Low-Rank Bias, Weight Decay, and Model Merging in Neural Networks

Ilja Kuzborskij
Google DeepMind

Yasin Abbasi Yadkori
Sapient Intelligence

Abstract

We explore the low-rank structure of the weight matrices in neural networks at the stationary points (limiting solutions of optimization algorithms) with $L2$ regularization (also known as weight decay). We show several properties of such deep neural networks, induced by $L2$ regularization. In particular, for a stationary point we show alignment of the parameters and the gradient, norm preservation across layers, and low-rank bias: properties previously known in the context of solutions of gradient descent/flow type algorithms. Experiments show that the assumptions made in the analysis only mildly affect the observations.

In addition, we investigate a multitask learning phenomenon enabled by $L2$ regularization and low-rank bias. In particular, we show that if two networks are trained, such that the inputs in the training set of one network are approximately orthogonal to the inputs in the training set of the other network, the new network obtained by simply summing the weights of the two networks will perform as well on both training sets as the respective individual networks. We demonstrate this for shallow ReLU neural networks trained by gradient descent, as well as deep linear networks trained by gradient flow.

1 INTRODUCTION

First-order optimization algorithms, such as Stochastic Gradient Descent (SGD) have emerged as a go-to tool for training machine learning models. Their popularity primarily stems from good scalability and empirical performance, while their theoretical properties and performance are reasonably well understood for learning problems with sufficient structure, such as convexity or even weak forms of non-convexity. However, in the recent years with the advent of overparameterized deep neural networks, there was a surge of interest in understanding the behavior of these algorithms on non-convex and often non-differentiable problems. Here, a combination of neural network architecture choice, initialization, and hyperparameter tuning often achieves near-zero training loss while enabling good test-time generalization ability.

Recently, some progress was made in an attempt to explain this phenomenon by looking for *implicit biases* in the algorithm (Vardi, 2023). While implicit biases were known before in the context of simpler learning problems such as overparameterized linear least-squares (solving this problem through pseudo-inverse gives an interpolating solution with the *minimal* $L2$ *norm*), observing the same phenomenon in neural network learning is rather recent. Lyu and Li (2020); Ji and Telgarsky (2020) showed that for a certain family of neural networks (*positive homogeneous* neural networks, such as deep linear or deep ReLU neural networks) trained by an idealized version of Gradient Descent (GD) known as Gradient Flow (GF), asymptotically converges to the predictor with the smallest parameter $L2$ norm. In another influential line of work, known as the Neural Tangent Kernel (NTK) approximation Jacot et al. (2018); Du et al. (2018); Ji and Telgarsky (2019b), it was shown that a sufficiently wide shallow non-linear neural network behaves similarly as a linear predictor on Reproducing kernel Hilbert space (RKHS). This enabled reduction to analysis of GD on RKHS, which is known to converge to the minimum $L2$ norm interpolating solution (or, which behaves as a regularized solution when stopped early (Yao et al., 2007)).

While $L2$ norm minimization (or regularization) has a clear interpretation in linear models, its role in deep neural network learning is less intuitive. On that front,

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

several works showed that this form of regularization induces a *low-rank* structure in weight matrices of neural networks. In fact, in deep linear neural networks weight matrices in such interpolants are shown to be rank-1 matrices (Ji and Telgarsky, 2019a). Similarly, Timor et al. (2023) showed that weight matrices in minimum $L2$ norm interpolating deep ReLU networks will have a low *stable rank* (ratio between Frobenius and spectral norms) as long as the depth is sufficiently large. See Section A for other related works.

Finally, while the appearance of low-rank weight matrix structure during training is interesting on its own, here we might ask a related question whether low-rank bias can explain other phenomena that we observe in neural network learning. In particular, in this paper we look at the connection between low-rank bias and a form of a multi-task learning known as *model merging*. Here, weight matrices of two neural networks trained on different tasks are summed to form a new neural network, which often performs well on both original tasks. This behaviour seems surprising at first glance, and the reason why model merging (Ilharco et al., 2023) (or, related, parameter averaging) might be effective is not well understood.

1.1 Our Contributions

In this paper we take a closer look at a low-rank bias and its effect on model merging. Our contribution is two-fold, yet at the core it is connected to low-rank bias induced by $L2$ regularization.

Low-rank bias and alignment through weight decay Model merging discussed earlier crucially relies on $L2$ regularization, and its success (good performance on both tasks) can be attributed to biases that arise in neural networks because of regularization. To this end, in Section 3 we take a closer look at such biases. In particular, we focus on depth-K neural networks with H-positive homogeneous activations (such as ReLU or powers of ReLU) and consider stationary points of the empirical logistic loss with $L2$ regularization. In this context, stationary points enjoy *alignment*, meaning that parameters converge in the direction of gradient of the (unregularized) loss. Asymptotic alignment was known before in case of GF and GD (without regularization) (Ji and Telgarsky, 2019a, 2020). Building upon this observation we show the following results:

- Near norm-preservation: weight matrices in neural networks we consider have the same Frobenius norm throughout layers, up to a scaling factor governed by the homogeneity parameter H and layer index.
- Deep linear neural networks have all weight matrices of rank one.
- Deep (possibly non-linear) neural networks have a low-rank weight matrix structure, in the following sense:

The weighted harmonic mean of stable ranks, or *pseudo-rank*[1] of weight matrices $(W_1, \dots, W_K)$ is controlled by regularization parameter λ and the training loss at the non-zero stationary point:

$$\frac{1}{\text{weight}_1 \left(\frac{\|W_1\|_F}{\|W_1\|_2}\right)^{-1} + \dots + \text{weight}_K \left(\frac{\|W_K\|_F}{\|W_K\|_2}\right)^{-1}} \leq \sqrt{\frac{(\text{Training loss})^{\alpha}}{\lambda}}$$

for some $\alpha > 1/2$ that depends only on (H, K), and where weights $(\text{weight}_k)_{k=1}^K$, depending only on $(H, [K])$, increase monotonically as the layer index k increases, entails that we effectively control the harmonic mean of the last layers.

The bound on the pseudo-rank is problem-dependent in a sense that it scales with the training loss. Therefore considering different solutions (stationary points of the regularized loss) we can observe different behaviours of the pseudo-rank. For instance, assume that the regularized loss at the stationary point is no greater than the regularized loss at the 0 vector, which is for instance satisfied by the global minimizer. Then, for any λ that ensures non-zero global minimizer,

$$\text{Pseudo-rank} \leq \sqrt{1/\lambda}$$

that is pseudo-rank decreases with increase of λ. Another example is to consider the limiting training error of some optimization algorithm with step size appropriate tuning, and Xavier-type initialization. In this case,

$$\text{Pseudo-rank} \lesssim \sqrt{1/\lambda + (\text{depth}) \cdot (\text{width})}\ .$$

Harmonic mean of stable ranks as a proxy for capturing the rank structure in deep neural networks was proposed by Timor et al. (2023). Since stable rank cannot be smaller than one, the harmonic mean is small whenever majority of weight matrices have small stable ranks. In particular, Timor et al. (2023) looked at the minimum $L2$-norm *interpolating* neural networks, in a sense that for all training examples $y_i \in \{\pm 1\}$, $y_i \cdot \text{prediction}(x_i) \geq 1$. Their conclusion was that the harmonic mean $\to \sqrt{2}$ as depth $K \to \infty$. In contrast,

[1] A stable rank of matrix is defined as a ratio of its Frobenius and spectral norms. It is small whenever matrix has few dominant eigenvectors.

we simply require GD to achieve a stationary point, in which case the harmonic mean is controlled by regularization parameter and the training loss at stationarity, even at a finite depth (see Lemma 3 and discussion therein). Indeed, this seems to be supported by some basic empirical evidence (see Section 5 and Fig. 2), which suggests that a low stable rank can be achieved even without interpolation but with weight decay.

Finally, we look at the behavior of a stable rank and norm preservation empirically for very deep neural networks (such as pre-trained large language models) in Section 5, and conclude that assumptions (such as homogeneity and stationarity) only mildly affect the observations.

Model merging enabled by weight decay. Next, we show that low-rank bias in neural networks enables effective *model merging*: after neural networks are trained on different datasets with mutually nearly-orthogonal inputs (but not necessarily orthogonal within the task), simply summing their weight matrices results in a predictor with combined weights that performs well simultaneously on *all tasks*. We explain this phenomenon through low-rank bias. Training biases different networks toward low-rank weight matrices that span non-overlapping subspaces. Consequently, summing their weight matrices results in a neural network that behaves as the original ones on inputs from respective tasks. In other words, two neural networks can be made to 'reside' in one parameter set.

More formally, given input-label pairs $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{S}^{d-1} \times \{\pm 1\}$ we obtain a predictor $f_{\theta(t)}$, whose parameters $\theta(t)$ are found by GD (or GF) run for t steps, while minimizing regularized loss $\theta \mapsto \frac{1}{n}\sum_i \mathrm{loss}(f_\theta(x_i), y_i) + \lambda\|\theta\|^2$. In a similar way, we obtain parameters $\theta'(t)$ given another data $(x'_1, y'_1), \dots, (x'_n, y'_n)$ belonging to the second task, such that inputs between these tasks are approximately orthogonal in a sense that $\max_{i,j} \left|\langle x_i, x'_j\rangle\right| \le \varepsilon$. Then, given an input x originating from the first task, we show that

$$|f_{\theta(t)}(x) - f_{\theta(t)+\theta'(t)}(x)| \le f_{\theta'(0)}(x)\, e^{-\lambda t} + C_2\, \varepsilon$$

for several scenarios.

Remark 1 (Approximate orthogonality in practice)**.** *The assumption* $\max_{i,j} \left|\langle x_i, x'_j\rangle\right| \le \varepsilon$ *with small* ε *is realistic in high-dimensional feature spaces. As shown by Luisto (2025), the number of nearly orthogonal directions in* $\mathbb{R}^d$ *grows rapidly with* d*, so that for typical embedding dimensions* ($d \ge 512$) *one can fit exponentially many almost orthogonal vectors. Empirically, semantic concepts in large language models are encoded as nearly orthogonal directions in the embedding space (Luisto, 2025). As a more basic example, normalized bag-of-words vectors for texts from different topics share very few non-zero coordinates and are therefore nearly orthogonal in high dimension.*

Here, exponentially vanishing term that appears because of $L2$ regularization, is responsible for contribution of initialization, which is typically non-zero in neural network training. The second, ε-dependent term captures the length of projection of the input (or activation vector in case of multilayer neural networks) from one task onto the weight matrix of the network trained on another task. Note that $L2$ regularization is essential here, since without it, the effect of initialization (appearing through a constant term $f_{\theta'(0)}(x)$) would not disappear. Interestingly, none of these results require convergence to the local minimum.

In particular, we show the above in case of linear prediction and shallow ReLU neural networks trained by GD, and deep linear neural networks trained by GF. Finally, in Section 5 we experimentally show that the same findings hold in case of fully-connected ReLU neural networks, which suggests that the same conclusions hold beyond our theoretical results.

2 PRELIMINARIES

Throughout, $\|\cdot\|$ is understood as the Euclidean norm for vectors and the Frobenius norm matrices. When, written explicitly for matrices, $\|\cdot\|_2$ is a spectral norm, and $\|\cdot\|_F$ is a Frobenius norm. For some matrix A its *stable rank* is defined as a ratio $\|A\|_F/\|A\|_2$. The Frobenius inner product between matrices A, B is denoted by $\langle A, B\rangle = \mathrm{tr}(A^\top B)$. Throughout $e_j = (\mathbf{1}(j=1), \mathbf{1}(j=2), \dots, \mathbf{1}(j=m)) \in \{0,1\}^m$ and $a \wedge b = \min(a,b)$ and $a \vee b = \max(a,b)$.

In the following we denote Rectified Linear Unit (ReLU) operation by $(x)_+ = \max(x, 0)$ for $x \in \mathbb{R}$. For vectors and matrices application of $(\cdot)_+$ is understood elementwise. ReLU has several useful properties. For $a, b \in \mathbb{R}$, we have $|(a)_+ - (b)_+| \le |a-b|$ and so for vectors $x, y \in \mathbb{R}^d$, $\|(x)_+ - (y)_+\|_2^2 = \sum_i ((x_i)_+ - (y_i)_+)^2 \le \sum_i (x_i - y_i)^2 = \|x-y\|_2^2$.

Function $f : \mathbb{R}^d \to \mathbb{R}$ is called *positive homogeneous* of degree K when $f(\alpha x) = \alpha^K f(x)$ for any $\alpha \ge 0$. Euler's homogeneous function theorem states that, if the chain rule holds, then $Kf(x) = \langle x, \nabla f(x)\rangle$. Note that ReLU is positive homogeneous, meaning that $(\alpha x)_+ = \alpha(x)_+$ for $\alpha \ge 0$. In particular, this implies that a K-layered ReLU or linear neural network is positive homogeneous and so $f_\theta(\alpha\, x) = \alpha^K f_\theta(x)$ and so Euler's theorem holds.

For the logistic loss function $\ell(x) = \ln(1+e^{-x})$ we have

$\ell'(x) = -e^{-x}/(1+e^{-x}) \in (-1,0)$, and moreover the following facts hold: $|\ell'(x)| = -\ell'(x) \le \ell(x)$, $x\,\ell'(x) \le \ell(x)$, and $-x\,\ell'(x) \le \sqrt{\ell(x)}$ for all x.

Differentiation We introduce some formalism for non-differentiable functions since occasionally we will work with the ReLU activation. For some $F : \mathbb{R}^p \to \mathbb{R}$ that is locally Lipschitz[2], we denote by ∂F its Clarke differential:

$$\partial F(\theta) = \mathrm{conv}\left(\{g \in \mathbb{R}^p : \exists (\theta_i)_i \to \theta,\ \nabla F(\theta_i) \to g\}\right).$$

Vectors in ∂F are called subgradients, while $\bar{\partial} f(x)$ will stand for a unique minimum-norm subgradient, that is $\bar{\partial} f(x) = \arg\min_{g \in \partial f(x)} \|g\|$. Throughout the paper it is understood that $\nabla f = \bar{\partial} f$. We will assume that the chain rule holds, that is $\dot{f}(\theta(t)) = \langle g(t), \dot{\theta}(t)\rangle$, for all $g(t) \in \partial F(\theta(t))$. It is possible to formally establish that the chain rule holds by assuming a technical notion of 'definability' (not covered here, see for instance (Ji and Telgarsky, 2019a)), which excludes functions that might result in a badly behaved optimization.

Neural networks In this paper we consider multilayer neural networks $f_\theta : \mathbb{R}^d \to \mathbb{R}$ given by a recursive relationship for $k \in [K-1]$,

$$f_\theta(x) = \langle w_K, h_{K-1}\rangle, \quad h_k = \sigma(W_k h_{k-1}), \quad h_0 = x$$

where x is the input, $W_1, \ldots, W_K \in \mathbb{R}^{m\times m}$ is collection of weight matrices, $\theta = (\mathrm{vec}(W_1), \ldots, \mathrm{vec}(W_K))$ is a parameter vector, and $\sigma : \mathbb{R}^m \to \mathbb{R}^m$ is activation function.

In practice parameters θ are tuned based on the training data by minimizing some loss function. In this paper we will focus on a binary classification problem, and so given inputs and labels $(x_i, y_i)_{i=1}^n \in (\mathbb{B}^d \times \{-1,1\})^n$, the training loss and its regularized version are defined as

$$L(\theta) = \frac{1}{n}\sum_{i=1}^n \ell\left(y_i\, f_\theta(x_i)\right) \ , \quad L_\lambda(\theta) = L(\theta) + \frac{\lambda}{2}\|\theta\|_2^2$$

where $\ell()$ is a logistic loss function $\ell(x) = \ln(1+e^{-x})$. In this work we consider standard *GD* algorithm which is often used to approximately minimize L_λ through recursive updates, for all $k \in [K]$ and $s = 0, 1, 2, \ldots, t$,

$$W_{k,s+1} = (1-\eta\lambda)W_{k,s} - \frac{\eta}{n}\sum_{i=1}^n y_i\,\ell'(y_i\, f_{\theta_s}(x_i))\,\frac{\mathrm{d}f_\theta(x_i)}{\mathrm{d}W_k}(\theta_s)$$

where $\eta > 0$ is a step size and θ_0 is an initial parameter vector.

[2] F is locally Lipschitz if for every point θ, there exists a neighborhood $B \supseteq \{\theta\}$ such that F is Lipschitz on B.

When we consider GF dynamics (we use (t) time indexing instead of $\cdot_t$ as in the discrete case), the update rule is replaced by time derivative

$$\dot{W}_k(t) = -\lambda W_k(t) - \frac{1}{n}\sum_{i=1}^n y_i\,\ell'(y_i\, f_{\theta(t)}(x_i))\,\frac{\mathrm{d}f_\theta(x)}{\mathrm{d}W_k}(\theta(t)) \ . \tag{1}$$

3 STATIONARY POINTS AND LOW-RANK BIAS

In this section we consider training deep neural networks with positive homogeneous activation functions (such as identity or ReLU or smoothed versions of ReLU, e.g. powers of ReLU), with weight decay regularization. Proofs for all statements in this section can be found in Section B.

In particular, we study solutions θ that are the *stationary points* of the regularization empirical loss L_λ. These solutions satisfy the following *alignment condition*:[3]

$$\lambda\theta = -\nabla L(\theta) \ . \tag{2}$$

Under standard smoothness and boundedness conditions, first-order methods such as SGD, Adam, and AdaGrad produce iterates whose gradients of the regularized objective L_λ vanish in expectation/asymptotically (Défossez et al., 2022; Li et al., 2023); hence alignment holds at their limit points. For non-smooth objectives (e.g., ReLU nets), these guarantees do not apply in full generality; one typically works with Clarke stationarity or adds additional structure/weak convexity. For the non-smooth L_λ there exist hardness results for deterministic algorithms to efficiently achieve (δ,ε)-stationary points (Kornowski and Shamir, 2021). These worst-case lower bounds apply to deterministic black-box methods and do not preclude the behavior observed with stochastic optimizers used in practice. However, here we treat Eq. (2) as an idealization when taking $\varepsilon \to 0, \delta \to 0$.

This type of alignment was first studied by (Ji and Telgarsky, 2019a, 2020) in the context of training with gradient flow, and under the extra condition that $L(\theta_0) < \ell(0)$. (They also show alignment for GD for deep linear networks using an adaptive step-size and under the extra conditions that the initialization has sufficiently small loss.) Here, weight decay enables a simpler argument to establish alignment. Next, we show several implications of this result.

[3] When L_λ is non-smooth (for example when activation function is ReLU), ∇ is understood the Clarke subgradient. In this case, formally $\lambda\theta = -g$ for $g \in \partial L_\lambda(\theta)$.

Lemma 1 (Deep linear networks). *Let f_θ be a deep linear network, and θ be a parameter vector satisfying the alignment condition of Eq. (2). Then all weight matrices $W_1, \dots, W_{K-1}$ are rank-1.*

Lemma 2 (Norm preservation). *Let θ be a parameter vector satisfying the alignment condition of Eq. (2). Further, assume that f_θ is locally Lipschitz and with H-positively homogeneous activations. Then weight matrices have Frobenius norm:*

$$\lambda \|W_k\|_F^2 = -\frac{H^{K-k}}{n} \sum_i \ell'(y_i f_\theta(x_i)) y_i f_\theta(x_i) \,.$$

A deep neural network with non-linear activations can satisfy the above conditions, for instance in case of ReLU $H = 1$.

Lemma 3 (Low-rank structure). *Consider logistic ($\ell(x) = \ln(1 + e^{-x})$) loss function. Assume activations are H-positive homogeneous and satisfy $\|\sigma(v)\| \le \|v\|^H$. Assume that $K \ge 2, H \ge 1$. For any $\lambda > 0$ such that θ is a non-zero stationary point of L_λ we have that*

$$\sum_k \frac{(H^{K-k})^{3/2}}{Z} \cdot \frac{\|W_k\|_2}{\|W_k\|_F} \ge \sqrt{\lambda}\, L(\theta)^{-(\frac{1}{4}+\frac{1}{2Z})}$$

where $Z = \sum_k H^{K-k}$. Note that $Z = K$ for $H = 1$ and $Z = (H^K - 1)/(H - 1)$ otherwise.

The above result implies that the average stable rank decreases as λ increases. In particular, to show that the lower bound increases with λ it is enough to upper bound $L(\theta)$.

Corollary 1. *Assume that $L_\lambda(\theta) \le L_\lambda(\theta_0)$ for some fixed θ_0 and that $L(\theta_0) \ge 1$. Then, under conditions of Lemma 3*[4]

$$\sum_k \frac{(H^{K-k})^{3/2}}{Z} \cdot \frac{\|W_k\|_2}{\|W_k\|_F} \ge \sqrt{\frac{\lambda}{L(\theta_0) + \lambda \|\theta_0\|^2}} \,.$$

For instance, when θ is a global minimizer of L_λ, it is easy to verify that $L(\theta) \le L_\lambda(\theta) \le L_\lambda(0) \le 1$, and so the lower bound implied by Lemma 3 becomes $\sqrt{\lambda}$ for $\lambda \le C_{H,K}$.[5]

Another very basic example is when θ_0 is initialization of some optimization algorithm which does not increase the objective (e.g. such as GD). In this case, the lower bound depends on width and depth, e.g. $\|\theta_0\|^2 \sim Km$ for Xavier initialization.

[4]Note that the lower bound holds since $L(\theta_0) \ge 1$ by assumption and since for $H \ge 1, K \ge 2$, we have $1/Z = (H-1)/(H^K - 1) \le 1/K \le 1/2$.

[5]Where $C_{H,K} = (\sum_k (H^{K-k})^{3/2}/Z)^2$.

Remark 2. *LHS in Lemma 3 is bounded by a constant whereas RHS is λ-dependent. One can imagine that by taking arbitrarily large λ on the RHS we could achieve inconsistency. This cannot happen because permissible range of λ is restricted by a 'non-zero θ' condition. Indeed, the following sketch argues that the unique global solution of L_λ for large enough λ must be $\theta = 0$.*

For simplicity consider the case $H = 1$ (ReLU or linear activation). By AM-GM inequality $|f_\theta(x)| \le \|\theta\|^K K^{-K/2}$ (see the first step of the proof of Lemma 3) and consider the proxy to L_λ, which employs the upper bound on $|f_\theta(x)|$:

$$g_\pm(\|\theta\|) = \frac{\lambda}{2}\|\theta\|^2 + \ln(1 + \exp(\pm \|\theta\|^K K^{-K/2}))$$

which 'sandwiches' L_λ. It is not hard to check that for $K \ge 2$, and $\lambda > C_{H,K}$ (constant that depends only on K and H), $g'(x) = 0$ has one solution which is 0. Note that this is specific for multilayer neural networks as this phenomenon does not occur with $K = 1$

The intuition is that the power K in $\|\theta\|^K$ makes the loss function concentrate around zero (because $\cdot^K$ makes predictions larger and logistic loss becomes tiny), but in the vicinity of zero ℓ_2 penalty is stronger and the only surviving global minimum is the one of the ℓ_2 penalty.

To this end, Timor et al. (2023) show low-rank structures for the global optimum of the minimum-rank interpolating solution assuming existence of a smaller "teacher" network with $K' < K$ layers that interpolates data and Frobenius norm of its weight matrices are bounded by C. More specifically, they show that $(1/K)\sum_k \|W_k\|_2/\|W_k\|_F \ge (1/C)^{K'/K}$. We can have an intuitive understanding of their result by noting that given a smaller interpolating network, the last layer of the network tends to be low-rank due to the *Neural Collapse* phenomena (Papyan et al., 2020). Then if we add more layers, the additional layers will remain low-rank as their activations will lie in a low-rank structure, and $(1/K)\sum_k \|W_k\|_2/\|W_k\|_F$ will approach one as we add more layers.

Results in Lemma 1 and Lemma 3 show a different phenomena. Remarkably, in both these results, irrespective of network capacity or performance, and as a consequence of weight decay, the weight matrices will become low-rank on average. Another interesting feature is that, unlike results of Timor et al. (2023), depth plays a minor role in establishing low-rank structures in Lemma 1 and Lemma 3. In Section 5, we present experiments that show that even if number of layers is small and network has a large number of classification mistakes, the average inverse stable-rank grows as weight decay parameter λ increases.

Finally, low-rank results we showed here also to some extent apply to modern architectures such as attention heads, which are fundamental building blocks of transformers (Vaswani et al., 2017). In particular, if we replace softmax with argmax, and remove residual connections and layer normalization, we will obtain a *homogeneous attention head*, and Lemmas 2 and 3 hold for such models. Transformers used in practice typically include residual connections, layer norms, softmax attention, and so on, that deviate from the above conditions. Yet, as we show empirically in Section 5, the conclusion of the above result holds to some extent.

4 MERGING PARAMETERS

Throughout this section, in addition to the training tuple $(x_i, y_i)_{i=1}^n$ we introduce $(x'_i, y'_i)_{i=1}^n$, and in addition to the regularized objective L_λ we will introduce L'_λ defined with respect to the second training set. As a warm-up example we first consider a linear prediction scenario where we are looking at predictors of a form $f_\theta(x) = \langle \theta, x \rangle$ and suppose that θ_t is obtained by GD minimizing L_λ, while θ'_t is obtained by minimizing L'_λ. Now given a test point x that is sufficiently different from inputs $(x'_i)_{i=1}^n$, or assuming that $\max_i |\langle x'_i, x \rangle| \le \varepsilon$ we have

$$f_{\theta_t + \theta'_t}(x) \approx f_{\theta_t}(x)$$

and the same holds for some point x' sufficiently different from $(x_i)_{i=1}^n$. This is derived in a straightforward way based on GD update rule $\theta_{t+1} = (1 - \lambda\eta)\theta_t - (\eta/n)\sum_{i=1}^n \ell'(y_i \langle \theta_t, x_i \rangle) x_i y_i$ while unrolling the recursive relationship we get

$$\theta_t = \theta_0 (1 - \eta\lambda)^t + \sum_{i=1}^n \alpha_i x_i$$
$$\text{where} \quad \alpha_i = \frac{1}{n} \sum_{s=0}^{t-1} \eta (1 - \eta\lambda)^{t-s-1} \ell'(y_i \langle \theta_s, x_i \rangle) y_i \ .$$

The above identity tells us that the solution θ_t resides in the span of inputs. It is easy to see that prediction on the inputs from the other task is close to ε,

$$|\langle \theta_t, x' \rangle| \le |\langle \theta_0, x' \rangle| (1 - \eta\lambda)^t + \varepsilon \frac{1 - (1 - \eta\lambda)^t}{\lambda}$$

for 1-Lipschitz loss function. In particular, the above implies that the gap of interest $|f_{\theta_t + \theta'_t}(x) - f_{\theta_t}(x)|$ is also controlled by the upper bound in the display above. Note that a possibly large term $|\langle \theta_0, x' \rangle|$ is attenuated exponentially quickly by the weight decay, so its effect is negligible at the end of optimization. While in the linear case we could set $\theta_0 = 0$, in case of neural network learning setting initialization at 0 is atypical and therefore weight decay seems to have an important role in such scenarios. Another summand on the right hand side is ε-dependent and captures similarity between inputs from different tasks. If inputs are orthogonal this term disappears.

Shallow ReLU networks Here we consider a shallow ReLU neural network

$$f_\theta(x) = \langle u, (Wx)_+ \rangle \qquad (x \in \mathbb{R}^d) \tag{3}$$

where hidden weight matrix $W \in \mathbb{R}^{m \times d}$ is a tunable parameters, and u with $\|u\| \le 1$ is fixed throughout training. In this scenario we obtain a merged predictor by simply summing hidden weight matrices of neural networks trained on different tasks. The intuition behind the argument in this case is that for an input x, the length $\|W'_t x\|$ must be small because rows of W'_t lie in the span of $(x'_i)_i$ meanwhile each of these points is sufficiently different from x. We formalize this in the following lemma, shown in Section C.1, which applies to GD iterates:

Lemma 4. *Suppose that inputs $(x_i)_i$ and x' are such that $\max_i |\langle x_i, x' \rangle| \le \varepsilon$. Suppose that W_t is a weight matrix of a shallow neural network obtained by running GD for t steps given $(x_i, y_i)_{i=1}^n$. Assume that the loss function is 1-Lipschitz (e.g. logistic loss). Then, for any $j \in [m], t \in \mathbb{N}$,*

$$\|W_t x'\| \le \|W_0 x'\| (1 - \eta\lambda)^t + \varepsilon \frac{1 - (1 - \eta\lambda)^t}{\lambda}$$

which implies

$$|f_{\theta_t + \theta'_t}(x) - f_{\theta_t}(x)| \le \|W'_0 x\| (1 - \eta\lambda)^t + \varepsilon \frac{1 - (1 - \eta\lambda)^t}{\lambda} \ .$$

Note that the bound requires only approximate orthogonality of the *inputs*, not of the hidden features: since rows of W_t lie in the span of $(x_i)_i$, it is sufficient that the test point x' is nearly orthogonal to those inputs.

Deep linear networks Next, we consider the predictor

$$f_\theta(x) = \langle w_K, W_{K-1} \cdots W_1 x \rangle$$

where each weight matrix is trained by GF dynamics as described in Eq. (1). To show the desired result we exploit a technical result of Arora et al. (2018) which translates GF dynamics for individual matrices into *implicit* GF dynamics for the *end-to-end* vector $w(t)^\top = w_K(t)^\top W_{K-1}(t) \cdots W_1(t)$. The proof, given in Section C.3, exploits the fact that $w(t)$ indeed lies in the span of inputs.

Theorem 1. *Assume that weight matrices are initialized such that for all* $k \in [K-1]$

$$W_{k+1}(0)^\top W_{k+1}(0) = W_k(0) W_k(0)^\top \ .$$

Then, for any $t \geq 0$ *and any input* x *such that* $\max_i |\langle x_i', x\rangle| \leq \varepsilon$ *we have*

$$|f_{\theta(t)+\theta'(t)}(x) - f_{\theta(t)}(x)| \leq |f_{\theta(0)}(x)| A_1 \, e^{-\lambda K t} + A_2 \, \varepsilon$$

where terms A_1, A_2, B *depend only on initialization,* λ*, and* K *(see Theorem 2 for precise constants).*

Theorem 1 gives the bound on the gap of the same form as in the shallow and linear case, that is an exponentially decaying term that arises because of the weight decay, and an ε-dependent term which captures similarity of inputs in different tasks. Unlike the previous cases, the theorem assumes a particular initialization, which is inherited from Arora et al. (2018), which is benign and is satisfied with high probability when entries of weight matrices are sampled from some symmetric distribution with equal layer widths (e.g. isotropic Gaussian). In expectation, standard He or Xavier initialization also satisfies this condition when all hidden layers have the same width; in practice a single realization may deviate, and large imbalances can slow convergence (Min et al., 2021).

5 EXPERIMENTS

Norm and rank structure In this section we aim to investigate whether some of the biases discussed in Section 3 extend to very deep neural networks. In particular, we aim to verify whether norm preservation and low stable-ranks can be found in some (smaller) LLMs. Even though assumptions of Section 3 might not be satisfied, we still find that in several LLMs we observe low stable-ranks (drastically smaller than the dimension of the matrix), while norm preservation appears in most of the layers (MLP and attention ones). In these experiments we look at publicly available pretrained BERT with $\approx$ 110M parameters, GPT2-Large with $\approx$ 774M , GPT2-XL with $\approx$ 1.5B , RoBERTa with $\approx$ 125M , Phi-2 with $\approx$ 2.8B , GPT-J with $\approx$ 6B (we used Hugging Face "Transformers" library (Wolf et al., 2020)). All experiments are performed on a computing platform with Nvidia A100 GPU.

Each of these 'transformer' models consists of a sequence of a so-called *encoder* layers, where each encoder layer consists of a *self-attention layer* followed by a fully-connected neural network. Self-attention layer is given by a matrix-to-matrix function $Q \mapsto \mathrm{softmax}\big(QK^T/\sqrt{\mathrm{columns}(K)}\big)V$ where (Q, V) are parameter matrices and softmax is taken row-wise. Each self-attention layer is followed by a feed-forward fully connected neural network consisting of two linear layers with a non-linear activation function (e.g., ReLU or a smooth activation function) in between. In the context of transformer architecture, Q, K, V are known as query, key, and value matrices. In Section D we provide a table Table 1 that summarizes architectural details of these models.

While some of the results are given in Section D, here in Figure 1 we provide results for two largest neural networks. In these plots, Q, K, V, O, and QKV denote Query, Key, Value, Output, and the concatenation of Query, Key, Value matrices of the attention layer, respectively. Weight matrices of MLP layers are denoted by M1 and M2.

The plots show that the Frobenius norms of QKV, M1, and M2 matrices are generally in the same order. Although this might appear puzzling at first, it is in fact consistent with Lemma 2: the lemma is a statement about norms of layer weights when the output of the network can be written as a product of those layers. To apply the lemma to a transformer architecture, we should consider the QKV matrix as a layer weight instead of the individual attention matrices. All weight matrices generally have small stable-ranks, although the values can be different for different parameter type and layer.

There are two observations for which we currently have no explanation: (1) The Frobenius norms of the same parameter type (even Q, K, V matrices) remain largely unchanged across layers. (2) Even though Lemma 2 suggests the O matrix should have similar Frobenius norms as QKV, M1, and M2 matrices, the plots show quite different values for norms of O matrices. Finally, Figure 2 shows that even if number of layers is small and network has a large number of classification mistakes, the average inverse stable-rank grows as weight decay parameter λ increases. In this experiment, 1000 inputs are drawn independently from $\mathcal{N}(0, I_d)$ with $d = 100$ and then projected onto a unit sphere. Given an input x_i, the label $y_i \in \{-1, +1\}$ is generated as $y_i = 2 \cdot \mathbf{1}\left(1/(1+e^{-f^\star(x)}) \geq 1/2\right) - 1$ where the labeling function is $f^\star(x) = \sin(10 W^\top x)$ (left plot) or $f^\star(x) = \sin(100 W^\top x)$ (right plot), and vector W is drawn from $\mathcal{N}(0, I_d)$ and fixed throughout the experiment. For each value of λ, we use SGD with weight decay to train a network with two hidden layers (so, $K = 3$), each of width 10. The number of epochs is 5000. Figure 2 shows the average inverse stable-rank of the final solution, along with its classification error, margin error (number of points with $y_i f_\theta(x_i) < 1$), and average loss. The plot shows that the average inverse stable-rank increases as λ increases, even though the network might have large errors.

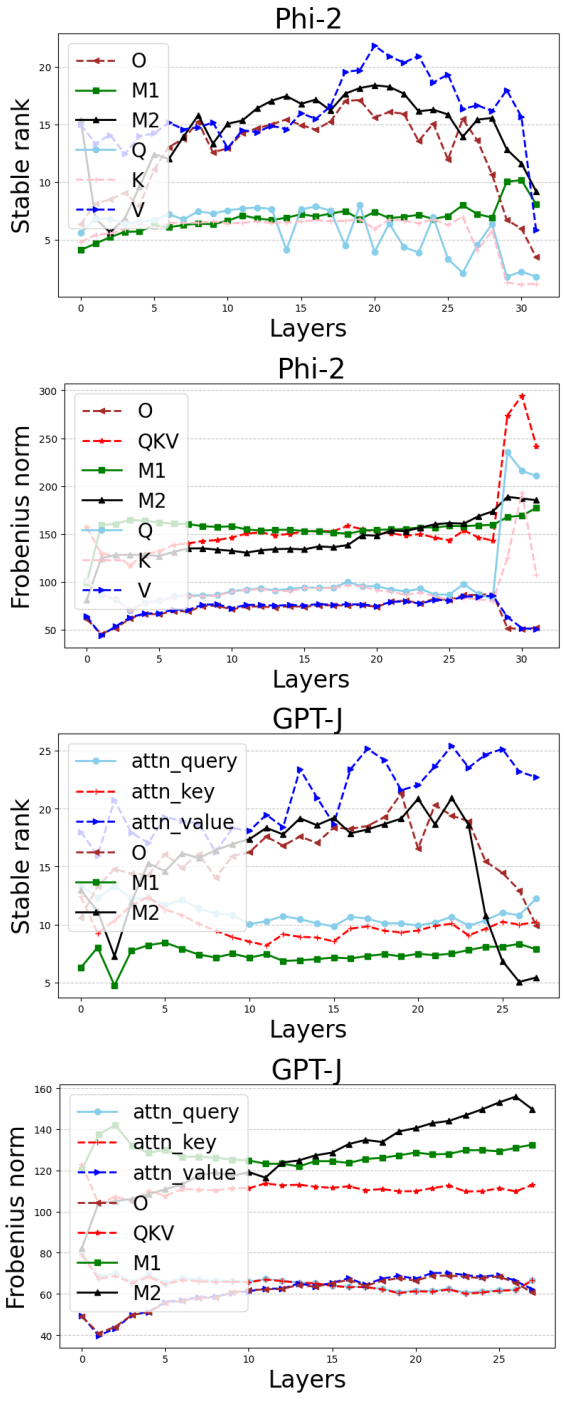

Figure 1: Stable ranks and Frobenius norms of different weight matrices in pretrained Phi-2 (first row) and GPT-J (second row) models.

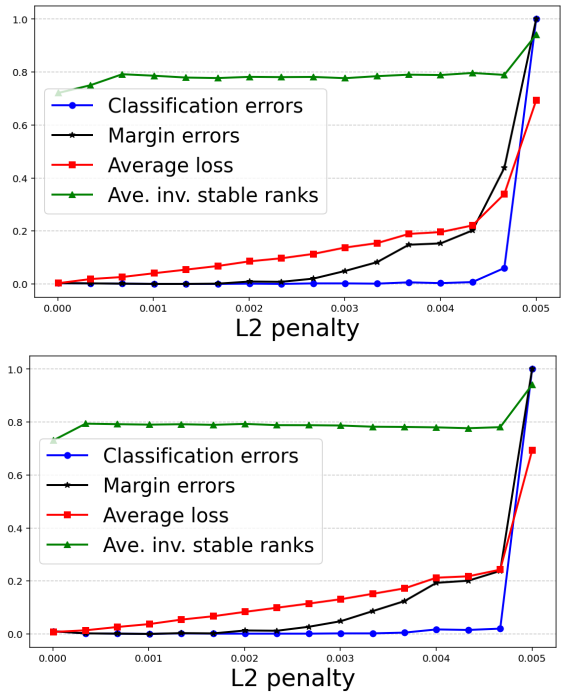

Figure 2: Low rank induced by weight decay.

Model merging In our experiments we aim to verify four hypotheses: (1) Training two neural networks on *different tasks* (with nearly orthogonal inputs), and summing the weights results in a combined neural network that performs nearly as well on each of the tasks. Here the performance is understood in terms of the training loss; (2) In contrast, training two neural networks on the *same task* and performing the merging as discussed above leads to the combined predictor that performs poorly on both tasks; (3) This behavior is enabled by the weight decay; (4) Using weight decay leads to a low stable rank of weight matrices.

Data. The first set of experiments is performed on synthetic data, constructed as follows: In case of independent tasks, inputs for the first task are drawn from isotropic Gaussian $\mathcal{N}(0, \Sigma_1)$, while for the second task inputs are drawn from $\mathcal{N}(0, \Sigma_2)$ where $\langle \Sigma_1, \Sigma_2 \rangle = 0$ (this corresponds to scenario $\varepsilon = 0$ in Section 4). Given an input x_i, the label $y_i \in \{-1, +1\}$ is generated as $y_i = 2\,\mathbf{1}\left(1/(1 + e^{-f^\star(x)}) \geq 1/2\right) - 1$ where the labeling function $f^\star(x) = \sin(W^\top x)$. For each task, vector W is drawn from $\mathcal{N}(0, I_d)$ and fixed throughout the experiment. In all experiments in this section inputs are normalized to lie on a unit sphere. We achieve similar observations also in case of a real data, see Section D.2.

Model and training. In all the experiments Fully-connected ReLU neural network with inputs $d = 100$, two hidden layers of sizes $(1000, 100)$, and a scalar output.In each experiment on synthetic data, models are

trained by GD with step size $\eta = 1$ over 10^5 steps. In all the experiments excepts the one where weight decay varies, weight decay parameter λ is set to 10^{-4}. On Fashion MNIST dataset, step size is set as $\eta = 0.1$ while $\lambda = 10^{-3}$. All experiments are repeated on 10 random draws of the sample, and we report standard deviations in all plots.

Discussion. On both datasets, we consistently observe that summing weight matrices originating from different tasks, enables small logistic loss for merged models trained on all tasks after sufficiently long training (first row in Figure 3). This seems to be in part enabled by orthogonality of inputs since, in contrast, when inputs are not orthogonal and labels (or labeling functions) are different, merged models do not get close in performance to distinct task-specific models. Another component that enables this gap is weight decay: In the second row of Figure 3 we observe that when weight decay strength is not sufficient, the gap between losses of merged and original models is substantial. Finally, we observe that the stable rank converges to a value much smaller than the actual rank at the stage when performance of merged model actually gets close to the performance of original model. This is another observation in support of our hypothesis that low-rank bias is crucial for model merging.

In this paper we focus on the training error, but one might wonder whether similar message about model merging carries over to *generalization*, or preserving performance on the unseen but similarly distributed data. In Section D.2 we include experimental results confirming that model merging not only retains training errors of the original models, but also preserves their generalization ability.

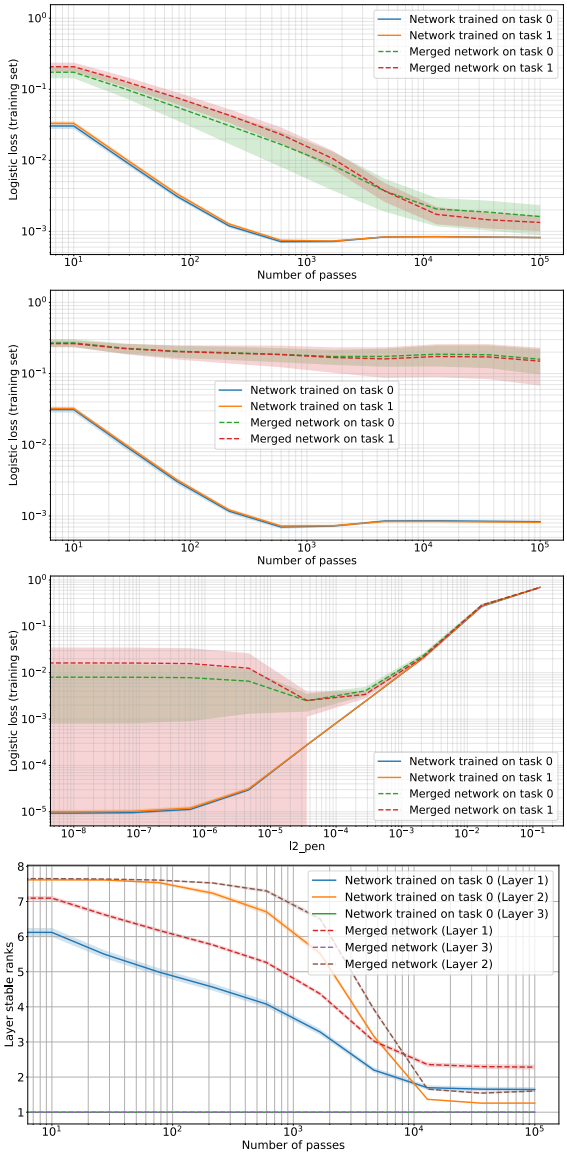

Figure 3: First and second plot: Training neural networks on different tasks (orthogonal inputs) (left) vs. the same task (right) and merging the parameters by adding weight matrices. The resulting network performs well on different tasks after sufficiently many iterations, while given the same task, it does not. Third and fourth plot: this effect manifests when weight decay strength is sufficiently large (left). Stable rank of each weight matrix converges to a small value, while stable rank of the merged network matches stable ranks of individual networks (right).

6 CONCLUSIONS

In this work we examined the role of $L2$ regularization in training of deep neural networks with logistic loss. We investigated a surprising phenomenon: merging two neural networks trained on sufficiently different tasks by simply adding their respective weight matrices results in a predictor that performs well on both tasks simultaneously. As we attributed the explanation of this to the low-rank bias arising in weight matrices, we also established that $L2$ regularization leads to weight matrices of a low stable rank.

These observations open up some interesting possibilities, especially in multitask learning with large models, such as large language models, and distributed optimization.

References

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learing (ICML)*, 2018.

Etienne Boursier, Loucas Pillaud-Vivien, and Nicolas Flammarion. Gradient flow dynamics of shallow relu networks for square loss and orthogonal inputs. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Ke Chen, Chugang Yi, and Haizhao Yang. Towards better generalization: Weight decay induces low-rank bias for neural networks. arXiv 2410.02176.

Woojin Chung, Hyowon Cho, James Thorne, and Se-Young Yun. Parameter averaging laws for multitask language models. In *International Workshop on Federated Learning in the Age of Foundation Models in Conjunction with NeurIPS 2023*, 2023.

Alexandre Défossez, Léon Bottou, Francis R Bach, and Nicolas Usunier. A simple convergence proof of adam and adagrad. *Transcations of Machine Learning Research*, 2022.

Arthur Douillard, Qixuan Feng, Andrei Alex Rusu, Rachita Chhaparia, Yani Donchev, Adhiguna Kuncoro, MarcAurelio Ranzato, Arthur Szlam, and Jiajun Shen. Diloco: Distributed low-communication training of language models. In *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ ICML 2024)*, 2024.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learing (ICML)*, 2020.

Spencer Frei, Gal Vardi, Peter Bartlett, Nathan Srebro, and Wei Hu. Implicit bias in leaky relu networks trained on high-dimensional data. In *International Conference on Learning Representations (ICLR)*, 2023.

Tomer Galanti, Zachary S. Siegel, Aparna Gupte, and Tomaso Poggio. Sgd and weight decay secretly minimize the rank of your neural network. In *Conference on Parsimony and Learning*, 2025.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *Uncertainty in Artificial Intelligence (UAI)*, 2018.

Arthur Jacot. Implicit bias of large depth networks: a notion of rank for nonlinear functions. In *International Conference on Learning Representations (ICLR)*, 2023.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: convergence and generalization in neural networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 8580–8589, 2018.

Arthur Jacot, François Ged, Berfin Şimşek, Clément Hongler, and Franck Gabriel. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity, 2026.

Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. In *International Conference on Learning Representations (ICLR)*, 2019a.

Ziwei Ji and Matus Telgarsky. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow relu networks. *International Conference on Learning Representations (ICLR)*, 2019b.

Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Guy Kornowski and Ohad Shamir. Oracle complexity in nonsmooth nonconvex optimization. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Haochuan Li, Alexander Rakhlin, and Ali Jadbabaie. Convergence of adam under relaxed assumptions. *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Rami Luisto. A short survey on almost orthogonal vectors in a few specific large dimensions. *arXiv 2510.23609*, 2025.

Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks.

In *International Conference on Machine Learing (ICML)*, 2020.

Hancheng Min, Salma Tarmoun, René Vidal, and Enrique Mallada. On the explicit role of initialization on the convergence and implicit bias of overparametrized linear networks. In *International Conference on Machine Learing (ICML)*, 2021.

Hancheng Min, Enrique Mallada, and Rene Vidal. Early neuron alignment in two-layer relu networks with small initialization. In *International Conference on Learning Representations (ICLR)*, 2024.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

Mary Phuong and Christoph H Lampert. The inductive bias of relu networks on orthogonally separable data. In *International Conference on Learning Representations (ICLR)*, 2021.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Alexandre Rame, Guillaume Couairon, Mustafa Shukor, Corentin Dancette, Jean-Baptiste Gaya, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Alexandre Ram'e, Nino Vieillard, L'eonard Hussenot, Robert Dadashi, Geoffrey Cideron, Olivier Bachem, and Johan Ferret. Warm: On the benefits of weight averaged reward models. In *International Conference on Learning Representations (ICLR)*, 2024.

Alexandre Ramé, Johan Ferret, Nino Vieillard, Robert Dadashi, Léonard Hussenot, Pierre-Louis Cedoz, Pier Giuseppe Sessa, Sertan Girgin, Arthur Douillard, and Olivier Bachem. Warp: On the benefits of weight averaged rewarded policies. arXiv, 2024.

Akshay Rangamani and Andrzej Banburski-Fahey. Neural collapse in deep homogeneous classifiers and the role of weight decay. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

Zafir Stojanovski, Karsten Roth, and Zeynep Akata. Momentum-based weight interpolation of strong zero-shot models for continual learning. In *NeurIPS Workshop*, 2022.

Matus Telgarsky. Deep learning theory lecture notes. https://mjt.cs.illinois.edu/dlt/, 2021. Version: 2021-10-27 v0.0-e7150f2d (alpha).

Nadav Timor, Gal Vardi, and Ohad Shamir. Implicit regularization towards rank minimization in relu networks. In *Algorithmic Learning Theory (ALT)*, 2023.

Joachim Utans. Weight averaging for neural networks and local resampling schemes. In *Conference on Artificial Intelligence (AAAI)*, 1996.

Gal Vardi. On the implicit bias in deep-learning algorithms. *Communications of the ACM*, 66(6):86–93, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, 2020.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael GontijoLopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learing (ICML)*, 2022.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv, 2017.

Yibo Yang, Shixiang Chen, Xiangtai Li, Liang Xie, Zhouchen Lin, and Dacheng Tao. Inducing neural collapse in imbalanced learning: Do we really need a learnable classifier at the end of deep neural network? In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive approximation*, 26(2):289–315, 2007.

David Yunis, Kumar Kshitij Patel, Samuel Wheeler, Pedro Savarese, Gal Vardi, Karen Livescu, Michael Maire, and Matthew R. Walter. Approaching deep learning through the spectral dynamics of weightsf. arXiv 2408.11804, 2024.

Siqi Zeng, Yifei He, Meitong Liu, Weiqiu You, Yifan Hao, Yao-Hung Hubert Tsai, Makoto Yamada, and Han Zhao. Task vector bases: A unified and scalable framework for compressed task arithmetic. arXiv 2502.01015, 2025.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] (Sections 2 to 4)
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] (Sections 3 and 4)
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No] Specification is provided. Source code sharing is prohibited by the policy of authors' organization.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] (Sections 3 and 4)
   (b) Complete proofs of all theoretical results. [Yes] (Sections B and C)
   (c) Clear explanations of any assumptions. [Yes] (Sections 2 to 4)

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] (Section 5)
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] (Section 5)
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes] (Section 5)
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] (Section 5)

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Low-rank bias, weight decay, and model merging in neural networks: Supplementary Materials

A Additional related work

A number of prior works have studied low-rank biases of neural networks in settings that are more restrictive than ours.

Jacot et al. (2026) study low-rank biases of deep linear networks trained with GF and with near-zero variance initialization. In contrast, we consider deep linear networks optimized by GD, and we have no requirement on near-zero variance initialization. Jacot (2023) study the low-rank phenomena in infinite-depth ReLU networks, while we study a finite-depth setting. Galanti et al. (2025) show low-rank updates due to a small batch size. But these low-rank updates can eventually lead to a high-rank final solution if they are in different directions. Finally, Yunis et al. (2024) empirically observe that a low-rank solution develops with weight-decay, and stable rank generally decreases during training.

A number of works have considered shallow networks. Phuong and Lampert (2021) show convergence of a hidden weight matrix in a shallow neural network to a rank-one matrix when it is trained by GF on *orthogonally-separable data* (in the context of classification all inputs with a matching label satisfy $\langle x_i, x_j \rangle > 0$, and otherwise for a non-matching one). Min et al. (2024) show convergence of stable rank to 2 for shallow ReLU networks trained by GF dynamics. Under mild assumptions on the inputs Frei et al. (2023) established that the hidden weight matrix of a shallow neural network convergences to the low-rank matrix (at most 2). Boursier et al. (2022) study implicit bias of shallow networks (under orthogonal inputs) trained with GF and with small initialization. Chen et al. show that for a two-layer ReLU network, training with SGD, weight decay, and minibatch leads to an approximately rank-two weight matrix. Our proof technique is different and is applicable to more general networks and the full-batch setting as well.

Weight averaging and task arithmetic Several prior works have considered building a final model by averaging weights of different models (Utans, 1996; Izmailov et al., 2018; Rame et al., 2022; Wortsman et al., 2022; Stojanovski et al., 2022; Ilharco et al., 2022; Chung et al., 2023; Douillard et al., 2024; Rame et al., 2023; Ram'e et al., 2024; Ramé et al., 2024). These works mostly consider settings where models are trained on the same or similar tasks. Their main idea is that randomization in data or parameter initialization leads to perturbations in the learned weights, and averaging leads to more stable solutions, a phenomenon known as "linear mode connectivity" (Frankle et al., 2020; Neyshabur et al., 2020). In contrast, we consider models trained on entirely different tasks and take the weights' sum instead of their average.

Closely related to the present work is Task Arithmetic (Ilharco et al., 2023), which builds *task vectors* $\tau_t = \theta_t - \theta_0$ by subtracting a shared pre-trained initialization θ_0 from the fine-tuned weights θ_t, and combines tasks by adding task vectors. Unlike weight averaging, Task Arithmetic operates on models trained on *different* tasks and uses weight summation rather than averaging. While Ilharco et al. (2023) empirically demonstrate that this succeeds for disjoint tasks, they do not provide a theoretical explanation. Our work offers such an explanation: $L2$ regularization confines each task's learned weights to the span of its training inputs, so that models trained on nearly orthogonal tasks do not interfere upon summation. Zeng et al. (2025) extend Task Arithmetic to compressed task-vector representations and provide error bounds for multi-task generalization under orthogonality conditions; their analysis is complementary to ours.

Neural Collapse Low-rank bias is intimately related to a so-called *Neural Collapse (NC)*, which is a form of clustering within the feature space of a neural network (Papyan et al., 2020). NC hypothesis suggests that the network refines its representations so that inputs belonging to the same class (in context of classification) are pulled closer together, forming distinct clusters. Simultaneously, the network pushes the cluster centers (class means) away from each other, maximizing their separation.

Some of the existing results are mostly in the Unconstrained Feature Model (UFM) where only the last-layer features and the output of linear classifier are learnable. This is in fact equivalent to learning a shallow linear network. Most studies show that the global optima of the loss function satisfies the conditions of neural collapse, without studying the dynamics of gradient descent.

Yang et al. (2022) demonstrated that in the Universal Feature Manifold (UFM) setting, fixing the linear output layer to satisfy a specific condition induces neural collapse in the feature vectors that minimize the loss, even with imbalanced data. This fixed output layer approach allows them to extend typical neural collapse results, which often assume balanced data, to the more challenging imbalanced case.

Rangamani and Banburski-Fahey (2022) showed that for ReLU deep networks trained on balanced datasets using gradient flow with weight decay and a squared loss, critical points satisfying a "Symmetric Quasi-interpolation" assumption also exhibit neural collapse. This assumption, which posits the existence of a classifier whose output depends only on the class label and not the specific data point, is presented as a key condition for their result.

B Proofs from Section 3

Proof of Lemma 1. Let $w^{(i)\top} = w_K^\top W_{K-1} \dots W_i$. Therefore, $f_\theta(x) = \langle w^{(i)}, W_{i-1} \dots W_1 x \rangle$. Let $r_i(\theta) := -y_i \ell'(y_i f_\theta(x_i))/(n\lambda)$. By alignment, we have

$$
\begin{aligned}
W_1 &= w^{(2)} \sum_{i=1}^n r_i(\theta) x_i^\top \,, \\
W_2 &= w^{(3)} \sum_{i=1}^n r_i(\theta) (W_1 x_i)^\top \,, \\
&\vdots \\
W_{K-1} &= w^{(K)} \sum_{i=1}^n r_i(\theta) (W_{K-2} \dots W_1 x_i)^\top \,, \\
w_K &= \sum_{i=1}^n r_i(\theta) (W_{K-1} \dots W_1 x_i)^\top \,.
\end{aligned}
$$

It's easy to see that all weight matrices above are rank-1. □

Proof of Lemma 2. Eq. (2) holds also for the parameters of layer k: $\lambda W_k = -\nabla_{W_k} L(\theta)$. Therefore,

$$
\begin{aligned}
\lambda \|W_k\|_F^2 &= \lambda \operatorname{tr}(W_k W_k^\top) \\
&= -\operatorname{tr}(\nabla_{W_k} L(\theta) W_k^\top) \\
&= -\frac{1}{n} \sum_{i=1}^n \ell'(y_i f_\theta(x_i)) y_i \operatorname{tr}(\nabla_{W_k} f_\theta(x_i) W_k^\top) \\
&= -\frac{H^{K-k}}{n} \sum_{i=1}^n \ell'(y_i f_\theta(x_i)) y_i f_\theta(x_i) \,,
\end{aligned}
$$

where the last step holds by the fact that for f_θ locally Lipschitz and positively homogeneous, $H^{K-k} f_\theta(x) = \operatorname{tr}(\nabla_{W_k} f_\theta(x_i) W_k^\top)$ (see e.g. Lemma 9.2 of Telgarsky (2021)). Given that the equation holds independently of layer index k, weight matrices all have the same Frobenius norm. □

Proof of Lemma 3. By the weighted AM-GM inequality, and the fact that activations are H-homogeneous, we get that

$$
|f_\theta(x)| \le \prod_k \|W_k\|_2^{H^{K-k}} \le \left(\frac{1}{Z} \sum_k^K H^{K-k} \|W_k\|_2 \right)^Z \,.
$$

assuming that $\|x\| \le 1$.

On the other hand, given that the loss is logistic, from Lemma 2 we have that,

$$
\begin{aligned}
\lambda \|W_{k'}\|_F^2 &= -\frac{H^{K-k'}}{n} \sum_i \ell'(y_i f_\theta(x_i)) y_i f_\theta(x_i) \\
&\le \frac{H^{K-k'}}{n} \sum_i |\ell'(y_i f_\theta(x_i))| \cdot \max_j |f_\theta(x_j)| \\
&\le H^{K-k'} L(\theta) \cdot \max_j |f_\theta(x_j)| \\
&\le H^{K-k'} L(\theta) \left(\frac{1}{Z} \sum_k^K H^{K-k} \|W_k\|_2 \right)^Z \,.
\end{aligned}
$$

Introduce $B^2 = (-1/n)\sum_i \ell'(y_i f_\theta(x_i)) y_i f_\theta(x_i)$, and so by Lemma 2 $\|W_k\|_F = \sqrt{H^{K-k}B^2/\lambda}$. Therefore,

$$\begin{aligned}
\frac{1}{Z}\sum_k (H^{K-k})^{3/2}\frac{\|W_k\|_2}{\|W_k\|_F} &= \frac{1}{Z}\sum_k (H^{K-k})^{3/2}\frac{\sqrt{\lambda}\|W_k\|_2}{\sqrt{H^{K-k}B^2}} \\
&= \frac{\sqrt{\lambda}}{B}\cdot\frac{1}{Z}\sum_k H^{K-k}\|W_k\|_2 \\
&\geq \frac{\sqrt{\lambda}}{B}\cdot\left(\frac{\lambda\|W_k\|_F^2}{H^{K-k}L(\theta)}\right)^{1/Z} && \text{(For any } k\text{)} \\
&= \frac{\sqrt{\lambda}}{B}\cdot\left(\frac{H^{K-k}B^2}{H^{K-k}L(\theta)}\right)^{1/Z} \\
&= \sqrt{\lambda}\cdot\frac{B^{2/Z-1}}{L(\theta)^{1/Z}} \\
&\geq \sqrt{\lambda}\cdot\frac{L(\theta)^{(2/Z-1)/4}}{L(\theta)^{1/Z}} \\
&= \sqrt{\lambda}L(\theta)^{-(\frac{1}{4}+\frac{1}{2Z})}
\end{aligned}$$

where the last inequality requires

$$B^2 = -\frac{1}{n}\sum_i \ell'(y_i f_\theta(x_i)) y_i f_\theta(x_i) \leq \sqrt{L(\theta)}$$

using the fact that $-x\ell'(x) \leq \sqrt{\ell(x)}$ for the logistic loss, and Jensen's inequality. In particular, assuming that for $K \geq 2, H \geq 1$, we have $2/Z - 1 \leq 0$, and so $B \mapsto B^{2/Z-1}$ is non-increasing and we have $B^{2/Z-1} \geq (\sqrt{L(\theta)})^{(2/Z-1)/2}$. □

C Proofs from Section 4

C.1 Proof of Lemma 4

Observe that GD update rule is

$$W_{s+1} = (1-\eta\lambda)W_s - \eta\,\frac{1}{n}\sum_{i=1}^{n} y_i\,\ell'(y_i\,f_{\theta_s}(x_i))\,D_{s,i}\,u\,x_i^\top$$

where $D_{s,i} = \mathrm{diag}(\mathbf{1}(W_s x_i > 0))$. Then

$$W_{s+1}x' = (1-\eta\lambda)W_s x' - \eta\,\frac{1}{n}\sum_{i=1}^{n} y_i\,\ell'(y_i\,f_{\theta_s}(x_i))D_{s,i}\,u\,\langle x_i, x'\rangle$$

which together with Cauchy-Schwartz inequality implies

$$\begin{aligned}\|W_{s+1}x'\| &\le (1-\eta\lambda)\|W_s x'\| + \eta\,\frac{1}{n}\sum_{i=1}^{n}|y_i\,\ell'(y_i\,f_{\theta_s}(x_i))|\|D_{s,i}\|_2\,\|u\|\,|\langle x_i, x'\rangle| \\ &\le (1-\eta\lambda)\|W_s x'\| + \varepsilon\,\eta\ .\end{aligned}$$

Observe that relation $x_{s+1} \le a_s x_s + b_s$ unwinds from t to t_0 as $x_t \le x_{t_0}\prod_{k=t_0}^{t-1} a_k + \sum_{s=t_0}^{t-1} b_s \prod_{k=s+1}^{t-1} a_k$. So,

$$\begin{aligned}\|W_t x'\| &\le \|W_0 x'\|\,(1-\eta\lambda)^t + \varepsilon\,\eta\sum_{s=0}^{t-1}\prod_{k=s+1}^{t-1}(1-\eta\lambda) \\ &= \|W_0 x'\|\,(1-\eta\lambda)^t + \varepsilon\,\eta\sum_{s=0}^{t-1}(1-\eta\lambda)^{t-s-1}\ .\end{aligned}$$

□

C.2 Proof of the shallow neural network case

First observe that

$$\begin{aligned}|f_{\theta_t+\theta'_t}(x) - f_{\theta_t}(x)| &\le |\langle u, (W_t x + W'_t x)_+\rangle - \langle u, (W_t x)_+\rangle| \\ &\le \|(W_t x + W'_t x)_+ - (W_t x)_+\| \qquad \text{(Cauchy-Schwartz inequality)} \\ &\le \|W'_t x\|\end{aligned}$$

where the last inequality comes by a basic fact about ReLUs (see Section 2). Now, since ℓ is 1-Lipschitz,

$$\begin{aligned}\ell\left(y f_{\theta_t+\theta'_t}(x)\right) - \ell\left(y f_{\theta_t}(x)\right) &= \ell\left(y\,\langle u, (W_t x + W'_t x)_+\rangle\right) - \ell\left(y\,\langle u, (W_t x)_+\rangle\right) \\ &\le |f_{\theta_t+\theta'_t}(x) - f_{\theta_t}(x)| \\ &\le \|W'_t x\|\ .\end{aligned}$$

At this point, we apply losses over $(x_i, y_i)_i$ and average to have

$$L(\theta_t + \theta'_t) \le L(\theta_t) + \frac{1}{n}\sum_{i=1}^{n}\|W'_t x_i\|\ .$$

Now we use Lemma 4 to control $\|W'_t x_i\|$. □

C.3 Proof of Theorem 1

Theorem 1 is a direct corollary of the following theorem which we show in Section C.3.1:

Theorem 2. *Assume that weight matrices are initialized such that for all* $k \in [K-1]$

$$W_{k+1}(0)^\top W_{k+1}(0) = W_k(0) W_k(0)^\top \ .$$

Let ℓ *be a logistic loss and let* $L(\theta(0)) \leq C$*. Then, for any* $t \geq 0$ *and any input* x' *such that* $\max_i |\langle x_i, x' \rangle| \leq \varepsilon$ *we have*

$$|\langle w(t), x' \rangle| \leq |\langle w(0), x' \rangle| A_1 \, e^{-\lambda K t} + A_2 \, \varepsilon$$

where

$$A_1 = \exp\left(\frac{B^{2-\frac{2}{K}} (K-1)(1+\lambda K) C}{2\lambda K} \right) \ ,$$
$$A_2 = 2B^{2-\frac{2}{K}} C \, A_1 \left(\frac{1 - e^{-\lambda K t/2}}{\lambda K} \right) \ ,$$
$$B = \|w(0)\|^2 + \frac{C}{\lambda} \ .$$

In the following for the end-to-end vector $w(t)^\top = w_K(t)^\top W_{K-1}(t) \cdots W_1(t)$ we use notation L^1 to denote its loss:

$$L^1(w(t)) = \frac{1}{n} \sum_{i=1}^n \ell(y_i \langle w(t), x_i \rangle) \ .$$

Note that $L^1(w(t)) = L(\theta(t))$. Proof relies on the following crucial result connecting per-layer updates to updates of the product matrix:

Theorem 3 (Arora et al. (2018, Theorem 1))**.** *Assume that weight matrices are initialized (at time* t_0*) in such a way that they satisfy for all* $k \in [K-1]$*.*

$$W_{k+1}(t_0)^\top W_{k+1}(t_0) = W_k(t_0) W_k(t_0)^\top$$

Then, under dynamics with updates as in Eq. (1) for the end-to-end matrix we have

$$\dot{W}(t) = -\lambda K \cdot W(t) - \sum_{k=1}^K \left(W(t) W(t)^\top \right)^{\frac{k-1}{K}} \cdot \nabla L^1(W) \cdot \left(W(t)^\top W(t) \right)^{\frac{K-k}{K}} \ .$$

C.3.1 Proof of Theorem 2

We start from adapting Theorem 3 to our case to get $w(t)$ and then get an identity for $\langle w(t), x' \rangle$ by solving the resulting differential equation. Once we get dependence on ε we must ensure that the remaining terms (arising from the solution to differential equation) are non-divergent, which we will do through the stationary point convergence argument.

Theorem 3 gives us

$$\begin{aligned}
\dot{w}(t)^\top &= -\lambda K \cdot w(t)^\top - \|w(t)\|^{\frac{2(K-1)}{K}} \cdot \nabla L^1(w(t))^\top \\
&\quad - \sum_{k=1}^{K-1} \|w(t)\|^{\frac{2(k-1)}{K}} \cdot \nabla L^1(w(t))^\top \left(w(t) w(t)^\top \right)^{\frac{K-k}{K}} \\
&= -\lambda K \cdot w(t)^\top - \|w(t)\|^{2-\frac{2}{K}} \cdot \nabla L^1(w(t))^\top \\
&\quad - \sum_{k=1}^{K-1} \|w(t)\|^{\frac{2(k-1)}{K} + \frac{2(K-k)}{K}} \cdot \nabla L^1(w(t))^\top \left(\frac{w(t) w(t)^\top}{\|w(t)\|^2} \right)^{\frac{K-k}{K}} \\
&= -\lambda K \cdot w(t)^\top - \|w(t)\|^{2-\frac{2}{K}} \cdot \nabla L^1(w(t))^\top \\
&\quad - (K-1) \|w(t)\|^{2-\frac{2}{K}} \cdot \left\langle \nabla L^1(w(t)), w(t) \right\rangle w(t)^\top \ .
\end{aligned}$$

Put another way,

$$\dot{w}(t) = \left(-\lambda K - (K-1)\|w(t)\|^{2-\frac{2}{K}} \left\langle \nabla L^1(w(t)), w(t)\right\rangle\right) w(t) - \|w(t)\|^{2-\frac{2}{K}} \nabla L^1(w(t)) \ . \tag{4}$$

Taking dot product with x' gives

$$\langle \dot{w}(t), x'\rangle = a(t) \ \langle w(t), x'\rangle + b(t) \tag{5}$$

where we introduce abbreviations

$$\begin{aligned} a(t) &:= -\lambda K - (K-1)\|w(t)\|^{2-\frac{2}{K}} \left\langle \nabla L^1(w(t)), w(t)\right\rangle \\ b(t) &:= -\|w(t)\|^{2-\frac{2}{K}} \left\langle \nabla L^1(w(t)), x'\right\rangle \ . \end{aligned}$$

Solving Eq. (5) we get

$$\langle w(t), x'\rangle = \langle w(t_0), x'\rangle \ e^{\int_{t_0}^t a(s)\,\mathrm{d}s} + \int_{t_0}^t b(s)\, e^{\int_s^t a(r)\,\mathrm{d}r}\,\mathrm{d}s \ .$$

Assuming for a moment that $\|w(t)\| \le B$ (where B will be determined later) Cauchy-Schwartz inequality gives

$$a(t) \le -\lambda K + (K-1)B^{1-\frac{1}{K}} \|w(t)\|^{1-\frac{1}{K}} \left|\left\langle \nabla L^1(w(t)), w(t)\right\rangle\right| \ .$$

On the other hand, using the fact that for logistic loss function $|\ell'(z)| \le \ell(z)$,

$$\begin{aligned} b(t) &= \|w(t)\|^{2-\frac{2}{K}} \ \frac{1}{n}\sum_{i=1}^n \ell'(y_i \langle w(t), x_i\rangle) y_i \langle x_i, x'\rangle \\ &\le \varepsilon \cdot B^{2-\frac{2}{K}} \ \frac{1}{n}\sum_{i=1}^n \ell(y_i \langle w(t), x_i\rangle) \\ &= \varepsilon \cdot B^{2-\frac{2}{K}} \ L^1(w(t)) \ . \end{aligned}$$

So, it is left to show that the term $\int_{t_0}^t a(s)\,\mathrm{d}s$ does not diverge. To this end, we show the following (with proof at the end of the section):

Lemma 5 (Stationary point convergence for dynamics in Eq. (4))**.**

$$(K-1)\int_s^t \|w(r)\|^{1-\frac{1}{K}} \left|\left\langle \nabla L^1(w(r)), w(r)\right\rangle\right| \mathrm{d}r \le \sqrt{(K-1)(1+\lambda K) L^1(w(0)) \cdot (t-s)} \ .$$

Using this lemma to bound $\int a(s)\,\mathrm{d}s$ gives

$$\begin{aligned} \left|\langle w(t), x'\rangle\right| \le \left|\langle w(0), x'\rangle\right| & \underbrace{\exp\left(-\lambda K \cdot t + B^{1-\frac{1}{K}} \sqrt{(K-1)(1+\lambda K) C \cdot t}\right)}_{(i)} \\ & + \varepsilon \cdot B^{2-\frac{2}{K}} C \underbrace{\int_0^t \exp\left(-\lambda K \cdot (t-s) + B^{1-\frac{1}{K}} \sqrt{(K-1)(1+\lambda K) C \cdot (t-s)}\right) \mathrm{d}s}_{(ii)} \end{aligned}$$

where we also assumed that $\sup_t L^1(w(t)) \le C$. At this point we will bound (i) and (ii) by using the fact that

$$e^{-at+b\sqrt{t}} \le e^{\frac{b^2}{2a}}\, e^{-at/2} \qquad (a, b, t > 0) \ .$$

that comes from (choosing p such that $\frac{b}{2\sqrt{p}} = a/2$):

Proposition 1. *For any $a, b, t, p > 0$,*

$$-at + b\sqrt{t} \le \left(-a + \frac{b}{2\sqrt{p}}\right) t + \frac{b}{2}\sqrt{p} \ .$$

In particular this gives

$$(ii) \leq 2\left(\frac{1-e^{-\lambda K t/2}}{\lambda K}\right)\exp\left(\frac{B^{2-\frac{2}{K}}(K-1)(1+\lambda K)C}{2\lambda K}\right)$$

As promised, the final bit is to give B. Since objective is non-increasing

$$\lambda\|w(t)\|^2 \leq \lambda\|w(t)\|^2 + L^1(w(t)) \leq \lambda\|w(0)\|^2 + L^1(w(0)) \qquad \Longrightarrow \qquad B = \|w(0)\|^2 + \frac{L^1(w(0))}{\lambda}\ .$$

Proof of Lemma 5. Using the chain rule together with Eq. (4) we have

$$\begin{aligned}\frac{\mathrm{d}L^1(w(t))}{\mathrm{d}t} &= \left\langle \nabla L^1(w(t)), \dot{w}(t)\right\rangle \\ &= -\lambda K\left\langle \nabla L^1(w(t)), \dot{w}(t)\right\rangle - (K-1)\|w(t)\|^{2-\frac{2}{K}}\left\langle \nabla L^1(w(t)), w(t)\right\rangle^2 \\ &\quad - \|w(t)\|^{2-\frac{2}{K}}\|\nabla L^1(w(t))\|^2\end{aligned}$$

and so

$$\begin{aligned}L^1(w(t)) - L^1(w(0)) &= -\lambda K\int_0^t \left\langle \nabla L^1(w(s)), \dot{w}(s)\right\rangle \mathrm{d}s \\ &\quad - (K-1)\int_0^t \|w(s)\|^{2-\frac{2}{K}}\left\langle \nabla L^1(w(s)), w(s)\right\rangle^2 \mathrm{d}s \\ &\quad - \int_0^t \|w(s)\|^{2-\frac{2}{K}}\|\nabla L^1(w(s))\|^2 \mathrm{d}s\ .\end{aligned}$$

Note also that the chain rule gives $\int_0^t \left\langle \nabla L^1(w(s)), \dot{w}(s)\right\rangle \mathrm{d}s = L^1(w(t)) - L^1(w(0))$ which gives

$$\begin{aligned}&(K-1)\int_0^t \|w(s)\|^{2-\frac{2}{K}}\left\langle \nabla L^1(w(s)), w(s)\right\rangle^2 \mathrm{d}s + \int_0^t \|w(s)\|^{2-\frac{2}{K}}\|\nabla L^1(w(s))\|^2 \mathrm{d}s \\ &\quad = (1+\lambda K)\left(L^1(w(0)) - L^1(w(t))\right)\ .\end{aligned}$$

Applying Jensen's inequality completes the proof. □

□

Model name	number of layers	model dimension	FF dimension
BERT	12	768	3072
RoBERTa	12	768	3072
GPT2-Large	36	1280	5120
GPT2-XL	48	1600	6400
Phi-2	32	2560	10240
GPT-J	28	4096	16384

Table 1: Specification of Transformer models.

D Additional empirical results from Section 5

D.1 Transformer results

Figure 4 summarizes more measurements in addition to those in Section 5. In Table 1 model dimension refers to dimensionality of self-attention weight matrices, while FF dimension refers to the size of the hidden layer of the feedforward network.

D.2 MLP results

We first include Section D.2 which presents results for misclassification test error (on the held out sample) for merged models (left figure corresponds to orthogonal datasets, while right to the same source), in the same setting as in Figure 3. Interestingly, merged models perform just as well as the original ones in term of the test error, whereas in the "same task" case the gap is apparent. This indicates that model merging not only retains training errors of the original models, but also preserves generalization ability.

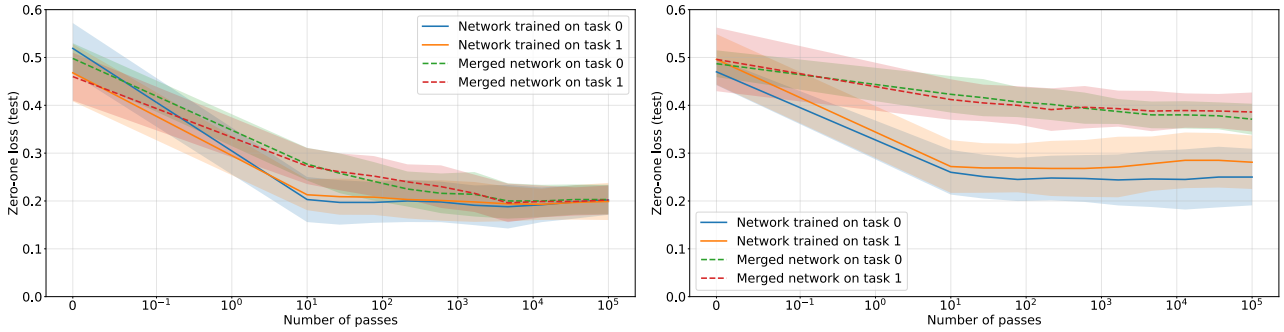

Figure 5: Misclassification test error (on the held out sample) for merged models (left figure corresponds to orthogonal datasets, while right to the same source), in the same setting as in Figure 3.

The second set of experiments is performed on 'Fashion MNIST' dataset (Xiao et al., 2017), which contains grayscale images of 28x28 pixels each, representing clothing items from 10 different categories. We adapt this dataset with sample size 10 for binary classification by grouping first 5 classes into class 0, and remaining into class 1. When we consider two different tasks, we append 784-dimensional zero vector for the first task, and prepend in case of the second task. This way, inputs from different tasks remain orthogonal, while the length of each input is preserved. Finally, in case of task one binary labels are preserved as is, while for the second task labels are inverted.

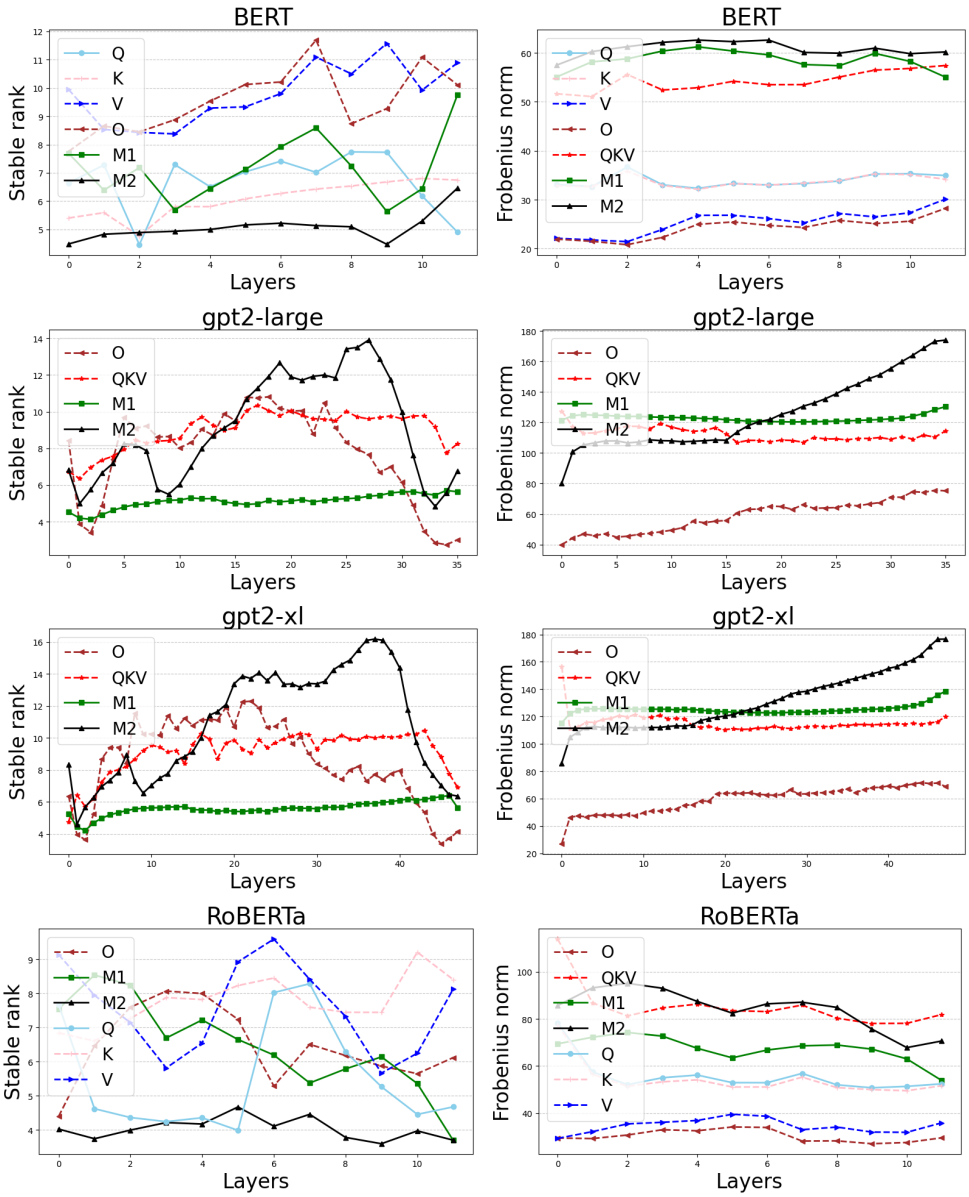

Figure 4: Stable ranks and Frobenius norms of different weight matrices in pretrained BERT (first row), GPT2-Large (second row), RoBERTa (third row), and GPT2-XL (fourth row) model.

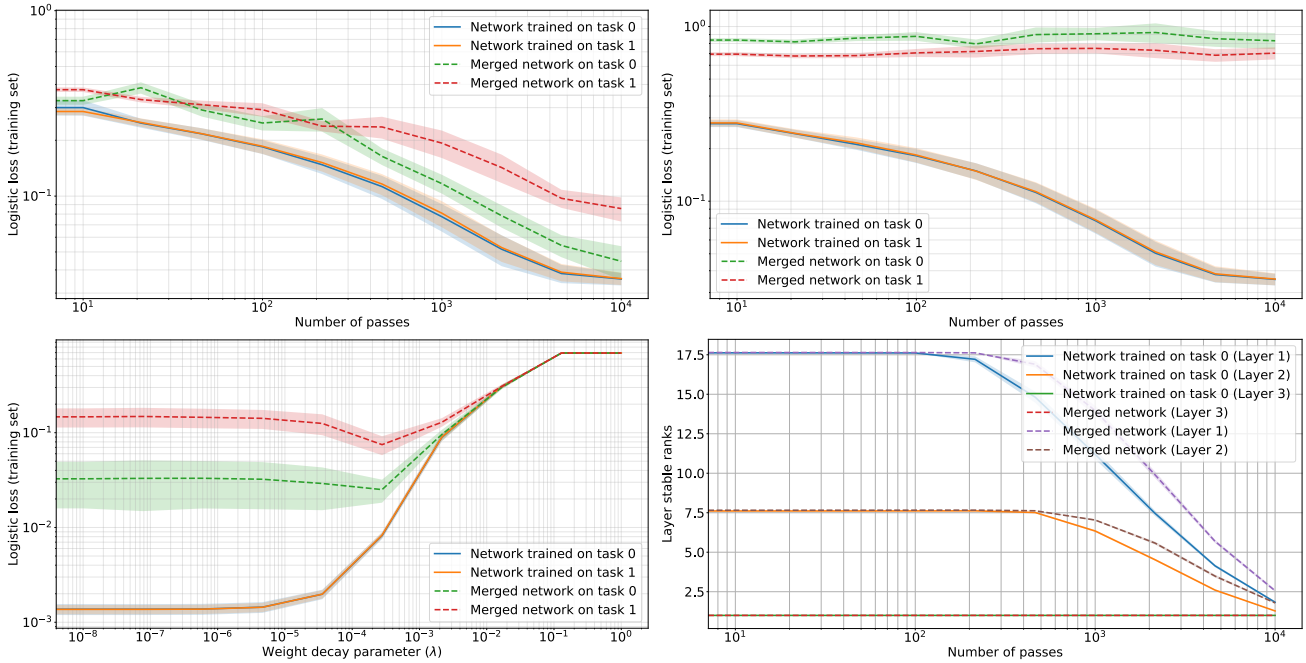

Figure 6: (Fashion MNIST dataset) First row: Training neural networks on different tasks (orthogonal inputs) (left) vs. the same task (right) and merging the parameters by adding weight matrices. The resulting network performs well on different tasks after sufficiently many iterations, while given the same task, it does not. Second row: this effect manifests when weight decay strength is sufficiently large (left). Stable rank of each weight matrix converges to a small value. Merged network matches the stable rank of individual networks (right).