Continual learning: a feature extraction formalization, an efficient algorithm, and barriers

Binghui Peng
Columbia University
bp2601@columbia.edu

Andrej Risteski
Carnegie Mellon University
aristesk@andrew.cmu.edu

Abstract

Continual learning is an emerging paradigm in machine learning, wherein a model is exposed in an online fashion to data from multiple different distributions (i.e. environments), and is expected to adapt to the distribution change. Precisely, the goal is to perform well in the new environment, while simultaneously retaining the performance on the previous environments (i.e. avoid "catastrophic forgetting"). While this setup has enjoyed a lot of attention in the applied community, there hasn't be theoretical work that even formalizes the desired guarantees. In this paper, we propose a framework for continual learning through the framework of feature extraction—namely, one in which features, as well as a classifier, are being trained with each environment. When the features are linear, we design an efficient gradient-based algorithm DPGrad, that is guaranteed to perform well on the current environment, as well as avoid catastrophic forgetting. In the general case, when the features are non-linear, we show such an algorithm cannot exist, whether efficient or not.

1 Introduction

In the last few years, there has been an increasingly large focus in the modern machine learning community on settings which go *beyond iid data*. This has resulted in the proliferation of new concepts and settings such as out-of-distribution generalization [16], domain generalization [3], multi-task learning [41], continual learning [25] and etc. *Continual learning*, which is the focus of this paper, concerns learning through a sequence of environments, with the hope of retaining old knowledge while adapting to new environments.

Unfortunately, despite a lot of interest in the applied community—as evidenced by a multitude of NeurIPS and ICML workshops [26, 12, 30]—approaches with formal theoretical guarantees are few and far between. The main reason, similar encountered as its cousin fields like out-of-distribution generalization or multi-tasks learning, usually come with some "intuitive" desiderata — but no formal definitions. What's worse, it's often times clear that without strong data assumptions—the problem is woefully ill-defined.

The intuitive desiderata the continual learning community has settled on is that the setting involves cases where an algorithm is exposed (in an online fashion) to data sequentially coming from different distributions (typically called "environments", inspired from a robot/agent interacting with different environments). Moreover, the goal is to keep the size of the model being trained fixed, and make sure the model performs well on the current environment *while simultaneously* maintaining a good performance in the previously seen environments. In continual learning parlance, this is termed "resistance to catastrophic forgetting".

It is clear that some of the above desiderata are shared with well-studied learning theory settings (e.g. online learning, lifelong learning), while some aspects differ. For example, in online learning,

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

we don't care about catastrophic forgetting (or we only do so in some averaged sense); in lifelong learning, it's not necessary to keep the size of the model fixed. It is also clear that absent some assumptions on the data and the model being trained, these desiderata cannot possibly be satisfied: why would there even exist a model of some fixed size that performs well on both past environments, and current ones — let alone one that gets updated in an online fashion.

A feature-extraction formalization of continual learning Our paper formalizes a setting for continual learning through the lens of *feature extraction*: the model maintains a *fixed* number of (trainable) features, as well as a *linear classifier* on top of said features. The features are updated for every new environment, with the objective that the features are such that a good linear classifier exists for the *new* environment, while the previously trained linear classifiers (on the updated features) are still good for the past environments. The reason the linear classifiers from previous rounds are not allowed to be updated is storage efficiency: in order to tune the prompts, one needs to store the training data from previous tasks, this would bring a storage overhead and potentially privacy concerns. This is a very common approach in practice—examples of this are systems involving large amounts of data of a streaming nature (e.g. Google searches, Youtube, a robotic agent interacting with a continual stream of environments), and it be would prohibitive to store it for later fine tuning. The number of features is kept fixed for the same reason: if we are to expand with new features for every new environment, the model size (and hence storage requirements) would grow.

We prove two main results for our setting.

1. When the features are a linear function of the input data, and a good set of features exist, we design *an efficient algorithm*, named doubly projected gradient descent, or DPGrad, that has a good accuracy on all environments, and resists catastrophic forgetting. Our algorithm, while being novel, bears some resemblance to a class of projection-based algorithms used in practice [11, 5] – we hope the theoretical analysis can shed insight onto large scale continual learning.
2. When the features are allowed to be a non-linear function of the input, we show that continual learning is not possible—in general. Namely, we construct an instance for which even if a good set of features exists, the online nature of the setting, as well as the fact that the linear classifiers for past environments are not allowed to be updated, makes it possible for the algorithm to "commit" to linear classifiers, such that either catastrophic forgetting, or poor performance on the current environment has to occur.

2 Our results

2.1 Problem formulation

In a continual learning problem, the learner has sequential access to k environments. In the i-th ($i \in [k]$) environment, the data is drawn i.i.d. from the underlying distribution $\mathcal{D}_i$ over $\mathbb{R}^d \times \mathbb{R}$, denoted as $(x, y) \sim \mathcal{D}_i$, where $x \in \mathbb{R}^d$ is the input and $y \in \mathbb{R}$ is the label. Motivated by the empirical success of representation learning [2, 9], we formulate the continual learning problem through the feature extraction view: The learner is required to learn a common feature mapping (also known as representation function) $R : \mathbb{R}^d \to \mathbb{R}^r$ that maps the input data $x \in \mathbb{R}^d$ to a low dimensional representation $R(x) \in \mathbb{R}^r$ ($r \ll d$), along with a sequence of task-dependent linear classifiers (also known as linear prompts) $v_1, \dots, v_k \in \mathbb{R}^r$ on top of the representation. Precisely, the prediction of the i-th environment is made by $f(x) = \langle v_i, R(x) \rangle$.

As this is the first-cut study, we focus on the *realizable* and the *proper* learning setting.[1] That is, we assume the existence of a feature mapping R in the function class $\mathcal{H}$ (which is known in advance) and a sequence of linear predictor $v_1, \dots, v_k$ such that for any $i \in [k]$ and any data $(x, y) \sim \mathcal{D}_i$, $y = \langle v_i, R(x) \rangle$ (realizable). The learner is required to output a function R that belongs to the hypothesis class $\mathcal{H}$ (i.e. the learner is proper).

Remark 2.1 (Known environment identity). *Our model requires the knowledge of environment identity at test time, and thus can be classified into the category of incremental task learning. We*

[1] We note it is possible to extend our algorithmic result to the non-realizable setting, provided the label has symmetric sub-gaussian noise.

note there is also empirical research focusing on unknown environment identity, which would be an interesting direction for future work (See Section 7).

The guarantee that we wish our learning algorithm to obtain is as follows:

Definition 2.2 (Goal of Continual Learning)**.** *Let $d, k, r \in \mathbb{N}$, $r \ll d, k$, $\epsilon \in (0, 1/2)$. Let $\mathcal{H}$ be a function class in which the feature mappings from $\mathbb{R}^d$ to $\mathbb{R}^r$ lie. The continual learning problem involves k environments $\mathcal{D}_1, \ldots, \mathcal{D}_k$. We assume there exists a function $R^\star \in \mathcal{H}$ and a sequence of linear classifiers $v_1^\star, \ldots, v_k^\star \in \mathbb{R}^r$ such that for any $(x, y) \sim \mathcal{D}_i$ ($i \in [k]$), the label satisfies $y = \langle v_i^\star, R^\star(x)\rangle$.*

The continual learner has sequential access to environments $\mathcal{D}_1, \ldots, \mathcal{D}_k$, as well as access to arbitrarily many samples per environment[2]*. The goal is to learn a representation function $R \in \mathcal{H}$ and a sequence of linear prompts $v_1, \ldots, v_k \in \mathbb{R}^r$ that achieve a good accuracy on the current task and do not suffer from catastrophic forgetting. Formally, in the i-th environment ($i \in [k]$), the learner optimizes the feature mapping R and the linear classifier v_i (without changing $v_1, \ldots, v_{i-1}$) and aims to satisfy*

- **Avoid catastrophic forgetting***: During the execution of the i-th task, the algorithm guarantees that*

$$L(R, v_j) := \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_j} (\langle v_j, R(x)\rangle - y)^2 \le \epsilon \quad \textit{for all } j = 1, \ldots, i-1,$$

- **Good accuracy on the current task:** *At the end of i-th task, the algorithm guarantees that*

$$L(R, v_i) := \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i} (\langle v_i, R(x)\rangle - y)^2 \le \epsilon.$$

For linear feature mapping, the representation function can be written in a linear form $R(x) = U^\top x$ for some $U \in \mathbb{R}^{d\times r}$, and it implies the i-th environment is generated by a linear model. That is, defining $w_i = Uv_i \in \mathbb{R}^d$, one can write $y = \langle v_i, U^\top x\rangle = \langle w_i, x\rangle$.

Remark 2.3 (The benefit of continual learning with linear feature)**.** *Note, for linear features, it's in principle possible to just learn a sequence of linear classifiers $w_1, \ldots, w_k \in \mathbb{R}^d$ separately—without learning a low-dimensional featurizer. Choosing an r-dimensional featurizer confers memory efficiency ($O(kr + dr)$ vs. $O(dk)$) and sample efficiency ($O(r)$ vs. $O(d)$ samples per task in the asymptotic regime $k \to \infty$). Furthermore, the linear case is a sandbox that can be mathematically analyzed and can generate insights for the nonlinear case as well.*

2.2 DPGrad: Efficient gradient based method for linear features

For the case of linear features, we propose an efficient algorithm which we term DPGrad (pseudocode in Algorithm 1), which is an efficient gradient based method and provably learns the representation while avoids catastrophic forgetting. Towards stating the result, we make a few technical assumptions.

Assumption 2.4 (Distribution assumption)**.** *For any $i \in [k]$, we assume $\mathcal{D}_i$ has zero means and it is in isotropic position, that is, $\mathbb{E}_{x\sim\mathcal{D}_i}[x] = \vec{0}$ and $\mathbb{E}_{x\sim\mathcal{D}_i}[xx^\top] = I$.*

Remark 2.5. *This assumption is largely for convenience. In fact, one can replace the isotropic condition with a general bounded covariance assumption, our algorithm still can work with extra preprocessing step, and the sample complexity scales with the condition number of covariance matrix.*

Assumption 2.6 (Range assumption)**.** *For any $i \in [k]$, w_i has bounded norm, i.e., $\|w_i\|_2 \le D$.*

Assumption 2.7 (Signal assumption)**.** *For any $i \in [k]$, let $\mathsf{W}_i = \mathsf{span}(w_1, \ldots, w_i)$, $\mathsf{W}_{i,\perp}$ be the space perpendicular to W_i and P_{W_i}, $P_{\mathsf{W}_{i,\perp}}$ be the projection operator. We assume either w_i belongs to W_{i-1} or it has non-negligible component orthogonal to W_{i-1}, i.e., $\|P_{\mathsf{W}_{i-1,\perp}} w_i\|_2 \in \{0\} \cup [1/D, D]$.*

Assumption 2.8 (Bit complexity assumption)**.** *Each coordinate of w_i is a multiple of $\nu > 0$.*

Remark 2.9. *The Range and Signal assumptions are standard in the statistical learning literature. The former ensures an upper bound on $\|w_i\|_2$ and the later ensures that a new task provides enough "signal" for new features. They are used to set up learning rate and number of gradient iterations.*

[2] The results easily extend to the finite sample case using standard techniques. We focus on the population results to keep the focus on the online nature of the environments.

Remark 2.10. *The Bit complexity assumption states that w_i can be described with a finite number of bits, and is mostly for convenience — namely so we can argue we exactly recover w_i—which makes calculations involving projections of features learned in the past cleaner. Since the number of gradient iterations only depends* logarithmically *on ν, one can relax the bit complexity restriction to only approximately recovering the ground truth features up to a polynomially small (in d, k, D, ϵ) precision. Our argument can still go through at the cost of some additional error analysis.*

The main result is then as follows:

Theorem 2.11 (Continual learning with linear features)**.** *Let $k, d, r \in \mathbb{N}$, $r \ll k, d$, $\epsilon \in (0, 1/2)$. When the features are a linear function over the input data, under Assumption 2.4 and Assumption 2.6-2.8, with high probability,* DPGrad *provably achieves good loss and avoids catastrophic forgetting. In particular, during the execution of i-th environment,* DPGrad *always ensures*

$$L(U, v_j) := \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_j} (x^\top U v_j - y)^2 \leq \epsilon, \quad \textit{for all } j = 1, 2, \ldots, i-1, \tag{1}$$

and at the end of i-th environment, DPGrad *ensures*

$$L(U, v_i) = \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i} (x^\top U v_i - y)^2 \leq \epsilon. \tag{2}$$

2.3 Fundamental obstructions for non-linear features

The continual learning setup with non-linear features turns out to be much more difficult — even without computational constraints. Our result rules out the existence of a proper continual learner, even when all environment distributions are uniform and the representation function is realizable by a two-layer convolutional neural network.

Theorem 2.12 (Barrier for Continual learning with non-linear feature)**.** *Let $k, r \geq 2, d \geq 3$. There exists a class of non-linear feature mappings and a sequence of environments, such that there is no (proper) continual learning algorithm that can guarantee to achieve less than $\frac{1}{1000}$-error over all environments with probability at least $1/2$,under the feature extraction formalization of Definition 2.2.*

3 Related work

Continual learning in practice The study of continual learning (or lifelong learning) dates back to the work of [36] and it receives a surge of research interest over recent years [15, 19, 11, 5, 14, 31, 34, 17, 29, 39, 18]. A central challenge in the field is to avoid *catastrophic forgetting* [24, 23], which the work of [15] observed happened for gradient-based training of neural networks. While there is a large amount of empirical work, we'll briefly summarize the dominant approaches. The *regularization based approach* alleviates catastrophic forgetting by posing constraints on the update of the neural weights. The elastic weight consolidation (EWC) approach [19] adds weighted ℓ_2 regularization to the objective function that penalizes the movement of neural weights. The *orthogonal gradient descent* (OGD) algorithm from [11, 5] enforces the gradient update being orthogonal to update direction (by viewing the gradients as a high dimensional vector). The *memory replay approach* restores data from previous tasks and alleviates catastrophic forgetting by rehearsing in the later tasks. [31] introduces experience replay to continual learning. [14] trains a deep generative model (a.k.a. GAN) to simulate past dataset for future use. The *dynamic architecture approach* dynamically adjusts the neural network architecture to incorporate new knowledge and avoid forgetting. The progressive neural network [34] blocks changes to the existing network and expands the architecture by allocating a new subnet to be trained with the new information. We refer the interested reader to more complete surveys [25, 7].

Continual learning in theory In comparison to the vast empirical literature, theoretical works are comparatively few. The work of [6] characterize the memory requirement of continual learning, when the environment identity is unknown. The works [35, 27, 1, 4] provide sample complexity guarantees on lifelong learning. Their approaches can be categorized roughly into the *duplicate and fine-tuning* paradigm: The algorithm maintains a weighted combination over a family of representation functions and the focus is on the sample complexity guarantee. By contrast, we focus on the *feature extraction* paradigm and learn linear prompts on top of a *single* representation function. Both the duplicate-and-fine-tuning and the feature extraction paradigm have been extensively investigated in the literature, detailed discussions can be found at [7] and we provide a brief comparison. From an algorithmic perspective, learning a weighted combination over a family of representation functions (i.e. the duplicate and fine-tuning) is much easier, as one can always initiates a new representation function for a new task. The algorithmic convenience allows previous literature focus more on the generalization and sample complexity guarantee, culminating with the recent work of [4]. We note again that learning a single representation function and task specific linear prompts is much more challenging, but has practical benefits, e.g. memory efficiency. For example, in the applications of NLP, the basic representation function (e.g. BERT [9]) is already overparameterized and contains billions of parameters. It is then formidable to maintain a large amount of the basic models and learn a linear combination over them. We mention several more works that are morally related in Appendix A.

4 Continual learning with linear feature

We restate our main result for linear feature mapping.

Theorem 2.11 (Continual learning with linear features). *Let $k, d, r \in \mathbb{N}$, $r \ll k, d$, $\epsilon \in (0, 1/2)$. When the features are a linear function over the input data, under Assumption 2.4 and Assumption 2.6-2.8, with high probability,* DPGrad *provably achieves good loss and avoids catastrophic forgetting. In particular, during the execution of i-th environment,* DPGrad *always ensures*

$$L(U, v_j) := \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_j} (x^\top U v_j - y)^2 \le \epsilon, \quad \textit{for all } j = 1, 2, \ldots, i-1, \tag{1}$$

and at the end of i-th environment, DPGrad *ensures*

$$L(U, v_i) = \frac{1}{2} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i} (x^\top U v_i - y)^2 \le \epsilon. \tag{2}$$

4.1 Algorithm

A complete and formal description of DPGrad is presented in Algorithm 1. DPGrad simultaneously updates the matrix of features U, as well as the linear classifier v_i using gradient descent—with the restriction that the update of U only occurs along directions that are orthogonal to the column and row span of the previous feature matrix. Intuitively, one wishes the projection guarantees that existing features that have been learned are not erased or interfered by new environments. Due to the quadratic nature of the loss, and the appearance of "cross-terms", this turns out to require both column and row orthogonality, and interestingly deviates from the practically common OGD method [11, 5].

In more detail, at the beginning of the i-th ($i \in [k]$) environment, DPGrad adds Gaussian noise to the feature matrix U and the linear classifier v_i, to generate a good initialization for U and v_i. Subsequently, we perform gradient descent to both the feature mapping matrix U and linear classifier v_i—except U is only updated along orthogonal directions w.r.t. the column span and the row span. At the end of each environment, DPGrad has a post-processing step to recover the ground truth w_i by rounding each entry of Uv_i to the nearest multiple of ν,[3] and then update the column and row span if the orthogonal component is non-negligible. The reason for the later step is that we only need to preserve row space when encountering new features.

Parameters We use σ to denote the initialization scale, η to denote the learning rate, and T to denote the number of iterations for each task. These are all polynomially small parameters, whose scaling is roughly $D, d, k \ll \sigma^{-1} \ll \eta^{-1} < T$.

Notation We write $[n] = \{1, 2, \ldots, n\}$, $[n_1 : n_2] = \{n_1, \ldots, n_2\}$. We use $\mathsf{rand}(n_1, n_2) \in \mathbb{R}^{n_1 \times n_2}$ to denote a size $n_1 \times n_2$ matrix whose entries are draw from random Gaussian $\mathsf{N}(0, 1)$. For each $i \in [k]$, $t \in [0 : T]$, denote $U_{i,t}$ to be the feature matrix in the t-th iteration of the i-th environment (after performing the gradient update), denote $v_{i,t}$ similarly. DPGrad includes a projection step at the end of i-th environment, we use $U_{i,\mathsf{end}}$ to denote the feature matrix after this projection. We use W_i (resp. V_i) to denote the column (resp. row) space maintained at the end of i-th environment. Let $\mathsf{W}_\perp \subseteq \mathbb{R}^n$ be the subspace orthogonal to W and define $\mathsf{V}_\perp$ similarly. Let P_W, P_V, $P_{\mathsf{W}_\perp}$, $P_{\mathsf{V}_\perp}$ be the projection onto W, V, $\mathsf{W}_\perp$, $\mathsf{V}_\perp$ separately.

[3] This is the only place where we use the Bit complexity assumption.

Algorithm 1 Doubly projected gradient descent (DPGrad)

```
$\mathsf{W} \leftarrow \emptyset, \mathsf{V} \leftarrow \emptyset, U \leftarrow \mathbf{0}$    $\triangleright\ U \in \mathbb{R}^{d\times r}$
$\sigma \leftarrow \widetilde{O}(\frac{\epsilon}{d^2 k D^4}), \eta \leftarrow O(\frac{\sigma^3}{k^2 D^5}), T \leftarrow O(\frac{D}{\eta}\log\frac{Dkd}{\epsilon\nu}) + O(\frac{D}{\eta}\log\frac{k}{\sigma})$
for $i = 1, \ldots, k$ do
    $U_{\mathrm{init}} \leftarrow \sigma \cdot P_{\mathsf{W}_\perp}\mathsf{rand}(d, r)P_{\mathsf{V}_\perp}$, $v_i \leftarrow \sigma \cdot \mathsf{rand}(r)$    $\triangleright\ U_{\mathrm{init}} \in \mathbb{R}^{d\times r}$, $v_i \in \mathbb{R}^r$
    $U \leftarrow U + U_{\mathrm{init}}$
    for $t = 1, \ldots, T$ do
        $\nabla_U \leftarrow \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[x(x^\top U v_i - y)v_i^\top], \nabla_{v_i} \leftarrow \mathbb{E}_{(x,y)\sim\mathcal{D}_i}[U^\top x(x^\top U v_i - y)]$
        $U = U - \eta P_{\mathsf{W}_\perp}\nabla_U P_{\mathsf{V}_\perp}$
        $v_i = v_i - \eta\nabla_{v_i}$
    end for
end for
$\widehat{w}_i \leftarrow \mathsf{Round}_\nu(Uv_i)$    $\triangleright$ Round to the nearest multiple of $\nu$, $\widehat{w}_i \in \mathbb{R}^d$
if $\|P_{\mathsf{W}_\perp}\widehat{w}_i\|_2 \geq 1/D$ then $\mathsf{W} \leftarrow \mathsf{span}(\mathsf{W} \cup \widehat{w}_i)$, $\mathsf{V} \leftarrow \mathsf{span}(\mathsf{V} \cup v_i)$
$U \leftarrow P_{\mathsf{W}} U P_{\mathsf{V}}$
```

4.2 Analysis

We sketch the analysis of DPGrad and prove Theorem 2.11. Due to space limitation, the detailed proof is deferred to Appendix B. The proof proceeds in the following four steps:

1. The first step, presented in Section 4.2.1, reduces continual learning to a problem of continual matrix factorization and it allows us to focus on a more algebraically friendly objective function.
2. We then present some basic linear-algebraic facts to decompose the feature mapping matrix U, its gradient, and the loss into orthogonal components. The orthogonality of gradient update allows us to decouple the process of *leveraging the existing features* and the process of *learning a new feature*, as reflected in the loss terms and gradient update rules. See Section 4.2.2 for details.
3. In Section 4.2.3, we zoom into one single environment, and prove DPGrad provably converges to a global optimum, assuming the feature matrix U from previous environment is well conditioned. This step contains the major bulk of our analysis: The objective function of continual matrix factorization is non-convex, and no regularization or spectral initialization used. (We cannot re-initialize, lest we destroy progress from prior environments.)
4. Finally, in Section 4.2.4, we inductively prove that DPGrad converges and the feature matrix is always well-conditioned. This wraps up the entire proof.

4.2.1 Reduction

We first recall the formal statement of the problem of continual matrix factorization.

Definition 4.1 (Continual matrix factorization)**.** *Let $d, k, r \in \mathbb{N}$, $r \ll d, k$, $\epsilon > 0$. Let $W = [w_1, \ldots, w_k] = U^\star(V^\star)^\top \in \mathbb{R}^{d\times k}$, where $U^\star \in \mathbb{R}^{d\times r}, V^\star \in \mathbb{R}^{k\times r}$. In an continual matrix factorization problem, the algorithm receives $w_i \in \mathbb{R}^d$ in the i-th step, and it is required to maintain a matrix $U \in \mathbb{R}^{d\times r}$ and output a vector $v_i \in \mathbb{R}^r$ such that*

$$\widehat{L}(U, v_i) = \frac{1}{2}\|Uv_i - w_i\|_2^2 \leq \epsilon, \tag{3}$$

and

$$\widehat{L}(U, v_j) = \frac{1}{2}\|Uv_j - w_j\|_2^2 \leq \epsilon \quad j = 1, \ldots, i-1. \tag{4}$$

The key observation is that running DPGrad on the original continual learning objective (2) implicitly optimizes the continual matrix factorization objective (3) (Lemma 4.2). Moreover, an ϵ-approximate solution of continual matrix factorization is also an ϵ-approximate solution of continual learning (Lemma 4.3).

Lemma 4.2 (Gradient equivalence). *Under Assumption 2.4, for any $i \in [k]$, the gradient update of objective* (2) *equals the gradient update of objective* (3).

Lemma 4.3 (Objective equivalence). *For any $w_1, \ldots, w_k \in \mathbb{R}^d$, $U \in \mathbb{R}^{d\times r}$ and $v_1, \ldots, v_k \in \mathbb{R}^r$, suppose $\widehat{L}(U, v_i) = \frac{1}{2}\|Uv_i - w_i\|_2^2 \leq \epsilon$ holds for all $i \in [k]$, then $L(U, v_i) = \frac{1}{2}\mathbb{E}_{(x,y)\sim\mathcal{D}_i}(x^\top Uv_i - y)^2 \leq \epsilon$.*

Combining the above observations, it suffices to analyse DPGrad for continual matrix factorization and prove Eq. (3) and Eq. (4).

4.2.2 Decomposition

We first provide some basic linear algebraic facts about orthogonal decompositions. For any $i \in [k]$, we decompose U_i, v_i, w_i along W_{i-1}, $\mathsf{W}_{i-1,\perp}$, V_{i-1} and $\mathsf{V}_{i-1,\perp}$.

Let $w_i = w_{i,A} + w_{i,B}$ where $w_{i,A} \in \mathsf{W}_{i-1}$ and $w_{i,B} \in \mathsf{W}_{i-1,\perp}$. Note this decomposition is unique. We focus on the case that $\|w_{i,B}\|_2 \in [1/D, D]$ in the following statements, and the case of $\|w_{i,B}\|_2 = 0$ carries over easily. (These are the only two cases, per Assumption 2.7). Similarly, let $U_i = U_{i,A} + U_{i,B}$, where each column of $U_{i,A}$ lies W_{i-1} and each column of $w_{i,B}$ lies in $\mathsf{W}_{i-1,\perp}$. (Note, again, $U_{i,A}$ and $U_{i,B}$ are unique.) We further write $U_{i,B} = w_{i,B}x_i^\top + U_{i,2}$, where the columns of $U_{i,2}$ lie in $\mathsf{W}_{i-1,\perp}\backslash\{w_{i,B}\}$. Finally, denote $v_i = v_{i,1} + v_{i,2}$ with $v_{i,1} \in \mathsf{V}_{i-1}$ and $v_{i,2} \in \mathsf{V}_{i-1,\perp}$. We summarize decompositions mentioned above, with a few additional observations, in the lemma below:

Lemma 4.4 (Orthogonal decomposition). *For any $i \in [k]$ and any $t \in [0 : T]$, there exists an unique decomposition of $U_{i,t}$, w_i and $v_{i,t}$ of the form*

$$\begin{aligned}
U_{i,t} &= U_{i,A,0} + U_{i,B,t}, & \mathsf{column}(U_{i,A,0}) \in \mathsf{W}_{i-1}, \mathsf{column}(U_{i,B,t}) \in \mathsf{W}_{i-1,\perp},\\
&& \mathsf{row}(U_{i,A,0}) \in \mathsf{V}_{i-1}, \mathsf{row}(U_{i,B,t}) \in \mathsf{V}_{i-1,\perp}\\
w_i &= w_{i,A} + w_{i,B}, & w_{i,A} \in \mathsf{W}_{i-1}, w_{i,B} \in \mathsf{W}_{i-1,\perp}\\
U_{i,B,t} &= w_{i,B}x_{i,t}^\top + U_{i,2,t}, & x_{i,t} \in \mathsf{V}_{i-1,\perp}, \mathsf{row}(U_{i,2,t}) \in \mathsf{V}_{i-1,\perp}, w_{i,B} \perp \mathsf{column}(U_{i,2,t})\\
v_{i,t} &= v_{i,1,t} + v_{i,2,t} & v_{i,1,t} \in \mathsf{V}_{i-1}, v_{i,2,t} \in \mathsf{V}_{i-1,\perp}.
\end{aligned}$$

Here we use $\mathsf{column}(A)$, $\mathsf{row}(A)$ to denote the column and row space of matrix A, and $\mathsf{column}(A) \in \mathsf{W}$ if the column space of A is a subspace of W.

Since $U_{i,A,t}$ remains unchanged for $t = [0 : T]$, we abbreviate it as $U_{i,A}$ hereafter. We next provide the exact gradient update of each component under loss function $\widehat{L}(U_i, v_i) = \frac{1}{2}\|U_iv_i - w_i\|_2^2$ and orthogonal projection.

Lemma 4.5 (Gradient formula). *For any $i \in [k]$, the gradient update (after projection) obeys the relations (1) $\nabla_{x_i}(\widehat{L}) = v_{i,2}(x_i^\top v_{i,2} - 1)$; (2) $\nabla_{U_{2,i}}(\widehat{L}) = U_{i,2}v_{i,2}v_{i,2}^\top$; (3) $\nabla_{v_{i,1}}(\widehat{L}) = U_{i,A}^\top U_{i,A}v_{i,1} - U_{i,A}^\top w_{i,A}$ and (4) $\nabla_{v_{i,2}}(\widehat{L}) = \|w_{i,B}\|_2^2(x_i^\top v_{i,2} - 1)x_i + U_{i,2}^\top U_{i,2}v_{i,2}$.*

We perform a similar decomposition to the loss function.

Lemma 4.6 (Loss formula). *For any $i \in [k], t \in [T]$, we have*

$$\widehat{L}(U_{i,t}, v_{i,t}) = \frac{1}{2}\|U_{i,A}v_{i,1,t} - w_{i,A}\|_2^2 + \frac{1}{2}\|w_{i,B}\|_2^2(x_{i,t}^\top v_{i,2,t} - 1)^2 + \frac{1}{2}\|U_{i,2,t}v_{i,2,t}\|_2^2. \tag{5}$$

Decoupling existing features from "new" features We now offer some intuitive explanation for the decomposition. The first loss term in Eq. (5) quantifies the error with already learned features. That is, the matrix $U_{i,A}$ stores existing features that have been learned, and it remains unchanged during the execution of the i-th environment; it remains to optimize $v_{i,1,t}$ such that $U_{i,A}v_{i,1,t}$ matches $w_{i,A}$. The second and last loss term quantify the loss on a new feature, where$w_{i,B}$ is the new feature component, and the matrix $U_{i,2,t}$ can be thought of as random noise. Intuitively, one should hope $x_{i,t}^\top v_{i,2,t} = 1$ and this matches the new component of $w_{i,B}$. At the same time, one hopes $U_{i,2,t}$ would disappear, or at least, $\|U_{i,2,t}v_{i,2,t}\|_2 \to 0$ when $t \to \infty$.

4.2.3 Convergence

For a fixed environment, we prove w.h.p. DPGrad converges and the loss approaches to zero, given the initial feature mapping matrix $U_{i,A}$ is well conditioned.

Lemma 4.7. *For any $i \in [k]$, suppose $U_{i,A}$ satisfies $\frac{1}{2\sqrt{D}} \leq \sigma_{\min}(U_{i,A}) \leq \sigma_{\max}(U_{i,A}) \leq 2\sqrt{D}$, where $\sigma_{\min}(U_{i,A})$ and $\sigma_{\max}(U_{i,A})$ denote the minimum and maximum non-zero singular value of matrix $U_{i,A}$. After $T = O(\frac{D}{\eta}\log\frac{Dkd}{\epsilon\nu}) + O(\frac{D}{\eta}\log\frac{k}{\sigma})$ iterations, with probability at least $1 - O(1/k)$, the loss $\widehat{L}(U_i, v_i) \leq \epsilon\nu/Dnk$.*

Outline of the proof DPGrad ensures existing features are preserved and it only optimizes the linear classifier, hence a linear convergence rate can be easily derived for the first loss term, given the feature matrix is well-conditioned. The key part is controlling the terms that capture learning with new features, i.e., the second and last loss term, where both the feature mapping $U_{i,B}$ and linear prompt v_i get updated. In this case, the objective is non-convex and non-smooth. Our analysis draws inspiration from the recent work of [40], and divides the optimization process into two stages. We prove DPGrad first approaches to a nice initialization position with high probability, and then show linear convergence.[4]

To be concrete, in the first stage, we prove (1) $x_{i,t}^\top v_{i,2,t}$ moves closer to 1, and (2) $\|x_{i,t} - \|w_{i,B}\|_2 v_{i,2,t}\|_2 \approx 0$. That is, the second loss term of Eq. (5) decreases to a small constant while the pairs $x_{i,t}, v_{i,2,t}$ remain balanced and roughly equal up to scaling. Meanwhile, we note $U_{i,2,t}$ is non-increasing, though the last loss term could still increase because $\|v_{i,2,t}\|_2$ increases. In the second stage, we prove by induction that $\|U_{i,2,t}^\top v_{i,2,t}\|_2$ and $|x_{i,t}^\top v_{i,2,t} - 1|$ decay with a linear rate (hence converging to a global optimal), and $\|x_{i,t} - \|w_{i,B}\|_2 v_{i,2,t}\|_2 \approx 0$.

4.2.4 Induction

Lemma 4.7 proves rapid convergence of DPGrad for one single environment. To extend the argument to the whole sequence of environments, we need to ensure (1) the feature matrix is always well-conditioned and (2) catastrophic forgetting does not happen. For (1), we need to analyse the limiting point of DPGrad (there are infinitely many optimal solutions to Eq. (3)), make sure it is well-balance and orthogonal to previous row/column space. For (2), we make use of the orthogonality of DPGrad.

Proof Sketch of Theorem 2.11. Thanks to the reduction established in Section 4.2.1, it suffices to prove Eq. (3) and Eq. (4). For each environment i ($i \in [k]$), we inductively prove (1) DPGrad achieves good accuracy on the current environment, i.e., $\|U_{i,T}v_i - w_i\|_2 \leq \epsilon\nu$; (2) The feature matrix U_i remains well conditioned, i.e. $\frac{1}{2\sqrt{D}} \leq \sigma_{\min}(U_{i,\mathsf{end}}) \leq \sigma_{\max}(U_{i,\mathsf{end}}) \leq 2\sqrt{D}$ and (3) The algorithm does not suffer from catastrophic forgetting, i.e., $\|U_{i,t}v_j - w_i\|_2 \leq \epsilon$ for any $j < i, t \in [T]$.

The first claim is already implied by Lemma 4.7. For the second claim, one first shows DPGrad exactly recovers w_i by taking $w_i = \widehat{w}_i = \mathsf{Round}_\nu(U_{i,T}v_i)$. When $w_{i,B} = 0$, one can prove the feature matrix does not change, i.e, $U_{i,\mathsf{end}} = U_{i-1,\mathsf{end}}$; when $w_{i,B} \in [1/D, D]$, then one can show $U_{i,\mathsf{end}} \approx U_{i,\mathsf{end}} + \frac{1}{\|v_{i,2,T}\|_2^2} w_{i,B} v_{i,2,T}^\top$, as $w_{i,B} \perp \mathsf{column}(U_{i-1,\mathsf{end}}), v_{i,2,T} \perp \mathsf{row}(U_{i-1,\mathsf{end}})$ and $\|\frac{1}{\|v_{i,2,T}\|_2^2} w_B v_{i,2,T}^\top\| \leq O(\sqrt{D})$, the feature matrix U remains well-conditioned. The last claim can be derived from the orthogonality. This wraps up the proof of Theorem 2.11. □

5 Lower bound for non-linear features

We next consider continual learning under a non-linear feature mapping. Learning with non-linear features turns out to be much more difficult, and our main result is to rule out the possibility of a (proper) continual learner. We restate the formal statement. The detailed proof are deferred to Appendix C.

[4] We note most existing works on matrix factorization or matrix sensing either require some fine-grained initialization (e.g. spectral initialization [8]) or adding a regularization term that enforces smoothness [13], none of which are applicable in our setting.

Theorem 2.12 (Barrier for Continual learning with non-linear feature)**.** *Let $k, r \geq 2, d \geq 3$. There exists a class of non-linear feature mappings and a sequence of environments, such that there is no (proper) continual learning algorithm that can guarantee to achieve less than $\frac{1}{1000}$-error over all environments with probability at least $1/2$,under the feature extraction formalization of Definition 2.2.*

Our lower bound is constructed on a simple family of two-layer convolutional neural network with quadratic activation functions. The input distribution is assumed to be uniform and the target function is a polynomial over the input. The first environment is constructed such that multiple global optimum exist (hence the optimization task is under-constrained). However, if a wrong optimum solution is picked, when the second environment is revealed, the non-linearity makes it impossible to switch back-and-forth.

Proof Sketch. It suffices to take $k = 2, d = 3, r = 2$ and we sketch the construction here. For both environments, we assume the input data are drawn uniformly at random from $\mathcal{B}_3(0,1)$, where $\mathcal{B}_3(0,1)$ denotes the unit ball in $\mathbb{R}^3$ centered at origin. The hypothesis class $\mathcal{H}$ consists of all two-layer convolutional neural network with a single kernel of size 2 and the quadratic activation function. That is, the representation function is parameterized by $w \in \mathbb{R}^2$ and takes the form of $R_w(x) = (\langle w, x_{1:2}\rangle^2, \langle w, x_{2:3}\rangle^2) \in \mathbb{R}^2$, where $x \in \mathbb{R}^3$, $x_{i:j} \in \mathbb{R}^{j-i+1}$ is a vector consists of the i-th entry to the j-th entry of x.

The hard sequence of environments are drawn from the following distribution: (1) The objective function f_1 of the first environment is $f_1(x) = x_2^2$; (2) The objective function f_2 of the second environment equals $f_2(x) = x_3^2$ with probability $1/2$, and equals $f_2(x) = x_1^2$ with probability $1/2$. Note the continual learning task is realizable and one can prove no (proper) continual learning algorithm can guarantee to achieve less than $1/1000$-error on both environments with probability at least $1/2$. □

Though our lower bound instance uses a polynomial activation function, this assumption is not essential – in Appendix C, we prove similar lower bounds with a ReLU activation function.

6 Experiments

For linear feature functions, we perform simulations on a synthetic dataset to verify the practicality of DPGrad and compare its performance with vanilla SGD and Orthogonal gradient descent (OGD), a close practical cousin of our algorithm. In our simulations, we set $d = 100$, $r = 20$, $k = 500$ and the ground truth $U^\star, V^\star$ is drawn from Gaussian. The input data are sampled from $N(0, I_d)$ and we draw 1000 samples for each task. Additional details about the setup can be found in Appendix D.

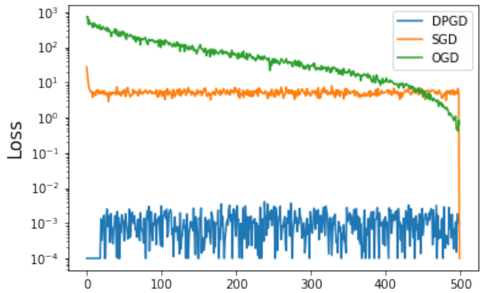


Figure 1: Continual learning with linear feature: comparative performance of DPGrad/OGD/SGD. Data is synthetically generated with $d = 100$, $r = 20$, $k = 500$ and the ground truth $U^\star, V^\star$ is drawn from Gaussian. Additional details about the setup can be found in Appendix D.

The results are presented at Figure 1. It indicates the (1) practicality of DPGrad and (2) DPGrad significantly outperforms the vanilla SGD and OGD (of course, DPGrad is designed for this kind of data). The population loss is measured at the end and the it equals $\|Uv_i - w_i\|_2$ for each task i. The average error of DPGrad is 0.001, the average error of OGD is 83.59, the average error of SGD is 5.16.

Moreover, in Appendix E, we provide additional experimental results on two popular benchmarks, Rotated MNIST and Permuted MNIST. Since DPGrad is designed specifically for linear regression, we provide two variants of DPGrad (without provable guarantees on their performance, of course)—one is a modification suitable for multi-class classification, the other is a modification suitable for non-linear featurizers. Detailed numbers and figures can be found in Appendix E. In brief, both algorithms alleviate catastrophic forgetting and perform much better than vanilla SGD. Furthermore, the performance of both is much more stable than OGD and the accuracy remains at a high level across tasks.

7 Conclusion

In this paper, we initiate a study of continual learning through *the feature extraction lens*, proposing an efficient gradient based algorithm, DPGrad, for the linear case, and a fundamental impossibility result in the general case. Our work leaves several interesting future directions. First, it would be interesting to generalize DPGrad to non-linear feature mappings (perhaps even without provable guarantees) and conduct an empirical study of its performance. Second, our impossibility result does not rule out an improper continual learner, and in general, one can always maintain a task specific representation function and achieve good performance over all environments. It would be thus interesting to investigate what are the fundamental memory-accuracy trade-offs.

Acknowledgement

BP acknowledges support of the NSF via CCF-1909756, CCF2007443, CCF-2134105, CCF-1703925, IIS-1838154, CCF-2106429 and CCF-2107187. AR acknowledges support of the NSF via IIS-2211907, the CMU/PwC DT&I Center, and an Amazon Research Award.

References

[1] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, pages 191–210. PMLR, 2015.

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[3] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[4] Xinyuan Cao, Weiyang Liu, and Santosh S Vempala. Provable lifelong learning of representations. *arXiv preprint arXiv:2110.14098*, 2021.

[5] Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33, 2020.

[6] Xi Chen, Christos Papadimitriou, and Binghui Peng. Memory bounds for continual learning. In *2022 IEEE 63th Annual Symposium on Foundations of Computer Science (FOCS)*, 2022.

[7] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

[8] Yuejie Chi, Yue M Lu, and Yuxin Chen. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269, 2019.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[10] Simon Shaolei Du, Wei Hu, Sham M Kakade, Jason D Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2020.

[11] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

[12] Haytham Fayek, Arslan Chaudhry, David Lopez-Paz, Eugene Belilovsky, Jonathan Schwarz, Marc Pickett, Rahaf Aljundi, Sayna Ebrahimi, Razvan Pascanu, and Puneet Dokania. Icml workshop on continual learning, 2020.

[13] Rong Ge, Chi Jin, and Yi Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *International Conference on Machine Learning*, pages 1233–1242. PMLR, 2017.

[14] Alexander Gepperth and Cem Karaoguz. A bio-inspired incremental learning architecture for applied perceptual problems. *Cognitive Computation*, 8(5):924–934, 2016.

[15] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[16] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[17] Khurram Javed and Martha White. Meta-learning representations for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[18] Ghassen Jerfel, Erin Grant, Tom Griffiths, and Katherine A Heller. Reconciling meta-learning and continual learning with online mixtures of tasks. *Advances in Neural Information Processing Systems*, 32, 2019.

[19] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[20] Andreas Maurer. Transfer bounds for linear feature learning. *Machine learning*, 75(3):327–350, 2009.

[21] Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *International conference on machine learning*, pages 343–351. PMLR, 2013.

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

[23] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[24] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[25] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[26] Razvan Pascanu, Yee Teh, Marc Pickett, and Mark Ring. Neurips workshop on continual learning, 2018.

[27] Anastasia Pentina and Ruth Urner. Lifelong learning with weighted majority votes. *Advances in Neural Information Processing Systems*, 29, 2016.

[28] Massimiliano Pontil and Andreas Maurer. Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory*, pages 55–76. PMLR, 2013.

[29] Vinay Venkatesh Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *International Conference on Learning Representations*, 2020.

[30] Amal Rannen-Triki, Arslan Chaudhry, Bogdan Mazoure, Xu He, Thang Doan, Rahaf Aljundi, and Vincenzo Lomonaco. Icml workshop on theory and foundation of continual learning, 2021.

[31] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[32] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. An online learning approach to interpolation and extrapolation in domain generalization. *arXiv preprint arXiv:2102.13128*, 2021.

[33] Elan Rosenfeld, Pradeep Kumar Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. In *International Conference on Learning Representations*, 2020.

[34] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[35] Paul Ruvolo and Eric Eaton. Ella: An efficient lifelong learning algorithm. In *International conference on machine learning*, pages 507–515. PMLR, 2013.

[36] Sebastian Thrun and Tom M Mitchell. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46, 1995.

[37] Nilesh Tripuraneni, Chi Jin, and Michael Jordan. Provable meta-learning of linear representations. In *International Conference on Machine Learning*, pages 10434–10443. PMLR, 2021.

[38] Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer learning: The importance of task diversity. *Advances in Neural Information Processing Systems*, 33:7852–7862, 2020.

[39] Gido M van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):1–14, 2020.

[40] Tian Ye and Simon S Du. Global convergence of gradient descent for asymmetric low-rank matrix factorization. *Advances in Neural Information Processing Systems*, 34, 2021.

[41] Yu Zhang and Qiang Yang. An overview of multi-task learning. *National Science Review*, 5(1):30–43, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Additional related work

Representation learning More broadly, our work is also closely related to representation learning. Some recent theoretical works [20, 21, 28, 38, 22, 37, 10] provide generalization and sample complexity guarantees for certain formalizations of multi-task learning based on the existence of a good representation. The work of [33, 32] formulate the problem of out-of-distribution generalization and provide theoretical guarantee, similarly, under the assumption of a good representation.

B Missing proof from Section 4

B.1 Missing proof from Section 4.2.1

We first present the proof of Lemma 4.2

Proof of Lemma 4.2. For any $i \in [k]$, the gradient of feature matrix U w.r.t. objective Eq. (2) equals

$$\nabla_U = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i}[x(x^\top U v_i - y)v_i^\top] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_i}[x(x^\top U v_i - x^\top w_i)v_i^\top] = (Uv_i - w_i)v_i^\top. \quad (6)$$

The first step follows from $y = x^\top w_i$ for any $(x, y) \sim \mathcal{D}_i$ and the second step follows from $\mathbb{E}_{x_i\sim\mathcal{D}_i}[xx^\top] = I_n$. The RHS of the above equation exactly equals the gradient of Eq. (3) for U (before and after projection to W_{i-1}).

We next observe

$$\nabla_{v_i} = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i}[U^\top x(x^\top U v_i - y)] = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_i}[U^\top x(x^\top U v - x^\top w_i)] = U^\top(Uv_i - w_i), \quad (7)$$

and the RHS of the above equation matches the gradient of Eq. (3) for v_i. We conclude the proof here. □

We then include the proof of Lemma 4.2

Proof of Lemma 4.3. We have

$$\begin{aligned}\frac{1}{2}\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i}(x^\top U v_i - y)^2 &= \frac{1}{2}\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_i}(x^\top U v_i - x^\top w_i)^2 \\ &= \frac{1}{2}(Uv_i - w)^\top \mathop{\mathbb{E}}_{x\sim\mathcal{D}_i}[xx^\top](Uv_i - w)^\top \\ &= \frac{1}{2}\|Uv_i - w_i\|_2^2 \leq \epsilon.\end{aligned}$$

where the first step follows from $y = x^\top w_i$ for any $(x, y) \sim \mathcal{D}_i$ and the third step follows from $\mathbb{E}_{x_i\sim\mathcal{D}_i}[xx^\top] = I_n$. This concludes the proof. □

B.2 Missing proof from Section 4.2.2

We first present the proof of Lemma 4.4

Proof of Lemma 4.4. For the first term, when $t = 0$, one has $\mathsf{column}(U_{i,A,0}) \in \mathsf{W}_{i-1}$ and $\mathsf{column}(U_{i,B,0}) \in \mathsf{W}_{i-1,\perp}$, and these indicate (1) $U_{i,A,0} = U_{i-1,\mathsf{end}}$, $\mathsf{row}(U_{i-1,\mathsf{end}}) \in \mathsf{V}_{i-1}$ and (2) $U_{i,B,0} = U_{i,\mathrm{init}}$, $\mathsf{row}(U_{i,\mathrm{init}}) \in \mathsf{V}_{i-1,\perp}$. Hence we conclude $\mathsf{row}(U_{i,A,0}) \in \mathsf{V}_{i-1}$ and $\mathsf{row}(U_{i,A,0}) \in \mathsf{V}_{i-1,\perp}$. Since the gradient update is perform along $\mathsf{W}_{i-1,\perp}$ and $\mathsf{V}_{i-1,\perp}$, one has $U_{i,A}$ remains unchanged, i.e., $U_{i,A,t} = U_{i,A,0}$ ($t \in [T]$), and the update of $U_{i,B,t}$ is along $V_{i-1,\perp}$, hence $\mathsf{row}(U_{i,B,t}) \in \mathsf{V}_{i-1,\perp}$ continues to hold.

For the third term, for any $t \in [0 : T]$, one has

$$\mathsf{V}_{i-1,\perp} \ni w_{i,B}^\top U_{i,B,t} = w_{i,B}^\top w_{i,B} x_{i,t}^\top + w_{i,B}^\top U_{i,2,t} = \|w_{i,B}\|_2^2 x_{i,t}^\top,$$

where the second step follows from $\mathsf{column}(U_{i,2,t}) \in \mathsf{W}_{i-1,\perp}\backslash\{w_{i,B}\}$. Hence we conclude $x_{i,t} \in V_{i-1,\perp}$. Since $\mathsf{row}(U_{i,B,t})$, $\mathsf{row}(w_{i,B}x_{i,t}^\top) \in \mathsf{V}_{i-1,\perp}$, one has $\mathsf{row}(U_{i,2,t}) \in V_{i-1,\perp}$. □

We then prove

Proof of Lemma 4.5. The gradient of U (before projection) satisfies

$$\begin{aligned}\nabla_{U_i} &= (U_i v_i - w_i) v_i^\top \\ &= U_{i,A} v_i v_i^\top + U_{i,B} v_i v_i^\top - w_{i,A} v_i^\top - w_{i,B} v_i^\top \\ &= (U_{i,A} v_i v_i^\top - w_{i,A} v_i^\top) + (w_{i,B} x_i^\top + U_{i,2}) v_i v_i^\top - w_{i,B} v_i^\top \\ &= (U_{i,A} v_i v_i^\top - w_{i,A} v_i^\top) + w_{i,B} v_i^\top (x_i^\top v_i - 1) + U_{i,2} v_i v_i^\top,\end{aligned}$$

where the first step follows from Eq. (6), the second and third steps follow from the first three terms of Lemma 4.4.

The actual update (after projection) obeys

$$\begin{aligned}P_{\mathsf{W}_{i-1,\perp}} \nabla_{U_i} P_{\mathsf{V}_{i-1,\perp}} &= P_{W_{i-1,\perp}} ((U_{i,A} v_i v_i^\top - w_{i,A} v_i^\top) + w_{i,B} v_i^\top (x_i^\top v_i - 1) + U_{i,2} v_i v_i^\top) P_{\mathsf{V}_{i-1\perp}} \\ &= (w_{i,B} v_i^\top (x_i^\top v_i - 1) + U_2 v_i v_i^\top) P_{\mathsf{V}_{i-1\perp}} \\ &= w_{i,B} v_{i,2}^\top (x_i^\top v_{i,2} - 1) + U_{i,2} v_{i,2} v_{i,2}^\top,\end{aligned}$$

where the second step follows from $w_{i,A}, \mathsf{column}(U_{i,A}) \in \mathsf{W}_{i-1}$, the third step follows from $\mathsf{row}(U_{i,2}) \in \mathsf{V}_{i-1,\perp}$ and $x_i \in \mathsf{V}_{i-1,\perp}$, see Lemma 4.4 for details.

Hence, we conclude

$$\nabla_{x_i} = v_{i,2}(x_i^\top v_{i,2} - 1) \quad \text{and} \quad \nabla_{U_{i,2}} = U_{i,2} v_{i,2} v_{i,2}^\top.$$

We next calculate the gradient of v, it satisfies

$$\begin{aligned}\nabla_{v_i} &= U_i^\top (U_i v_i - w_i) \\ &= U_{i,A}^\top U_{i,A} v_i + U_{i,B}^\top U_{i,B} v_i - U_{i,A}^\top w_{i,A} - U_{i,B}^\top w_{i,B} \\ &= U_{i,A}^\top U_{i,A} v_{i,1} - U_{i,A}^\top w_{i,A} + U_{i,B}^\top U_{i,B} v_{i,2} - U_{i,B}^\top w_{i,B}.\end{aligned}$$

The first step follows from Eq. (7), the second step follows from the first two terms of Lemma 4.4. The third step uses the fact that $\mathsf{row}(U_{i,A}) \in \mathsf{V}_{i-1}$, $v_{i,1} \in \mathsf{V}_{i-1}$, $v_{i,2} \in \mathsf{V}_{i-1,\perp}$ and $\mathsf{row}(U_{i,B}) \in \mathsf{V}_{i-1,\perp}$

Hence, we have

$$\nabla_{v_{i,1}} = U_{i,A}^\top U_{i,A} v_{i,1} - U_{i,A}^\top w_{i,A}$$

and

$$\begin{aligned}\nabla_{v_{i,2}} &= U_{i,B}^\top U_{i,B} v_{i,2} - U_{i,B}^\top w_{i,B} \\ &= (w_{i,B} x_i^\top + U_{i,2})^\top (w_{i,B} x_i^\top + U_{i,2}) v_{i,2} - (w_{i,B} x_i^\top + U_{i,2})^\top w_{i,B} \\ &= x_i \|w_{i,B}\|_2^2 x_i^\top v_{i,2} + U_{i,2}^\top U_{i,2} v_{i,2} - x_i \|w_{i,B}\|_2^2 \\ &= \|w_{i,B}\|_2^2 (x_i^\top v_i - 1) x_i + U_{i,2}^\top U_{i,2} v_{i,2},\end{aligned}$$

where the third step holds due to $w_{i,B} \perp \mathsf{column}(U_{i,2})$. We conclude the proof here. □

Finally, we prove

Proof of Lemma 4.6. For any $i \in [k], t \in [T]$, we have

$$\begin{aligned}\|U_{i,t} v_{i,t} - w_i\|_2^2 &= \|(U_{i,A} + U_{i,B,t})(v_{i,1,t} + v_{i,2,t}) - w_{i,A} - w_{i,B}\|_2^2 \\ &= \|U_{i,A} v_{i,1,t} + U_{i,B,t} v_{i,2,t} - w_{i,A} - w_{i,B}\|_2^2 \\ &= \|U_{i,A,t} v_{i,1,t} - w_{i,A}\|_2^2 + \|U_{i,B,t} v_{i,2,t} - w_{i,B}\|_2^2 \\ &= \|U_{i,A} v_{i,1,t} - w_{i,A}\|_2^2 + \|(w_{i,B} x_{i,t}^\top + U_{i,2,t}) v_{i,2,t} - w_{i,B}\|_2^2 \\ &= \|U_{i,A} v_{i,1,t} - w_{i,A}\|_2^2 + \|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1)^2 + \|U_{i,2,t} v_{i,2,t}\|_2^2.\end{aligned}$$

The second step follows from $\mathsf{row}(U_{i,A}) \in \mathsf{V}_{i-1}$, $\mathsf{row}(U_{i,B}) \in \mathsf{V}_{i-1,\perp}$, $v_{i,1,t} \in \mathsf{V}_{i-1}$, $v_{i,2,t} \in \mathsf{V}_{i-1,\perp}$, the third step follows from $U_{i,A} v_{i,1,t} - w_{i,A} \in \mathsf{W}_{i-1}$ and $U_{i,B,t} v_{2,t} - w_{i,B} \in \mathsf{W}_{i-1,\perp}$. The last step follows from $w_{i,B} \perp \mathsf{column}(U_{i,2,t})$. □

B.3 Missing proof from Section 4.2.3

In the proof, we write $x = y \pm z$ if $x \in [y - z, y + z]$. For simplicity, we assume $\log(1/\epsilon\nu) \ll k, d$.

First, we prove linear convergence for the first loss term.

Lemma B.1 (Fast learning on existing features)**.** *For any $i \in [k]$ and $t \in [T]$, we have*

$$\|U_{i,A}v_{i,1,t} - w_{i,A}\|_2 \leq \left(1 - \frac{\eta}{4D}\right)^t \|U_{i,A}v_{i,1,0} - w_{i,A}\|_2.$$

Proof of Lemma B.1. This follows easily from the standard analysis of gradient descent for least square regressions. For any $t \in [0 : T-1]$, one has

$$\begin{aligned}
\|U_{i,A}v_{i,1,t+1} - w_{i,A}\|_2 &= \|U_{i,A}(v_{i,1,t} - \eta(U_{i,A}^\top U_{i,A}v_{i,1,t} - U_{i,A}^\top w_{i,A})) - w_{i,A}\|_2 \\
&= \|(I - \eta U_{i,A}U_{i,A}^\top)(U_{i,A}v_{i,1,t} - w_{i,A})\|_2 \\
&\leq (1 - \frac{\eta}{4D})\|(U_{i,A}v_{i,1,t} - w_{i,A})\|_2.
\end{aligned}$$

The first step follows from the gradient update formula (see Lemma 4.5), the third step follows from $U_{i,A}v_{i,1,t} - w_{i,A} \in \mathsf{column}(U_{i,A})$, and $2\sqrt{D} \geq \sigma_{\max}(U_{i,A}) \geq \sigma_{\min}(U_{i,A}) \geq \frac{1}{2\sqrt{D}}$ and $\eta < \frac{1}{4D}$. We conclude the proof here. □

We next focus on the second and last loss terms. One can show that $x_{i,t}^\top v_{i,2,t}$ moves to 1 while $\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2$ remains small in the first $T_1 = O(\frac{D}{\eta}\log\frac{k}{\sigma})$ iterations.

Lemma B.2. *With probability at least $1 - O(1/k)$ over the random initialization, there exists $T_1 = O(\frac{D}{\eta}\log\frac{k}{\sigma})$, such that for any $t \leq T_1$, one has (1) $\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2 \leq O(r\sigma\log(k/\sigma))$; (2) $x_{i,t}^\top v_{i,2,t} < 0.9$ when $t < T_1$ and $0.9 < x_{i,T_1}^\top v_{i,2,T_1} < 1$; (3) $U_{i,2,t}^\top U_{i,2,t} \preceq U_{i,2,0}^\top U_{i,2,0}$.*

Proof of Lemma B.2. Recall our goal is to prove

1. $\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2 \leq O(r\sigma\log(k/\sigma))$,
2. $x_{i,t}^\top v_{i,2,t} < 0.9$ when $t < T_1$ and $0.9 < x_{i,T_1}^\top v_{i,2,T_1} < 1$,
3. $U_{i,2,t}^\top U_{i,2,t} \preceq U_{i,2,0}^\top U_{i,2,0}$.

We inductively prove these three claims. For the base case, we have that

$$\begin{aligned}
x_{i,0} = \frac{1}{\|w_{i,B}\|_2^2} w_{i,B}^\top U_{i,B,0} &= \frac{1}{\|w_{i,B}\|_2^2} w_{i,B}^\top P_{\mathsf{W}_{i-1,\perp}} U_{i,\mathrm{init}} P_{\mathsf{V}_{i-1,\perp}} = \frac{1}{\|w_{i,B}\|_2^2} w_{i,B}^\top U_{i,\mathrm{init}} P_{\mathsf{V}_{i-1,\perp}} \\
&\approx \frac{\sigma}{\|w_{i,B}\|_2} \cdot \mathsf{rand}(r,1) P_{\mathsf{V}_{i-1\perp}},
\end{aligned} \tag{8}$$

where in the first step we use the fact that $U_{i,B,0} = w_{i,B}x_{i,0}^\top + U_{i,2,0}$, $w_{i,B} \perp \mathsf{column}(U_{i,2,0})$, in the third step, we use $w_{i,B} \in \mathsf{W}_{i-1,\perp}$. The fourth step follows from $\frac{1}{\|w_{i,B}\|_2^2} w_{i,B}^\top U_{i,\mathrm{init}}$ is a random Gaussian vector with variance $\frac{\sigma}{\|w_{i,B}\|_2}$. Similarly, we have

$$v_{i,2,0} = \sigma \cdot \mathsf{rand}(r,1) P_{\mathsf{V}_{i-1,\perp}}. \tag{9}$$

Hence, with probability at least $1 - O(1/k)$, we have

$$\|\|w_{i,B}\|_2 x_{i,0} - v_{i,2,0}\|_2 \leq O(r\sigma\log(k)) \quad \text{and} \quad x_{i,0}^\top v_{i,2,0} < O(\sigma^2 rD\log(k)) \ll 1.$$

We have proved the base case. Now suppose the induction holds up to time t, for the $(t+1)$-th iteration, we first go over the first claim. One has

$$\begin{aligned}
&\|\|w_{i,B}\|_2 x_{i,t+1} - v_{i,2,t+1}\|_2^2 - \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 \\
={}& \|\|w_{i,B}\|_2 (x_{i,t} - \eta v_{i,2,t}(x_{i,t}^\top v_{i,2,t} - 1)) - (v_{i,2,t} - \eta x_{i,t}\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) - \eta U_{i,2,t}^\top U_{i,2,t} v_{i,2,t})\|_2^2 \\
&- \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 \\
={}& \|(\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}) - \eta v_{i,2,t}\|w_{i,B}\|_2 (x_{i,t}^\top v_{i,2,t} - 1) + \eta x_{i,t}\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) + \eta U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\|_2^2 \\
&- \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 \\
={}& 2\eta\langle \|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}, x_{i,t}\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) - v_{i,2}\|w_{i,B}\|_2 (x_{i,t}^\top v_{i,2,t} - 1) + U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\rangle \\
&\pm O(\eta^2 D^4) \\
={}& 2\eta \|w_{i,B}\|_2 (x_{i,t}^\top v_{i,2,t} - 1)\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 + \eta\langle \|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}, U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\rangle \\
&\pm O(\eta^2 D^4) \qquad (10) \\
\le{}& 2\eta\langle \|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}, U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\rangle \pm O(\eta^2 D^4) \\
\le{}& \widetilde{O}(\eta r \sigma^3 d^2 D) + O(\eta^2 D^4). \qquad (11)
\end{aligned}$$

The first step follows from the gradient update formula (see Lemma 4.5), the third step follows from that

$$\|U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\|_2 \ll 1, \quad \|\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) x_{i,t}\|_2 \le O(D^2)$$

and

$$\|\|w_{i,B}\|_2 (x_{i,t}^\top v_{i,2,t} - 1) v_{i,2,t}\|_2 \le O(D^2),$$

which can be derived easily from the induction hypothesis. The fifth step follows from $x_{i,t}^\top v_{i,2,t} < 1$ when $t \le T_1$. The last step follows from

$$\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\| \le \widetilde{O}(r\sigma), \|U_{i,2,t}^\top U_{i,2,t}\| \le \|U_{i,2,0}^\top U_{i,2,0}\| \le \widetilde{O}(d^2\sigma^2), \|v_{i,2,t}\|_2 \le O(D), \tag{12}$$

which can be derived easily from the induction hypothesis. Combining with $\eta \le \frac{\sigma^2}{D^5}$, $\sigma \le \frac{1}{D^2 d^2}$ and the total number of iteration is $T_1 \le O(\frac{D}{\eta}\log\frac{k}{\sigma})$, one can proved the first claim.

For the second claim, we have that

$$\begin{aligned}
&x_{i,t+1}^\top v_{i,2,t+1} - x_{i,t}^\top v_{i,2,t} \\
={}& (x_{i,t} - \eta v_{i,2,t}(x_{i,t}^\top v_{i,2,t} - 1))^\top (v_{i,2,t} - \eta x_{i,t}\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) - \eta U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}) - x_{i,t}^\top v_{i,2,t} \\
={}& -\eta(\|w_{i,B}\|_2^2 \|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2)(x_{i,t}^\top v_{i,2,t} - 1) - \eta x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t} v_{i,2,t} \pm O(\eta^2 D^3) \qquad (13) \\
\ge{}& \frac{1}{2}\eta(\|w_{i,B}\|_2^2 \|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2)(x_{i,t}^\top v_{i,2,t} - 1) - O(\eta^2 D^3) \\
\ge{}& \frac{1}{20}\eta(\|w_{i,B}\|_2^2 \|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2) - O(\eta^2 D^3). \qquad (14)
\end{aligned}$$

The first step follows from the gradient update formula (see Lemma 4.5), the second step holds since

$$\|v_{i,2,t}(x_{i,t}^\top v_{i,2,t} - 1))\|_2 \le O(D), \quad \|\|w_{i,B}\|_2^2 (x_{i,t}^\top v_{i,2,t} - 1) x_{i,t}\|_2 \le O(D^2) \quad \text{and} \quad \|U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\|_2 \ll 1.$$

Again, these inequalities can be derived easily from the inductive hypothesis. The third step holds since $U_{i,2,t}^\top U_{i,2,t} \preceq U_{i,2,t}^\top U_{i,2,t} \preceq \widetilde{O}(d^2\sigma^2) \cdot I$, and therefore,

$$|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t} v_{2,t}| \le \widetilde{O}(d^2\sigma^2) \cdot \|x_{i,t}\|_2 \|v_{i,2,t}\|_2 \ll |(\|w_{i,B}\|_2^2 \|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2)(x_{i,t}^\top v_{i,2,t} - 1)|.$$

The last step uses the fact that $x_{i,t}^\top v_{2,t} < 0.9$ when $t < T_1$.

We next bound the RHS of Eq. (14) and prove it can not be too small. We focus on $\|\|w_{i,B}\|_2 x_{i,t+1} + v_{i,2,t+1}\|_2$ and prove it monotonically increasing. In particular, at initialization, with probability at least $1 - O(1/k)$, due to anti-concentration of Gaussian, we have

$$\|\|w_{i,B}\| x_{i,0} + v_{i,2,0}\|_2 \approx \sigma\|\mathsf{rand}(r,1) P_{\mathsf{V}_{i-1,\perp}}\|_2 \ge \sigma/k. \tag{15}$$

Furthermore, we have

$$\begin{aligned}
&\|\|w_{i-1,B}\|_2 x_{i,t+1} + v_{i,2,t+1}\|_2^2 \\
=\ & \|\|w_{i-1,B}\|_2(x_{i,t} - \eta v_{i,2,t}(x_{i,t}^\top v_{i,2,t} - 1)) + (v_{i,2,t} - \eta x_{i,t}\|w_{i,B}\|_2^2(x_{i,t}^\top v_{i,2,t} - 1) - \eta U_{i,2,t}^\top U_{i,2,t} v_{i,2,t})\|_2^2 \\
=\ & \|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2 + 2\eta\|w_{i,B}\|_2(1 - x_{i,t}^\top v_{i,2,t})\|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2 \\
& + \eta\langle \|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}, U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\rangle \pm O(\eta^2 D^4) \\
\geq\ & (1 + \frac{1}{20}\eta\|w_{i,B}\|_2)\|w_{i,B} x_{i,t} + v_{i,2,t}\|_2^2, \qquad (16)
\end{aligned}$$

where the first step holds due to the gradient update formula (see Lemma 4.5), the second step holds due to Eq. (12). The last step holds since

$$\begin{aligned}
\|U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\|_2 \leq \widetilde{O}(d^2\sigma^2 D) \ll \frac{\sigma}{40kD} &\leq \frac{1}{40}\|w_{i,B}\|_2 \cdot \|\|w_{i,B}\|_2 x_{i,0} + v_{i,2,0}\|_2 \\
&\leq \frac{1}{40}\|w_{i,B}\|_2 \cdot \|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2
\end{aligned}$$

and

$$O(\eta D^4) \ll \frac{\sigma^2}{40k^2 D} \leq \frac{1}{40}\|w_{i,B}\|_2 \cdot \|\|w_{i,B}\|_2 x_{i,0} + v_{i,2,0}\|_2^2 \leq \frac{1}{40}\|w_{i,B}\|_2 \cdot \|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2.$$

Hence, we conclude that $\|\|w_{i,B}\|_2 x_t + v_{2,t}\|_2$ is monotonically increasing, and in particular,

$$\begin{aligned}
&\|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2 \geq \|\|w_{i,B}\|_2 x_{i,0} + v_{i,2,0}\|_2^2 = \Omega(\sigma^2/k^2) && \forall t \in [T_1] \\
&\|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2 \geq \Omega(1) && t \geq O(\frac{D}{\eta}\log\frac{k}{\sigma})
\end{aligned}$$

The second inequality follows from Eq. (16). Plugging into Eq. (14), one has

$$\begin{aligned}
x_{i,t+1}^\top v_{i,2,t+1} - x_{i,t}^\top v_{i,2,t} &\geq \frac{1}{20}\eta(\|w_{i,B}\|_2^2\|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2) - O(\eta^2 D^3) \\
&\geq \frac{1}{40}\eta(\|\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t}\|_2^2) - O(\eta^2 D^3) \\
&\geq \begin{cases} 0 & t \in [T] \\ \Omega(\eta) & t \geq O(\frac{D}{\eta}\log\frac{k}{\sigma}) \end{cases}
\end{aligned}$$

Hence, after at most $T_1 \leq O(\frac{D}{\eta}\log\frac{k}{\sigma})$ iterations, we have $0.9 \leq x_{i,T_1}^\top v_{i,2,T_1} < 1$. It would not exceed 0.9 too much since by Eq. (13), the change per iteration is at most

$$\begin{aligned}
|x_{i,t+1}^\top v_{i,2,t+1} - x_{i,t}^\top v_{i,2,t}| &\lesssim \eta(\|w_{i,B}\|_2^2\|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2) \\
&\leq \eta(\|\|w_{i,B}\|_2 x_t - v_{i,2,t}\|_2^2 + 2\|w_{i,B}\|_2 x_{i,t}^\top v_{i,2,t}) \leq 4\eta D \ll 1 \qquad (17)
\end{aligned}$$

For the third claim, we have

$$U_{i,2,t+1}^\top U_{i,2,t+1} = (U_{i,2,t} - \eta U_{i,2,t} v_{i,2,t} v_{i,2,t}^\top)^\top (U_{i,2,t} - \eta U_{i,2,t} v_{i,2,t} v_{i,2,t}^\top) \preceq U_{i,2,t}^\top U_{i,2,t}.$$

The last step holds since $(I - v_{i,2,t}v_{i,2,t}^\top)$ is a PSD matrix and $(I - v_{i,2,t}v_{i,2,t}^\top) \preceq I$. We have proved all three claims. □

A linear convergence of the second and the last loss terms can be shown, after the first T_1 iterations.

Lemma B.3. *Let $T_2 = O(\frac{D}{\eta}\log(\frac{kdD}{\epsilon\nu}))$. After $T = T_1 + T_2$ iterations, we have (1) $|x_{i,T}^\top v_{i,2,T} - 1| \leq \epsilon\nu/kdD$; (2) $\|U_{i,2,T}^\top U_{i,2,T}^\top v_{i,2,T}\|_2 \leq \epsilon\nu$.*

Proof of Lemma B.3. For the t-th iteration ($t \in [T_1 : T_2]$), we prove the following claims inductively.

1. $|x_{i,t}^\top v_{i,2,t} - 1| \leq \frac{1}{2}(1 - \frac{\eta}{4D})^{t-T_1}$,

2. $\|U_{i,t}v_{i,t}\|_2 \le (1-\frac{\eta}{4D})^{t-T_1}$,

3. $\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2 \le \widetilde{O}(r\sigma)$.

The inductive base ($t = T_1$) holds trivially. Assuming the hypothesis holds up to time t, we start from the first claim. We have that

$$
\begin{aligned}
&(1 - x_{i,t+1}^\top v_{i,2,t+1}) - (1 - x_{i,t}^\top v_{i,2,t})\\
=&\ -(x_{i,t} - \eta v_{i,2,t}(x_t^\top v_{i,2,t} - 1))^\top (v_{i,t} - \eta x_{i,t}\|w_{i,B}\|_2^2(x_{i,t}^\top v_{i,2,t} - 1) - \eta U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}) + x_{i,t}^\top v_{i,2,t}\\
=&\ \eta(\|w_{i,B}\|_2^2\|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2)(x_{i,t}^\top v_{i,2,t} - 1) + \eta x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_{i,2} \pm O(\eta^2 D^3|x_{i,t}^\top v_{i,2,t} - 1|).
\end{aligned}
$$

The first step follows from the gradient update formula (see Lemma 4.5), the second step follows from

$$
\|w_{i,B}\|_2^2\|x_{i,t}\| \le O(D^2), \quad \|v_{i,2,t}\|_2 \le D \quad \text{and} \quad \|U_{i,2,t}v_{i,2,t}\|_2 \ll 1.
$$

Since

$$
\|w_{i,B}\|_2^2\|x_{i,t}\|_2^2 + \|v_{i,2,t}\|_2^2 = \|\|w_{i,B}\|x_{i,t} - v_{i,2,t}\|_2^2 + 2\langle \|w_{i,B}\|_2 x_{i,t}, v_{i,2,t}\rangle \ge \frac{1}{D}
$$

holds due to our inductive hypothesis, we further have that

$$
|1 - x_{i,t+1}^\top v_{i,2,t+1}| \le (1 - \frac{\eta}{D})|1 - x_{i,t}^\top v_{i,2,t}| + \eta|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}| \pm O(\eta^2 D^3|x_{i,t}^\top v_{i,2,t} - 1|). \tag{18}
$$

Case 1. Suppose $\frac{1}{2}(1-\frac{\eta}{4D})^{t+2-T_1} \le |x_{i,t}^\top v_{i,2,t} - 1| \le \frac{1}{2}(1-\frac{\eta}{4D})^{t-T_1}$, then we have

$$
|1 - x_{i,t+1}^\top v_{i,2,t+1}| \le (1 - \frac{\eta}{4D})|1 - x_{i,t}^\top v_{i,2,t}| \le \frac{1}{2}(1-\frac{\eta}{4D})^{t+1-T_1}.
$$

This holds due to Eq. (18), $\eta D^3 \ll \frac{1}{4D}$ and

$$
|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}| \le \|x_{i,t}^\top U_{i,2,t}^\top\|_2\|U_{i,2,t}v_{i,2,t}\|_2 \le \widetilde{O}(Dd\sigma)\cdot 2|x_{i,t}^\top v_{i,2,t} - 1| \le \frac{1}{4D}|x_{i,t}^\top v_{i,2,t} - 1|,
$$

where the second step holds due to the induction hypothesis.

Case 2. Suppose $|x_{i,t}^\top v_{i,2,t} - 1| \le \frac{1}{2}(1-\frac{\eta}{4D})^{t+2-T_1}$, then we have

$$
\begin{aligned}
\eta|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_2| \pm O(\eta^2|x_{i,t}^\top v_{i,2,t} - 1|D^3) &\le \eta\cdot\widetilde{O}(Dd\sigma)\cdot(1-\frac{\eta}{4D})^{t-T_1} + O(\eta^2D^3)\cdot(1-\frac{\eta}{4D})^{t-T_1}\\
&\le \frac{1}{2}(1-\frac{\eta}{4D})^{t+1-T_1}\cdot\frac{\eta}{4D},
\end{aligned}
$$

where the first step holds due to induction hypothesis and

$$
|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_{i,2}| \le \|x_{i,t}^\top U_{i,2,t}^\top\|_2\|U_{i,2,t}v_{i,2,t}\|_2 \le \widetilde{O}(Dd\sigma)\cdot(1-\frac{\eta}{4D})^{t-T_1}.
$$

Therefore

$$
|1 - x_{i,t+1}^\top v_{i,2,t+1}| \le \frac{1}{2}(1-\frac{\eta}{4D})^{t+2-T_1} + \frac{1}{2}(1-\frac{\eta}{4D})^{t+1-T_1}\cdot\frac{\eta}{4D} = \frac{1}{2}(1-\frac{\eta}{4D})^{t+1-T_1}.
$$

Next, we prove the second claim. We have

$$
\begin{aligned}
&\|U_{i,2,t+1}v_{i,2,t+1}\|_2\\
=&\ \|(U_{i,2,t} - \eta U_{i,2,t}v_{i,2,t}v_{i,2,t}^\top)(v_{i,2,t} - \eta x_{i,t}\|w_{i,B}\|_2^2(x_{i,t}^\top v_{i,2,t} - 1) - \eta U_{i,2,t}^\top U_{i,2,t}v_{i,2,t})\|_2\\
\le&\ \|U_{i,2,t}v_{i,2,t}(1 - \eta v_{i,2,t}^\top v_{i,2,t}) - \eta U_{i,2,t}U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}\|_2 + \eta\|w_{i,B}\|_2^2|x_{i,t}^\top v_{i,2,t} - 1|\|U_{i,2,t}x_{i,t}\|_2\\
&\quad \pm O(\eta^2D^3)\cdot\|U_{i,2,t}v_{i,2,t}\|_2\\
\le&\ (1 - 5\eta\|v_{i,2,t}\|_2^2/6)\|U_{i,2,t}v_{i,2,t}\|_2 + \eta\|w_{i,B}\|_2^2|x_{i,t}^\top v_{i,2,t} - 1|\|U_{i,2,t}x_{i,t}\|_2\\
\le&\ (1 - \frac{\eta}{3D})\|U_{i,2,t}v_{i,2,t}\|_2 + \eta\|w_{i,B}\|_2^2|x_{i,t}^\top v_{i,2,t} - 1|\|U_{i,2,t}x_{i,t}\|_2,
\end{aligned} \tag{19}
$$

where the first step follows from the gradient update rule (Lemma 4.5), the second step holds due to triangle inequality and

$$\|v_{i,2,t}\|_2 \le O(D), \quad |x_{i,t}^\top v_{i,2,t} - 1|\|w_{i,B}\|_2^2\|x_{i,t}\|_2 \le O(D^2) \quad \text{and} \quad \|U_{i,2,t}^\top U_{i,2,t} v_{i,2,t}\| \ll 1,$$

the third step holds due to $\eta D^3 \le \|v_{i,2,t}\|_2^2/6$ and the last step holds since

$$\begin{aligned}\|v_{i,2,t}\|_2^2 &= v_{i,2,t}^\top(\|w_{i,B}\|_2 x_{i,t} + v_{i,2,t} - \|w_{i,B}\|_2 x_{i,t})\\ &\ge \|w_{i,B}\|_2 x_{i,2,t}^\top v_{i,2,t} - \|v_{i,2,t}\|_2 \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2\\ &\ge \frac{1}{2D} - \widetilde{O}(Dr\sigma) \ge \frac{2}{5D}.\end{aligned}$$

Case 1. Suppose $(1-\frac{\eta}{4D})^{t+2-T_1} \le \|U_{i,2,t}v_{2,t}\|_2 \le (1-\frac{\eta}{4D})^{t-T_1}$, then

$$\begin{aligned}\|U_{i,2,t+1}v_{i,2,t+1}\|_2 &\le (1-\frac{\eta}{3D})\|U_{i,2,t}v_{i,2,t}\|_2 + \eta\|w_{i,B}\|_2^2|x_{i,t}^\top v_{i,2,t} - 1|\|U_{i,2,t}x_{i,t}\|_2\\ &\le (1-\frac{\eta}{4D})\|U_{i,2,t}v_{i,2,t}\|_2 \le (1-\frac{\eta}{4D})^{t+1-T_1},\end{aligned}$$

where the first step comes from Eq. (19), the second step comes from

$$\begin{aligned}\eta\|w_{i,B}\|_2^2|x_{i,t}^\top v_{i,2,t} - 1|\|U_{i,2,t}x_{i,t}\|_2 &\le \eta D^2 \cdot \frac{1}{2}(1-\frac{\eta}{4D})^{t-T_1} \cdot \widetilde{O}(Dd\sigma)\\ &\le \frac{\eta}{12D}(1-\frac{\eta}{4D})^{t+2-T_1} \le \frac{\eta}{12D}\|U_{i,2,t}v_{i,2,t}\|_2.\end{aligned}$$

Case 2. Suppose $\|U_t v_{2,t}\|_2 \le (1-\frac{\eta}{4D})^{t+2-T_1}$, then

$$\begin{aligned}\|U_{i,2,t+1}v_{i,2,t+1}\|_2 &\le \|U_{i,2,t}v_{i,2,t}\|_2 + \eta\|w_{i,B}\|_2^2 \cdot |x_{i,t}^\top v_{i,2,t} - 1| \cdot \|U_{i,2,t}x_{i,t}\|_2\\ &\le (1-\frac{\eta}{4D})^{t+2-T_1} + \frac{1}{2}\eta D^2(1-\frac{\eta}{4D})^{t-T_1} \cdot \widetilde{O}(Dd\sigma)\\ &\le (1-\frac{\eta}{4D})^{t+2-T_1} + (1-\frac{\eta}{4D})^{t+1-T_1} \cdot \frac{\eta}{4D}\\ &= (1-\frac{\eta}{4D})^{t+1-T_1},\end{aligned}$$

where the first step comes from Eq. (19), the second step follows from the induction hypothesis and $\|U_{i,2,t}x_{i,t}\|_2^2 \le \widetilde{O}(Dd\sigma)$. We have proved the second claim.

Now we move to the third claim. One has

$$\begin{aligned}&\|\|w_{i,B}\|_2 x_{i,t+1} - v_{i,2,t+1}\|_2^2 - \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2\\ &= 2\eta(x_{i,t}^\top v_{i,2,t} - 1)\|w_{i,B}\|_2\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 + \eta\langle\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}, U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}\rangle \pm O(\eta^2 D^4)\\ &\lesssim 2\eta \cdot d^2D^3\sigma^2 \cdot D \cdot (r\sigma)^2 + \eta \cdot r\sigma \cdot d^2\sigma^2 D + \eta^2 D^4\\ &\lesssim \eta D d^2 r\sigma^3.\end{aligned}$$

The first step comes from Eq. (10), the second step follows from

$$\|w_{i,B}\|_2 \le D, \quad \|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2 \le \widetilde{O}(r\sigma), \quad \|U_{i,2,t}^\top U_{i,2,t}v_{i,2,t}\|_2 \le \widetilde{O}(d^2\sigma^2 D)$$

and

$$x_{i,t}^\top v_{i,2,t} - 1 \le \widetilde{O}(d^2D^3\sigma^2).$$

Here the last term holds since (i) $|x_{i,\tau+1}^\top v_{i,2,\tau+1} - x_{i,\tau}^\top v_{i,2,\tau}| \le O(\eta D)$, i.e., the step size is at most ηD (see Eq. (13) (17)); (ii) $x_{i,T_1}^\top v_{i,2,T_1} < 1$ and (iii) $|x_{i,\tau+1}^\top v_{i,2,\tau+1} - 1| < |x_{i,\tau}^\top v_{i,2,\tau} - 1|$ whenever

$$\eta\|x_{i,t}^\top U_{i,2,t}^\top U_{i,2,t}v_2\|_2 \le \eta \cdot \widetilde{O}(d^2D^2\sigma^2) \lesssim \frac{\eta}{2D}|x_{i,\tau}^\top v_{i,2,\tau} - 1| \quad \Rightarrow \quad |x_{i,\tau}^\top v_{i,2,\tau} - 1| \gtrsim d^2D^3\sigma^2.$$

That is, combining (i) (ii), we know that the first time $x_{i,\tau}^\top v_{i,\tau}$ being greater 1 must obey $x_{i,\tau}^\top v_{i,\tau} < 1 + O(\eta D)$, (iii) implies that whenever $x_{i,\tau+1}^\top v_{i,2,\tau+1} - 1 \gtrsim d^2D^3\sigma^2$, it value should decrease, hence we conclude

$$x_{i,T_1}^\top v_{i,2,T_1} - 1 \lesssim \eta D + d^2D^3\sigma^2 \lesssim d^2D^3\sigma^2.$$

Taking a telescopic summation, one has

$$\begin{aligned}\|\|w_{i,B}\|_2 x_{i,t} - v_{i,2,t}\|_2^2 - \|\|w_{i,B}\|_2 x_{i,T_1} - v_{i,2,T_1}\|_2^2 &\le (t - T_1) \cdot O(\eta D d^2 r^2 \sigma^3) \\ &\le \widetilde{O}(D^2 d^2 r \sigma^3) \le r^2 \sigma^2.\end{aligned}$$

This concludes the third claim. We conclude the proof here. □

Combining Lemma 4.6, Lemma B.1 – B.3, one can conclude the proof of Lemma 4.7.

B.4 Missing proof from Section 4.2.4

Proof of Theorem 2.11. Due to the reduction established in Section 4.2.1, it suffices to prove Eq. (3) and Eq. (4). For each environment i ($i \in [k]$), we inductively prove

1. DPGrad achieves good accuracy on the current environment, i.e., $\|U_{i,T}v_i - w_i\|_2 \le \epsilon\nu$;
2. The feature matrix U_i remains well conditioned, i.e. $\frac{1}{2\sqrt{D}} \le \sigma_{\min}(U_{i,\mathsf{end}}) \le \sigma_{\max}(U_{i,\mathsf{end}}) \le 2\sqrt{D}$.
3. The algorithm does not suffer from catastrophic forgetting, i.e., $\|U_{i,t}v_j - w_i\|_2 \le \epsilon$ for any $j < i$ and $t \in [T]$;

The base case ($i = 0$) holds trivially as at the beginning of CL, we have $\mathsf{W}, \mathsf{V} = \emptyset$ and $U = 0$. Suppose the induction holds up to the $(i-1)$-th environment, we focus on the second and last claim, as the first claim holds directly due to Lemma 4.7.

For the second claim, we have already proved $\|U_{i,T}v_i - w_i\|_2 \le \epsilon\nu$, this indicates that each coordinate of $U_{i,T}v_i - w_i$ is less than $\nu/2$. Since we assume each coordinate of w_i is a multiple of ν, therefore, we have $\widehat{w}_i = \mathsf{Round}_\nu(U_{i,T}v_i) = w_i$. That is, we exact recover w_i. We divide into two cases.

Case 1. If $\|w_{i,B}\|_2 = 0$, i.e., $w_i \in \mathsf{W}$, then $\|P_{\mathsf{W}_\perp}\widehat{w}_i\|_2 = \|P_{\mathsf{W}_\perp}w_i\|_2 = 0$, Therefore, we do not update W and V, and

$$U_{i,\mathsf{end}} = P_{\mathsf{W}}U_{i,T}P_{\mathsf{V}} = P_{\mathsf{W}}(U_{i,A,0} + U_{i,B,T})P_{\mathsf{V}} = P_{\mathsf{W}}U_{i,A,0}P_{\mathsf{V}} = U_{i-1,\mathsf{end}},$$

where the second and the third step holds to Lemma 4.4 and the last step just holds due to definition. Hence U_i continues to be well-conditioned (since it does not change).

Case 2. If $\|w_{i,B}\|_2 \in [1/D, D]$, then $\|P_{\mathsf{W}_\perp}\widehat{w}_i\|_2 = \|P_{\mathsf{W}_\perp}w_i\|_2 = \|w_{i,B}\| \ge 1/D$. Hence, we augment $\mathsf{W}_i = \mathsf{W}_{i-1} \cup \{w_i\}$ and $\mathsf{V}_i = \mathsf{V}_{i-1} \cup \{v_i\}$ and have

$$\begin{aligned}U_{i,\mathsf{end}} &= P_{\mathsf{W}}U_{i,T}P_{\mathsf{V}} = P_{\mathsf{W}}(U_{i,A,0} + U_{i,B,T})P_{\mathsf{V}} \\ &= U_{i,A,0} + (\frac{1}{\|w_{i,B}\|_2^2}w_{i,B}w_{i,B}^\top)U_{i,B,T}(\frac{1}{\|v_{i,2,T}\|_2^2}v_{i,2,T}v_{i,2,T}^\top) \\ &= U_{i,A,0} + (\frac{1}{\|w_B\|_2^2}w_B w_B^\top)(w_B x_{i,T}^\top + U_{i,2,T})(\frac{1}{\|v_{i,2,T}\|_2^2}v_{i,2,T}v_{i,2,T}^\top) \\ &= U_{i,A,0} + w_B v_{i,2,T}^\top \frac{x_{i,T}^\top v_{i,2,T}}{\|v_{i,2,T}\|_2^2} \\ &= U_{i,A,0} + (1 \pm o(\epsilon/D))\frac{1}{\|v_{i,2,T}\|_2^2}w_B v_{i,2,T}^\top. \end{aligned} \tag{20}$$

The third step holds since $\mathsf{row}(U_{i,A,0}) \in \mathsf{V}$, $\mathsf{column}(U_{i,A,0}) \in \mathsf{W}$, $\mathsf{column}(U_{i,B,T}) \cap \mathsf{W} = w_{i,B}$, $\mathsf{row}(U_{i,B,T}) \cap \mathsf{V} = v_{i,2,T}$ (see Lemma 4.4), the later two imply the projection operation essentially boils to projection on $w_{i,B}$ and $v_{i,2,T}$. The fifth step follows from $\mathsf{column}(U_{i,2,T}) \perp w_B$ (see Lemma 4.4), the sixth step follows from $x_{i,T}^\top v_{i,T} = 1 \pm o(\epsilon/D)$ (see Lemma B.3). To bound the condition number, it suffices to note that $w_{i,B} \perp \mathsf{W}_{i-1}$, $v_{i,2,T} \perp \mathsf{V}_{i-1}$ (see Lemma 4.4), and therefore, $w_B \perp \mathsf{column}(U_{i,A,0})$, $v_{i,2,T} \perp \mathsf{row}(U_{i,A,0})$ (i.e., we add an orthogonal basis) and

$$(1 \pm o(\epsilon/D))\frac{1}{\|v_{i,2,T}\|_2^2}\|w_B\|_2\|v_{i,2,T}^\top\|_2 = (1 \pm o(\epsilon/D))\frac{\|w_B\|_2}{\|v_{i,2,T}\|_2} = (1 + o(1))\sqrt{\|w_B\|_2} \in \left[\frac{1}{2\sqrt{D}}, \frac{\sqrt{D}}{2}\right]$$

where the last step is derived from $x_{i,t}^\top v_{i,2,T} \approx 1 + o(1)$ and $\|\|w_B\|_2 x_{i,t} - v_{i,2,t}\|_2 \approx 1 \pm o(1/D^3)$. We have proved the second claim.

For the last claim, fix an index $j < i$, we prove the accuracy of j-th environment would not drop significantly and remain good. Note by inductive hypothesis, we already have $\|U_{j,T} v_j - w_j\|_2 \leq \epsilon\nu/kd$ before the final projection step of j-th environment. After the projection step, one has

$$\|U_{j,\mathsf{end}} v_j - w_j\|_2 = \|P_{\mathsf{W}} U_{j,T} P_{\mathsf{V}} v_j - w_j\|_2 = \|P_{\mathsf{W}}(U_{j,A,T} + w_{j,B} x_{j,T}^\top + U_{j,2,T}) P_{\mathsf{V}} v_j - w_j\|_2$$

We divide into two cases.

Case 1. Suppose $\|w_{j,B}\|_2 = 0$. We have $\mathsf{W}_j = \mathsf{W}_{j-1}, \mathsf{V}_j = \mathsf{V}_{j-1}$ and

$$\begin{aligned}\|U_{j,\mathsf{end}} v_j - w_j\|_2 &= \|P_{\mathsf{W}_j}(U_{j,A,T} + U_{j,2,T}) P_{\mathsf{V}_j} v_j - w_j\|_2 = \|U_{j,A,T} v_j - w_j\|_2 \\ &\leq \|U_{j,A,T} v_{j,1,T} - w_j\|_2 \leq \epsilon\nu.\end{aligned}$$

The second step follows from $\mathsf{column}(U_{j,A,T}) \in \mathsf{W}_j$, $\mathsf{row}(U_{j,A,T}) \in \mathsf{V}_k$ and $\mathsf{row}(U_{j,2,T}) \in \mathsf{V}_{j,\perp}$ (see Lemma 4.4), the third step follows from $\mathsf{row}(U_{j,A,T}) \in \mathsf{V}_{i-1}$. Hence, we have that the error remains small after the projection.

During the i-th environment, for any $t \in [T]$, we decompose $U_{i,t} = U_{j,\mathsf{end}} + \widehat{U}_{i,t}$. We have

$$\begin{aligned}\|U_{i,t} v_j - w_j\|_2 &= \|(U_{j,\mathsf{end}} + \widehat{U}_{i,t}) v_j - w_j\|_2 \\ &\leq \|U_{j,\mathsf{end}} v_j - w_j\|_2 + \|\widehat{U}_{i,t} v_j\|_2 \\ &= \|U_{j,\mathsf{end}} v_j - w_j\|_2 + \|\widehat{U}_{i,t} v_{j,2,T}\|_2 \\ &\leq \epsilon\nu + \widetilde{O}(\sqrt{D} \cdot r\sigma) \\ &\leq \epsilon.\end{aligned}$$

The third step holds due to the fact that $\mathsf{row}(\widehat{U}_{i,t}) \in V_{j,\perp}$, the fourth step holds due to (1) $\|v_{j,2,t}\|_2$ is non-decreasing during the j-th environment (see the gradient update formula in Lemma 4.5) and therefore $\|v_{j,2,T}\|_2 \leq \|v_{j,2,0}\|_2 \leq \widetilde{O}(r\sigma)$ w.h.p.; (2) the spectral norm $\|\widehat{U}_{i,t}\| \leq O(\sqrt{D})$, since

$$\begin{aligned}\|\widehat{U}_{i,t}\| &\leq \|U_{i,t}\| + \|U_{j,\mathsf{end}}\|_2 \leq \|U_{i,A}\| + \|w_{i,B} x_{i,t}^\top + U_{i,2,T}\| + \|U_{j,\mathsf{end}}\|_2 \\ &\leq 2\sqrt{D} + 2\sqrt{D} + 2\sqrt{D} = O(\sqrt{D}).\end{aligned}$$

Here the first step and the second step hold due to triangle inequality, the second step holds due to the inductive hypothesis and $\|w_{i,B} x_{i,t}^\top + U_{i,2,T}\| \leq 2\sqrt{D}$. We finished the proof of the first case.

Case 2. Suppose $\|w_{j,B}\|_2 \in [1/D, D]$. Then we augment $\mathsf{W}_j = \mathsf{W}_{j-1} \cup \{w_j\}$ and $\mathsf{V}_j = \mathsf{V}_{j-1} \cup \{v_j\}$. We first prove the loss remains small after the final projection step of j-th environment. In particular, we have

$$\begin{aligned}\|U_{j,\mathsf{end}} v_j - w_j\|_2 &= \|\big(U_{j,A,0} + (1 \pm o(\epsilon/D)) \frac{1}{\|v_{j,2,T}\|_2^2} w_B v_{j,2,T}^\top\big)(v_{j,1,T} + v_{j,2,T}) - w_{j,A} - w_{j,B}\|_2 \\ &= \|(U_{j,A,0} v_{j,1,T} - w_{j,A}) + (1 \pm o(\epsilon/D)) \frac{1}{\|v_{j,2,T}\|_2^2} w_{j,B} v_{j,2,T}^\top v_{j,2,T} - w_{j,B}\|_2 \\ &\leq \|(U_{j,A,0} v_{j,1,T} - w_{j,A})\|_2 + o(\epsilon/D)\|w_{j,B}\|_2 \\ &\leq \epsilon\nu + o(\epsilon) \leq \epsilon.\end{aligned}$$

The first step holds due to Eq. (20), the third step holds due to triangle inequality, the fourth step holds due to the inductive hypothesis and $\|w_{j,B}\|_2 \leq D$.

During the i-th environment, since the update is performed in the orthogonal space, we expect $U v_j$ does not change. Formally, let $U_{i,t} = U_{j,\mathsf{end}} + \widehat{U}_{i,t}$, where $\mathsf{column}(\widehat{U}_{i,t}) \perp \mathsf{W}_j$ and $\mathsf{row}(\widehat{U}_{i,t}) \perp \mathsf{V}_j$, then

$$U_{i,t} v_j = (U_{j,\mathsf{end}} + \widehat{U}_{i,t}) v_j = U_{j,\mathsf{end}} v_j,$$

Hence $\|U_{i,t} v_j - w_j\| \leq \epsilon$ continues to hold. We conclude the proof here.

□

C Missing proof from Section 5

Proof of Theorem 2.12. We take $k = 2, n = 3, d = 2$. For both environments, we assume the input data are drawn uniformly at random from $\mathcal{B}_3(0,1)$, where $\mathcal{B}_3(0,1)$ denotes the unit ball in $\mathbb{R}^3$ centered at origin. The hypothesis class $\mathcal{H}$ consists of all two-layer convolutional neural network with a single kernel of size 2 and the quadratic activation function. That is, the representation function is parameterized by $w \in \mathbb{R}^2$ and takes the form of $R_w(x) = (\langle w, x_{1:2}\rangle^2, \langle w, x_{2:3}\rangle^2) \in \mathbb{R}^2$, where $x \in \mathbb{R}^3$, $x_{i:j} \in \mathbb{R}^{j-i+1}$ is a vector consists of the i-th entry to the j-th entry of x.

The hard sequence of environments are drawn from the following distribution.

- The objective function f_1 of the first environment is $f_1(x) = x_2^2$
- The objective function f_2 of the second environment equals $f_2(x) = x_3^2$ with probability $1/2$, and equals $f_2(x) = x_1^2$ with probability $1/2$.

First, the continual learning task is realizable: (1) if $f_2(x) = x_3^2$, then one can take $w = (0,1)$ and $v_1 = (1,0), v_2 = (0,1)$; (2) if $f_2(x) = x_1^2$, then one can take $w = (1,0)$, $v_1 = (0,1)$, $v_2 = (1,0)$.

We then prove no (proper) continual learning algorithm can guarantee to achieve less than $1/1000$-error on both environments with probability at least $1/2$. Suppose the algorithm takes $v_1 = (v_{1,1}, v_{1,2})$ for the first environment. Due to symmetry, one can assume $|v_{1,1}| \geq |v_{1,2}|$. With probability $1/2$, the objective function of the second environment is $f_2(x) = x_1^2$. Let $v_2 = (v_{2,1}, v_{2,2})$ be the linear prompt and $w = (w_1, w_2)$ be the parameter of neural network. We prove by contradiction and assume

$$\mathop{\mathbb{E}}_{x\sim\mathcal{B}_3(0,1)}[|\langle v_1, R_w(x)\rangle - x_2^2|^2] \leq 1/1000 \text{ and } \mathop{\mathbb{E}}_{x\sim\mathcal{B}_3(0,1)}[|\langle v_2, R_w(x)\rangle - x_1^2|^2] \leq 1/1000.$$

Let Π_n^d be the space of all polynomial of degree at most d in n variables. By Lemma C.1, notice that $\langle v_1, R_w(x)\rangle, \langle v_2, R_w(x)\rangle \in \Pi_3^2$, we must have that their coefficients match well with x_2^2 and x_1^2 respectively (in the sense that the absolute deviation is no larger than $1/4$).

First, compare the polynomials of $\langle v_2, R_w(x)\rangle$ and x_1^2, we must have (1) $v_{2,1}w_1^2 \geq 3/4$ due to the x_1^2 term, and due to the $x_1x_2^2$ term, one has (2) $|v_{2,1}w_1w_2| \leq 1/4$. These two indicate (3) $|w_1| \geq 3|w_2|$. Then compare the polynomials of $\langle v_1, R_w(x)\rangle$ and x_2^2, we have (4) $|v_{1,1}w_1^2| \leq 1/4$ due to the x_1^2 term. Combining (3) and (4), one has (5) $|v_{1,1}w_2^2| \leq \frac{1}{9}|v_{1,1}w_1^2| \leq \frac{1}{36}$. Since the x_2^2 term is roughly matched, one must have (6) $|v_{1,2}w_1^2| \geq 1 - \frac{1}{4} - \frac{1}{36} = \frac{13}{18}$. However, note that (4) and (6) contradicts with the assumption that $|v_{1,1}| \geq |v_{1,2}|$. We conclude the proof. □

We provide the proof of a technical Lemma used in proving Theorem 2.12

Lemma C.1 (Technical tool)**.** *Let Π_n^d be the space of all polynomial of degree at most d in n variables. For any two polynomials $p_1(x), p_2(x) \in \Pi_3^2$, if*

$$\mathop{\mathbb{E}}_{x\sim\mathcal{B}_3(0,1)}[(p_1(x) - p_2(x))^2] \leq \frac{1}{1000},$$

then the absolute deviation of each coefficient is at most $1/4$.

Proof. Let $p(x) = (p_1(x) - p_2(x))^2$, taking an integral over $B_3(0,1)$, we can only need to consider all quadratic terms, since all odd terms would be canceled due to symmetry. We divide into cases. (1) The coefficient of the constant term is greater than $1/4$, then $p(x) \geq 1/16$. (2) The coefficient of x_1 is greater than $1/4$, then $p(x) \geq \mathbb{E}_{x\sim\mathcal{B}_3(0,1)} \frac{1}{16}x_1^2 = \frac{1}{16}\cdot\frac{1}{5} = \frac{1}{80}$. (3) The coefficient of x_1x_2 is greater than $1/4$, then then $p(x) \geq \mathbb{E}_{x\sim\mathcal{B}_3(0,1)} \frac{1}{16}x_1^2x_2^2 = \frac{1}{16}\cdot\frac{1}{35} = \frac{1}{560}$. (4) The coefficient of x_1^2 is greater than $1/4$, then then $p(x) \geq \mathbb{E}_{x\sim\mathcal{B}_3(0,1)} \frac{1}{16}x_1^4 = \frac{1}{16}\cdot\frac{3}{35} = \frac{3}{560}$. Hence we conclude no coefficient has difference greater than $1/4$. □

Lower bound with ReLU activation There is nothing particularly special about the activation quadratic activation function: here, we provide a similar lower bound for features are represented via one-layer convolutional neural network with ReLU activation.

Theorem C.2. *Let $k, r, d \geq 2$. There exists a class of non-linear feature mappings and a sequence of environments, such that there is no (proper) continual learning algorithm that can guarantee to achieve less than $1/3$-error over all environments with probability at least $1/16$. The lower bound is constructed on a single family of two-layer neural network with ReLU activation.*

Proof. It suffices to take $k = 2, d = 2, r = 2$. The input distribution is uniform over $(1, 1), (-1, -1), (1, -1), (-1, 1)$ for both tasks. The hypothesis class $\mathcal{H}$ contains all two-layer convolutional neural network with a single kernel of size 1 and ReLU activation. That is, the representation function is parameterized by $w \in \mathbb{R}$ and $R_w(x) = (\max\{wx_1, 0\}, \max\{wx_2, 0\}) \in \mathbb{R}^2$ for input $x = (x_1, x_2) \in \mathbb{R}^2$.

The hard sequence of environment are drawn as follow: (1) The objective value in first environment is always $(0, 0, 1, -1)$; (2) The objective value of the second environment equals $(2, 0, 1, 1)$ with probability $1/2$ and $(0, 2, 1, 1)$ with probability $1/2$.

One can easily verify that the continual learning task is realizable: in the first case, one takes $w = 1$, $v_1 = (1, -1), v_2 = (1, 1)$ while in the second case, one takes $w = -1, v_1 = (-1, 1), v_2 = (1, 1)$.

We next prove any proper continual learning algorithm makes error at least $1/10$. We prove by contradiction and assume the continual learning algorithm takes value w for the convolutional layer in the first task. It is easy to verify that $w \neq 0$, otherwise the first environment suffers loss at least $1/2$. When $w > 0$, then representation function equals $(w, w), (0, 0), (w, 0), (0, w)$ and we have $v_1 = (\frac{1}{w} \pm \frac{1}{2w}, -\frac{1}{w} \pm \frac{1}{2w}) \in (\mathbb{R}_+, \mathbb{R}_-)$. For the second environment, suppose the objective value equals $(0, 2, 1, 1)$ (note this happens with probability $1/2$). Then the algorithm must change the parameter to $w' < 0$, otherwise the loss on point $(-1, -1)$ is at least 4. Then for the first task, the loss on point $(1, -1)$ is at least 1. The case of $w < 0$ is similar and we conclude the proof here. □

D Additional details of simulation

We provide further details for our simulations. In our simulations, we set input dimension $d = 100$, the number of features $r = 20$ and the continual learning setup uses $k = 500$ tasks. Each entry of the ground truth $U^\star \in \mathbb{R}^{d \times r}$ (and $V^\star \in \mathbb{R}^{k \times r}$) is drawn from the Gaussian $N(0, 1)$, and $W = U^\star (V^\star)^\top \in \mathbb{R}^{d \times k}$. The input data x of each task is drawn from the multivariate gaussian $N(0, I_d)$, and the label is set to be $y = \langle w_i, x \rangle$. For each task, we drawn $N = 1000$ samples, and perform DPGrad/OGD/SGD for $T = 3000$ iterations, with learning rate $\eta = 0.01/0.0001/0.01$ respectively, and the initialization scale $\sigma = 0.01$. We omit the rounding step of DPGrad and simply takes $\widehat{w}_i = Uv_i$ (Line 12 in Algorithm 1), and the result of simulation empirically verifies that our Bit complexity assumption is indeed for convenience of analysis and one does not need it for practice. The OGD algorithm does not always converge in our simulations and we (1) decrease the learning rate and (2) perform early stopping: the projection is only w.r.t. the first 20 tasks. Our experiments are executed on an Apple M1 CPU.

E Additional Experiments

In addition to our synthetic data experiments, we also perform experiments on two common benchmark datasets: Permuted MNISTs and Rotated MNISTs. We do this both to verify the behavior of our proposed algorithm, as well as compare with two baseline approaches: Vanilla Stochastic Gradient Descent (SGD) and Orthogonal gradient descent (OGD) [11].

Datasets We consider two datasets, Permuted MNIST and Rotated MNIST. In the Permuted MNIST dataset, a task is created by performing a random permutation to the input pixels; in the Rotated MNISTs a task is created by randomly rotating the input image. We generated 10 tasks for both benchmark datasets and the continual learning algorithm is sequentially exposed to these 10 tasks. The permutation/rotation is same within each task but different across tasks. Each task contains 60000 training samples and the test set contains 10000 images.

Our methods Our DPGrad algorithm is tailored to the linear regression setting we consider, so it has to be modified to apply it to multi-class classification problems like Rotated/Permuted MNIST and/or to handle non-linear representations. We consider two natural generalizations of DPGrad.

To adapt it to a multi-class classification problem, we view each task as having 10 linear predictors—one for each class. Recall the key idea of DPGrad is to perform (fine-grained) column/row projection for the gradient of weight matrix and the column/row space is increased by (at most) 1 after each task. In the multi-class case, we force the increase of the row/column space to be (at most) 10 dimensions per task. In the tables/figures below we just call this DPGrad.

To adapt it to non-linear representations, we use a modified approach we call DPGrad+. In the linear setting, the column/row space increases by (at most) one dimension after each task and the newly added column/row is essentially the top eigenvector of the feature matrix U as it is close to a rank-one matrix (see Lemma B.3) after projection. For non-linear feature, there is no reason to hope the weight matrix is rank-one, but instead, we perform singular value decomposition (SVD) to the matrix and take the top-h eigenvectors and then add them to the column/row space. In other words, the only difference between DPGrad+ and DPGrad is that DPGrad+ augment the column/row space by the top-h eigenvector instead of the top-1 eigenvector. We take $h = 15$ in both experiments.

Hyperparameter choices We use a two-layer fully connected neural network, where the hidden layer contains 300 neurons and uses ReLU activation (for DPGrad+; for DPGrad the activation is linear). The parameters of the first layer are shared across tasks, while the weights of the second layer are different across tasks (i.e. the linear predictor). We perform 5 epochs of training for each task, the learning rate is fixed to be 0.1 and the batch size is 100.

Experimental results The experimental results on Permuted MNIST can be found at Figure 3, Figure 5 and Table 4, the results on Rotated MNIST can be found at Figure 2, Figure 4 and Table 3. Figure 2-5 plot the test accuracy on the 10 tasks over time, Table 4 and Table 3 record the test accuracy of each tasks at the end of training (i.e., after the 10-th task). The average accuracy is reported in Table 1. The deviations of the values and confidence intervals are gotten from 5 runs, randomizing over the order of the tasks, as well as the randomness of the algorithm (i.e. seed).

Both DPGrad+ and DPGrad alleviate catastrophic forgetting and perform much better than vanilla SGD. Both outperform OGD, which is a strong baseline approach and outperforms classical approaches like elastic weight consolidation [19]. The performance of DPGrad+ and DPGrad is much more stable than OGD and the accuracy remains at a high level across tasks. By contrast, OGD has large variance across tasks—it obtains high accuracy in recent tasks but much lower accuracy in early tasks (especially in Rotated MNIST).

	Rotated MNIST	Permuted MNIST
DPGrad+	76.6% (±2.1%)	89.5% (±0.2%)
DPGrad	74.3% (±1.5%)	86.3% (±0.1%)
OGD	73.0% (±2.4%)	88.7% (±0.6%)
SGD	66.8% (±2.9%)	81.6% (±1.6%)

Table 1: Average Accuracy

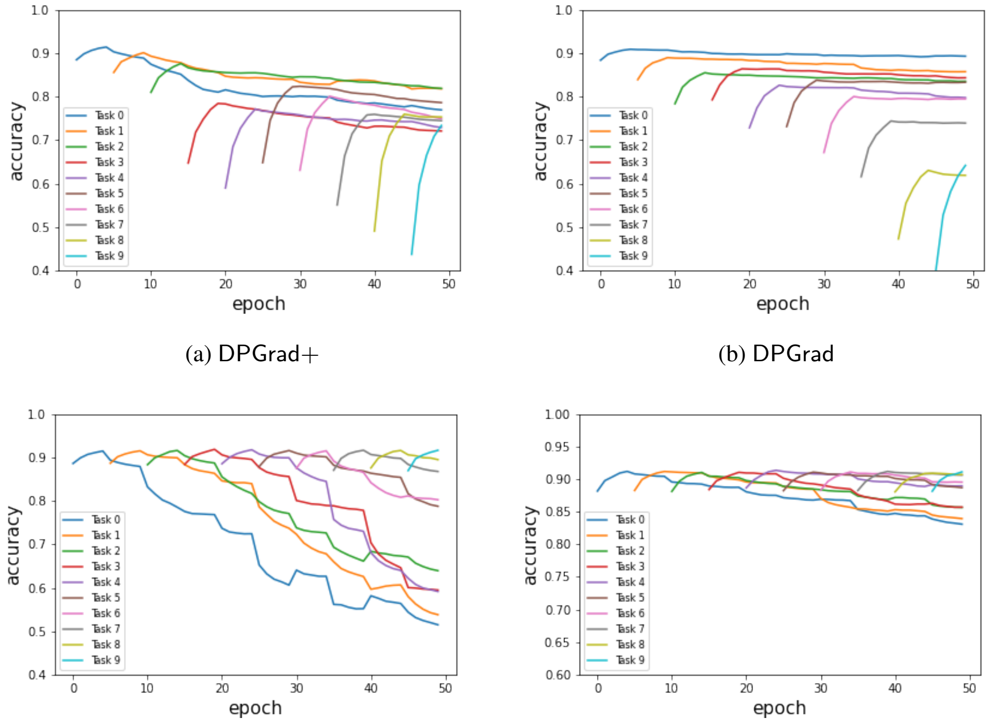
Finally, we provide values for a common metric for quantifying forgetting: the *backward transfer value*, defined as

$$\frac{1}{k-1}\sum_{i=1}^{k-1} \mathsf{ACC}_{k,i} - \mathsf{ACC}_{i,i}$$

where $\mathsf{ACC}_{i,j}$ is the test accuracy of task j, after training with task i. A large negative backward transfer value means the algorithm suffers from catastrophic forgetting, a small or even positive backward transfer value indicates the algorithm avoids catastrophic forgetting. We report the backward transfer value in Table 2. In brief, OGD is more plastic than DPGrad, however at the expense of incurring a larger forgetting ratio (or negative backward transfer).

	Rotated MNIST	Permuted MNIST
DPGrad+	-0.04 (±0.01)	-0.01 (±0.01)
DPGrad	-0.01 (±0.01)	0 (±0.01)
OGD	-0.20 (±0.06)	- 0.03 (±0.01)
SGD	-0.28 (±0.09)	-0.10 (±0.05)

Table 2: Backward Transfer



(a) DPGrad+

(b) DPGrad

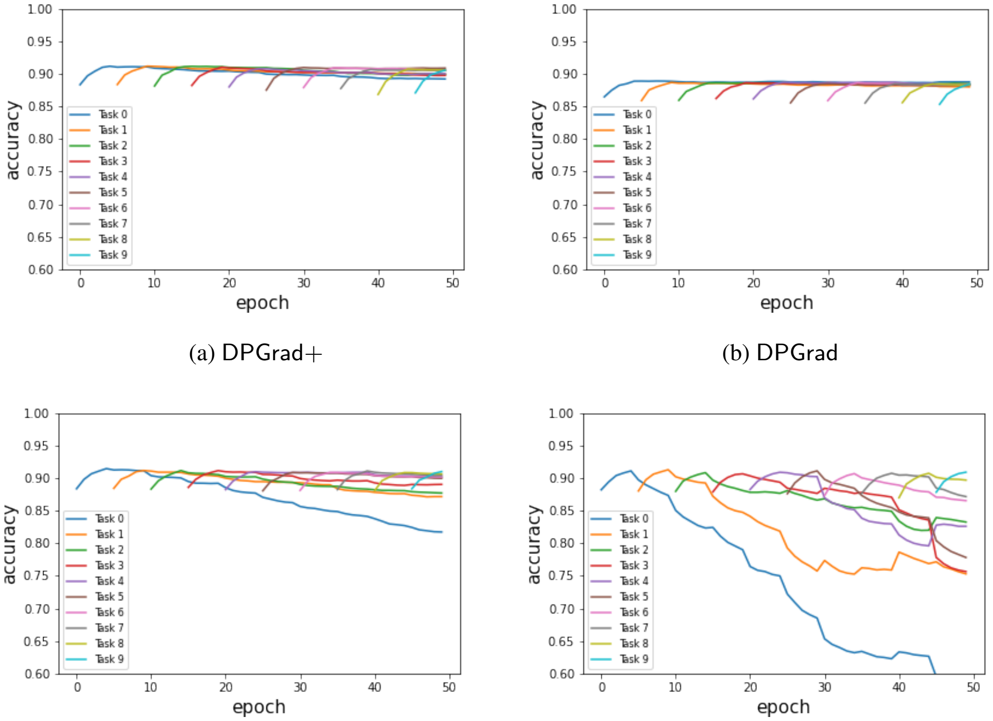
(c) OGD

(d) SGD

Figure 2: Rotated MNIST

	Task 1	Task 2	Task 3	Task 4	Task 5
DPGrad+	80.4% (±4.59%)	83.2% (±1.98%)	82.6% (±2.86%)	81.8% (±1.31%)	78.5% (±2.75%)
DPGrad	82.6% (±1.19%)	79.5% (±1.46%)	77.3% (±6.02%)	76.6% (±2.97%)	76.3% (±2.85%)
OGD	47.0% (±9.31%)	59.9% (±6.56%)	69.1% (±6.25%)	65.1% (±5.62%)	70.2% (±14.4%)
SGD	49.7% (±9.37%)	52.8% (±12.8%)	56.5% (±14.9%)	55.3% (±15.0%)	63.5% (±8.45%)
	Task 6	Task 7	Task 8	Task 9	Task 10
DPGrad+	74.8% (±9.24%)	71.7% (±4.68%)	73.2% (±3.27%)	70.6% (±4.79%)	69.5% (±5.22%)
DPGrad	72.6% (±3.66%)	71.3% (±5.02%)	71.5% (±3.29%)	70.2% (±6.16%)	64.7% (±8.49%)
OGD	77.0% (±4.80%)	83.6% (±4.73%)	78.9% (±5.64%)	87.6% (±2.33%)	91.7% (±0.29%)
SGD	67.6% (±8.54%)	69.1% (±12.1%)	77.3% (±4.31%)	84.3% (±2.69%)	91.5% (±0.12%)

Table 3: Rotated MNIST



(a) DPGrad+

(b) DPGrad

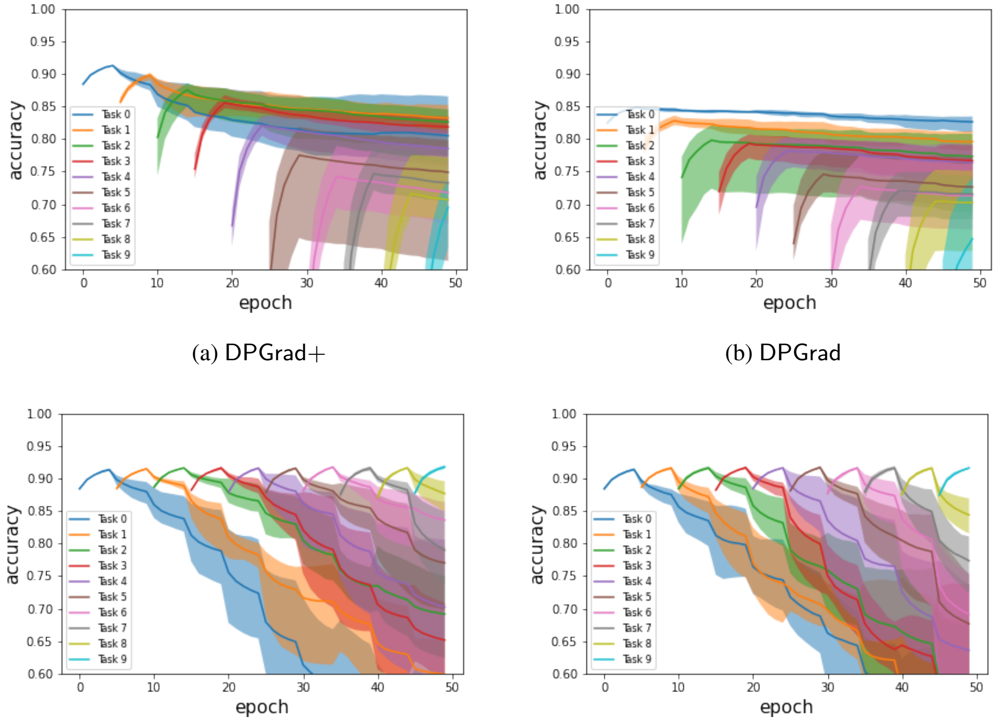
(c) OGD

(d) SGD

Figure 3: Permuted MNIST

	Task 1	Task 2	Task 3	Task 4	Task 5
DPGrad+	87.1% (±2.14%)	88.8% (±1.08%)	87.9% (±1.12%)	89.5% (±0.57%)	89.2% (±1.32%)
DPGrad	86.3% (±0.36%)	86.1% (±0.64%)	86.3% (±0.18%)	86.3% (±0.36%)	86.3% (±0.23%)
OGD	82.9% (±4.37%)	86.3% (±1.17%)	88.2% (±1.05%)	88.5% (±1.38%)	89.2% (±0.59%)
SGD	76.0% (±6.83%)	67.9% (±9.03%)	75.1% (±7.88%)	78.6% (±3.85%)	84.5% (±2.11%)
	Task 6	Task 7	Task 8	Task 9	Task 10
DPGrad+	90.3% (±0.19%)	90.5% (±0.16%)	90.4% (±0.24%)	90.4% (±0.11%)	90.4% (±0.24%)
DPGrad	86.2% (±0.32%)	86.2% (±0.12%)	86.3% (±0.05%)	86.3% (±0.25%)	86.3% (±0.13%)
OGD	90.0% (±0.50%)	90.3% (±0.50%)	90.5% (±0.40%)	90.6% (±0.13%)	91.0% (±0.12%)
SGD	82.9% (±7.05%)	85.6% (±3.20%)	86.6% (±2.33%)	88.6% (±2.87%)	90.7% (±0.24%)

Table 4: Permuted MNIST



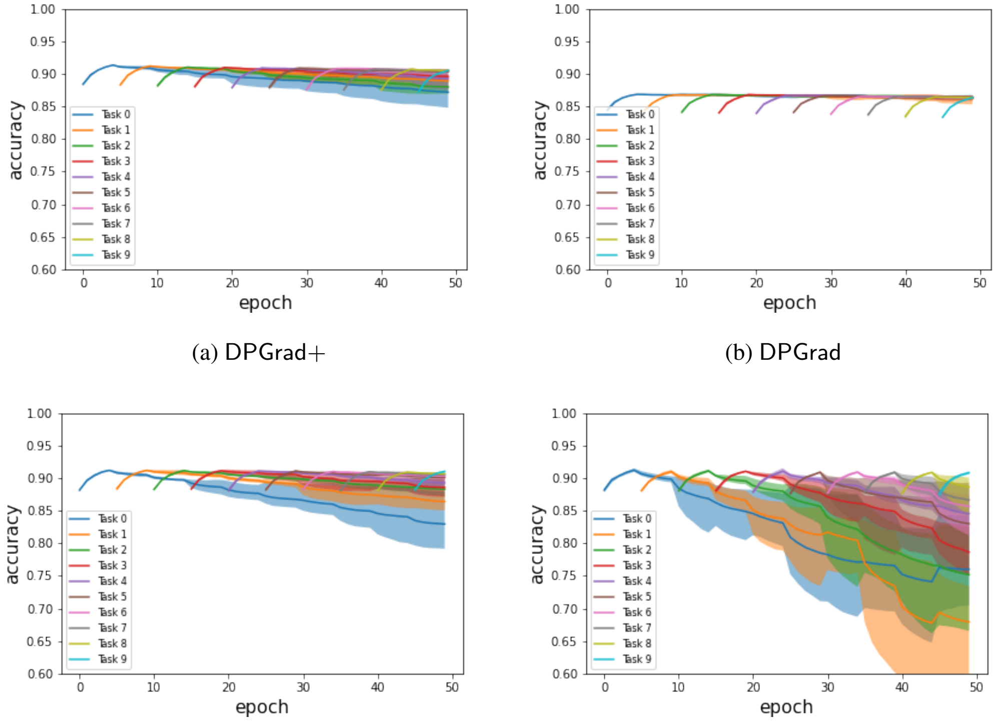
(a) DPGrad+

(b) DPGrad

(c) OGD

(d) SGD

Figure 4: Rotated MNIST (with error bar)



(a) DPGrad+

(b) DPGrad

(c) OGD

(d) SGD

Figure 5: Permuted MNIST (with error bar)