CONTINUAL LEARNING IN LOW-COHERENCE SUBSPACE: A STRATEGY TO MITIGATE LEARNING CAPACITY DEGRADATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Methods using gradient orthogonal projection, an efficient strategy in continual learning, have achieved promising success in mitigating catastrophic forgetting. However, these methods often suffer from the learning capacity degradation problem following the increasing number of tasks. To address this problem, we propose to learn new tasks in low-coherence subspaces rather than orthogonal subspaces. Specifically, we construct a unified cost function involving regular DNN parameters and gradient projections on the Oblique manifold. We finally develop a gradient descent algorithm on a smooth manifold to jointly minimize the cost function and minimize both the inter-task and the intra-task coherence. Numerical experimental results show that the proposed method has prominent advantages in maintaining the learning capacity when tasks are increased, especially on a large number of tasks, compared with baselines. Li & Lin (2018)

1 INTRODUCTION

Deep **N**eural **N**etworks (DNNs) have achieved promising performance on many tasks. However, they lack the ability for continual learning, i.e., they suffer from catastrophic forgetting French (1999) when learning sequential tasks, where catastrophic forgetting is a phenomenon of new knowledge interfering with old knowledge. Research on continual learning, also known as incremental learning Aljundi et al. (2018a); Chaudhry et al. (2018a); Chen & Liu (2018); Aljundi et al. (2017), and sequential learning Aljundi et al. (2018b); McCloskey & Cohen (1989), aims to find effective algorithms that enable DNNs to simultaneously achieve plasticity and stability, i.e., to achieve both high learning capacity and high memory capacity.

Various methods have been proposed to avoid or mitigate the catastrophic forgetting De Lange et al. (2019), either by replaying training samples Rolnick et al. (2019); Ayub & Wagner (2020); Saha et al. (2021), or reducing mutual interference of model parameters, features or model architectures between different tasks Zenke et al. (2017); Mallya & Lazebnik (2018); Wang et al. (2021). Among these methods, **G**radient **O**rthogonal **P**rojection (GOP) Chaudhry et al. (2020); Zeng et al. (2019); Farajtabar et al. (2020); Li et al. (2021) is an efficient continual learning strategy that advocates projecting gradients with the orthogonal projector to prevent the knowledge interference between tasks. GOP-based methods have achieved encouraging results in mitigating catastrophic forgetting. However, from Fig. 1, we observe that these methods suffer from the learning capacity degradation problem: their learning capacity is gradually degraded as the number of tasks increases and eventually becomes unlearnable. Specifically, when learning multiple tasks, e.g., more than 30 tasks in Fig. 1, their performance on new tasks dramatically decreases. These results suggest that the GOP-based methods focus on the stability and somewhat ignore the plasticity. Ignoring the plasticity may limit the task learning capacity of models, i.e., the number of tasks that a model can learn without forgetting.

To address this issue, we propose to learn new tasks in low-coherence subspaces rather than orthogonal subspaces. Specifically, Low-coherence projectors are utilized for each layer to project features and gradients into low-coherence subspaces. To achieve this, we construct a unified cost function to find projectors and develop a gradient descent algorithm on the Oblique manifold to jointly minimize inter-task coherence and intra-task coherence. Minimizing the inter-task coherence can reduce the mutual interference between tasks, and minimizing the intra-task coherence can enhance the model's expressive power. Restricting projectors on the Oblique manifold can avoid the scale ambiguity

Aharon et al. (2006); Wei et al. (2017), i.e., preventing the parameters of the projector from being extremely large or extremely small.

The main contributions of this work are summarized as follows. First, to address the learning capacity degradation problem of GOP, we propose a novel method, namely, **L**ow-**c**oherence **S**ubspace **P**rojection (LcSP), that replaces the orthogonal projectors with the low-coherence gradient projectors, allowing the DNN to maintain both plasticity and stability. Additionally, our work observes that the GOP models with **B**atch **N**ormalization (BN) Ioffe & Szegedy (2015) layers could cause catastrophic forgetting. This paper proposes two strategies in LcSP to solve this problem, i.e., replacing BN with **G**roup **N**ormalization (GN) Wu & He (2018) or learning specific BN for each task.

2 Related Work

In this section, we briefly review some existing works of continual learning and the GOP based methods.

Replay-based Strategy The basic idea of this type of approach is to use limited memory to store small amounts of data (e.g., raw samples) from previous tasks, called episodic memory, and to replay them when training a new task. Some of the existing works focused on selecting a subset of raw samples from the previous tasks Rolnick et al. (2019); Isele & Cosgun (2018); Chaudhry et al. (2019); Zhang et al. (2020). In contrast, others concentrated on training a generative model to synthesize new data that can substitute for the old data Shin et al. (2017); Van de Ven & Tolias (2018); Lavda et al. (2018); Ramapuram et al. (2020).

Regularization-based Strategy This strategy prevents catastrophic forgetting by introducing a regularization term in the loss function to penalize the changes in the network parameters. Existing works can be divided into data-focused, and prior-focused methods De Lange et al. (2021). The Data-focused methods take the previous model as the teacher and the current model as the student, transferring the knowledge from the teacher model to the student model through knowledge distillation. Typical methods include LwF Li & Hoiem (2017), LFL Jung et al. (2016), EBLL Rannen et al. (2017), DMC Zhang et al. (2020) and GD-WILD Lee et al. (2019). The prior-focused methods estimate a distribution over the model parameters, assigning an importance score to each parameter and penalizing the changes in significant parameters during learning. Relevant works include SI Zenke et al. (2017), EWC Kirkpatrick et al. (2017), RWalk Chaudhry et al. (2018a), AGS-CL Jung et al. (2020) and IMM Lee et al. (2017).

Parameter Isolation-based Strategy This strategy considers dynamically modifying the network architecture by pruning, parameter mask, or expansion to greatly or even completely reduce catastrophic forgetting. Existing works can be roughly divided into two categories. One is dedicated to isolating separate sub-networks for each task from a large network through pruning techniques and parameter mask, including PackNet Mallya & Lazebnik (2018), PathNet Fernando et al. (2017), HAT Serra et al. (2018) and Piggyback Mallya et al. (2018). Another class of methods dynamically expands the network architecture, increasing the number of neurons or sub-network branches, to break the limits of expressive capacity (Rusu et al., 2016; Aljundi et al., 2017; Xu & Zhu, 2018; Rosenfeld & Tsotsos, 2018). However, as the number of tasks growing, this approach also complicates the network architecture and increases the computation and memory consumption.

Gradient Orthogonal Projection-based Strategy Methods based on GOP strategies, which reduce catastrophic forgetting by projecting gradient or features with orthogonal projectors, have been shown to be effective in continual learning with encouraging results Farajtabar et al. (2020); Zeng et al. (2019); Saha et al. (2021); Wang et al. (2021); Chaudhry et al. (2020). According to the different ways of finding the projector, we can further divide the existing works into **C**ontext **O**rthogonal **P**rojection (COP) and **S**ubspace **O**rthogonal **P**rojection (SOP). Methods based on COP, such as OWM Zeng et al. (2019), Adam-NSCL Wang et al. (2021), and GPM Saha et al. (2021), always rely on the context of previous tasks to build projectors. In contrast to COP, SOP-based methods such as ORTHOG-SUBSPACE Chaudhry et al. (2020) use hand-crafted, task-specific orthogonal projectors and yield competitive results.

This paper proposes a novel approach to continual learning called LcSP. Compared with other methods based on GOP, LcSP trains the network on the low-coherence subspaces to balance plasticity and stability and overcomes the learning capacity degradation problem, which significantly decreases the performance of GOP methods with the increasing number of tasks.

3 Continual Learning Setup

This work adopts the Task-Incremental Learning (TIL) setting, where multiple tasks are learned sequentially. Let us assume that there are T tasks, denoted by $\mathcal{T}_t$ for $t = 1, \ldots, T$ with its training data $\mathcal{D}_t = \{(x_i, y_i, \tau_t)_{i=1}^{N_t}\}$. Here, the data $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}_t$ is assumed to be drawn from some independently and identically distributed random variable, and $\tau_t \in \mathcal{T}$ denotes the task identifier. In the TIL setting, the data $\mathcal{D}_t$ can be accessed if and only if task $\mathcal{T}_t$ arrives. When episodic memory is adopted, a limited number of data samples drawn from old tasks can be stored in the replay buffer $\mathcal{M}$ so that $\mathcal{D}_t \cup \mathcal{M}$ can be used for training when task $\mathcal{T}_t$ arrives.

Assuming that a network f parameterized with $\Phi = \{\theta, \varphi\}$ consists of two parts, where θ denotes the parameters of the backbone network and φ denotes the parameters of the classifier. Let $f(x; \theta) : \mathcal{X} \times \mathcal{T} \to \mathcal{H}$ denote the backbone network parameterized with $\theta = \{W_l\}_{l=1}^{L}$, which encodes the data samples x into feature vector. Let $f(x; \varphi) : \mathcal{H} \to \mathcal{Y}$ denote the classifier parameterized with $\varphi = w$ which returns the classification result of the feature vector obtained by $f(x; \theta)$. The goal of TIL is to learn T tasks sequentially with the network f and finally achieve the optimal loss on all tasks.

Evaluation Metrics Once the training on all tasks is finished, we evaluate the performance of algorithm by calculating the average accuracy $\mathcal{A}$ and forgetting $\mathcal{F}$ Chaudhry et al. (2020) of the network on the T tasks $\{\mathcal{T}_1, ..., \mathcal{T}_T\}$. Suppose all tasks come sequentially, let $Acc_{i,j}$ denote the test accuracy of the network on task $\mathcal{T}_i$ after learning task $\mathcal{T}_j$, where $i \le j$. The average accuracy is defined as

$$\mathcal{A} = \frac{1}{T} \sum_{i=1}^{T} Acc_{i,T}, \tag{1}$$

and the forgetting is defined as

$$\mathcal{F} = \frac{1}{T-1} \sum_{i=1}^{T-1} \max_{j \in \{i, \ldots, T-1\}} (Acc_{i,j} - Acc_{i,T}). \tag{2}$$

4 Continual Learning in Low-coherence Subspaces

In this section, we first introduce how to find task-specific, low-coherence projectors for LcSP on the Oblique manifold. We then describe how to use it in a specific DNN architecture to project features and gradients. Finally, we analyze the factors that enable LcSP to maintain plasticity and stability.

4.1 Constructing low-coherence Projectors on Oblique Manifold

Here, we first introduce the concept of coherence metric. The coherence metric is usually used in compressed sensing and sparse signal recovery to describe the correlation of the columns of a measurement matrix Candes et al. (2011); Candes & Romberg (2007). Formally, the coherence of a matrix $\boldsymbol{M}$ is defined as

$$\mu(\boldsymbol{M}, \boldsymbol{N}) = \begin{cases} \max_{j<k} \frac{|\langle \boldsymbol{M}_j, \boldsymbol{M}_k \rangle|}{\|\boldsymbol{M}_j\|_2 \|\boldsymbol{M}_k\|_2}, & \boldsymbol{M} = \boldsymbol{N} \\ \max_{i,j} \frac{|\langle \boldsymbol{M}_i, \boldsymbol{N}_j \rangle|}{\|\boldsymbol{M}_i\|_2 \|\boldsymbol{N}_j\|_2}, & \boldsymbol{M} \neq \boldsymbol{N} \end{cases}. \tag{3}$$

where $\boldsymbol{M}_j$ and $\boldsymbol{M}_k$ denote the column vectors of matrix $\boldsymbol{M}$. Without causing confusion, we use $\mu(\boldsymbol{M})$ denote $\mu(\boldsymbol{M}, \boldsymbol{M})$. To measure the coherence between different projectors, we introduce the

Babel function Li & Lin (2018), measuring the maximum total coherence between a fixed atom and a collection of other atoms in a dictionary, which can be described as follows.

$$\mathbf{B}(\boldsymbol{M}) = \max_{\Lambda, |\Lambda| = \boldsymbol{M}} \max_{i \notin \Lambda} \sum_{j \in \Lambda} \frac{|\langle \boldsymbol{M}_i, \boldsymbol{M}_j \rangle|}{\|\boldsymbol{M}_i\| \, \|\boldsymbol{M}_j\|} \tag{4}$$

With the concept of a coherence metric in mind, we then introduce the main optimization objective in finding projectors. Specifically, suppose that the DNN has learned the task $\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_{t-1}$ in the subspace $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_{t-1}$, respectively, and $\boldsymbol{P}_1, \boldsymbol{P}_2, .., \boldsymbol{P}_{t-1}$ denote the projectors of all previous tasks. When learning task $\mathcal{T}_t$, we project features and gradients into a d_t-dimensional low-coherence subspace $\mathcal{S}_t$ with projector $\boldsymbol{P}_t$ so that the LcSP can prevent catastrophic forgetting. The projector $\boldsymbol{P}_t$ can be found by optimizing

$$\begin{aligned} &\arg\min \mathbf{B}(\boldsymbol{P}_t), \\ &\text{s.t.} \quad \boldsymbol{P}_t \in \mathbb{R}^{m \times m}, \quad \operatorname{rank}(\boldsymbol{P}_t) = d_t. \end{aligned} \tag{5}$$

Two considerations need to be taken in solving Eq. (5), i.e., considering the rank constraint and avoiding the entries of $\boldsymbol{P}_t$ being extremely large or extremely small. With these considerations in mind, we can rephrase the rank and scale constrained problem as a problem on the Riemannian manifold, more specifically on the Oblique manifold $\mathcal{OM}(m, d_t)$, by setting $\boldsymbol{P}_t = \boldsymbol{O}_t \boldsymbol{O}_t^\top$ and normalizing the columns of $\boldsymbol{O}_t$, i.e, $\operatorname{diag}(\boldsymbol{O}_t^\top \boldsymbol{O}_t) = \boldsymbol{I}_n$, where $\operatorname{diag}(\cdot)$ represents the diagonal matrix and $\boldsymbol{I}_n$ is the $n \times n$ identity matrix. With these settings, the new cost function $\mathbf{J}(\cdot)$ and optimization problem can be described as follows:

$$\begin{aligned} \mathbf{J}(\boldsymbol{O}_t) &= \begin{cases} \lambda \cdot \mathbf{B}(\boldsymbol{O}_t \boldsymbol{O}_t^\top) + \gamma \cdot \mu(\boldsymbol{O}_t \boldsymbol{O}_t^\top), & t > 1 \\ \mu(\boldsymbol{O}_t \boldsymbol{O}_t^\top), & t = 1 \end{cases}, \\ \boldsymbol{O}_t &= \arg\min \mathbf{J}(O_t), \quad \text{s.t.} \quad \boldsymbol{O}_t \in \mathcal{OB}(m, d_t). \end{aligned} \tag{6}$$

In the cost function $\mathbf{J}(\boldsymbol{O}_t)$, we further divided the optimization objective into inter-task $\mathbf{B}(\boldsymbol{O}_t \boldsymbol{O}_t^\top)$ and intra-task $\mu(\boldsymbol{O}_t \boldsymbol{O}_t^\top)$, and utilize parameters λ and γ to provide a trade-off between them. Here, intra-task coherence is optimized to maintain the full rank of $\boldsymbol{O}_t$. Meeting the full-rank constraint helps to balance plasticity and stability in the case of increasing tasks. Relevant ablation studies and numerical analyses are given in the appendix.

Optimization on the Oblique manifold, i.e., the solution lies on the Oblique manifold, is a well-established area of research Absil et al. (2009); Absil & Gallivan (2006); Selvan et al. (2012). Here, we briefly summarize the main steps of the optimization process. Formally, the Oblique manifold $\mathcal{OM}(n, p)$ is defined as

$$\mathcal{OM}(n, p) \triangleq \{\boldsymbol{X} \in \mathbb{R}^{n \times p} : \operatorname{diag}(\boldsymbol{X}^\top \boldsymbol{X}) = \boldsymbol{I}_p\}, \tag{7}$$

representing the set of all $n \times p$ matrices with normalized columns. $\mathcal{OM}$ can also be considered as an embedded Riemannian manifold of $\mathbb{R}^{n \times p}$, endowed with the canonical inner product

$$\langle \boldsymbol{X}_1, \boldsymbol{X}_2 \rangle = \operatorname{trace}\left(\boldsymbol{X}_1^\top \boldsymbol{X}_2\right), \tag{8}$$

where $\operatorname{trace}(\cdot)$ represents the sum of the diagonal elements of the given matrix. For a given point $\boldsymbol{X}$ on $\mathcal{OM}$, the tangent space at $\boldsymbol{X}$, denoted by $T_{\boldsymbol{X}}\mathcal{OM}$, is defined as

$$T_{\boldsymbol{X}}\mathcal{OM}(n, p) = \{\boldsymbol{U} \in \mathbb{R}^{n \times p} : \operatorname{diag}(\boldsymbol{X}^\top \boldsymbol{U}) = 0\}. \tag{9}$$

Further, the tangent space projector $\mathbf{P}_{\boldsymbol{X}}$ at $\boldsymbol{X}$ which projects $\boldsymbol{H} \in \mathbb{R}^{n \times p}$ into $T_{\boldsymbol{X}}\mathcal{OM}$, is represented as

$$\mathbf{P}_{\boldsymbol{X}}(\boldsymbol{H}) = \boldsymbol{H} - \boldsymbol{X} \operatorname{ddiag}\left(\boldsymbol{X}^\top \boldsymbol{H}\right), \tag{10}$$

where ddiag sets all off-diagonal entries of a matrix to zero. When optimizing on $\mathcal{OM}$, the kth iterate $\boldsymbol{X}_k$ must move along a descent curve on $\mathcal{OM}$ for the cost function, such that the next iterate $\boldsymbol{X}_{k+1}$ will be fixed on the manifold. This is achieved by the retraction

$$\mathbf{R}_{\boldsymbol{X}_k}(\boldsymbol{U}) = \operatorname{normalize}(\boldsymbol{X}_k + \boldsymbol{U}), \tag{11}$$

where normalize scales each column of the input matrix to have unit length. Finally, with this knowledge, we can extend the gradient descent algorithm to solve any unconstrained optimization problems on $\mathcal{OM}$, which can be summarized as

$$\begin{aligned} \boldsymbol{U} &= \mathbf{P}_{\boldsymbol{X}_k}(\nabla_{\boldsymbol{X}_k} \mathbf{J}), \\ \boldsymbol{X}_{k+1} &= \mathbf{R}_{\boldsymbol{X}_k}(-\alpha \boldsymbol{U}), \end{aligned} \tag{12}$$

where $\nabla_{\boldsymbol{X}_k}\mathbf{J}$ denotes the Euclidean gradient at the kth iterate and α is the step size. Finally, our algorithm for finding $\boldsymbol{O}_t$ in $\mathcal{OM}$ to task $\mathcal{T}_t$ is summarized in Algorithm 1.

Algorithm 1 Construct the $\boldsymbol{O}_t$ on $\mathcal{OM}$ for Task $\mathcal{T}_t$

```
Input: $\boldsymbol{O}_1, \ldots, \boldsymbol{O}_{t-1}$
Output: $\boldsymbol{O}_t$
$\mathbf{R}_{\boldsymbol{X}}(\boldsymbol{U}) := \mathrm{normalize}(\boldsymbol{X} + \boldsymbol{U})$    ▷ normalize scales each column of the input matrix to have norm 1
$\boldsymbol{X}_0 \leftarrow$ random initialization on $\mathcal{OM}$
$k \leftarrow 0$
Set tolerance error $0 \leq \mathcal{E} \ll 1$
while True do
    $\boldsymbol{G} \leftarrow \nabla f(\boldsymbol{X}_k)$    ▷ Calculate Euclidean Gradient
    $\boldsymbol{U} \leftarrow \boldsymbol{G} - \boldsymbol{X}_k \,\mathrm{ddiag}(\boldsymbol{X}_k^\top \boldsymbol{G})$    ▷ Calculate Riemann Gradient
    if $\|\boldsymbol{U}\| \leq \mathcal{E}$ then
        break
    end if
    $\alpha \in (0, 0.5), \quad \beta \in (0, 1)$
    $t \leftarrow 1$    ▷ Initial step size
    while $\mathbf{J}(\mathbf{R}_{\boldsymbol{X}_k}(-t \cdot \boldsymbol{U})) > \mathbf{J}(\boldsymbol{X}_k) - \alpha \cdot t \cdot \|\boldsymbol{U}\|_2^2$ do    ▷ Searching the step size for calculating next iterate
        $t \leftarrow \beta \cdot t$
    end while
    $\boldsymbol{X}_{k+1} \leftarrow \mathbf{R}_{\boldsymbol{X}_k}(-t \cdot \boldsymbol{U})$    ▷ Updating the next iterate $\boldsymbol{X}_{k+1}$ and fixing it on manifold
    $k \leftarrow k + 1$
end while
$\boldsymbol{O}_t = \boldsymbol{X}_k$
return $\boldsymbol{O}_t$
```

4.2 The Application of Low-coherence Projectors in DNNs

With the LcSP at hand, the following introduces some technical details of applying LcSP in DNNs. When learning task $\mathcal{T}_t$, LcSP first constructs task-specific projector $\boldsymbol{P}_t^l$ for each layer before training, and freezes them during training. These projectors are used to project the features and gradients, ensuring that the DNN learns in the low-coherence subspace. Specifically, suppose that a network f with L linear layers is used as DNN architecture, let $\boldsymbol{W}_t^l$, $\boldsymbol{x}_t^l$, $\boldsymbol{z}_t^l$, σ^l, and $\boldsymbol{P}_t^l$ denote the model parameters, the input features, the output features, the activation function, and the introduced low-coherence projector in layer $l \in \{1, ..., L\}$, respectively. LcSP introduces $\boldsymbol{P}_t^l$ immediately after $\boldsymbol{W}_t^l$ such that the pre-activation features are projected into the subspace, i.e.,

$$\begin{aligned} \boldsymbol{z}_t^l &= (\boldsymbol{x}_t^l \boldsymbol{W}_t^l)\boldsymbol{P}_t^l, \\ \boldsymbol{x}_t^{l+1} &= \sigma^l(\boldsymbol{z}_t^l). \end{aligned} \tag{13}$$

According to the chain rule of derivation, the gradients at $\boldsymbol{W}_t^l$ will also be multiplied with $\boldsymbol{P}_l^t$ in backpropagation, as follows

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial (W_t^l)_{(i,:)}} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}_t^l} \frac{\partial \boldsymbol{z}_t^l}{\partial (\boldsymbol{W}_t^l)_{(i,:)}} \\ &= \frac{\partial \mathcal{L}}{\boldsymbol{z}_t^L} \prod_{k=l}^{L-1} \frac{\partial \boldsymbol{z}_t^{k+1}}{\partial \boldsymbol{z}_t^k} \cdot (\boldsymbol{x}_t^l)_i \cdot \boldsymbol{P}_t^l, \end{aligned} \tag{14}$$

where $(\boldsymbol{W}_t^l)_{(i,:)}$ represents the ith row of $\boldsymbol{W}_t^l$ and $(\boldsymbol{x}_t^l)_i$ is the ith element of $\boldsymbol{x}_t^l$. In **C**onvolutional **N**eural **N**etworks (CNNs), the input and the output typically represent the image features and have more than two dimensions, e.g., input channel, output channel, height, and width. In this case, we reshape $\boldsymbol{z}^l \in \mathbb{R}^{c_{\text{out}} \times (c_{in} \cdot h \cdot w)}$ to $\boldsymbol{z}^l \in \mathbb{R}^{(c_{in} \cdot h \cdot w) \times c_{\text{out}}}$ and align the dimension of projector with the output channel so that $\boldsymbol{P}_t^l \in \mathbb{R}^{c_{\text{out}} \times c_{\text{out}}}$. After the projection, we recover the shape of $\boldsymbol{z}_t^l$ so that it can be used as input for the next layer.

Overcoming the Catastrophic Forgetting in BN based models BN is a widely used module in DNNs to make training of DNNs faster and more stable through normalization of the layers' features by re-centering and re-scaling Ioffe & Szegedy (2015). However, re-centering and re-scaling of the layers' features changes the data distribution (e.g., the mean and the variance) of features of previous tasks, which often leads to the catastrophic forgetting of LcSP. For example, when learning the new task $\mathcal{T}_t$, $\boldsymbol{W}^l_t$ may not work for $\mathcal{T}_t$ due to the change in data distribution caused by BN. To solve this problem, we propose two strategies in LcSP: **the strategy (1)** learning specific BN for each task, or **the strategy (2)** using GN instead of BN. We verify the effectiveness of these two strategies in experiments and compare their performance in §5.

4.3 Method Analysis

In this section, we provide analysis on plasticity and stability of LcSP.

Stability Analysis Let $\theta = \{\boldsymbol{W}^l_t\}_{l=1}^L$ denote the parameter set of f; $\Delta\theta = \{\Delta\boldsymbol{W}^1_t, \dots, \Delta\boldsymbol{W}^L_t\}$ denote set of variation values of parameters after learning task $\mathcal{T}_t$; $\boldsymbol{P}_t = \{\boldsymbol{P}^l_t\}_{l=1}^L$ denote the projectors set obtained by Algorithm 1; $\boldsymbol{x}^l_{q,t}$ and $\boldsymbol{z}^l_{q,t}$ denote the input and ouput when feeding the data of task $\mathcal{T}_q$ $(q \le t)$ into the network f , which has been optimized in learning task $\mathcal{T}_t$.

Lemma 1. *Assume that f is fed the data of task $\mathcal{T}_t$ $(q < t)$, then f can effectively overcomes catastrophic forgetting if*

$$\boldsymbol{z}^l_{q,q} \approx \boldsymbol{z}^l_{q,t}, \quad \forall q \le t \tag{15}$$

holds for $l \in \{1, 2, ..., L\}$.

Lemma 1 suggests that f can overcome catastrophic forgetting if the output of f to previous tasks is invariant. In the following, we prove that LcSP achieves approximate invariance to the output of previous tasks.

Proof. *Suppose $q = t - 1$. When $l = 1$, $\boldsymbol{x}^l_{q,t} = \boldsymbol{x}^l_{q,q}$. Then*

$$\begin{aligned}\boldsymbol{z}^l_{q,t} &= \boldsymbol{x}^l_{q,t}(\boldsymbol{W}^l_q + \Delta\boldsymbol{W}^l_t)\boldsymbol{P}^l_q \\ &= \boldsymbol{x}^l_{q,t}\boldsymbol{W}^l_q P^l_q + \boldsymbol{x}^l_{q,t}\Delta\boldsymbol{W}^l_t\boldsymbol{P}^l_q \\ &= \boldsymbol{z}^l_{q,q} + \boldsymbol{x}^l_{q,t}\Delta\boldsymbol{W}^l_t\boldsymbol{P}^l_q.\end{aligned} \tag{16}$$

Let g^l_t denote the gradient when training the network on task $\mathcal{T}_t$. In backpropagation, $\Delta\boldsymbol{W}^l_t = \boldsymbol{g}^l_t\boldsymbol{P}^l_t$. Then $\boldsymbol{x}^l_{q,t}\Delta\boldsymbol{W}^l_t\boldsymbol{P}^l_q = \boldsymbol{x}^l_{q,t}\boldsymbol{g}^l_t\boldsymbol{P}^l_t\boldsymbol{P}^l_q$. If the inter-task coherence $\mu(\boldsymbol{P}^l_t, \boldsymbol{P}^l_q) \approx 0$, then $\boldsymbol{P}^l_t\boldsymbol{P}^l_q \approx \boldsymbol{0}$. Projectors satisfying this condition can be found by Algorithm 1. We can prove that $\boldsymbol{z}^l_{q,q} \approx \boldsymbol{z}^l_{q,t}$ holds for all layers by repeating the above process. This proof can also be generalized to any previous task $\mathcal{T}_q$.

Plasticity Analysis Let $\tilde{\boldsymbol{g}}^l_t = \boldsymbol{g}^l_t\boldsymbol{P}^l_t$ denote the projected gradient at $\boldsymbol{W}^l_t$. f can achieve optimal loss on task $\mathcal{T}_t$ if $\langle\boldsymbol{g}^l_t, \tilde{\boldsymbol{g}}^l_t\rangle > 0$ holds for each $l \in \{1, \dots, L\}$, where $\langle\cdot,\cdot\rangle$ represents the inner product. Here, we prove that $\langle\boldsymbol{g}^l_t, \tilde{\boldsymbol{g}}^l_t\rangle > 0$ holds for each $l \in \{1, \dots, L\}$.

Proof. *Let $\tilde{\boldsymbol{g}}^l_t = \boldsymbol{g}^l_t\boldsymbol{P}^l_t$ denote the projected gradient, we have*

$$\begin{aligned}\langle\boldsymbol{g}^l_t, \tilde{\boldsymbol{g}}^l_t\rangle &= \boldsymbol{g}_t\tilde{\boldsymbol{g}}^l_{t^\top} = \boldsymbol{g}^l_t\boldsymbol{O}^l_t\boldsymbol{O}^{l\top}_t\boldsymbol{g}^{l\top}_t \\ &= \langle\boldsymbol{g}^l_t\boldsymbol{O}^l_t, \boldsymbol{g}^l_t\boldsymbol{O}^l_t\rangle = \|\boldsymbol{g}^l_t\boldsymbol{O}^l_t\| > 0.\end{aligned} \tag{17}$$

Note that $\|\boldsymbol{g}^l_t\boldsymbol{O}^l_t\|$ is always positive unless $\boldsymbol{g}^l_t\boldsymbol{O}^l_t$ is $\boldsymbol{0}$. This result is easy to generalize to each layer.

5 Experiments

In this section, we evaluate our approach on several popular continual learning benchmarks and compare LcSP with previous state-of-the-art methods. The result of accuracy and forgetting demonstrate the effectiveness of our LcSP, especially when the number of tasks is large.

5.1 Benchmarks

Benchmarks for Learning 20 Tasks We conducted this experiment on four image classification datasets: Permuted MNIST, Rotated MNIST, Split CIFAR100 and Split miniImageNet. Permuted MNIST is constructed by randomly rearranging MNIST LeCun (1998) image pixels, using different seeds for different tasks. Rotated MNIST is constructed by rotating the MNIST image at a certain

Table 1: The average accuracy and forgetting results of the proposed LcSP and baselines. **Memory** denotes whether the method is trained using a replay strategy with episodic memory.

Methods	**Memory**	**Permuted MNIST**		**Rotated MNIST**	
		Accuracy(%)	Forgetting	Accuracy(%)	Forgetting
EWC Kirkpatrick et al. (2017)	✗	$68.4(\pm 0.76)$	$0.25(\pm 0.01)$	$43.6(\pm 0.81)$	$0.53(\pm 0.01)$
AGEM Chaudhry et al. (2018b)	✓	$78.3(\pm 0.42)$	$0.15(\pm 0.01)$	$60.5(\pm 1.77)$	$0.36(\pm 0.01)$
ER-RING Chaudhry et al. (2019)	✓	$79.5(\pm 0.31)$	$0.12(\pm 0.01)$	$70.9(\pm 0.38)$	$0.24(\pm 0.01)$
KCL Derakhshani et al. (2021)	✗	$85.5(\pm 0.78)$	**0.02(±0.00)**	$81.8(\pm 0.60)$	**0.01(±0.00)**
OWM Zeng et al. (2019)	✗	46.35	0.01	44.78	0.01
Adam-NSCL Wang et al. (2021)	✗	26.44	0.70	42.73	0.56
GPM Saha et al. (2021)	✗	80.54	0.16	80.34	0.16
ORTHOG-SUBSPACE Chaudhry et al. (2020)	✗	$86.6(\pm 0.91)$	$0.04(\pm 0.01)$	$80.1(\pm 0.95)$	$0.14(\pm 0.01)$
LcSP	✗	**92.2(±0.65)**	$0.05(\pm 0.01)$	**86.6(±0.96)**	$0.08(\pm 0.01)$

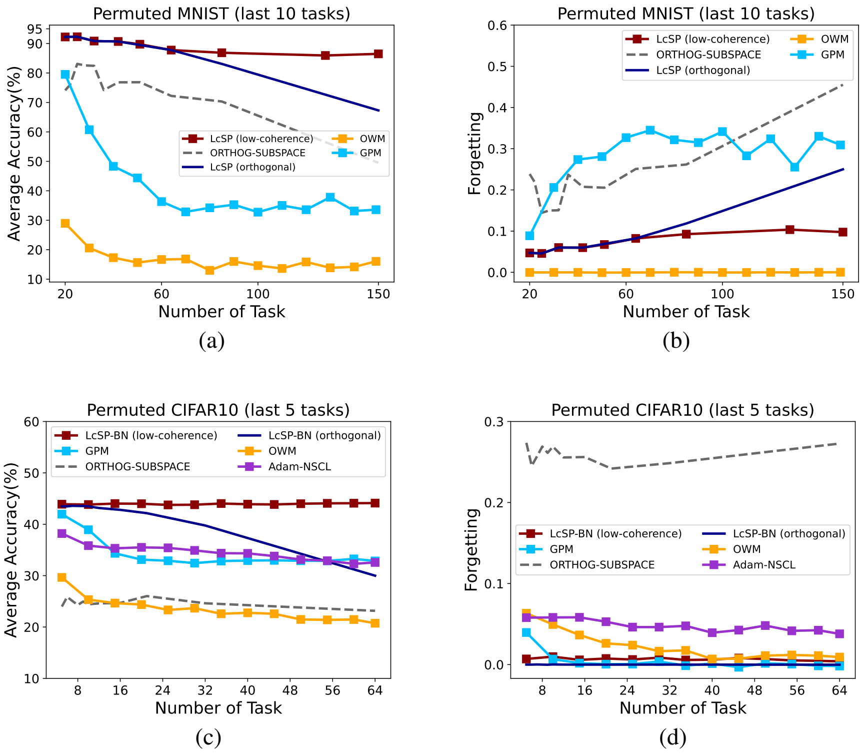
Methods	**Memory**	**Split CIFAR100**		**Split miniImageNet**	
		Accuracy(%)	Forgetting	Accuracy(%)	Forgetting
EWC Kirkpatrick et al. (2017)	✗	$43.2(\pm 2.77)$	$0.26(\pm 0.02)$	$34.8(\pm 2.34)$	$0.24(\pm 0.04)$
AGEM Chaudhry et al. (2018b)	✓	$51.3(\pm 3.49)$	$0.18(\pm 0.03)$	$42.3(\pm 1.42)$	$0.17(\pm 0.01)$
ER-RINGChaudhry et al. (2019)	✓	$59.6(\pm 1.19)$	$0.14(\pm 0.01)$	$49.8(\pm 2.92)$	$0.12(\pm 0.01)$
HAT Serra et al. (2018)	✗	72.06	**0.00**	59.78	0.03
KCL Derakhshani et al. (2021)	✗	$62.7(\pm 0.89)$	$0.06(\pm 0.01)$	$53.3(\pm 0.57)$	$0.04(\pm 0.00)$
OWM Zeng et al. (2019)	✗	50.94	0.30	-	-
ORTHOG-SUBSPACE Chaudhry et al. (2020)	✓	$64.3(\pm 0.59)$	$0.07(\pm 0.01)$	$51.4(\pm 1.44)$	$0.10(\pm 0.01)$
GPM Saha et al. (2021)	✗	72.48	**0.00**	60.41	**0.00**
Adam-NSCL Wang et al. (2021)	✗	**75.95**	0.04	63.27	0.06
ORTHOG-SUBSPACE	✗	$29.1(\pm 1.99)$	$0.53(\pm 0.03)$	$34.6(\pm 1.12)$	$0.22(\pm 0.01)$
ORTHOG-SUBSPACE-BN	✗	$34.3(\pm 0.31)$	$0.35(\pm 0.01)$	$31.2(\pm 1.30)$	$0.23(\pm 0.02)$
ORTHOG-SUBSPACE-GN	✗	$40.9(\pm 2.31)$	$0.11(\pm 0.03)$	$38.6(\pm 1.23)$	$0.35(\pm 0.02)$
LcSP	✗	$24.1(\pm 0.76)$	$0.52(\pm 0.01)$	$23.0(\pm 0.84)$	$0.50(\pm 0.00)$
LcSP-GN	✗	$66.7(\pm 0.54)$	**0.00(±0.00)**	$65.0(\pm 1.14)$	**0.00(±0.00)**
LcSP-BN	✗	72.7(±**1.18**)	**0.00(±0.00)**	**67.9(±0.40)**	**0.00(±0.00)**

angle. The rotation angle is a random value between $[0, \pi]$ and arbitrary selection for different tasks. In this experimental setup, both of the above MNIST datasets contain 20 tasks, each task containing 10,000 samples from 10 classes. Split CIFAR is constructed by splitting CIFAR100 into multiple tasks, where each task contains the data pertaining to five random classes (without replacement) out of the total 100 classes. Split miniImageNet Vinyals et al. (2016) is a subset of ImageNet. In Split miniImageNet, each task contains the data from five random classes (without replacement) out of 100 classes. Both CIFAR100 and miniImageNet contain 20 tasks, each contains 250 samples from each of the five classes.

Benchmarks for Learning 150 Tasks and 64 Tasks This experiment was conducted on two image classification datasets: Permuted MNIST and Permuted CIFAR10. Both Permuted MNIST and Permuted CIFAR10 are obtained by randomly rearranging image pixels. In this experimental setup, Permuted MNIST contains 150 tasks, and Permuted CIFAR10 contains 64 tasks, each containing 10,000 samples from 10 classes.

5.2 Baselines

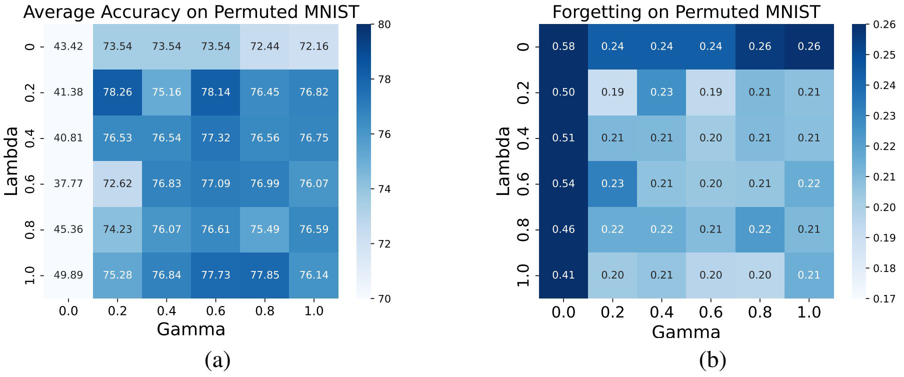
We compare the proposed method with SOTA based on GOP. As aforementioned, we generalize GOP into COP and SOP. For methods based on the COP strategy, we compare proposed method with OWM Zeng et al. (2019), Adam-NSCL Wang et al. (2021) and GPM Saha et al. (2021). For methods based on the SOP strategy, we compare the proposed method with ORTHOG-SUBSPACE Chaudhry et al. (2020). Moreover, we also compare our method with HAT Serra et al. (2018), EWC Kirkpatrick et al. (2017), ER-Ring Chaudhry et al. (2019), AGEM Chaudhry et al. (2018b) and **K**ernel **C**ontinual **L**earning (KCL) Derakhshani et al. (2021).



(a) (b)

(c) (d)

Figure 1: Fig. 1(a) and 1(b) show the average accuracy and forgetting of the last 10 tasks on Permuted MNIST when learning 150 tasks. Fig. 1(c) and 1(d) show the average accuracy and forgetting of the last 5 tasks on Permuted CIFAR10 when learning 64 tasks.



(a) (b)

Figure 2: Average accuracy and forgetting for different λ and γ on Permuted MNIST. A fully connected network with 2 hidden layers, each with 64 neurons, is used for this experiment.

5.3 Implementation Details

Learning 20 Tasks For experiments on Permuted MNIST and Rotated MNIST, all methods use a fully connected network with two hidden layers, each with 256 neurons, using ReLU activations. For experiments on CIFAR and miniImageNet, all methods use standard ResNet18 architecture except OWM, HAT, and GPM which use AlexNet Krizhevsky et al. (2012). As described in § 4.2, the proposed LcSP makes two changes to the BN layer in standard ResNet18: (1) learning specific BN for each task and (2) replacing BN with GN. We also apply these strategies to ORTHOG-SUBSPACE for additional comparison. For experiments on MNIST, all tasks share the same classifier. For experiments on CIFAR and miniImageNet, each task requires a task-specific classifier. For all experiments, LcSP does not use episodic memory to store data samples for data replay. For all methods, We use **S**tochastic **G**radient **D**escent (SGD) uniformly. The learning rate is set to 0.01 for experiments on MNIST and 0.003 for experiments on CIFAR and ImageNet. Both λ and γ in Eq. (**??**) are set to 1. All experiments were run five times with five different random seeds, with the batch size besing 10.

Learning 150 **Tasks and** 64 **Tasks** For experiments on Permuted CIFAR10, LcSP uses ResNet18 architecture and applies **the strategy (1)** to change BN in ResNet18. To compare the performance of different GOP strategies, we did not use episodic memory for ORTHOG-SUBSPACE. Except for these changes, other experimental settings are the same as described above.

5.4 Experimental Results

Comparisons of Learning 20 Tasks Table 1 compares the average accuracy and forgetting results of the proposed **LcSP** and its variants (**LcSP-BN** and **LcSP-GN**) with baselines on the four continual learning benchmarks. Therein, **LcSP-BN** and **LcSP-GN** adopt *strategy* (1) and (2) described in § 4.2, respectively. First, as shown in Table 1, the proposed methods outperform all baselines on MNIST and miniImageNet. On Permuted MNIST and Rotated MNIST, the average accuracy of LcSP surpasses the baselines by $23.8\% \sim 5.6\%$ and $43\% \sim 4.8\%$, respectively. On miniImageNet, the average accuracy of **LcSP-BN** surpasses the baselines by $44.9\% \sim 4.63\%$. On CIFAR100, the proposed **LcSP-BN** achieved a competitive performance with the second highest average accuracy, 3.25% lower than Adam-NSCL, and a forgetting rate of 0. The average accuracy of **LcSP-GN** also outperforms most baselines, being lower than Adam-NSCL, HAT and GPM on CIFAR100 but higher than compared methods on miniImageNet. These results suggest that minimizing the inter-task and intra-task coherence with low-coherence projectors is an effective strategy for solving catastrophic forgetting. Secondly, results on CIFAR100 and miniImageNet also show that BN in ORTHOG-SUBSPACE Chaudhry et al. (2020) and LcSP may change previous tasks' data distribution and lead to catastrophic forgetting. Both strategies (1) and (2) described in § 4.2 can effectively solve this problem.

Comparisons of Learning 150 Tasks and 64 Tasks In order to demonstrate the high advantage of proposed methods in learning a long sequence of tasks, the following experiments compare the results with 64 tasks and 150 tasks. Note that, in Fig. 1, LcSP (orthogonal) and LcSP-BN (orthogonal) use orthogonal projectors as comprisons while LcSP (low-coherence) and LcSP-BN (low-coherence) use low-coherence projectors. Fig. 1(a) and 1(b) report the average accuracy and forgetting of last 10 tasks, with learning 150 tasks on Permuted MNIST. Fig. 1(c) and 1(d) report the average accuracy and forgetting of last 5 tasks with learning 64 tasks on Permuted CIFAR10. The average accuracy of all methods, except the proposed LcSP-BN (low-coherence), dramatically degrades or is consistently low as the number of tasks increases. Furthermore, it can be seen from 1(d) that all methods except ORTHOG-SUBSPACE have almost no forgetting. This result indicates that methods using orthogonal projectors gradually lose their learning capacity with increasing number of tasks. The proposed method uses the low-coherence projector to relax the orthogonal restriction, effectively solving this problem. Fig. 2 gives the ablation study and shows the performance of our method with different λ and γ. When λ equals γ, the result of average accuracy on Permuted MNIST reached the highest. Results reach the worst when either λ or γ equals zero. These results indicate that both inter-task and intra-task coherence should be minimized to solve the plasticity and stability dilemma.

6 Conclusion

This paper proposed a novel gradient projection approach for continual learning to address gradient orthogonal projection's learning capacity degradation problem. Instead of learning in orthogonal subspace, we propose projecting features and gradients via low-coherence projectors to minimize inter-task and intra-task coherence. Additionally, two strategies have been proposed to mitigate the catastrophic forgetting caused by the BN layer, i.e., replacing BN with GN or learning specific BN for each task. Extensive experiments show that our approach works well in alleviating forgetting and has a significant advantage in maintaining learning capacity, especially in learning long sequence tasks.

REFERENCES

P-A Absil and Kyle A Gallivan. Joint diagonalization on the oblique manifold for independent component analysis. 5:V–V, 2006.

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. Optimization algorithms on matrix manifolds. 2009.

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11): 4311–4322, 2006.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. pp. 3366–3375, 2017.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. pp. 139–154, 2018a.

Rahaf Aljundi, Marcus Rohrbach, and Tinne Tuytelaars. Selfless sequential learning. *arXiv preprint arXiv:1806.05421*, 2018b.

Ali Ayub and Alan R Wagner. Storing encoded episodes as concepts for continual learning. *arXiv preprint arXiv:2007.06637*, 2020.

Emmanuel Candes and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.

Emmanuel J Candes, Yonina C Eldar, Deanna Needell, and Paige Randall. Compressed sensing with coherent and redundant dictionaries. *Applied and Computational Harmonic Analysis*, 31(1):59–73, 2011.

Miguel A Carreira-Perpinán. A review of dimension reduction techniques. *Department of Computer Science. University of Sheffield. Tech. Rep. CS-96-09*, 9:1–69, 1997.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. pp. 532–547, 2018a.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018b.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and M Ranzato. Continual learning with tiny episodic memories. 2019.

Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33:9900–9911, 2020.

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

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2(6), 2019.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Mohammad Mahdi Derakhshani, Xiantong Zhen, Ling Shao, and Cees Snoek. Kernel continual learning. pp. 2621–2631, 2021.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. pp. 3762–3773, 2020.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3 (4):128–135, 1999.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. pp. 448–456, 2015.

David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. 32(1), 2018.

Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. *Advances in Neural Information Processing Systems*, 33:3647–3658, 2020.

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.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Frantzeska Lavda, Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Continual classification learning using generative models. *arXiv preprint arXiv:1810.10612*, 2018.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. pp. 312–321, 2019.

Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

Elizaveta Levina and Peter Bickel. Maximum likelihood estimation of intrinsic dimension. *Advances in neural information processing systems*, 17, 2004.

Huan Li and Zhouchen Lin. Construction of incoherent dictionaries via direct babel function minimization. In *Asian Conference on Machine Learning*, pp. 598–613. PMLR, 2018.

Yanni Li, Bing Liu, Kaicheng Yao, Xiaoli Kou, Pengfan Lv, Yueshen Xu, and Jiangtao Cui. Defeating catastrophic forgetting via enhanced orthogonal weights modification. *arXiv preprint arXiv:2111.10078*, 2021.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. pp. 7765–7773, 2018.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. pp. 67–82, 2018.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. 24:109–165, 1989.

Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *Neurocomputing*, 404:381–400, 2020.

Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. pp. 1320–1328, 2017.

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

Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 42(3):651–663, 2018.

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.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

Suviseshamuthu Easter Selvan, Umberto Amato, Kyle A Gallivan, Chunhong Qi, Maria Francesca Carfora, Michele Larobina, and Bruno Alfano. Descent algorithms on oblique manifold for source-adaptive ica contrast. *IEEE Transactions on Neural Networks and Learning Systems*, 23 (12):1930–1947, 2012.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. pp. 4548–4557, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Gido M Van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.

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

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. pp. 184–193, 2021.

Xian Wei, Yuanxiang Li, Hao Shen, Weidong Xiang, and Yi Lu Murphey. Joint learning sparsifying linear transformation for low-resolution image synthesis and recognition. *Pattern Recognition*, 66: 412–424, 2017.

Yuxin Wu and Kaiming He. Group normalization. pp. 3–19, 2018.

Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. pp. 3987–3995, 2017.

Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. pp. 1131–1140, 2020.

A Appendix

In this section, we give the implementation details about the experiments on Permuted MNIST and Permuted CIFAR10, to help readers reproduce these experiments. Additionally, more ablation studies and experimental results are provided here to further support the conclusions and contributions.

A.1 Implementation details

The main hyperparameter settings are listed in Table 2 and 3. For the baselines, we adopt the default settings provided in their code to bring out the proper performance. For fair comprision, we use the uniform **Batch size** for all methods.

Table 2: Implementation details for the experiments on Permuted MNIST.

Method	**Optimizer**	**Momentum**	**Learning rate**	**Max epochs**	**Early stop**	**Batch size**
OWM	SGD	0.9	2.0	100	✗	10
GPM	SGD	0.9	0.01	5	✗	10
ORTHOG-SUBSPACE	SGD	0.9	0.1	15	✓	10
LcSP(low-coherence)	SGD	-	0.1	15	✓	10
LcSP(orthogonal)	SGD	-	0.1	15	✓	10

Table 3: Implementation details for the experiments on Permuted CIFAR10.

Method	**Optimizer**	**Momentum**	**Learning rate**	**Max epochs**	**Early stop**	**Batch size**
OWM	SGD	0.9	0.02	25	✗	64
GPM	SGD	0.9	0.01	200	✓	64
ORTHOG-SUBSPACE	SGD	0.9	0.1	80	✓	64
LcSP(low-coherence)	SGD	-	0.003	80	✓	64
LcSP(orthogonal)	SGD	-	0.003	80	✓	64

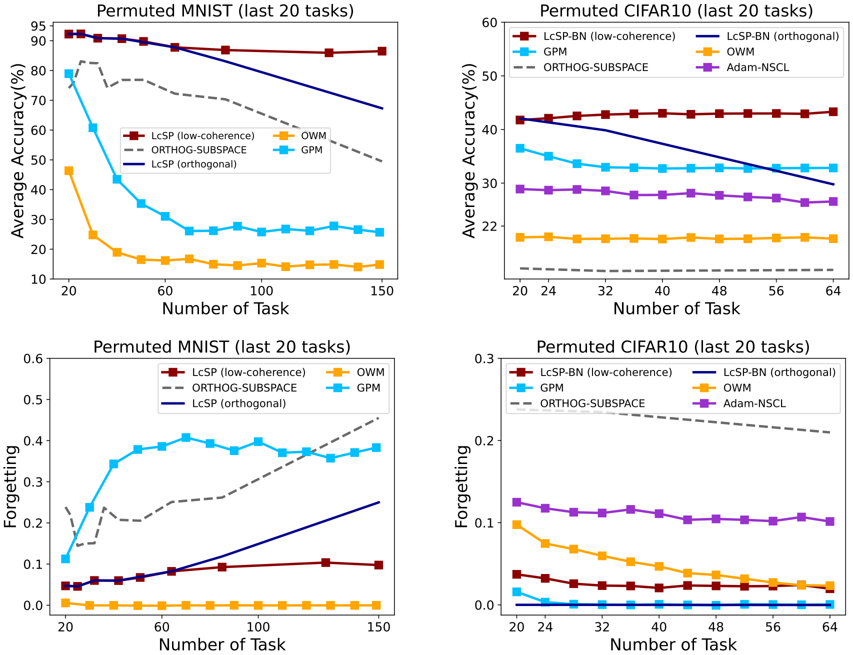


Figure 3: The average accuracy and forgetting of the last 20 tasks on Permuted MNIST (left) and Permuted CIFAR10 (right). The x-axis shows the number of tasks learned and y-axis represents the corresponding average accuracy and forgetting on the last 20 tasks.

A.2 ABLATION STUDIES AND ADDITIONAL RESULTS

Additional results on Permuted MNIST and Permuted CIFAR10 Readers may wonder whether our conclusion holds if we evaluate the average performance with more tasks (e.g., the average accuracy and forgetting on the last 20 tasks). As shown in Fig.3, LcSP still outperforms all baselines with a significant advantage. However, the phenomenon of learning capacity degradation in baselines becomes more imperceptible, e.g., the average accuracy of OWM on Permuted CIFAR10 is consistently low, rather than significantly decreasing. To further investigate the learning capacity degradation problem, we give the accuracy of baselines for the last task on Permuted MNIST and Permuted CIFAR10. As shown in Fig.7, all baselines, except LcSP, suffer from this problem with different degrees and show some decrease in accuracy compared to the initial ($66.16\% \sim 24.63\%$ on Permuted MNIST and $24.8\% \sim 3.48\%$ on Permuted CIFAR10). These results suggest that this

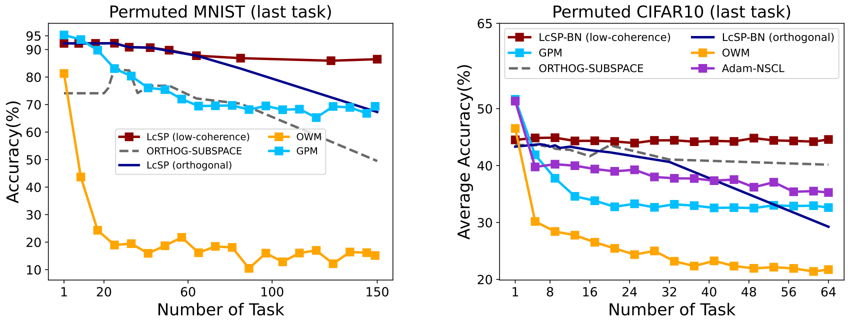
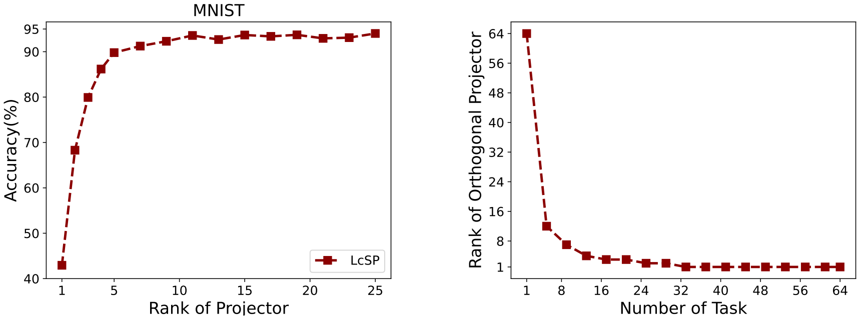


Figure 4: The accuracy of the last task on Permuted MNIST (left) and Permuted CIFAR10 (right), respectively.



(a)

(b)

Figure 5: Fig.(a) gives the ablation study for different rank of projectors on MNIST. Fig.(b) shows the average rank of all projectors when the number of tasks increasing. Here, the dimension m of features is 64.

problem is the critical factor that results in degrading the performance of GOP-based methods in the case of a large number of tasks.

Ablation studies and experiments for rank and scale constraints Further ablation studies and experiments are conducted to investigate the effects of the rank and scale constraints on the expressive power (plasticity) and stability of DNNs.

The result in Fig.5(a) suggests that projecting features or gradients into subspaces with low dimensions (e.g., lower than 5 in Fig.5(a)) will decrease the expressive power of DNN. GOP methods and LcSP rely on the projectors to project the features or gradients (or both) into a d-dimensional subspace, which can also be considered as a form of dimension reduction. The dimension reduction is motivated by a consensus in the high-dimensional data analysis community, i.e., the data can be summarized in a low-dimensional space embedded in a high-dimensional space, such as a nonlinear manifold Levina & Bickel (2004). The dimension of this low-dimensional space is also known as the intrinsic dimension D Carreira-Perpinán (1997). If the d is too small, e.g., $d \ll D$, important data features will be ”collapsed” onto the same dimension. From the perspective of training a DNN, if the gradients are projected into a subspace with a dimension lower than D, the DNN cannot activate sufficient parameters to learn the presentation of this task.

Due to the unknown number of tasks to be learned, the limited dimensions of the features, and the strict orthogonality constraint between projectors, the orthogonal projector cannot be constrained to a fixed-rank manifold. As shown in Fig.5(b), the rank of the orthogonal projector decreases as the number of tasks increases. Therefore, methods using orthogonal projectors usually ignore the intrinsic dimension of data (features or gradients) and finally lead to suffering from the learning capacity degradation problem. In contrast to GOP methods, LcSP relaxes the orthogonality constraint

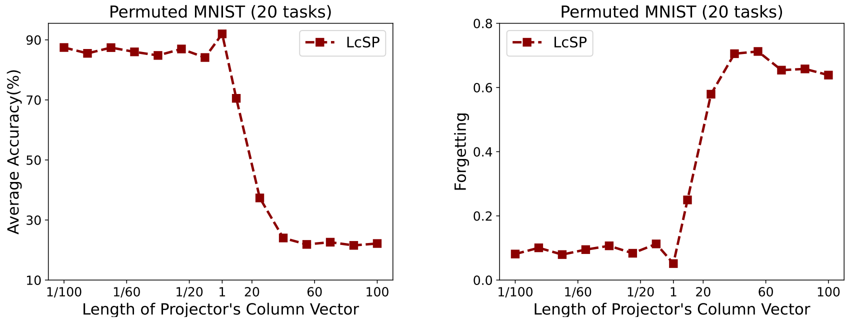


Figure 6: The average accuracy (left) and forgetting (right) of the last 20 tasks with different scales of columns on Permuted MNIST.

and meets the rank constraint by optimizing intra-task coherence on the Oblique manifold and thus does not suffer from this problem.

Finally, Fig.6 gives an ablation study for scale constraints on the projector's columns. In Fig.6, when the columns of the projector have unit length, the average accuracy reaches the highest. Result gets worse when the length of the projector's columns is too small or too large.

Table 4: Efficiency analysis.

	OWM	**GPM**	**Adam-NSCL**	**ORTHOG-SUBSPACE**	**LcSP**
FLOPs	26,676,256	23,202,880	558,538,752	556,708,864	528,965,376
Time(s)	47.614	5.174	47.685	87.623	25.496
Epochs	25	200	80	80	80
Mean_inference_time(ms)	1.249	1.152	10.089	3.957	5.893

Based on the RTX3060 12G, we tested the efficiency of LcSP and compared methods on Permuted CIFAR10 (which contains 64 tasks, and each image's size is $3 \times 32 \times 32$), using AlexNet for OWM and GPM, and resnet18 for the other methods. We provide 4 metrics to assess the efficiency of the method: floating point operations **FLOPs**, time spent per epoch during training **Time (s)**, the number of epochs needed to train to convergence **Epochs**, and the average time spent on inference **Mean_inference_time (ms)**. The detailed results are listed in Table.4.

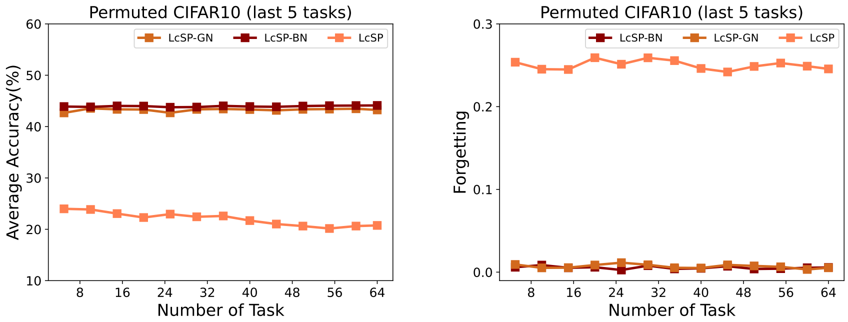


Figure 7: The average accuracy (left) and forgetting (right) of the last 5 tasks on Permuted CIFAR10. All methods use the low-coherence projector.

A.3 Method analysis

The mechanism of low-coherence learning and the difference with orthogonal learning Recall the Lemma 1 described in Section 4.3. **Lemma 1.** *Assume that f is fed the data of task $\mathcal{T}_t$ $(q < t)$, then f can effectively overcomes catastrophic forgetting if*

$$\boldsymbol{z}_{q,q}^{l} \approx \boldsymbol{z}_{q,t}^{l}, \quad \forall q \leq t$$

. Here, $z_{q,t}^{l}$ denotes the output feature of l-th layer when data from task $\mathcal{T}_q$ are fed into the DNN that has been trained on task $\mathcal{T}_t$. Let us consider the case where there are only two tasks, i.e., $q = t - 1$. By appling the LcSP's projection strategy, we have

$$\boldsymbol{z}_{q,t}^{l} = \boldsymbol{z}_{q,q}^{l} + \boldsymbol{x}_{q,t}^{l} \Delta \boldsymbol{W}_{t}^{l} \boldsymbol{P}_{q}^{l} = \boldsymbol{z}_{q,q}^{l} + \boldsymbol{x}_{q,t}^{l} \boldsymbol{g}_{t}^{l} \boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$$

. Here, $\boldsymbol{g}_t^l$, $\boldsymbol{x}_{q,t}^l$ and $\boldsymbol{P}_t^l$ denote the gradient, input data, and projection matrix (which is a symmetric matrix) of task $\mathcal{T}_t$ in l-th layer, respectively. The term $\boldsymbol{x}_{q,t}^{l} \boldsymbol{g}_{t}^{l} \boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$ can be thought of as the forgetting, which changes the output of DNN to previous data. Since we cannot change $\boldsymbol{x}_{q,t}^l$ and $\boldsymbol{g}_t^l$, the key to reduce $\boldsymbol{x}_{q,t}^{l} \boldsymbol{g}_{t}^{l} \boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$ is to minimize $\boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$. If $\boldsymbol{P}_t^l$ is orthogonal to $\boldsymbol{P}_q^l$, the forgetting is equal to 0. Let us consider reducing forgetting when $\boldsymbol{P}_t^l$ cannot be orthogonal to $\boldsymbol{P}_q^l$ (e.g., the number of tasks is large and the dimensionality of the space is limited). If we optimize $\boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$ directly, we may get $\boldsymbol{P}_t^l = 0$, or the column scale of $\boldsymbol{P}_t^l$ is too small. To find a useful $\boldsymbol{P}_t^l$, LcSP constrains the length of the column vector to be 1, i.e., to find $\boldsymbol{P}_t^l$ on the Oblique manifold ($\mathcal{OM}$). To reduce forgetting, we optimize the inter-task coherence $\mu(\boldsymbol{P}_t^l, \boldsymbol{P}_q^l)$ on $\mathcal{OM}$ so that the maximum entry of $\boldsymbol{P}_{t}^{l} \boldsymbol{P}_{q}^{l}$ is minimized.

In the worst case, we may obtain $\boldsymbol{P}_t^l$ consisting of only one column vector (i.e., $\boldsymbol{P}_t^l$ with rank 1). To avoid this situation, we optimize the intra-task coherence $\mu(\boldsymbol{P}_t^l)$ along with inter-task coherence, forcing the projector to satisfy the rank constraint in order to maintain the learning capacity of DNN (as shown in Fig. 5(a), the rank of projector affects the plasticity of DNN).

In general, low-coherence projection, which can be seen as a relaxed orthogonal constraint, aims to balance plasticity and stability better, motivated by our observation that orthogonal projections reduce the plasticity of the model and make the DNN unable to adapt to new environments.