrm{Logits}(\hat{\mathbf{y}}_t^x, M(\mathbf{x}, \theta_t))$ // Logits (directly computed while decoding)
          $\hat{\mathbf{p}}_t \leftarrow \mathrm{softmax}(\hat{\boldsymbol{\ell}}_t)$ // Probability vectors
      end for
  end for
  // Compute coefficients
  for each epoch do
      for $B \sim \cup_t X_t$ do // Batch sampling
          for $\mathbf{x} \in B$ do
              $\tilde{\boldsymbol{\ell}} \leftarrow \mathrm{Logits}(\hat{\mathbf{y}}_t^{\mathbf{x}}, M(\mathbf{x}, f_\Gamma(\theta_0, \{\theta_t \,|\, t \in \mathcal{T}\})))$ // Logits: 1 forward step
              $\tilde{\mathbf{p}}_t \leftarrow \mathrm{softmax}(\tilde{\boldsymbol{\ell}}_t)$ // Probability vectors
          end for
          $L_\Gamma \leftarrow 0$
          for each $t \in \mathcal{T}$ do
              for $\mathbf{x} \in b$ do
                  // Choose right task
                  // Compute pointwise Loss
                  $L_\Gamma \leftarrow L_\Gamma + \mathrm{D}(\hat{\mathbf{p}}_t \| \tilde{\mathbf{p}}_t)$
              end for
          end for
          $\Gamma \leftarrow \Gamma - \nabla_\Gamma L_\Gamma$ // Gradient update
      end for
  end for
  return $\Gamma$
```

B Theoretical results

B.1 Notations

For the different proofs we introduce some notations. Random variables are denoted by capital letters (*e.g.* X), their spaces by calligraphic letters (*e.g.* $\mathcal{X}$), and elements by lowercase letters (*e.g.* $x \in \mathcal{X}$). In this study, we suppose that every considered space is Borel standard (Crauel, 2002) and if we consider the space $\mathcal{X}$, we denote by $\mathcal{B}(\mathcal{X})$ its Borel σ-algebra. $\mathcal{P}(\mathcal{X})$ is the set of probability measures on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$, and $\mathcal{P}(\mathcal{Y}|\mathcal{X})$ the set of conditional probabilities on $\mathcal{Y}$ given $\mathcal{X}$. For $X \in \mathcal{X}$, $\mathbb{P}_X \in \mathcal{P}(\mathcal{X})$ is its law, and $\mathcal{S}_X \subset \mathcal{X}$ its support. A task t is a probability measure $\mathbb{P}_{X_t,Y_t} \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$, following the formalism of (Fosse et al., 2025). For sake of simplicity, we hypothesis that all tasks share the same space $\mathcal{X} \times \mathcal{Y}$, and that $\mathcal{X} = \mathcal{Y}$, which is true in most generative tasks: both inputs and outputs are texts. We note a language model with parameters θ with $M(\cdot|\cdot\,;\theta) \in \mathcal{P}(\mathcal{Y}|\mathcal{X})$. For task t, the specialized model on t is $M(\cdot|\cdot\,;\theta_t)$ or $M_t(\cdot|\cdot)$.

B.2 Proof of Proposition 1

Proof. Since the divergences we use are non-negative, we have the following equivalence:

$$\sum_{t\in\mathcal{T}} \mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) = 0 \Leftrightarrow \forall t \in \mathcal{T},\ \mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) = 0.$$

Moreover, by the properties of the KL and JS divergences, we have:

$$\mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) = 0 \Leftrightarrow \forall x \in \mathcal{S}_{X_t},\ M(x;\theta_t) = M(x; \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})),$$

where the last equality is understood in the sense of equality of measures. By transitivity, we obtain:

$$\sum_{t\in\mathcal{T}} \mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) = 0 \Leftrightarrow \forall t \in \mathcal{T},\ \forall x \in \mathcal{S}_{X_t},\ M(x;\theta_t) = M(x; \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})),$$

which concludes the proof. □

Remark 1. In this demonstration, we stated that this was due thanks to some properties of the KL or JS divergence. However, we have the same result if we use any f-divergence, any divergence than can be expressed as following,

$$\mathrm{D}_f(\mu\|\nu) = \int f\left(\frac{d\mu}{d\nu}\right) d\nu,$$

which is of course the case of the Jensen Shannon and the Kullback ones. In fact this proof is valid for any divergence D which satisfies the following property,

$$\mathrm{D}_f(\mu\|\nu) = 0 \ \Leftrightarrow\ \mu = \nu.$$

B.3 Proof of Proposition 2

Definition 2 (Standard Multi Task Objective)**.** Let $\{(X_t, Y_t) \mid t \in \mathcal{T}\}$ be a set of tasks, $\mathcal{H}$ the cross-entropy loss function, and $M(\cdot;\theta)$ a model parametrized by θ. We define the multi-task loss function as follows:

$$\mathcal{L}_{\mathrm{MT}}(\theta) \triangleq \frac{1}{|\mathcal{T}|} \sum_{t\in\mathcal{T}} \mathcal{H}\left(\mathbb{P}_{Y_t|X_t}, M(\cdot \mid X_t, \theta)\right),$$

Lemma 1 (Boudiaf et al. (2020))**.** *Let $\mathbb{P}_{X_t,Y_t}$ be a task, $\mathcal{H}$ the cross-entropy loss function, and $M(\cdot \mid \cdot, \theta)$ a model parametrized by θ. Then, the following relation holds:*

$$\begin{aligned}\mathcal{H}(\mathbb{P}_{Y_t|X_t}, M_\theta) &= H(Y_t|X_t) + \mathbb{E}_{X_t}\left[\mathrm{KL}(\mathbb{P}_{Y_t|X_t}(\cdot \mid X_t) \| M(\cdot \mid X_t, \theta))\right] \\ &= H(Y_t|X_t) + \mathrm{KL}_{X_t}\left(\mathbb{P}_{Y_t|X_t} \| M_\theta\right)\end{aligned}$$

where H denotes Shannon's entropy.

We now provide the proof of Proposition 2:

Proof. By hypothesis, we have

$$\theta_t = \arg\min_\theta \mathcal{H}(\mathbb{P}_{Y_t|X_t}, M(\cdot \mid X_t, \theta)).$$

By Lemma 1 we have,

$$\theta_t = \arg\min_\theta \mathrm{KL}_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| M(\cdot \mid X_t, \theta)).$$

which can be seen as an extension of the moment projection (M-projection) of Csiszár (1975) of $\mathbb{P}_{Y_t|X_t}$ onto the set $\{M(\cdot \mid \cdot, \theta) \mid \theta \in \mathbb{R}^d\}$. Based on this, we define the M-projection multi-task objective as follows:

$$\mathcal{L}_{\mathrm{MT}}^{\mathrm{M}}(\theta) \triangleq \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathcal{H}\left(M_{\theta_t} \| M_\theta\right).$$

Again, by Lemma 1, we have

$$\arg\min_\theta \mathcal{L}_{\mathrm{MT}}^{\mathrm{M}}(\theta) = \arg\min_\theta \sum_{t \in \mathcal{T}} \mathrm{KL}_{\mathbb{P}_{X_t}}\left(M_{\theta_t} \| M_\theta\right),$$

which concludes the proof. □

Remark 2. From Lemma 1 we can clearly see the causation effect between the KL divergence and the cross-entropy. If KL is low then automatically the cross entropy function will be low enforcing the link between KL and performance.

B.4 Proof of Proposition 3

For the sake of this proof, we need to first introduce the notion of Bregman divergence (Bregman, 1967), between two probability measures μ and ν on the same space. Let F a strictly convex differentiable function. A Bregman divergence between μ and ν can be expressed as following:

$$\mathrm{D}^F(\mu, \nu) \triangleq F(\mu) - F(\nu) - \langle \mu - \nu, \nabla F(\nu) \rangle.$$

The following result will be central for our proof.

Theorem 1 (Weighted centroid (Banerjee et al., 2005; Nielsen and Nock, 2007))**.** *Let $\{\mu_1, \dots, \mu_n\} \subset \mathcal{P}(\mathcal{X})$ and $\{w_1, \dots, w_n\}$ such that $w_i \geqslant 0$ and $\sum_i w_i = 1$. Then for any Bregman divergence D_F we have,*

$$\arg \min_{\nu \in \mathcal{P}(\mathcal{X})} \sum_{i=1}^n w_i \mathrm{D}^F(\mu_i, \nu) = \sum_{i=1}^n w_i \mu_i$$

We then introduce the following result which is an extension of Theorem 1 for the case of Markov Kernels.

Proposition 4 (Markov centroid)**.** *Given a set of tasks $\{\mathbb{P}_{X_t, Y_t}\}_{t \in \mathcal{T}}$ and λ a measure such that $\mathbb{P}_{X_t} \ll \lambda \; \forall t \in \mathcal{T}$.*

$$K^* \triangleq \arg \min_{K \in \mathcal{P}(\mathcal{Y}|\mathcal{X})} \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K),$$

then we have,

$$\forall x \in \mathcal{X}, \quad K^*(\cdot \mid x) = \sum_t \frac{h_t(x)}{c(x)} \mathbb{P}_{Y_t|X_t}(\cdot \mid x),$$

with,

$$h_t = \frac{d\mathbb{P}_{X_t}}{d\lambda} \quad \textit{and} \quad c = \sum_t h_t.$$

where $\frac{d}{d\lambda}$ denote the Radon-Nikodym derivative operator.

Proof.

$$\begin{aligned}
\min_K \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K) &= \min_K \sum_t \int_x \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot|x) \| K(\cdot \mid x)) d\mathbb{P}_{X_t}(x) \\
&= \min_K \sum_t \int_x \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot|x) \| K(\cdot \mid x)) d\mathbb{P}_{X_t}(x) \\
&\underset{\text{Lin.}}{=} \min_K \int_x \sum_t \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot|x) \| K(\cdot \mid x)) h_t(x) d\lambda(x) \\
&\geqslant \int_x \left(\min_K \sum_t \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot|x) \| K(\cdot \mid x)) h_t(x) \right) d\lambda(x) \\
&= \int_x \left(\min_K \sum_t \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot|x) \| K(\cdot \mid x)) \frac{h_t(x)}{c(x)} \right) c(x) d\lambda(x).
\end{aligned}$$

Where Lin. stands for the fact that the Lebesgue integral is a linear operator, and $c(x) = \sum_k h_k(x)$. Then by Theorem 1 the last term is minimized for K^* such that,

$$K^*(\cdot \mid x) = \sum_t \frac{h_t(x)}{c(x)} \mathbb{P}_{Y_t|X_t}(\cdot \mid x) \quad \forall x \in \mathcal{X}.$$

We then have,

$$\min_K \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K) \geqslant \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K^*)$$

However, by definition of a minimum we also have,

$$\min_K \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K) \leqslant \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K^*).$$

Which proves that,

$$K^* = \arg\min_K \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K).$$

If we explicit the normalization term, we have,

$$\forall x \in \mathcal{X} \quad \frac{h_t(x)}{c(x)} = \frac{\frac{d\mathbb{P}_{X_t}}{d\lambda}(x)}{\sum_k \frac{d\mathbb{P}_{X_k}}{d\lambda}(x)}$$

□

Remark 3. In the case of NLP, λ is identified as the counting measure since the space of natural language texts is at least discrete. In this specific case, we have,

$$\frac{h_t(x)}{c(x)} = \frac{\mathbb{P}\{X_t = x\}}{\sum_{k \in \mathcal{T}} \mathbb{P}\{X_k = x\}}, \quad \forall x \in \cup_{t \in \mathcal{T}} \mathcal{S}_{X_t}$$

We can now give the proof of Proposition 3.

Proof. Let,

$$K^* \triangleq \arg \min_{K \in \mathcal{P}(\mathcal{Y}|\mathcal{X})} \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t} \| K)$$
$$K^\dagger \triangleq \arg \min_{K \in \mathcal{P}(\mathcal{Y}|\mathcal{X})} \sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(M_t \| K)$$

Then by Proposition 4, for all $x \in \mathcal{X}$, if we suppose that the Bregman divergence is convex in its both arguments:

$$\begin{aligned}\mathrm{D}^F(M^*(\cdot \mid x), M^\dagger(\cdot \mid x)) &= \mathrm{D}_F\left(\sum_t \frac{h_t(x)}{c(x)}\mathbb{P}_{Y_t|X_t}(\cdot \mid x), \sum_t \frac{h_t(x)}{c(x)} M_t(\cdot \mid x)\right) \\ &\leqslant \sum_t \frac{h_t(x)}{c(x)} \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot \mid x), M_t(\cdot \mid x)).\end{aligned}$$

We consider the measure $\mu \triangleq \frac{1}{|\mathcal{T}|}\sum_t \mathbb{P}_{X_t}$. This measure can be seen as a mixture measure over the different domain of our tasks. It seems to be the most natural measure to use when dealing with a multi-task model. By noticing that $\frac{d\mu}{d\lambda} = \frac{1}{|\mathcal{T}|}c$, by unicity of the Radon-Nikodym derivative up to a λ-null measure space (*cf.* Radon-Nikodym-Lebesgue theorem (Bermudez et al., 2025, Theorem 6)), then we have,

$$\begin{aligned}\mathrm{D}^F_\mu(M^*, M^\dagger) &= \int_x \mathrm{D}^F(M^*(\cdot \mid x), M^\dagger(\cdot \mid x)) d\mu(x) \\ &= \frac{1}{|\mathcal{T}|}\int_x \mathrm{D}^F(M^*(\cdot \mid x), M^\dagger(\cdot \mid x)) c(x) d\lambda(x) \\ &\leqslant \frac{1}{|\mathcal{T}|}\sum_t \int_x \frac{h_t(x)}{c(x)} \mathrm{D}^F(\mathbb{P}_{Y_t|X_t}(\cdot \mid x), M_t(\cdot \mid x)) c(x) d\lambda(x) \\ &= \frac{1}{|\mathcal{T}|}\sum_t \mathrm{D}^F_{\mathbb{P}_{X_t}}(\mathbb{P}_{Y_t|X_t}, M_t).\end{aligned}$$

Since the KL divergence is a Bregman divergence convex in its both arguments it concludes the proof. Moreover it has been demonstrated in (Amari, 2009) that the KL divergence is the only f-divergence that is also a Bregman divergence. □

Remark 4. In Proposition 3 the measure μ seems to be a bit abstract and not directly related to our multi-task problem. However, we can give an interpretation of μ in terms of multi-tasking. In fact, let $T \in \mathcal{T}$ a random variable such that $\mathbb{P}_T(t) = \frac{1}{|\mathcal{T}|}$ and let $X \in \mathcal{X}$ a random variable such that $\mathbb{P}_{X|T=t} = \mathbb{P}_{X_t}$. Then by the law of total probabilities and Bayes theorem, for all $b \in \mathcal{B}(\mathcal{X})$ we have:

$$\begin{aligned}\mathbb{P}_X(b) &= \sum_t \mathbb{P}_{X|T=t}(b) \times \mathbb{P}_T(t) \\ &= \frac{1}{|\mathcal{T}|}\sum_t \mathbb{P}_{X_t}(b),\end{aligned}$$

from which we can conclude that $\mathbb{P}_X = \mu$. Then μ represents the probability measure on $\mathcal{X}$ that will draw samples randomly from the different measures $\{\mathbb{P}_{X_t}\}_{t\in\mathcal{T}}$ which is what a multi-task model is confronted to in an operational set-up.

B.5 Distribution shift (*cf.* Sec. 6)

The objective function we proposed in Eq. 4 supposed that for each task we have access to the input data distribution denoted as $\mathbb{P}_{X_t}$. However, in some cases we can have no access to $\mathbb{P}_{X_t}$ but to an approximation of it, denoted as $\mathbb{P}_{\tilde{X}_t}$. For example, we have a model trained on sentiment analysis and we do not have access the true data. We can thus use existing data for such task as an approximation. We show in the following that we can in fact control the behaviour of our method with respect to the quality of the approximation.

Proposition 5. *Considering a set of approximated distribution* $\left\{\mathbb{P}_{\tilde{X}_t}\right\}_{t\in\mathcal{T}}$, *for* $\mathrm{D} = \mathrm{JS}$, *our method will converge in a uniform way with* $\left\{\mathbb{P}_{\tilde{X}_t}\right\}_{t\in\mathcal{T}}$.

Proof. For sake of simplicity, in this proof we refer to a model by its parameter, *i.e.* $M(\cdot \mid \cdot, \theta) \equiv \theta(\cdot \mid \cdot)$ We recall that in Eq. 4 for a given task t we have the following,

$$\mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) = \int_x \mathrm{D}\left(\theta_t(.|x) \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})(.|x)\right) \mathbb{P}_{X_t}(dx).$$

Then,

$$\begin{aligned}
&\left|\mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) - \mathrm{D}_{\mathbb{P}_{\tilde{X}_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right)\right| \\
&= \left|\int_x \mathrm{D}\left(\theta_t(.|x) \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})(.|x)\right) \left(\mathbb{P}_{X_t}(dx) - \mathbb{P}_{\tilde{X}_t}(dx)\right)\right| \\
&\leqslant \int_x \mathrm{D}\left(\theta_t(.|x) \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})(.|x)\right) \left|\mathbb{P}_{X_t}(dx) - \mathbb{P}_{\tilde{X}_t}(dx)\right|
\end{aligned}$$

If we use the Jensen Shannon divergence we then have,

$$\left|\mathrm{JS}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) - \mathrm{JS}_{\mathbb{P}_{\tilde{X}_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right)\right| \leqslant \log(2) \int_x \left|\mathbb{P}_{X_t}(dx) - \mathbb{P}_{\tilde{X}_t}(dx)\right|$$

Then by Scheffe's Theorem (Scheffé, 1947), we have:

$$\left|\mathrm{JS}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) - \mathrm{JS}_{\mathbb{P}_{\tilde{X}_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right)\right| \leqslant 2\log(2)\mathrm{TV}(\mathbb{P}_{X_t}, \mathbb{P}_{\tilde{X}_t}),$$

where TV stands for total variation distance. Then we have,

$$\left|\sum_t \left(\mathrm{D}_{\mathbb{P}_{X_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right) - \mathrm{D}_{\mathbb{P}_{\tilde{X}_t}}\left(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})\right)\right)\right| \leqslant 2\log(2) \sum_t \mathrm{TV}(\mathbb{P}_{X_t}, \mathbb{P}_{\tilde{X}_t}),$$

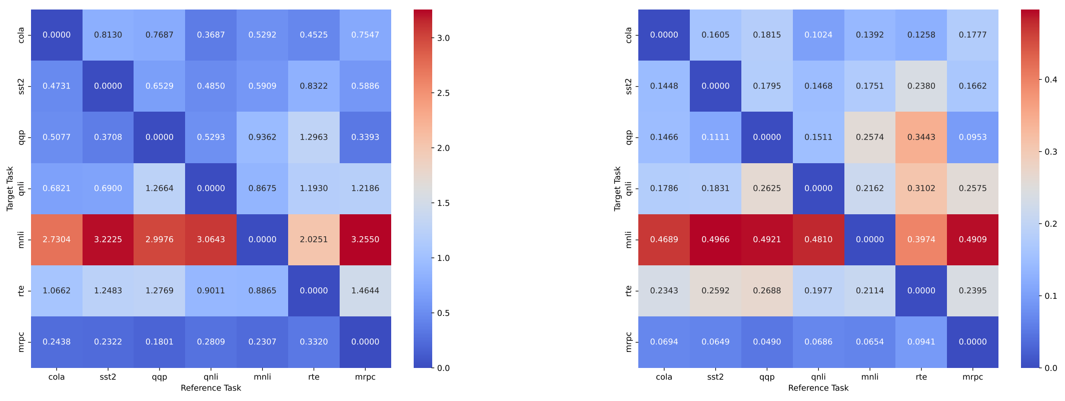
which concludes the proof. □

Remark 5. In Proposition 5 we state that the convergences is uniform in the sense that if the approximations we have converge uniformally to the true distribution *i.e.* in the sense of the total variation, then we have a convergence of our objective function.

C Correlation between Divergence Variants and Model Relatedness.

In this section, we propose experiments to support the fact that divergences are interesting proxies for model performances which is linked to Proposition 2. Given a set of tasks $\mathcal{T}$, for each pair of tasks $(i, j) \in \mathcal{T} \times \mathcal{T}$, we compute $\mathrm{D}_{\mathbb{P}_{X_i}}(\theta_i \| \theta_j)$ on a development set, and the performance of the model θ_j on task i on a test set, denoted as $\mathrm{PERF}(\theta_j, i)$. Then, for all tasks $i \in \mathcal{T}$ we compute the correlation between $\left\{-\mathrm{D}_{\mathbb{P}_{X_i}}(\theta_i \| \theta_j),\ \forall j \in \mathcal{T}\right\}$ and $\{\mathrm{PERF}(\theta_j, i),\ \forall j \in \mathcal{T}\}$. In Figure 5, we propose the heat map defined by $\left\{\mathrm{D}_{\mathbb{P}_{X_i}}(\theta_i \| \theta_j)\right\}_{(i,j) \in \mathcal{T} \times \mathcal{T}}$, and in Table 6, we proposed the matrix of values $\{\mathrm{PERF}(\theta_j, i)\}_{(i,j) \in \mathcal{T} \times \mathcal{T}}$.

Base on Sec. 5.1, we decided to focus on classification tasks *i.e.* tasks from the GLUE benchmark. We report Spearman's correlations in Table 7. Correlations computed in Table 7 correspond to correlations computed between rows of Figure 5 and rows of Table 6. We can clearly observe that we obtain high correlations indicating that KL and JS are interesting proxies for performance: if $\mathrm{D}_{\mathbb{P}_{X_i}}(\theta_i \| \theta_j)$ is low, it may suggest that $\mathrm{PERF}(\theta_j, i)$ will be high (without stating that this is a causation relation). We also observe that the JS divergence achieves the highest correlation. Since our method involves minimizing $\sum_{i=1}^n \mathrm{D}_{\mathbb{P}_{X_i}}(\theta_i \| \mathrm{TA}_n^\Gamma)$, it thus can be viewed as aiming for a merging model that performs well on each task. This aligns directly with Proposition 2, which states that our method is an approximation of the classical multi-task learning objective.



(a) KL

(b) JS

Figure 5: $\mathrm{D}_{X_i}(\theta_i \| \theta_j)$ values for different divergences. (i corresponds to the row index, while j corresponds to the column index.)

Eval Dataset → Model ↓	**CoLA**	**SST-2**	**QQP**	**QNLI**	**MNLI**	**RTE**	**MRPC**
CoLA	**82.20**	77.80	50.60	44.60	8.80	28.88	70.00
SST-2	44.00	**92.80**	71.20	37.00	10.20	26.71	66.75
QQP	33.00	50.20	**84.60**	40.80	8.60	28.88	69.75
QNLI	46.20	76.40	39.20	**86.40**	9.20	43.68	69.00
MNLI	33.80	61.40	65.40	45.00	**78.00**	28.16	67.50
RTE	34.80	48.20	63.60	50.20	14.00	**75.81**	69.00
MRPC	32.80	57.40	66.20	41.20	11.60	45.85	**85.00**

Table 6: Accuracies (%) of each model checkpoint (rows) evaluated on the seven GLUE tasks (columns). Each row corresponds to a model fine-tuned on a specific task. The highest accuracy for each task is highlighted in bold, and corresponds each time to the specialized model.

D Additional experiments

D.1 Task Vectors Cosine Similarities

As is well known in the model merging literature, and more specifically within the task arithmetic framework, cosine similarities between task vectors are typically close to zero. This indicates that the tasks are sufficiently disentangled and can be effectively merged using task arithmetic methods. In Figure 6, we present the cosine similarity matrices of the task vectors used in our experiments.

D.2 Details on Figure 2

On Table 8, we propose the values that are plotted on Figure 2a, and on Table 9 we propose the values that are plotted on Figure 2b.

D.3 Parameter convergence

In Sec. 5.2 we provided an analysis of the convergence of our method by displaying the evolution of our loss function through training iterations and we concluded that our method smoothly converges to an local

	CoLA	**SST-2**	**QQP**	**QNLI**	**MNLI**	**RTE**	**MRPC**	**Avg.**
KL	0.82	**0.92**	0.31	0.77	0.95	0.77	**0.88**	0.77
JS	**0.92**	0.89	**0.34**	**0.80**	**0.99**	**0.91**	0.87	**0.82**

Table 7: Spearman's correlation between (negations of) the divergences between dedicated models and cross-performances of each model on each task. The "Avg." column reports the mean correlation across all tasks.

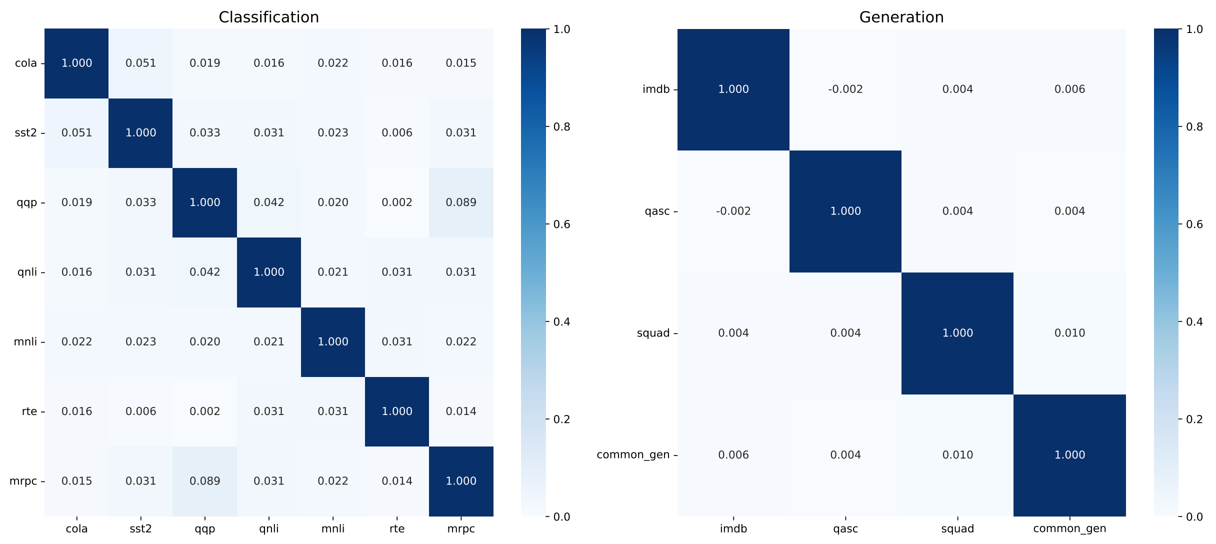


Figure 6: Cosine similarity matrices of task vectors between different tasks. Left: Similarity between GLUE benchmark tasks (CoLA, SST2, QQP, QNLI, MNLI, RTE, MRPC). Right: Similarity between diverse tasks from the mixed scenario (IMDB, QASC, SQuAD 2.0, CommonGen). Lower similarity values indicate greater orthogonality between task vectors, suggesting less interference when merging models fine-tuned on these tasks.

# Tasks	**Model Averaging**	**Multi-SLERP**	**TIES**	Task-wise			Layer-wise		
				Adamerging	KL (ours)	JS (ours)	Adamerging	KL (ours)	JS (ours)
2	93.89	94.20	96.02	91.16	98.18	98.28	92.41	98.85	98.85
3	89.10	92.18	94.08	90.86	97.40	97.39	90.26	98.42	98.26
4	75.12	79.31	79.37	78.73	80.12	79.21	78.47	95.20	95.19
5	60.92	66.86	73.61	66.84	83.82	85.37	64.93	95.60	95.50
6	56.98	70.48	67.23	67.97	83.45	84.45	62.85	93.11	93.42
7	60.51	68.89	68.39	67.26	83.45	84.70	63.05	92.53	93.06
Average	72.75	78.65	79.78	77.14	87.73	88.23	75.33	95.62	95.71

Table 8: ANP for merged tasks obtained via different merging methods. Values are normalized as percentages, with separate evaluations for KL and JS divergence variants.

optimum value. We decided to go further and analyse the evolution of the coefficients associated to each task. As a recall we used the framework of task arithmetic and in this framework the merged model is given by the following,

$$\theta_0 + \sum_i \Gamma_i \times \tau_i,$$

and we are here interested into the evolution of the coefficients Γ_i. In Figure 7, we provide the evolution of Γ_1 (left) and Γ_2 (right) through training iterations, on different pairwise merging set-up on the benchmark GLUE. We can mainly observe that the dynamic of our method is also smooth in the coefficients Γ_i, with an interesting convergence of the parameters. We can also go further by observing in some settings that the values of Γ_1 and Γ_2 seem to be independent meaning that when merging two tasks the merging coefficient associated to one task seems to strongly depend on the task itself and not the task with which we merge. To better support this fact, we decided to add a visualization. In the framework of task arithmetic, each merging experiment can be represented by a point in an euclidean space defined by the following coordinates $(\Gamma_1, \Gamma_2, \ldots, \Gamma_n)^t$. In the case of pairwise merging experiments, these points are in a plan and we decided to visualize this plan on Figure 8, for classification tasks, and Figure 9 for mixed tasks. On these figures, we can mainly observe that we have different scenarios. For tasks such as QNLI, the factor associated with the QNLI task seems not to depend on the other tasks, while for some other tasks such as MRPC and CoLA we have another scenario where the value of the coefficient associated to the task seems to depend on the value associated to the other tasks. This seems to be an interesting observation, to

# Tasks	Model Averaging	Multi-SLERP	TIES	Task-wise			Layer-wise		
				Adamerging	Forward (ours)	JS (ours)	Adamerging	Forward (ours)	JS (ours)
2	97.92	98.96	98.43	99.74	96.34	98.96	100.00	98.96	99.48
3	82.62	67.89	87.87	79.92	92.99	96.36	89.26	95.91	97.79
4	53.38	64.85	56.42	60.88	87.52	91.75	83.15	94.97	97.44
Average	77.97	77.23	80.91	80.18	92.28	95.69	90.80	96.61	98.24

Table 9: ANP for merged tasks obtained via different merging methods. Values are normalized as percentages, with separate evaluations for KL and JS Divergence variants.

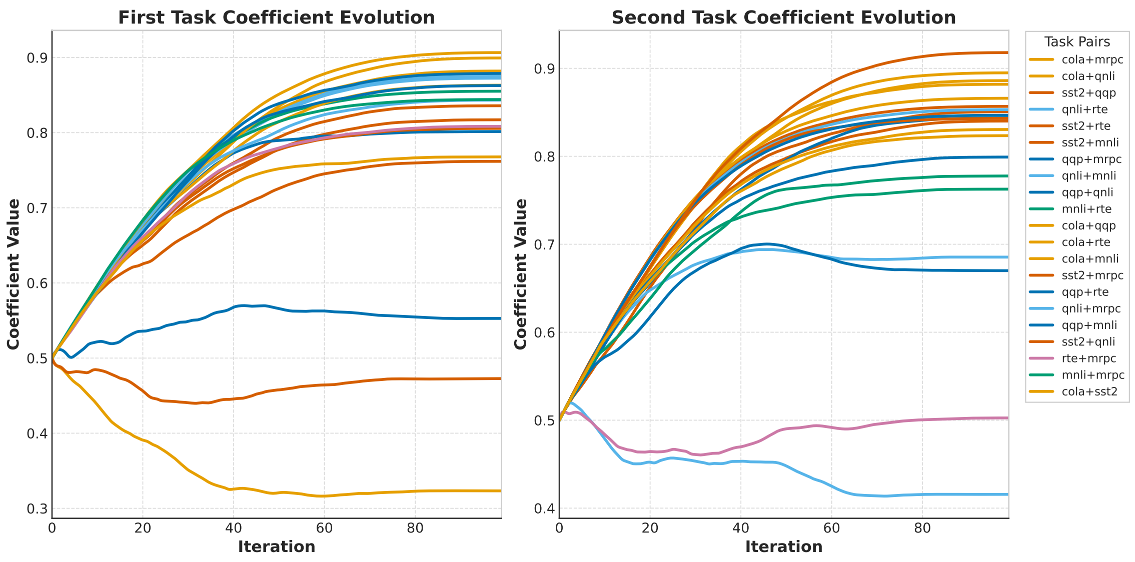


Figure 7: Evolution of the task coefficients across training iterations. The first graph shows the coefficient assigned to the first task in each task pair (as indicated in the legend), while the second graph shows the coefficient assigned to the second task.

be considered alongside the fact that some tasks may be independent, while others may have a statistical dependency, *i.e.* completing one task may have a positive or negative impact on another.

E Training Settings

E.1 Data details

As explained in Sec. 5.1, we used the GLUE Benchmark (Wang et al., 2018) to perform our experiments. We recall on Table 10 the description of tasks from this benchmark.

E.2 Training details

We propose in Table 11 training hyper-parameters we chose for our method, as well as for the Adamerging one since it also requires a training procedure. All the optimizations were done using the Adam opti-

CoLA	detection of the linguistic acceptability of a sentence
MNLI	natural language inference
MRPC	paraphrase detection
QNLI	question answering converted into natural language inference
QQP	detection of equivalence between questions
RTE	natural language inference
SST2	sentiment analysis

Table 10: Description of the GLUE Benchmark

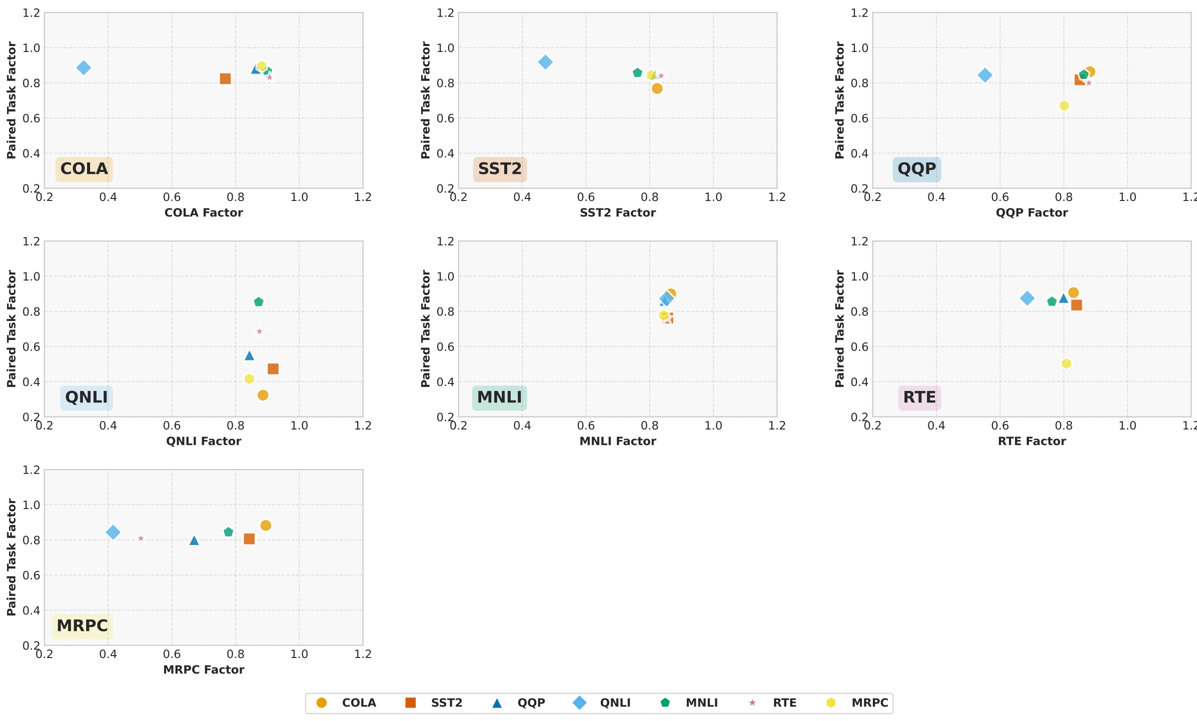


Figure 8: Visualization of coefficient values for a fixed reference task versus coefficient values for the remaining GLUE tasks. Each subplot corresponds to a different reference task.

Method	Level	Batch Size	Epochs	Dataset Size	Scheduler (LR)	Init. Param.
KL/JS (Classif)	*task*	$4\times$ # tasks	4	200	1e−2	0.5
KL/JS (Classif)	*layer*	$4\times$ # tasks	4	400	1e−2	0.5
KL/JS (Gen.)	*task*	$4\times$ # tasks	4	200	1e−2	0.5
KL/JS (Gen.)	*layer*	$4\times$ # tasks	4	400	1e−2	0.5
AdaMerging (Classif)	*task*	$4\times$ # tasks	5	200	1e−3	0.5
AdaMerging (Classif)	*layer*	$4\times$ # tasks	5	400	1e−3	0.5
AdaMerging (Gen.)	*task*	$4\times$ # tasks	5	200	1e−2	0.5
AdaMerging (Gen.)	*layer*	$4\times$ # tasks	5	400	1e−2	0.5

Table 11: Training configurations.

mizer (Kingma and Ba, 2014) with default moments hyper-parameters. From a practical standpoint, the hyperparameters we choose, both for the Adamerging method and for our own, allow us to maximize multi-task performance on evaluation sets. While the choice of hyperparameters is relatively sensitive in the Adamerging method, our method appears to be more robust in terms of hyperparameter selection. The other methods we used that had hyper-parameters were TIES and Multi-SLERP, for which we basically used the recommended recipes:

- **TIES:** We used the recommended recipe from (Yadav et al., 2023), with $\lambda = 1$ and a mask rate of 0.2 (i.e., 80% zeros in the mask).
- **Multi-SLERP:** The weights were set to $1/N$, where N is the number of tasks.

F Everything is task arithmetic

Many different merging methods have emerged in the landscape of machine learning. Among them, task arithmetic (Ilharco et al., 2023) is probably the most widely used. As a reminder, the merging function in the case of task arithmetic is defined as follows:

$$\mathrm{TA}_{\Gamma}\left(\theta_0, \mathbf{T}\right) = \theta_0 + \sum_{t \in \mathcal{T}} \Gamma_t \times \tau_t.$$

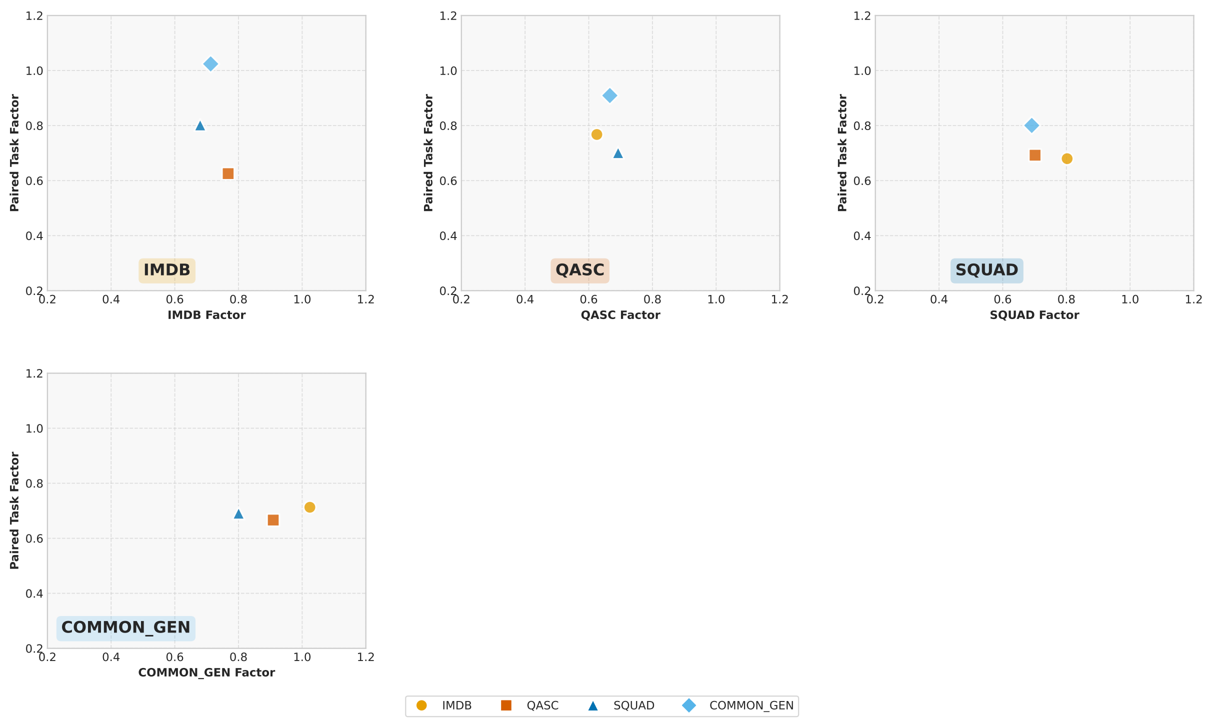


Figure 9: Coefficient values for each task at different T5 checkpoints. Each plot fixes a reference task and compares its coefficient to those of the other tasks.

An interesting question that naturally arises is the following: Given a merging method $g_\Delta(\theta_0, \mathbf{T})$, can we find coefficients Γ such that

$$g_\Delta(\theta_0, \mathbf{T}) = \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})?$$

If this is true, then we can state that

$$\min_\Gamma \sum_{t\in\mathcal{T}} \mathrm{D}_{X_t}(\theta_t \,\|\, \mathrm{TA}_\Gamma(\theta_0, \mathbf{T})) \leqslant \min_\Delta \sum_{t\in\mathcal{T}} \mathrm{D}_{X_t}(\theta_t \,\|\, g_\Delta(\theta_0, \mathbf{T}))\,.$$

This inequality would highlight the strength of our method, as it encompasses the entire range of task arithmetic. As stated in Sec. 3, our method can be applied to any hyper-parameter differentiable merging approach. As pointed out in (Goddard et al., 2024), a wide range of merging methods are based on task arithmetic, with the main differences lying in the estimation of the merging coefficients. In the following, we provide an analysis of other merging methods to show that they can be expressed as model merging. This demonstrates that our method has the potential to achieve better results.

SLERP. Spherical linear interpolation (SLERP) (Wortsman et al., 2022) is a classical method used to combine vectors on a spherical manifold. For this method, we introduce a hyperparameter $t \in [0, 1]$, and define SLERP as follows:

$$f_t(\theta_0, \{\tau_1, \tau_2\}) \triangleq \theta_0 + \frac{\sin((1-t)\Omega)}{\sin\Omega}\frac{\tau_1}{\|\tau_1\|} + \frac{\sin(t\Omega)}{\sin\Omega}\frac{\tau_2}{\|\tau_2\|},$$

where Ω is the angle between τ_1 and τ_2.

Fisher Weight Averaging. The Fisher weight averaging method, introduced in (Matena and Raffel, 2022), is a merging technique that reduces to task arithmetic in the case of linear interpolation between specialized models. The merging coefficients for each model are based on the Fisher Information matrix and are determined by solving an optimization problem related to finding a centroid between models. One limitation, as pointed out in the original paper, is the high computational cost of estimating the Fisher Information matrix to obtain the merging coefficients. This estimation can also be numerically unstable, as the coefficients in the matrix can be close to zero. Additionally, this method introduces extra scaling hyper-parameters that must be tuned.

RegMean. The RegMean merging method, proposed in (Jin et al., 2023), also reduces to task arithmetic, as it performs a linear interpolation between specialized models. This interpolation aims to minimize the L_2 distance between the merged model and the individual models, whereas our method is designed to minimize the JS (or KL) divergence between models. The L_2 distance is a restrictive measure. Moreover, as stated in (Blau and Michaeli, 2018, 2019; Zhang, 2023), L_2 distance is a distortion measure, while KL and JS are perception measures. Minimizing perception distance appears to be more suitable for downstream applications, such as performing other tasks.

Karcher Mean. The Karcher mean (or Riemannian centroid or Fréchet mean), originally formulated in (Grove and Karcher, 1973) can be used as a merging method which consists in finding some sort of centroid of a finite set of task vectors, denoted as $\{\tau_t\}$. To do so, we suppose that task vectors lies in a Finite dimension Hilbert Space $(H, \langle\cdot,\cdot\rangle)$, where $\langle\cdot,\cdot\rangle$ is the standard dot product onto this space and thus $\|\cdot\|$ is the associated norm. The Kracher mean is defined as following,

$$\tau_{\mathrm{F}} \triangleq \arg\min_{\tau\in H} \sum_t \|\tau - \tau_t\|^2.$$

The following proposition holds,

Proposition 6. *Let $(H, \langle\cdot,\cdot\rangle)$ be a Hilbert space. Let $\{\tau_t\}_{t\in\mathcal{T}} \subset H$, a finite set of points in this hilbert space. Then,*

$$\arg\min_{h\in H} \sum_{t\in\mathcal{T}} \|\tau - \tau_t\|^2 \in \mathrm{Span}(\{\tau_t\}_{t\in\mathcal{T}})$$

Proof. Let $\{\tau_t\} \subset H$. Let $F \triangleq \mathrm{Span}\left(\{\tau_t\}\right)$. Let $\tau \in H$. We have the following result,

$$\tau = p_1 + p_2, \text{ s.t. } p_1 \in F, p_2 \in F^{\perp}.$$

Then we have,

$$\begin{aligned}\psi(\tau) &\triangleq \sum_t \|\tau - \tau_t\|^2,\\ &= \sum_t \langle \tau - \tau_t, \tau - \tau_t\rangle,\\ &= \sum_t \|\tau\|^2 - 2\langle\tau,\tau_t\rangle + \|\tau_t\|^2,\\ &= \sum_t \|p_1\|^2 + \|p_2\|^2 - 2\langle p_1,\tau_t\rangle + \|\tau_t\|^2.\end{aligned}$$

Then by taking, $\tau' = p_1$, we have $\psi(\tau') \leqslant \psi(\tau)$, which leads to the following statement: $\forall\tau \in H$, $\exists\tau' \in F$, such that,

$$\psi(\tau') \leqslant \psi(\tau).$$

Then $\arg\min_{\tau\in H} \psi(\tau) \in F$, which concludes the proof. □

Remark 6. As stated in Sec. 4, the method we proposed in this study can also be viewed as finding a centroid. However the framework we used does not allow to connect directly to the theory of Karcher mean. In fact, if one would want to formulate our method as a Karcher mean, the "distance" defined over the space of task vectors would be the following,

$$d(\tau_i, \tau_j) = \mathrm{D}_{X_i}\left(\theta_i \| \theta_j\right).$$

However, even in the case of the Jensen Shannon divergence this "distance" is not a mathematical one as it does not respect the property of the distance. Consequently it does not define a metric space and therefore even less a Hilbert space. Then an interesting line of research would be to identify the possible distances one could define over the space of task vectors. From the result we just demonstrated, if we can verify that the distance can be derived from a dot product and thus induce a Hilbert Space, then we can conclude that the optimal solution lies in the framework of task arithmetic, giving thus added weight to this method and possibly offering more theoretical explanations as to why this method is in many cases the state of the art.