FEW-SHOT MULTI-TASK LEARNING VIA IMPLICIT REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern machine learning is highly data-intensive. Few-shot learning (FSL) aims to resolve this sample efficiency problem by learning from multiple tasks and quickly adapt to new tasks containing only a few samples. However, FSL problems proves to be significantly more challenging and require more compute expensive process to optimize. In this work, we consider multi-task linear regression (MTLR) as a canonical problem for few-shot learning, and investigate the source of challenge of FSL. We find that the MTLR exhibits local minimum problems that are not present in single-task problem, and thus making the learning much more challenging. We also show that the problem can be resolved by overparameterizing the model by increasing both the width and depth of the linear network and initializing the weights with small values, exploiting the implicit regularization bias of gradient descent-based learning.

1 INTRO

Despite the recent success of large deep neural-networks models, training these models typically require extremely large dataset. In contrast, human cognition exhibits an impressive feature that enables quickly learning new concept from a small set of experience and to robustly generalize to unseen tasks. An active area of research to bridge this sample efficiency gap is few-shot learning and multi-task learning, which aims to aggregate learning from many number of related tasks to extract the shared feature/information across tasks, and leverages it to quickly learn new tasks (Argyriou et al., 2006; 2008; Santoro et al., 2016; Vinyals et al., 2016; Snell et al., 2017; Finn et al., 2017; Nichol & Schulman, 2018).

However, learning to solve these few-shot/multi-task problems proves to be considerably more challenging than single-task problems. Popular approaches include meta-learning setup that uses separate inner-loop and outer-loop optimization that are both compute and memory intensive (Finn et al., 2017; Nichol & Schulman, 2018) , or require additional unsupervised learning steps that learn the relatedness metric between objects (Santoro et al., 2016; Vinyals et al., 2016; Snell et al., 2017).

In this work, we investigate the challenge of few-shot learning problems by considering a multi-task version of linear regression problem as a canonical example, in which the tasks are related by sharing a small common feature space (Argyriou et al., 2006; 2008). Our analysis reveals that despite the simplicity of the problem, the few-shot/multi-task learning (MTL) setting makes the linear regression problem significantly more difficult to solve. We find that multi-task setting makes the linear regression problem highly non-convex and introduces poor local minima, which hinders the performance of gradient-descent learning. However, we show that the problem can be resolved by overparameterizing the linear network model with increased the width and depth and initializing the weights with small values. This setting exploits the recently investigated implicit regularization bias of gradient descent-based learning.

Out contribution includes: 1. Characterization and complexity analysis of multi-task linear regression problem, 2. Derivation of an analytic expression that quantifies the few-shot generalization performance of the model, and 3. a simple learning method to find the solution.

2 PROBLEM FORMULATION

Here, we describe the few-shot multi-task learning problem considered in this work, which follows a similar construction as in Argyriou et al. (2006). We are given T supervised learning tasks, where each task t is identified by a function $f_t : \mathbb{R}^{d_x} \to \mathbb{R}$ that describes the input-label relationship of the corresponding dataset. Each dataset contains k samples of input-label examples $\mathcal{D}_t = (\boldsymbol{x}_{t1}, y_{t1}), \cdots, (\boldsymbol{x}_{tk}, y_{tk}) \in \mathbb{R}^{d_x} \times \mathbb{R}$, which can also be described by using matrix notations as $\boldsymbol{X}_t \in \mathbb{R}^{d_x \times k}$ and $\boldsymbol{y}_t \in \mathbb{R}^{1 \times k}$.

The multi-task learner trains on T tasks $\{\mathcal{D}_t\}_{t=1}^{T}$ to uncover the underlying structure shared across the task distribution that generalizes to unseen tasks. Specifically, we consider that the tasks are related by sharing a small set of features. Formally, we assume that the task can be represented by the function

$$f_t(\boldsymbol{x}) = \sum_{i=1}^{d_f} a_{it} h_i(\boldsymbol{x}) = \boldsymbol{a}_t \cdot \boldsymbol{h}(\boldsymbol{x}) \tag{1}$$

where $h_i : \mathbb{R}^{d_x} \to \mathbb{R}$ are the features, $a_{it} \in \mathbb{R}$ are the task-specific regression parameters and d_f is the total number of features. For simplicity, we consider the case that the features are linear homogeneous functions, *i.e.* they are of the form $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{d_f \times d_x}$. Without loss of generality, we further assume that $\boldsymbol{U}$ is an orthogonal matrix, which simplifies the derivations later. Thus, the final form of the task function is

$$f_t(\boldsymbol{x}) = \boldsymbol{a}_t \boldsymbol{U} \boldsymbol{x} \tag{2}$$

The input data is assumed to be drawn from a standard zero-mean unit Normal distribution.

2.1 COMPLEXITY OF MULTI-TASK PROBLEM

The simplicity of our setup makes it straightforward to quantify all the degrees of freedom (DOF) involved in the multi-task learning problem. The parameter count involved in the task description equation 2 is d_f for $\boldsymbol{c}_t$ per task and $d_f \times d_x$ shared across the tasks. However, the orthogonality condition $\boldsymbol{U}\boldsymbol{U}^\intercal = \boldsymbol{I}^{d_f}$ introduces d_f^2 constraints, which restricts DOF. Therefore, the total DOF for characterizing a set of T structured tasks is $(T \times d_f + d_f \times (d_x - d_f))$. Note that the shared structure reduces the DOF by $(T - d_f) \times (d_x - d_f)$, which is what ultimately allows the few-shot learning feasible.

From this, we can estimate the minimum number of tasks that is needed to completely characterize the MTL problem. Since each datapoint $(\boldsymbol{x}_{tm}, \boldsymbol{y}_{tm})$ provides one constraint, the total data in T tasks with k datapoints contributes to $T \times k$ constraints, which needs to be greater or equal to the above DOF: *i.e.* $T \times k \geq T \times d_f + d_x \times (d_x - d_f)$. Therefore, the minimum required number of tasks that makes the k-shot multi-task learning feasible is

$$T^* \equiv \frac{d_f \times (d_x - d_f)}{k - d_f}, \tag{3}$$

which proves to be critical for controlling the difficulty of solving the MTL problem. Note that T^* can dramatically increase for MTL with small $k \sim d_f$, since then most of the information of each task's dataset gets utilized for identifying task-specific parameters, only leaving $k - d_f$ constraints per task to characterize the shared parameters (Fig 3).

2.2 LINEAR NETWORK MODEL

The task structure in equation 2 implies that it can be naturally modeled by two-layer linear neural networks.

$$g_t(\boldsymbol{x}) = \boldsymbol{c}_t \boldsymbol{W} \boldsymbol{x} \tag{4}$$

where $\boldsymbol{W} \in \mathbb{R}^{d_h \times d_x}$, $\boldsymbol{c}_t \in \mathbb{R}^{d_h}$, and d_h is the dimension of the hidden layer, which may differ from the task's feature dimension d_f . For deeper versions of the network model, we consider the weight $\boldsymbol{W}$ being composed of a product of l matrices, $\boldsymbol{W} = \boldsymbol{W}_l \cdots \boldsymbol{W}_2 \boldsymbol{W}_1$, with $\boldsymbol{W}_l \in \mathbb{R}^{d_h \times d_x}$ and $\boldsymbol{W}_{i \neq l} \in \mathbb{R}^{d_x \times d_x}$.

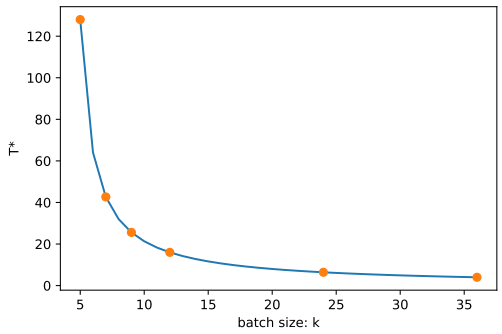

Figure 1: The minimum required task T^* as a function of per task batch-size k, shown for input and feature dimensions of $d_x = 36$, $d_f = 4$. Dots indicate the values used in the experiments.

Note that most deep network models used in multi-task learning settings indeed takes the same architecture as equation 4 that they also use the different top-level layer is used for each specific task, while the earlier feature-encoding layers are shared across all tasks.

3 METHOD

For training, we use gradient descent algorithm with momentum to update both the task-specific $\boldsymbol{c}_t$ and the shared parameters $\boldsymbol{W}_i$ to minimize the empirical risk $\mathbb{E}_{\mathbb{D}}[L_t(\boldsymbol{c}_t, W)]$, where $\mathbb{D} \equiv \{\mathcal{D}_t\}_{t=1}^T$ and

$$L_t(\boldsymbol{c}_t, W) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_t}[L(y, g_t(\boldsymbol{x}))] + \epsilon\|\boldsymbol{c}_t\|^2 \tag{5}$$

where L is the square-loss and the second term is a regularization loss on $\boldsymbol{c}_t$. A small regularization term on $\boldsymbol{W}$ may also be applied, which is suppressed here for notational simplicity.

For evaluating the generalization capacity of the learned weight $\boldsymbol{W}$ to unseen tasks, we derive an analytic lower-bound of the expected loss on a d_f-shot learning task. The derivation assumes first adapting/optimizing $\boldsymbol{c}$ over a d_f batch of datapoints of an evaluation task, called the *support set*. However, due to the small batch-size, the quality of the estimation can critically depend on the specifics of the sampled batch; For example, the support set can be completely non-informative if all input data $\boldsymbol{X}$ are from the null space of $\boldsymbol{U}$, in which all label data are $\boldsymbol{y} = 0$. Here, we remove such dependency by considering the most informative batch as the support set, which exactly spans the range of $\boldsymbol{U}$, *i.e.*

$$\boldsymbol{X}\boldsymbol{X}^\intercal/d_f = \boldsymbol{U}^\intercal\boldsymbol{U} \tag{6}$$

With $\boldsymbol{c}$ adapted to the support set of a evaluation task, taking expectation of the loss over the task and input distributions yields the following lower bound

$$L^{\text{eval}}(\boldsymbol{W}) = \mathbb{E}_t[\mathbb{E}_{(\boldsymbol{X}_t,\boldsymbol{y}_t)}[L_t(\boldsymbol{c}_t^{\text{opt}}, \boldsymbol{W}; \boldsymbol{X}_t, \boldsymbol{y}_t)]] \tag{7}$$

$$= \text{Tr}[\boldsymbol{U}\boldsymbol{U}^\intercal] + \text{Tr}[(\boldsymbol{W}\boldsymbol{U}^\intercal\boldsymbol{U}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{I})^{-1}(\boldsymbol{W}\boldsymbol{W}^\intercal - 2\boldsymbol{W}\boldsymbol{U}^\intercal\boldsymbol{U}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{I})] \tag{8}$$

which is expressed entirely in terms of $\boldsymbol{W}$ and $\boldsymbol{U}$. See Appendix for the derivation.

This evaluation loss quantifies how much $\boldsymbol{W}$ aligns with the low-rank feature space, i.e. the the row space of $\boldsymbol{U}$. To see this, consider $\boldsymbol{W}$ whose singular vectors are either entirely inside or outside the feature space. That is, given the singular value decomposition $\boldsymbol{W} = \boldsymbol{O}\sigma\boldsymbol{V}^\intercal$, the singular vectors projected onto $\boldsymbol{U}$ have norm $\sum_{ln}(\sum_m \boldsymbol{U}_{lm}\boldsymbol{V}_{mn})^2$ that is either zero or one. In this case, the above loss reduces to

$$L^{\text{eval,opt}} = \sum_{i\in\mathbb{I}} \frac{2\epsilon}{\sigma_i^2 + \epsilon} + \sum_{i\notin\mathbb{I}} \frac{\sigma_i^2 + \epsilon}{\epsilon} \tag{9}$$

$$\tag{10}$$

where $\mathbb{I}$ is the indicator function for distinguishing whether the i'th singular mode is within the range of $\boldsymbol{U}$. (See Appendix.) Note that the first term promotes large singular values within the range of $\boldsymbol{U}$, whereas the second term penalizes singular values outside the subspace. Therefore, this evaluation loss is minimized when $\boldsymbol{W}$ exhibits the same low-rank structure as $\boldsymbol{U}$.

4 RESULTS

We investigate the efficacy of solving the few-shot MTL problem using gradient decent learning in various data regimes. We control the difficulty of the problem by varying the per task batch size k and the number of tasks T in the training dataset in relation to the given input dimension and feature dimension of the problem. For each batch size, we use the number of task T to be around the theoretical minimum required number T^* with increment size of 25%.

With this setup, we investigate the effect of width, depth, and initial value of the linear network model on its learning dynamics. The weights $\boldsymbol{W}_l$ are initialized to be orthogonal matrix with the singular vectors of adjacent layer being aligned, which yields the fastest learning dynamics, as shown/introduced in Saxe et al. (2013) and widely used in linear network research (Arora et al., 2018a;b; 2019; Lampinen & Ganguli, 2018; Advani & Saxe, 2017).

4.1 LOW-RANK ARCHITECTURE

First, let's consider the case in which the network's hidden dimension matches the task's feature dimension, *i.e.* $d_h = d_f$. This condition may appear to be most suitable for solving the MTL problem, since the model architecture is explicitly constrained to have the same rank as the task structure, thus can achieve good generalization performance on unseen tasks.

Fig **??** shows that the learning dynamics can fully fit the training data when the problem is highly underdetermined ($T < T^*$, two left columns), or when the batch-size approaches the input dimension ($k = d_x$, bottom row), which is no longer a few-shot learning problem. In other conditions, however, the learning dynamics of the model fail to perfectly fit the training data and instead converges to the poor local minimum solutions. Moreover, the model shows poor generalization performance in all conditions (except for the bottom row), and thus has failed to learn the correct task structure.

This result shows that even linear models can exhibit poor local minima when considered in few-shot multi-task learning problems. This is indeed very surprising, since all previous studies on linear networks, primarily conducted in single-task settings, have shown that the loss landscapes of linear models only exhibit global minima and their learning dynamics are guaranteed to converge to those global minima in almost all initialization (Arora et al., 2018a;b; Bah et al., 2019; Geyer et al., 2020; Kawaguchi, 2016) . To our best knowledge, this is the first demonstration of local minima problem in linear models.

4.2 FULL-RANK MODEL

Next, we consider the over-parameterized condition in which the hidden dimension matches the input dimension $d_h = d_x$. This full rank condition makes learning of the training dataset trivial, since the task-specific parameters $\boldsymbol{c}$ can fullly fit the data without ever learning $\boldsymbol{W}$. Consequently, learning dynamics in this model exhibits smooth decay toward zero training loss. However, it also means the shared parameter does not learn the generalizable, low rank tasks structure. Consequently, this model exhibits even worse generalization performance than the narrow model. See Fig 3A. Note the scale of the y-axis.

Can the overparameterized models be trained to learn the task structure of the MTL problem? Recent investigations have shown that deep linear networks show implicit bias toward low rank solutions when trained with gradient descent algorithm, which becomes stronger for deeper networks that are initialized with small weight values (Arora et al., 2018a;b; 2019; Lampinen & Ganguli, 2018; Advani & Saxe, 2017). Next, we investigate whether such implicit regularization bias can be exploited to rescue the generalization performance of the over-parameterized model.

Our simulation result with confirms that indeed deep linear networks that are initialized with small weights can indeed greatly enhance the generalization performance in the MTL problem (See Fig 3B

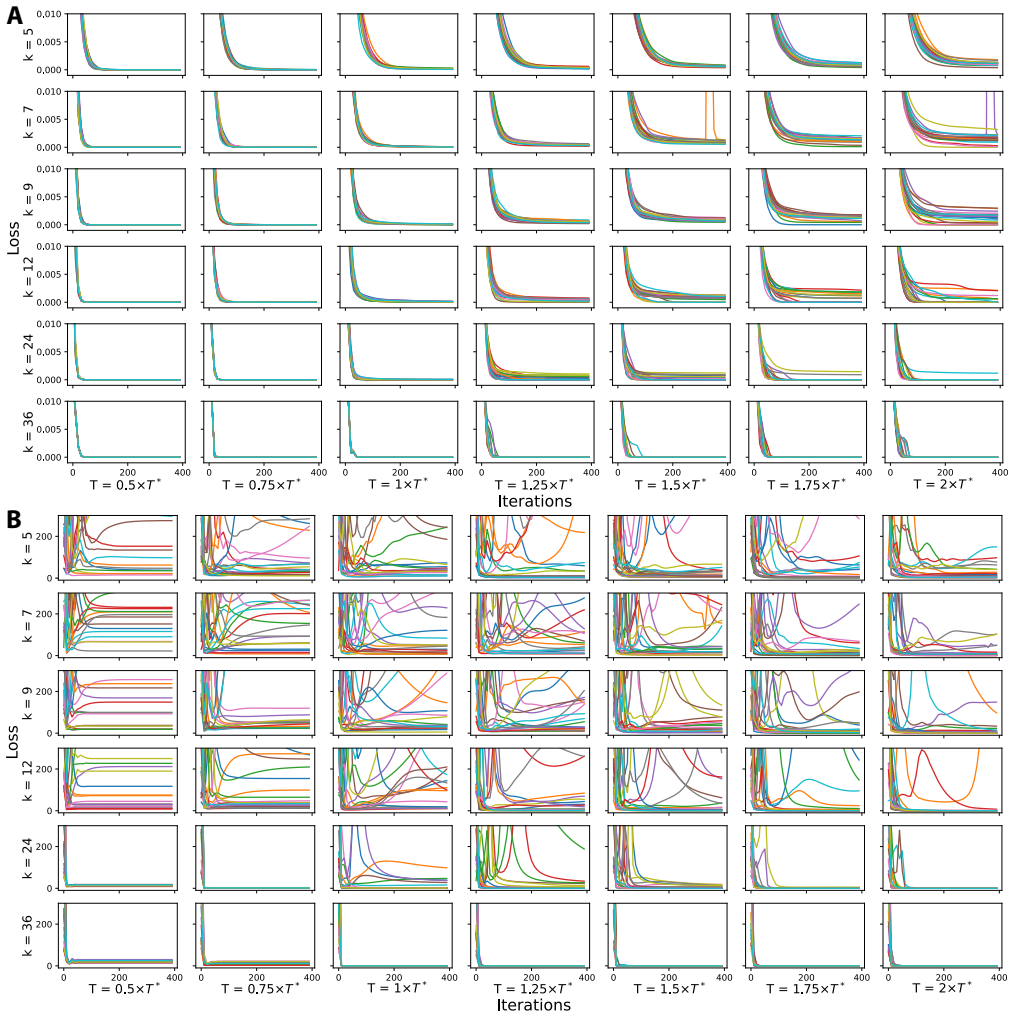

Figure 2: Training loss (A) and evaluation loss (B) profiles of narrow 2-layer models with initial weight scale 0.1. Each plot shows the result of 20 learning results of different initializations.

and Fig 4). The resulting model show smooth decay in the training loss, while also keeping the evaluation loss small.

The singular value decomposition of the network weight $\boldsymbol{W}$ reveals that the deep linear network models indeed learns the low rank task structure. The singular values exhibit large gap between d_f number of main singular modes (Fig 5A), and the corresponding singular vectors shows increasing alignment with the row space of $\boldsymbol{U}$ (Fig 5B) during learning. Our result confirms that the implicit bias for regularization in gradient descent can be exploited to discover low-rank solutions that generalizes across tasks.

5 Conclusion

Few-shot/multi-task learning exhibits many challenging aspects to solve. In this work, we made a step toward understanding those challenges by investigating a simplified canonical model of multi-task regression problem. We found that the multi-task version introduces increased non-convexity of loss landscape with poor local minima, which hinders learning by simple gradient descent algorithm. We then showed that this problem can be resolved by considering larger, overparameterized models that are well initialized to exploit the implicit regularization bias of gradient-descent learning. This

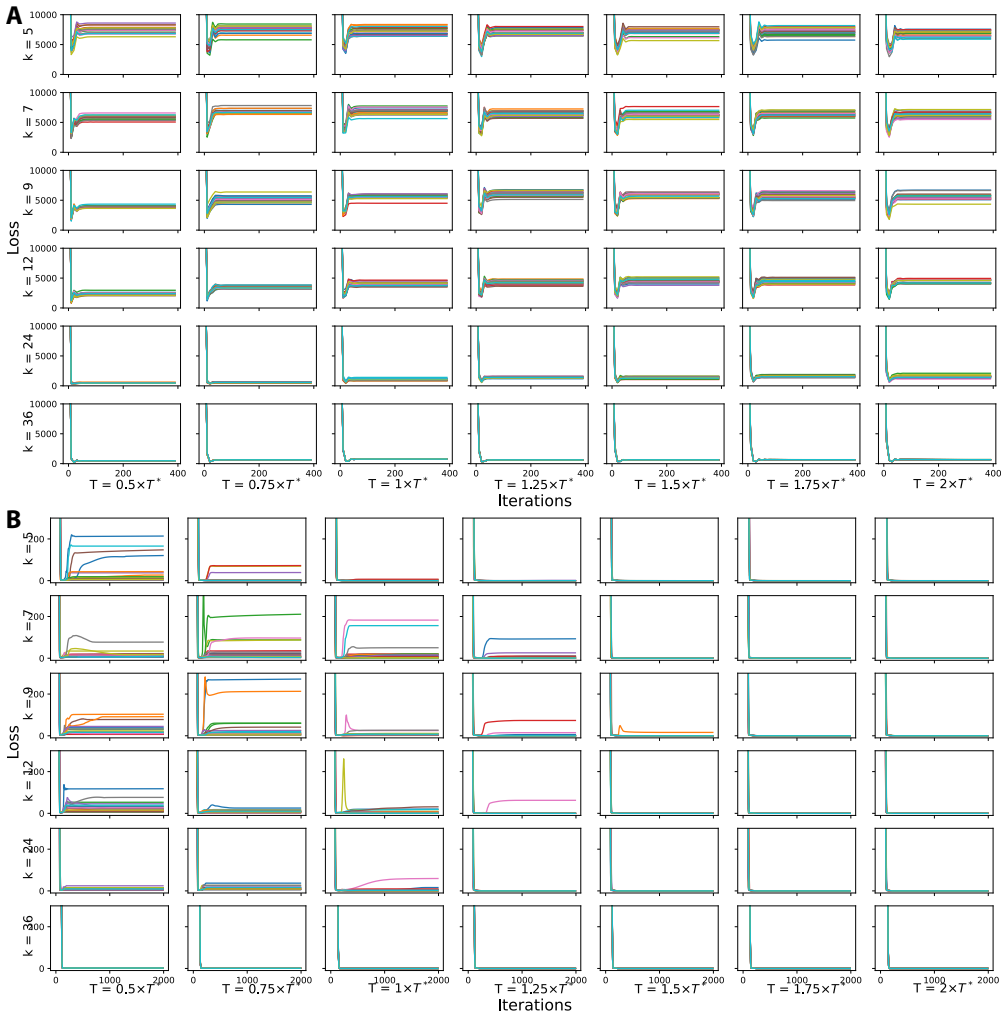

Figure 3: Evaluation loss of wide 2-layer model with initial weight scale 0.1 (A), and evaluation loss of wide 4-layer model with initial weight scale 0.001 (B). Note the difference in scale.

result could be further investigated to yield more efficient approaches for solving few-shot learning and meta-learning problems.

REFERENCES

Madhu S Advani and Andrew M Saxe. High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*, 2017.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *NIPS*, 2006.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine learning*, 73(3):243–272, 2008.

Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*, 2018a.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *arXiv preprint arXiv:1802.06509*, 2018b.

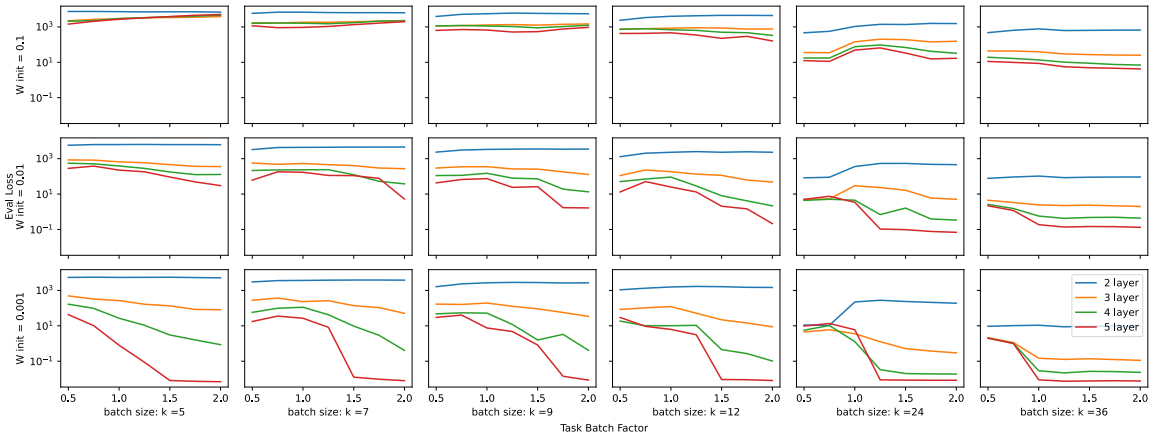

Figure 4: Summary figure showing the evaluation loss of wide linear network models across all training conditions (averged over 20 runs). Deeper network with smaller initial weights indeed helps with generalization performance.

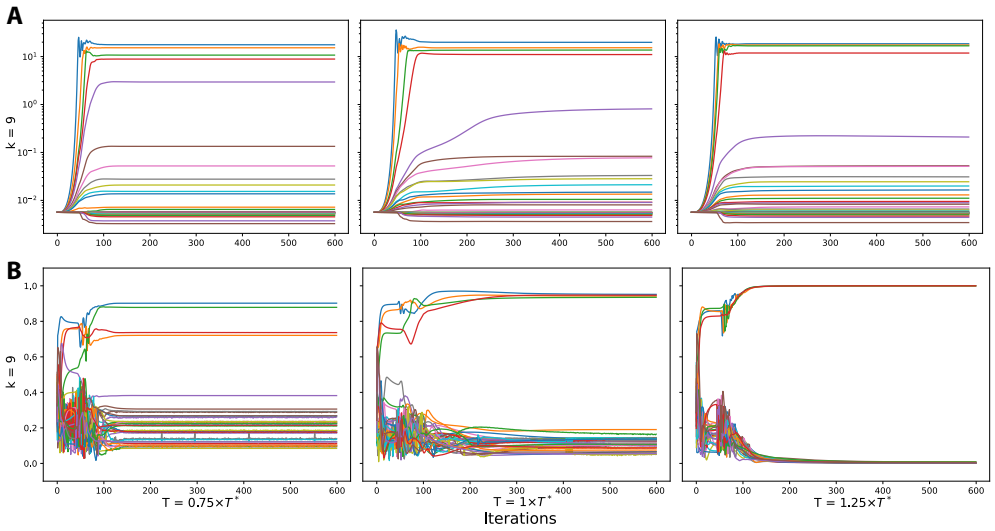

Figure 5: Trajectories of singular values (A) and singular vectors (B) of learning dynamics of a wide 3-layer network (initial weight scale 0.001). The right column $T = 1.25T^*$ shows the case of successful learning.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *arXiv preprint arXiv:1905.13655*, 2019.

Bubacarr Bah, Holger Rauhut, Ulrich Terstiege, and Michael Westdickenberg. Learning deep linear neural networks: Riemannian gradient flows and convergence to global minimizers. *arXiv preprint arXiv:1910.05505*, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135. PMLR, 2017.

Kelly Geyer, Anastasios Kyrillidis, and Amir Kalev. Low-rank regularization and solution uniqueness in over-parameterized matrix sensing. In *International Conference on Artificial Intelligence and Statistics*, pp. 930–940. PMLR, 2020.

Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in neural information processing systems*, pp. 586–594, 2016.

Andrew K Lampinen and Surya Ganguli. An analytic theory of generalization dynamics and transfer learning in deep linear networks. *arXiv preprint arXiv:1809.10374*, 2018.

Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850. PMLR, 2016.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016.

A Appendix

Here, we derive the analytic form of the lower-bound of the generalization loss, defined as

$$L^{\text{eval}}(\boldsymbol{W}) \equiv \mathbb{E}_t[\mathbb{E}_{\boldsymbol{X}_t,\boldsymbol{y}_t}[L(\boldsymbol{W}, \boldsymbol{c}_t^{\text{opt}}; \mathcal{D}_t))]] \tag{11}$$

where

$$L(\boldsymbol{W}, \boldsymbol{c}; \mathcal{D}) = \|\boldsymbol{c}\boldsymbol{W}\boldsymbol{X} - \boldsymbol{y}\|_2^2 + \epsilon\|\boldsymbol{c}\|^2 \tag{12}$$

$$= (\boldsymbol{c}\boldsymbol{W} - \boldsymbol{a}\boldsymbol{U})\frac{\boldsymbol{X}\boldsymbol{X}^\intercal}{k}(\boldsymbol{W}^\intercal\boldsymbol{c}^\intercal - \boldsymbol{U}^\intercal\boldsymbol{a}^\intercal) \tag{13}$$

where $\boldsymbol{y}$ is expanded as $y_k = \boldsymbol{a}\boldsymbol{U}\boldsymbol{x}_k$.

The adapted $\boldsymbol{c}_t^{\text{opt}}$ is defined as the optimum of eq 13 for the given weight $\boldsymbol{W}$ and data $\boldsymbol{c}_t^{\text{opt}} = \text{argmin}_{\boldsymbol{c}_t} L_t$. This optimum can be identified by the zero gradient condition wrt $\boldsymbol{c}_t$:

$$0 = \boldsymbol{c}_t^{\text{opt}}\boldsymbol{W}\frac{\boldsymbol{X}_t\boldsymbol{X}_t^\intercal}{k}\boldsymbol{W}^\intercal - \boldsymbol{a}_t\boldsymbol{U}\frac{\boldsymbol{X}_t\boldsymbol{X}_t^\intercal}{k}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{c}_t^{\text{opt}},$$

which yields

$$\boldsymbol{c}_t^{\text{opt}} = \boldsymbol{a}_t(\boldsymbol{U}\frac{\boldsymbol{X}_t\boldsymbol{X}_t^\intercal}{k}\boldsymbol{W}^\intercal)(\boldsymbol{W}\boldsymbol{U}^\intercal\frac{\boldsymbol{X}_t\boldsymbol{X}_t^\intercal}{k}\boldsymbol{U}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{I})^{-1} \tag{14}$$

We consider the case of few-shot adaptation in which the batch-size of the support set matches the feature dimension $k = d_f$. As explained in the main text, the lower bound of the few-shot performance is achieved by having a maximally informative support set, which spans the row space of $\boldsymbol{U}$: *i.e.*

$$\frac{1}{K}\boldsymbol{X}_t\boldsymbol{X}_t^\intercal = \boldsymbol{U}^\intercal\boldsymbol{U}$$

By denoting $\boldsymbol{P} \equiv \boldsymbol{W}\boldsymbol{U}^\intercal$, the adapted $\boldsymbol{c}$ equation A can be expressed as

$$\boldsymbol{c}_t^{\text{opt}} = \boldsymbol{a}_t\boldsymbol{P}_\epsilon^\dagger$$

where $\boldsymbol{P}_\epsilon^\dagger \equiv \boldsymbol{P}^\intercal(\boldsymbol{P}\boldsymbol{P}^\intercal + \epsilon\boldsymbol{I})^{-1}$ is pseudo-inverse of $\boldsymbol{P}$ with ϵ regularization, which reduces to the true pseudo-inverse in the limit $\epsilon \to 0$.

The lower-bound evaluation loss is then computed as (suppressing the regularization term $\epsilon\|\boldsymbol{c}_t\|^2$ for the simplicity of notation)

$$L^{\text{eval}} = \mathbb{E}_{\boldsymbol{a}_t}[\mathbb{E}_{\boldsymbol{X}_t}[\|\boldsymbol{a}_t(\boldsymbol{P}_\epsilon^\dagger \boldsymbol{W} - \boldsymbol{U})\boldsymbol{X}_t\|_2^2] \tag{15}$$

$$= \text{Tr}[(\boldsymbol{P}_\epsilon^\dagger \boldsymbol{W} - \boldsymbol{U})(\boldsymbol{W}^\intercal \boldsymbol{P}_\epsilon^{\dagger\intercal} - \boldsymbol{U}^\intercal)] \tag{16}$$

$$\approx \text{Tr}[\boldsymbol{U}\boldsymbol{U}^\intercal - 2\boldsymbol{P}^\intercal \boldsymbol{P}_\epsilon^{\dagger\intercal} + \text{Tr}[\boldsymbol{W}^\intercal(\boldsymbol{P}\boldsymbol{P}^\intercal + \epsilon\boldsymbol{I})^{-1}\boldsymbol{W}] \tag{17}$$

where the identities of the expectation $\mathbb{E}_{\boldsymbol{a}_t}[\boldsymbol{a}_t^\intercal \boldsymbol{a}_t] = \boldsymbol{I}$ and $\mathbb{E}_{\boldsymbol{X}}[\boldsymbol{X}\boldsymbol{X}^\intercal] = \boldsymbol{I}$ are used at equation 16 and the approximation $\boldsymbol{P}_\epsilon^{\dagger\intercal}\boldsymbol{P}_\epsilon^\dagger \approx (\boldsymbol{P}\boldsymbol{P}^\intercal + \epsilon\boldsymbol{I})^{-1}$ which holds in the limit $\epsilon \to 0$, is used at equation 17.

Expanding $\boldsymbol{P}$ and rearranging terms in equation 17 yields

$$L^{\text{eval,opt}} = \text{Tr}[\boldsymbol{U}\boldsymbol{U}^\intercal] + \text{Tr}[(\boldsymbol{W}\boldsymbol{U}^\intercal\boldsymbol{U}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{I})^{-1}(\boldsymbol{W}\boldsymbol{W}^\intercal - 2\boldsymbol{W}\boldsymbol{U}^\intercal\boldsymbol{U}\boldsymbol{W}^\intercal + \epsilon\boldsymbol{I})] \tag{18}$$

A.1 SVD ANALYSIS OF EQUATION 18

Given the singular value decomposition of $\boldsymbol{W}$: $\boldsymbol{W} = \boldsymbol{O}\sigma\boldsymbol{V}^\intercal$. we consider the case in which each of the right singular vectors in $\boldsymbol{V}$ is either completely inside or outside the the row space of $\bar{\boldsymbol{W}}$. That is, the norm of the projected vector $\sum_{ln}(\sum_m \boldsymbol{U}_{lm}\boldsymbol{V}_{mn})^2$ is either zero or one, which is described by a indicator function: $\mathbb{I}_j = 1$ if the ith singular mode is overlaps with $\bar{\boldsymbol{W}}$, or 0 otherwise. Expanding equation 18 with the SVD decomposition $\boldsymbol{W} = \boldsymbol{O}\sigma\boldsymbol{V}^\intercal$ simplifies to

$$L^{\text{eval,opt}} = \text{Tr}[\boldsymbol{I} + (\sigma^2\mathbb{I} + \epsilon\boldsymbol{I})^{-1}(\sigma^2 - 2\sigma^2\mathbb{I} + \epsilon)] \tag{19}$$

$$= \sum_i \mathbb{I}_i + \frac{\sigma_i^2 - 2\sigma_i^2\mathbb{I}_i + \epsilon}{\sigma_i^2\mathbb{I}_i + \epsilon} \tag{20}$$

$$= \sum_i \frac{\sigma_i^2\mathbb{I}_i + \epsilon\mathbb{I}_i + \sigma_i^2 - 2\sigma_i^2\mathbb{I}_i + \epsilon}{\sigma_i^2\mathbb{I}_i + \epsilon} \tag{21}$$

$$= \sum_i \frac{\sigma_i^2 + \epsilon + (-\sigma_i^2 + \epsilon)\mathbb{I}_i}{\sigma_i^2\mathbb{I}_i + \epsilon} \tag{22}$$

$$= \sum_{i\in\mathbb{I}} \frac{2\epsilon}{\sigma_i^2 + \epsilon} + \sum_{i\notin\mathbb{I}} \frac{\sigma_i^2 + \epsilon}{\epsilon} \tag{23}$$