A Theoretical Analysis of Catastrophic Forgetting through the NTK Overlap Matrix

Thang Doan [1] [2] **Mehdi Bennani** [3] **Bogdan Mazoure** [1] [2]

Guillaume Rabusseau [2] [4] **Pierre Alquier** [5]

Abstract

Continual learning (CL) is a setting in which an agent has to learn from an incoming stream of data during its entire lifetime. Although major advances have been made in the field, one recurring problem which remains unsolved is that of *Catastrophic Forgetting* (CF). While the issue has been extensively studied empirically, little attention has been paid from a theoretical angle. In this paper, we show that the impact of CF increases as two tasks increasingly align. We introduce a measure of task similarity called the *NTK overlap matrix* which is at the core of CF. We analyze common projected gradient algorithms and demonstrate how they mitigate forgetting. Then, we propose a variant of Orthogonal Gradient Descent (OGD) which leverages structure of the data through Principal Component Analysis (PCA). Experiments support our theoretical findings and show how our method can help reduce CF on classical CL datasets.

1 Introduction

Continual learning (CL) or lifelong learning [Thrun, 1995, Chen and Liu, 2018] has been one of the most important milestone on the path to building artificial general intelligence [Silver, 2011]. This setting refers to learning from an incoming stream of data, as well as leveraging previous knowledge for future tasks (through forward-backward transfer [Lopez-Paz and Ranzato, 2017]). While the topic has seen increasing interest in the past years [De Lange et al., 2019, Parisi et al., 2019] and a number of sohpisticated methods have been developed [Kirkpatrick et al., 2017, Lopez-Paz and Ranzato, 2017, Chaudhry et al., 2018, Aljundi et al., 2019b], a yet unsolved central challenge remains: Catastrophic Forgetting (CF) [Goodfellow et al., 2013, McCloskey and Cohen, 1989].

CF occurs when past solutions degrade while learning from new incoming tasks according to non-stationary distributions. Previous work either investigated this phenomenon empirically at different granularity levels (task level [Nguyen et al., 2019], neural network representations level [Ramasesh et al., 2020]), or proposed a quantitative metric [Farquhar and Gal, 2018, Kemker et al., 2017, Nguyen et al., 2020].

Despite the vast set of existing works on CF, there is still few theoretical works studying this major topic. Recently, Bennani et al. [2020] propose a framework to study Continual Learning in the NTK regime then derive generalization guarantees of CL under the Neural Tangent Kernel [Jacot et al., 2018, NTK] for Orthogonal Gradient Descent [Farajtabar et al., 2020, OGD]. Following on this work, we propose a theoretical analysis of Catastrophic Forgetting for a family of projection algorithms including OGD, GEM [Lopez-Paz and Ranzato, 2017]. Our contributions can be summarized as follows:

- We provide a general definition of Catastrophic Forgetting, and examine the special case of CF under the Neural Tangent Kernel (NTK) regime. Our definition leverages the similarity between the source and target task.
- We derive the expression of the forgetting error for a family of orthogonal projection methods based on the *NTK overlap matrix.* This matrix reduces

Proceedings of the 24$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

[1]McGill University [2]Mila [3]Aqemia [4]Université de Montréal [5]RIKEN AIP
corresponding author: thang.doan@mail.mcgill.ca

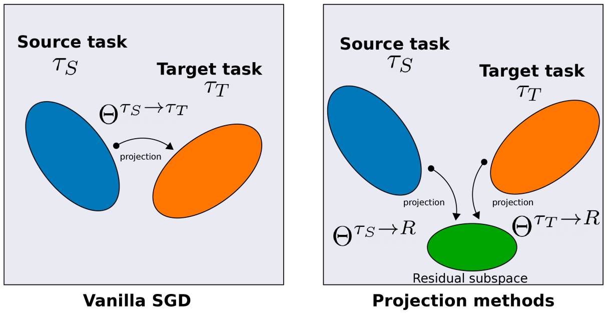

Figure 1: Unlike SGD, the projection methods reduce the forgetting by projecting the source and target tasks on a residual subspace.

to the angle between the source and target tasks and is a critical component responsible for the Catastrophic Forgetting.

- For these projection methods, we analyze their mechanisms to reduce Catastrophic Forgetting and how they differ from each other.
- We propose PCA-OGD, an extension of OGD which mitigates the CF issue by compressing the relevant information into a reduced number of principal components. We show that our method is advantageous whenever the dataset has a dependence pattern between tasks.

2 Related Works

Defying Catastrophic Forgetting [McCloskey and Cohen, 1989] has always been an important challenge for the Continual Learning community. Among different families of methods, we can cite the following: regularization-based methods [Kirkpatrick et al., 2017, Zenke et al., 2017], memory-based and projection methods [Chaudhry et al., 2018, Lopez-Paz and Ranzato, 2017, Farajtabar et al., 2020] or parameters isolations [Mallya and Lazebnik, 2018, Rosenfeld and Tsotsos, 2018]. In [Pan et al., 2020], the authors propose a method to identify memorable example from the past that must be stored. An exhaustive list can be found in [De Lange et al., 2019]. Although theses methods achieve more or less success in combating Catastrophic Forgetting, its underlying theory remains unclear.

Recently, a lot of efforts has been put toward dissecting CF [Toneva et al., 2018]. While Nguyen et al. [2019] empirically studied the impact of tasks similarity on the forgetting, [Ramasesh et al., 2020] analyzed this phenomenon at the neural network layers level. [Xie et al., 2020] studied the designed *artificial neural variability* and studied its impact on the forgetting. Mirzadeh et al. [2020] investigated how different training regimes affected the forgetting. Other streams of works investigated different evaluation protocol and measure of CF [Farquhar and Gal, 2018, Kemker et al., 2017]. That being said, there is still few theoretical work confirming empirical evidences of CF.

Yin et al. [2020] provide an analysis of CL from a loss landscape perspective through a second-order Taylor approximation. Recent advances towards understanding neural networks behavior [Jacot et al., 2018] has enabled a better understanding through Neural Tangent Kernel (NTK) [Du et al., 2018, Arora et al., 2019]. Latest work and important milestone towards better theoretical understand of CL is from Bennani et al. [2020]. The authors provide a theoretical framework for CL under the NTK regime for the infinite memory case. Our work relaxes this constraint to the *finite memory* case, which is more applicable in the empirical setting.

3 Preliminaries

3.1 Notations

We use $\|\cdot\|_2$ to denote the Euclidian norm of a vector or the spectral norm of a matrix. We use $\langle \cdot, \cdot \rangle$ for the Euclidean dot product, and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ the dot product in the Hilbert space $\mathcal{H}$. We index the task ID by τ. Learnable parameter are denoted ω and when indexing as ω_τ correspond to the training during task τ. Moreover $\star$ represents the variable at the end of a given task, i.e w_τ^* represents the learned parameters at the end of task τ.

We denote $\mathbb{N}$ the set of natural numbers, $\mathbb{R}$ the space of real numbers and $\mathbb{N}^\star$ for the set $\mathbb{N} \setminus \{0\}$.

3.2 Continual Learning

Let $\mathcal{X}$ be some feature space of interest (we take $\mathcal{X} = \mathbb{R}^p$), and let $\mathcal{Y}$ be the space of labels (we let $\mathcal{Y} = \mathbb{R}$, but $\mathcal{Y} = \Delta^K$ can be used for classification[1]). In CL, we receive a stream of supervised learning tasks $\mathcal{T}_\tau, \tau \in [T]$ where $\mathcal{T}_\tau = \{x_j^\tau, y_j^\tau\}_{j=1}^{n_\tau}$, with $T \in \mathbb{N}^*$. While $X^\tau \in \mathbb{R}^{n_\tau \times p}$ (p being the number of features) represents the dataset of task $\mathcal{T}_\tau$ and $x_j^\tau, j = 1, ..., n_\tau \in \mathcal{X}$ is a sample with its corresponding label $y_j^\tau \in \mathcal{Y}$. The goal is to learn a predictor $f_\omega : \mathcal{X} \times \mathcal{T} \to \mathcal{Y}$ with $\omega \in \mathbb{R}^p$ the parameters that will perform a prediction as accurate as possible. In the framework of CL, one cannot recover samples from previous tasks unless storing them in a memory buffer [Lopez-Paz and Ranzato, 2017, Parisi et al., 2019].

[1] Δ^K denotes the vertices of the probability simplex of dimension K

3.3 NTK framework for Continual Learning

Lee et al. [2019] recently proved that under the NTK regime neural networks evolve as a linear model:

$$f^*_\tau(x) = f^*_{\tau-1}(x) + \langle \nabla_\omega f_0(x), \omega^*_\tau - \omega^*_{\tau-1} \rangle$$

with $\omega^\star_\tau$ being the final weight after training on task τ. The latter formulation implies the feature maps $\phi(x) = \nabla_\omega f_0(x) \in \mathbb{R}^{1\times p}$ is constant over time. Under that framework, [Bennani et al., 2020] show that CL models can be expressed as a recursive kernel regression and prove generalization and performance guarantee of OGD under infinite memory setting. We build up on this theoretical framework to study CF and quantify how the tasks similarity imply forgetting through the lens of eigenvalues and singular values decomposition (PCA and SVD).

4 Analysis of Catastrophic Forgetting in finite memory

In this section, we propose a general definition of Catastrophic Forgetting (CF). Casted in the NTK framework, this definition allows to understand what are the main sources of CF. Namely, CF is likely to occur when two tasks align significantly. Finally, we investigate CF properties for the vanilla case (SGD) and projection based methods such as OGD and a variant of GEM. We then introduce a new algorithm called PCA-OGD, an extension of OGD which reduces CF .

4.1 A definition of Catastrophic Forgetting under the NTK regime

A natural quantity to characterize CF is the change in predictions for the same input between a source task τ_S and target task τ_T.

Definition 1 (Drift)**.**

Let τ_S (respectively τ_T) be the source task (respectively target task), $\mathcal{D}_{\tau_S}$ the source test set, the CF of task τ_S after training on all the subsequent tasks up to the target task τ_T is defined as:

$$\delta^{\tau_S \to \tau_T}(X^{\tau_S}) = \Big(f^\star_{\tau_T}(x) - f^\star_{\tau_S}(x) \Big)_{(x,y)\in\mathcal{D}_{\tau_S}} \qquad (1)$$

Note that $\delta^{\tau_S \to \tau_T}(X^{\tau_S})$ is a vector in $\mathbb{R}^{n_{\tau_S}}$ that contains the changes of predictions for any input x in the task τ_S. In the case of classification, we take the k-output of $f^\star_\tau$ such that $y_k = 1$. In order to quantify the overall forgetting on this task, we use the squared norm of this vector.

Definition 2 (Catastrophic Forgetting)**.**

Let τ_S (respectively τ_T) be the source task (respectively target task), $\mathcal{D}_{\tau_S}$ the source test set, the CF of task τ_S after training on all subsequent tasks up to task τ_T is defined as:

$$\begin{aligned}\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) &= \|\delta^{\tau_S \to \tau_T}(X^{\tau_S})\|_2^2 \\ &= \sum_{(x,y)\in\mathcal{D}_{\tau_S}} (f^\star_{\tau_T}(x) - f^\star_{\tau_S}(x))^2 \qquad (2)\end{aligned}$$

The above expression is very general but has an interesting linear form under the NTK regime and allows us to get insight on the behavior on the variation of the forgetting.

Lemma 1 (CF under NTK regime)**.**

Let $\{\omega^\star_\tau, \forall \tau \in [T]\}$ be the weight at the end of the training of task τ, the Catastrophic Forgetting of a source task τ_S with respect to a target task τ_T is given by:

$$\begin{aligned}\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) &= \|\delta^{\tau_S \to \tau_T}(X^{\tau_S})\|_2^2 \qquad (3)\\ &= \left\|\phi(X^{\tau_S})(\omega^*_{\tau_T} - \omega^*_{\tau_S})\right\|_2^2 \qquad (4)\end{aligned}$$

Proof. See Appendix Section 8.1. □

Lemma 1 expresses the forgetting as a linear relation between the kernel $\phi(X^{\tau_S})$ (which is assumed to be constant) and the variation of the weights from the source task τ_S until the target task τ_T.

Remark 1. *Note that, from Equation 4, two cases are possible when $\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) = 0$. The trivial case happens when $\forall \tau \in [T]$:*

$$\Big(f^\star_{\tau+1}(x) - f^\star_\tau(x) \Big)_{(x,y)\in\mathcal{D}_{\tau_S}} = 0$$

In this case, there is no drift at all. However, it is also possible that some tasks induce a drift on X^{τ_S} that is compensated by subsequent tasks. Indeed, for $\forall \tau \in [T]$:

$$\begin{aligned}0 &= \delta^{\tau_S \to \tau_T}(X^{\tau_S}) \\ &= \Big(f^\star_{\tau_T}(x) - f^\star_{\tau_S}(x) \Big)_{(x,y)\in\mathcal{D}_{\tau_S}} \\ &= \Big(f^\star_{\tau_T}(x) - f^\star_\tau(x) + f^\star_\tau(x) - f^\star_{\tau_S}(x) \Big)_{(x,y)\in\mathcal{D}_{\tau_S}}\end{aligned}$$

simply implies, for any $(x, y) \in \mathcal{D}_{\tau_S}$,

$$f^\star_{\tau_T}(x) - f^\star_\tau(x) = -(f^\star_\tau(x) - f^\star_{\tau_S}(x)).$$

This would be an example of no forgetting due to a forward/backward transfer in the sense of Lopez-Paz and Ranzato [2017].

Now that we have defined the central quantity of this study, we will gain deeper insights by investigating SGD which is the vanilla algorithm.

4.2 High correlations across tasks induce forgetting for vanilla SGD

In this section, we derive the Catastrophic Forgetting expression for SGD. This will be the starting point to derive CF for the projection based methods (OGD, GEM and PCA-OGD).

Theorem 1. *(Catastrophic Forgetting for SGD) Let $U_\tau \Sigma_\tau V_\tau^T$ be the SVD of $\phi(X^\tau)$ for each $\tau \in [T]$, and let $\lambda > 0$ the weight decay regularizer. The CF from task τ_S up until task τ_T is then given by:*

$$\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) = \left\| \sum_{k=\tau_S+1}^{\tau_T} U_{\tau_S} \Sigma_{\tau_S} O_{SGD}^{\tau_S \to k} M_k \tilde{y}_k \right\|_2^2 \quad (5)$$

where:

$$\begin{aligned} O_{SGD}^{\tau_S \to k} &= V_{\tau_S}^\top V_k \\ M_k &= \Sigma_k [\Sigma_k^2 + \lambda I_{n_k}]^{-1} U_k^\top \\ \tilde{y}_k &= y_k - f_{k-1}^\star(x^k) \end{aligned}$$

Proof. See Appendix Section 8.2. □

Theorem 1 describes the Catastrophic Forgetting for SGD on the task $\mathcal{T}_{\tau_S}$ after training on the subsequent tasks up to the task $\mathcal{T}_{\tau_T}$. The CF is expressed as a function of the overlap between the subspaces of the subsequent tasks and the reference task, through what we call the **NTK overlap matrices** $\{O_{SGD}^{\tau_S \to k}, k \in [\tau_S+1, \tau_T]\}$. High overlap between tasks increases the norm of the NTK overlap matrix which implies high forgetting.

More formally, the main elements of Catastrophic Forgetting are :

- Σ_{τ_S} encodes the importance of the principal components of the source task. Components with high magnitude contribute to forgetting since they imply high variation along thoses directions.
- $\{O_{SGD}^{\tau_S \to k}, k \in [\tau_S + 1, \tau_T]\}$ encodes the similarity of the principal components between the source task and a subsequent task k. High norm of this matrix means high overlap between tasks and leads to high risk of forgetting. This forgetting occurs because the previous knowledge along a given component may be erased by the new dataset.
- $\tilde{y}_k$ encodes the residual that remains to be learned by the current model. A null residual implies that the previous model predicts perfectly the new task, therefore there is no learning hence no forgetting.
- M_k is a rotation of the residuals weighted by the principal components space. The rotated residuals $M_k \tilde{y}_k$ can be interpreted as the residuals along each principal component.
- $\left\|\sum_{k=\tau_S+1}^{\tau_T} \cdot\right\|$ encodes that the forgetting can be canceled by other tasks by learning again forgotten knowledge.

We will see in what follows that the matrix $O_{SGD}^{\tau_S \to \tau_T}$ captures the alignment between the source task τ_S and the target task τ_T. More formally, the singular values of $O_{SGD}^{\tau_S \to \tau_T}$ are the cosines of the *principal angles* between the spaces spanned by the source data $\phi(X^{\tau_S})$ and the target data $\phi(X^{\tau_T})$ [Wedin, 1983].

Corollary 1 (Bounding CF with angle between source and target subspace)**.**

Let $\Theta^{\tau_S \to \tau_T}$ be the diagonal matrix of singular values of $O_{SGD}^{\tau_S \to \tau_T}$ (each diagonal element $\cos(\theta_{\tau_S,\tau_T})_i$ is the cosine of the i-th principal angle between $\phi(X^{\tau_S})$ and $\phi(X^{\tau_T})$). Let $\sigma_{\tau_S,1} \geq \sigma_{\tau_S,2} \geq ... \geq \sigma_{\tau_S,n_{\tau_S}}$ be the singular values of $\phi(X^{\tau_S})$ (i.e. the diagonal elements of Σ_{τ_S}).

The bound of the forgetting from a source task τ_S up until a target task τ_T is given by:

$$\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) \leq \sigma_{\tau_S,1}^2 \sum_{k=\tau_S+1}^{\tau_T} \left\|\Theta^{\tau_S \to k}\right\|_2^2 \left\|M_k \tilde{y}_k\right\|_2^2 \quad (6)$$

Proof. See Appendix Section 8.3. □

Corollary 1 bounds the CF by the sum of the cosines of the first principal angles between the source task τ_S and each subsequent task until the target task τ_T (represented by the diagonal matrix $\Theta^{\tau_S \to k}$) and a coefficient $\sigma_{\tau_S,1}^2$ from the source task τ_S.

- $\{\Theta^{\tau_S \to k}, k \in [\tau_S + 1, \tau_T]\}$ is the diagonal matrix where each element represents the cosine angle between subspaces τ_S and k: $\cos(\theta_{\tau_S,k})_i$. If the principal angle between two tasks is small (i.e. the two tasks are aligned), the cosine will be large which implies a high risk of forgetting.
- $\sigma_{\tau_S,1}$ is the variance of the data of task τ_S along its principal direction of variation. Intuitively, $\sigma_{\tau_S,1}$ measures the spread of the data for task τ_S.

In the end, a potential component responsible for CF in the Vanilla SGD case is the projection from the source task onto the target task. This phenomenon is best characterized by the eigenvalues of $O_{SGD}^{\tau_S \to \tau_T}$ which acts as a similarity measure between the tasks. One avenue to mitigate the CF can be to project orthogonally to the source task subspace which are the main insight from OGD [Farajtabar et al., 2020] and GEM [Lopez-Paz and Ranzato, 2017].

4.3 The effectiveness of the orthogonal projection against Catastrophic Forgetting

Now, we study the GEM and OGD algorithms, we identify these two algorithms as projection based algorithms. We extend the previous analysis to study the effectiveness of these algorithms against Catastrophic Forgetting.

Recap OGD [Farajtabar et al., 2020] stores the feature maps of arbitrary samples from each task, then projects the update gradient orthogonally to these feature maps. The idea is to preserve the subspace spanned by the previous samples ([Yu et al., 2020] proposed a similar variant for multi-task learning).

GEM [Lopez-Paz and Ranzato, 2017] computes the gradient of the train loss over each previous task, by storing samples from each task. While OGD performs an orthogonal projection to the **gradients** of the model, GEM projects orthogonally to the space spanned by the **losses gradients**. The idea is to update the model under the constraint that the train loss over the previous tasks does not increase.

GEM-NT : Decoupling Forward/Backward Transfer from Catastrophic Forgetting OGD has been extensively studied by Bennani et al. [2020]), therefore we perform the analysis for the GEM algorithm, then highlight the similarities with OGD. Also, in order to decouple CF from Forward/Backward Transfer, we study a variant of GEM with no transfer at all, which we call GEM No Transfer (GEM-NT).

Similarly to GEM, GEM-NT maintains an episodic memory containing d samples from each previous tasks seen so far. During each gradient step of task $\tau+1$, GEM samples from the memory d elements from each previous task then compute the average loss function gradient:

$$g_k = \frac{1}{d}\sum_{j=1}^{d} \nabla_\omega \mathcal{L}_\lambda^k(x_j^k), \quad \forall k = 1, .., \tau$$

If the proposed update during task $\tau+1$ can potentially degrades former solutions (i.e $\langle g_{\tau+1}, g_k\rangle < 0, \forall k \leq \tau$) then the proposed update is projected orthogonally to these gradients g_k, $\forall k \leq \tau$.

As opposed to GEM, which performs the orthogonal projection conditionally on the impact of the gradient update on the previous training losses, GEM-NT project orthogonally to g_k, $\forall k \leq \tau$ at each step **irrespectively** of the sign of the dot product. The algorithm pseudo-code can be found in Appendix Section 2.

The effectiveness of GEM-NT against CF Denote $G_\tau \in \mathbb{R}^{p\times\tau}$ the matrix where each columns represents $g_k, \forall k = 1, .., \tau$, the orthogonal projection matrix is then defined as $T_\tau = I_p - G_\tau G_\tau^\top = \overline{G}_\tau \overline{G}_\tau^\top$. This represents an orthogonal projection whatever the sign of the dot product $\langle g_{\tau+1}, g_k\rangle$ in order to decouple the forgetting from transfer.

We are now ready to provide the CF of GEM-NT.

Corollary 2 (CF for GEM-NT)**.**

Using the previous notations. The CF from task τ_S up until task τ_T for GEM-NT given by:

$$\Delta^{\tau_S\to\tau_T}(X^{\tau_S}) = \left\| \sum_{k=\tau_S+1}^{\tau_T} U_{\tau_S}\Sigma_{\tau_S}\mathbf{O}^{\tau_\mathbf{S}\to\mathbf{k}}_{GEM\text{-}NT} M_k \tilde{y}_k \right\|_2^2 \tag{7}$$

where:

$$\begin{aligned} O^{\tau_S\to k}_{GEM\text{-}NT} &= V_{\tau_S}^\top \overline{G}_{k-1}\overline{G}_{k-1}^\top V_k \\ M_k &= \Sigma_k U_k^\top[\overline{\phi}(X^k)\overline{\phi}(X^k)^\top + \lambda I_{n_k}]^{-1} \\ \overline{\phi}(X^k) &= \phi(X^k)T_{k-1} \end{aligned}$$

(Differences with the vanilla case SGD are highlighted in color)

Proof. See Appendix Section 8.4. □

The difference for GEM-NT lies in the **double** projection of the source and target task onto the subspace $\overline{G}_\tau$ which contain elements orthogonal to $g_k, \forall k = 1, .., \tau - 1$.

Similarly to Corollary 1, we can bound each projection matrix ($V_{\tau_S}^\top \overline{G}_{k-1}$ and $\overline{G}_{k-1}^\top V_k$,$\forall k \in [\tau_S+1, \tau_T]$) by their respective matrices of singular values ($\Theta^{\tau_S\to G_{k-1}}$ and $\Theta^{k\to G_{k-1}}$, $\forall k \in [\tau_S+1, \tau_T]$). This leads us to the following upper-bound for the CF of GEM-NT:

$$\Delta^{\tau_S\to\tau_T}(X^{\tau_S}) \leq \tag{8}$$
$$\sigma^2_{\tau_S,1}\sum_{k=\tau_S+1}^{\tau_T} \left\|\Theta^{\tau_S\to\overline{G}_{k-1}}\right\|_2^2 \left\|\Theta^{k\to\overline{G}_{k-1}}\right\|_2^2 \|M_k\tilde{y}_k\|_2^2$$

Connection of GEM-NT to OGD For the analysis purpose, let's suppose that the memory per task is 1, $\lambda = 0$, $\forall\tau \in [T]$ and assume a mean square loss error function. In that case:

$$g_k = \begin{cases} \nabla_\omega f_k(x)(f_k(x) - y_k) & \text{(GEM-NT)} \\ \nabla_\omega f_k(x) & \text{(OGD)} \end{cases} \tag{9}$$

- unlike OGD, GEM-NT weights the orthogonal projection with the residuals $(f_\tau(x^k) - y_k) = (\tilde{y}_k + \delta^{k\to\tau}(x^k))$ which represents the difference between the new prediction (due to the drift) for x^k under model τ and the target y_k.

- Previous tasks that are well learned (small residuals) will contribute less to the orthogonal projection to the detriment of tasks with large residuals (badly learned then). This seems counter-intuitive because by doing so, the projection will not be orthogonal to well learned tasks (in the edge case of zero residuals) then unlearning can happen for those tasks.

While OGD and GEM-NT are more robust to CF than SGD through the orthogonal projection, they do not leverage explicitely the structure in the data [Farquhar and Gal, 2018]. We can then compress this information through dimension reduction algorithms such as SVD in order to both maximise the information contained in the memory as well as mitigating the CF.

4.4 PCA-OGD: leveraging structure by projecting orthogonally to the top d principal directions

Unlike OGD that stores randomly d samples from each task $k = 1, .., \tau$ of $\{\nabla_\omega f(x_j^k)\}_{j=1}^d$, at the end of each task τ, PCA-OGD samples randomly $s > d$ elements from X^τ then stores the top d eigenvectors of $\{\nabla_\omega f(X^\tau)\}$ denoted as $v_{\tau,i}$, $i = 1, .., d$. These are the directions that capture the most variance of the data. If we denote by $P_{\tau,:d}$ the matrix where each columns represents $v_{k,i}$, $k = 1, .., \tau$, $i = 1, .., d$ then the orthogonal matrix projection can be written as:

$$T_{\tau,:d} = I_p - P_{\tau,:d} P_{\tau,:d}^\top = R_{\tau,d:} R_{\tau,d:}^\top \tag{10}$$

where the columns of $R_{\tau,d:}$ form an orthonormal basis of the orthogonal complement of the span of $P_{\tau,:d}$. For the terminology, $P_{\tau,:d} \in \mathbb{R}^{p \times (\tau \cdot d)}$ (respectively $R_{\tau,d:} \in \mathbb{R}^{p \times p - (\tau \cdot d)}$) represents the **top subspace** (respectively the **residuals subspace**) of order d for task 1 until τ. A pseudo-code of PCA-OGD is given in Alg. 1 (the computational overhead can be found in the Appendix). We are now ready to provide the CF of PCA-OGD.

Corollary 3 (Forgetting for PCA-OGD)**.**

For each $\tau \in [T]$, let $\tilde{\phi}(X^\tau) = \phi(X^\tau) T_{\tau-1,:d}$ and let $U_\tau \Sigma_\tau V_\tau^T$ be the SVD of $\phi(X^\tau)$. The CF for PCA-OGD is given by:

$$\Delta^{\tau_S \to \tau_T}(X^{\tau_S}) = \left\| \sum_{k=\tau_S+1}^{\tau_T} \mathbf{U}_{\tau_\mathbf{S}} \mathbf{\Sigma}_{\tau_\mathbf{S}} \mathbf{O}_{\mathbf{PCA}}^{\tau_\mathbf{S} \to \mathbf{k}} M_k \tilde{y}_k \right\|_2^2 \tag{11}$$

where:

$$\begin{aligned} O_{PCA}^{\tau \to k} &= V_{\tau_S}^\top R_{k-1,d:} R_{k-1,d:}^\top V_k \\ M_k &= \Sigma_k U_k^\top [\tilde{\phi}(X^k) \tilde{\phi}(X^k)^\top + \lambda I_{n_k}]^{-1} \\ \tilde{\phi}(X^k) &= \phi(X^k) T_{k-1,:d} \end{aligned}$$

Algorithm 1: PCA-OGD (Differences with OGD in red)

```
Input    : A task sequence $\mathcal{T}_1, \mathcal{T}_2, \ldots$, learning rate
           $\eta$, PCA samples $s$, components to keep $d$
1. Initialize $S_J \leftarrow \{\}$ ; $\omega \leftarrow \omega_0$
2. for Task ID $\tau = 1, 2, 3, \ldots$ do
     repeat
         $\mathbf{g} \leftarrow$ Stochastic Batch Gradient for $\mathcal{T}_\tau$ at $\omega$;
         // Orthogonal updates
         $\tilde{\mathbf{g}} = \mathbf{g} - \sum_{\mathbf{v} \in \mathcal{S}_J} \text{proj}_\mathbf{v}(\mathbf{g})$;
         $\omega \leftarrow \omega - \eta \tilde{\mathbf{g}}$
     until convergence;
     // Gram-Schmidt orthogonalization
     for $(\mathbf{x}, y) \in \mathcal{D}_\tau$ and $k \in [1, c]$ s.t. $y_k = 1$ do
         $\mathbf{u} \leftarrow \nabla f_\tau(\mathbf{x}; \omega) - \sum_{\mathbf{v} \in \mathcal{S}_J} \text{proj}_\mathbf{v}(\nabla_\omega f_\tau(\mathbf{x}; \omega))$
         $\mathcal{S}_J \leftarrow \mathcal{S}_J \bigcup \{\mathbf{u}\}$
     end for
     // PCA
     Sample $s$ elements from $\mathcal{T}_\tau$
     top $d$ eigenvectors $\leftarrow PCA(\{\nabla_\omega f_\tau(x_j^\tau)\}_{j=1}^s)$
     $\mathcal{S}_J \leftarrow \mathcal{S}_J \bigcup$ { top $d$ eigenvectors }
   end for
```

Proof. See Appendix Section 8.5. □

Corollary 3 underlines the difference with GEM-NT as this time the double projection are on the residuals subspace $R_{k-1,:d}$ containing the orthogoanl vector to the features map $\nabla_\omega f(x)$ instead of the loss function gradient.

Remark 2.

- *PCA is helpful in datasets where the eigenvalues are decreasing exponentially since keeping a small number of components can leverage a large information and explain a great part of the variance. Projecting orthogonally to these main components will lead to small forgetting if $\sigma_{\tau,d+1}$ is small.*

- *On the other hand, unfavourable situations where data are spread uniformly along all directions (i.e, eigenvalues are uniformly equals) will requires to keep all components and a larger memory. As an example, we build a worst-case scenario in Appendix Section 8.12 where OGD is performing better than PCA-OGD.*

Similarly as the previous case, we can bound the double projection on $R_{k-1,:d}$ with the corresponding diagonal matrix $\Theta^{\tau_S \to R_{k,:d}}$. Additionally, the CF is bounded by $\sigma_{\tau_S, d+1}$ which is due to the orthogonal projection to the first d principal directions. The upper bound of the CF is given by:

$$\Delta^{\tau_S \to \tau_T}(X^\tau) \leq \tag{12}$$

$$\boldsymbol{\sigma^2_{\tau_S, d+1}} \sum_{k=\tau_S+1}^{\tau_T} \left\| \boldsymbol{\Theta^{\tau_S \to R_{k-1,:d}}} \right\|_{\mathbf{2}}^{\mathbf{2}} \left\| \boldsymbol{\Theta^{k \to R_{k-1,:d}}} \right\|_{\mathbf{2}}^{\mathbf{2}} \left\| M_k \tilde{y}^k \right\|_2^2$$

Note that in contrast with Eq. (8), the first term in the upper bound is the $(d+1)$-th singular value of $\phi(X^{\tau_S})$, which is due to the PCA step of PCA-OGD. A summary of the forgetting properties of the described methods can be found in Table 2 in Appendix.

5 Experiments

In this section, we study the impact of the NTK overlap matrix on the forgetting by validating Corollary 1. We then illustrates how PCA-OGD efficiently captures and compresses the information in datasets (Corollary 3. Finally, we benchmark PCA-OGD on standard CL baselines.

5.1 Low eigenvalue of the NTK overlap matrix induces smaller drop in performance

Objective : As presented in Corollary 1, we want to assess the effect of the eigenvalues of the NTK overlap matrix on the forgetting.

Experiments : We measure the drop in accuracy for task 1 until task 15 on Rotated MNIST with respect to the maximum eigenvalue of the NTK overlap matrix $O^{1 \to 15}$.

Results : Figure 2 shows the drop in accuracy between task 1 and task 15 for Rotated MNIST versus the largest eigenvalue of $O^{1 \to 15}$. As expected low eigenvalues leads to a smaller drop in accuracy and thus less forgetting. PCA-OGD improves upon OGD, having from 7% to 10% less drop in performance.

5.2 PCA-OGD reduces forgetting by efficiently leveraging structure in the data

Objective : We show how capturing the top d principal directions helps reducing Catastrophic Forgetting (Corollary 3).

Experiments : We compare the spectrum of the NTK overlap matrix for different methods: SGD, GEM-NT, OGD and PCA-OGD, for different memory sizes.

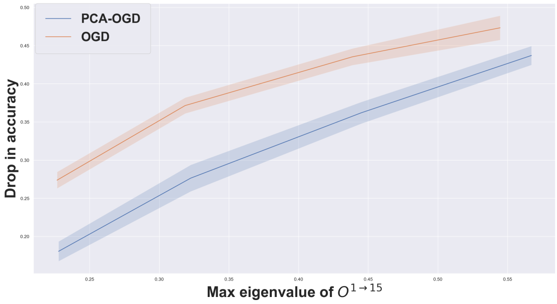

Figure 2: Drop in performance with respect to the maximum eigenvalue for Rotated MNIST (averaged over 5 seeds ±1 std).

Results: We visualize the effect of the memory size on the forgetting through the eigenvalues of the NTK overlap matrix $O^{\tau_S \to \tau_T}$. To unclutter the plot, Figure 3 only shows the results for memory sizes of 25 and 200. Because PCA-OGD compresses the information in a few number of components, it has lower eigenvalues than both OGD and GEM-NT and the gap gets higher when increasing the memory size to 200. Table 9 in the Appendix confirms those findings by seeing that with 200 components one can already explain 90.71% of the variance.

Finally, the eigenvalues of SGD are higher than those of projection methods since it does not perform any projection of the source or target task.

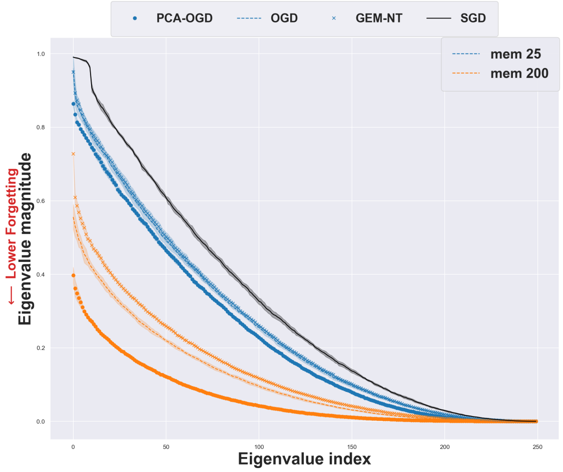

Figure 3: Comparison of the eigenvalues of $O^{1 \to 2}$ on **Rotated MNIST** with increasing memory size. Lower values imply less forgetting (averaged over 5 seeds ±1 std).

Finally, the final accuracies on Rotated and Permuted MNIST are reported in Figure 4 for the first seven tasks. In Rotated MNIST, we can see that PCA-OGD is twice more memory efficient than OGD: with a memory size of 100 PCA-OGD has comparable results to OGD with a memory size 200. Interestingly, while the marginal increase for PCA-OGD is roughly constant going from memory size 25 to 50 or 50 to 100, OGD incurs a high increase from memory size 100 to 200 while below that threshold the improvement is relatively small.

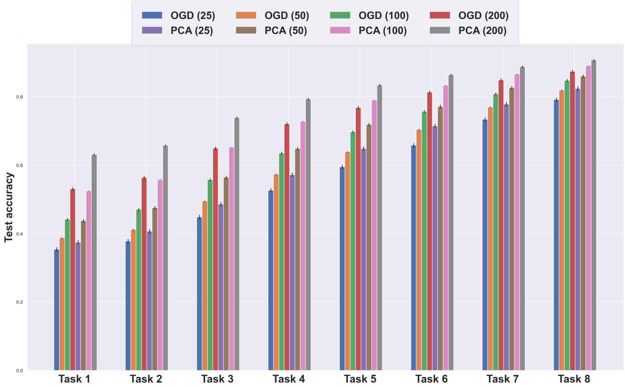

Figure 4: Final accuracy on **Rotated MNIST** for different memory size (averaged over 5 seeds ±1 std). OGD needs twice as much memory as PCA-OGD in order to achieve the same performance (i.e compare OGD (200) and PCA (100).

We ran OGD and PCA-OGD on a counter-example dataset (Permuted MNIST), where there is no structure within the dataset (see Appendix 8.7). In this case, PCA-OGD is less efficient since it needs to keep more principal components than in a structured dataset setting.

5.3 General performance of PCA-OGD against baselines

Objective and Experiments : We compare PCA-OGD against other baseline methods: SGD, A-GEM [Chaudhry et al., 2018] and OGD [Farajtabar et al., 2020]. Additionally to the final accuracies, we report the **Average Accuracy** A_T and **Forgetting Measure** F_T [Lopez-Paz and Ranzato, 2017, Chaudhry et al., 2018]. We run AGEM instead of GEM-NT which is faster with comparable results [Chaudhry et al., 2018] (since GEM-NT is solving a quadratic programming optimization at each iteration step). Definition of these metrics and full details of the experimental setup can be found in Appendix 8.11.

Results : The results are summarized in Table 1 (additional results are presented in Appendix 8.11). Overall, PCA-OGD obtains comparable results to A-GEM. A-GEM has the advantage of accounting for the NTK changes by updating it while PCA-OGD and OGD are storing the gradients from previous iteration. The later therefore project updates orthogonally to outdated gradients. This issue has also been mentioned in [Bennani et al., 2020]. Note the good performance of PCA-OGD in Split CIFAR where the dataset size is $2,500$ (making the NTK assumption more realistic) and similar patterns are seen across tasks (CIFAR100 dataset is divided into 20 superclasses within which we can count 5 subfamilies hence having a pattern across tasks. To examine this hypothesis, we plot the NTK changes for different datasets in Appendix 8.10. We can indeed see that the NTK does not vary anymore after 1 task for Split CIFAR while it increases linearly for MNIST datasets which confirms our hypothesis.

	SGD	EWC	A-GEM	OGD	PCA-OGD
			Permuted MNIST		
A_T	76.81 ± 1.36	79.71 ± 0.52	**83.4 ± 0.43**	80.95 ± 0.5	81.44 ± 0.62
F_T	14.88 ± 1.64	**3.81 ± 0.47**	7.29 ± 0.45	9.72 ± 0.51	9.11 ± 0.65
			Rotated MNIST		
A_T	66.07 ± 0.47	76.2 ± 0.62	**83.52 ± 0.22**	77.42 ± 0.35	82.05 ± 0.58
F_T	29.57 ± 0.56	13.44 ± 0.82	**9.86 ± 0.28**	16.52 ± 0.46	11.67 ± 0.65
			Split MNIST		
A_T	95.1 ± 1.08	95.06 ± 1.15	94.25 ± 1.62	**96.05 ± 0.34**	95.96 ± 0.29
F_T	2.02 ± 1.48	2.08 ± 1.56	2.82 ± 1.72	0.37 ± 0.21	**0.28 ± 0.15**
			Split CIFAR		
A_T	56.11 ± 1.65	65.45 ± 0.97	47.55 ± 2.4	69.77 ± 0.72	**72.7 ± 0.97**
F_T	22.69 ± 2.18	6.56 ± 1.49	31.36 ± 2.58	8.27 ± 0.31	**5.39 ± 0.85**

Table 1: Average Accuracy and Forgetting for all baselines considered across the datasets (5 seeds).

6 Conclusion

We present a theoretical analysis of CF in the NTK regime, for SGD and the projection based algorithms OGD, GEM-NT and PCA-OGD. We quantify the impact of the tasks similarity on CF through the NTK overlap matrix. Experiments support our findings that the overlap matrix is crucial in reducing CF and our proposed method PCA-OGD efficiently mitigates CF. However, our analysis relies on the core assumption of overparameterisation, an important next step is to account for the change of NTK over time. We hope this analysis opens new directions to study the properties of Catastrophic Forgetting for other Continual Learning algorithms.

7 Acknowledgments

The authors would like to thank Joelle Pineau for useful discussions and feedbacks. Finally, we thank Compute Canada for providing computational ressources used in this project.

References

Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems*, pages 11849–11860, 2019a.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pages 11816–11825, 2019b.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pages 8141–8150, 2019.

Mehdi Abbana Bennani, Thang Doan, and Masashi Sugiyama. Generalisation guarantees for continual learning with orthogonal gradient descent, 2020.

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

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*, 2019.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes overparameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

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, 2020.

Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

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.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. *arXiv preprint arXiv:1708.02072*, 2017.

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 et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

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.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *arXiv preprint arXiv:2006.06958*, 2020.

Cuong V Nguyen, Alessandro Achille, Michael Lam, Tal Hassner, Vijay Mahadevan, and Stefano Soatto. Toward understanding catastrophic forgetting in continual learning. *arXiv preprint arXiv:1908.01091*, 2019.

Giang Nguyen, Shuan Chen, Tae Joon Jun, and Daeyoung Kim. Explaining how deep neural networks forget by deep visualization, 2020.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past. *arXiv preprint arXiv:2004.14070*, 2020.

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.

Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *arXiv preprint arXiv:2007.07400*, 2020.

Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

Daniel L Silver. Machine lifelong learning: challenges and benefits for artificial general intelligence. In *International conference on artificial general intelligence*, pages 370–375. Springer, 2011.

Sebastian Thrun. A lifelong learning perspective for mobile robot control. In *Intelligent robots and systems*, pages 201–214. Elsevier, 1995.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Per Åke Wedin. On angles between subspaces of a finite dimensional inner product space. In *Matrix Pencils*, pages 263–285. Springer, 1983.

Zeke Xie, Fengxiang He, Shaopeng Fu, Issei Sato, Dacheng Tao, and Masashi Sugiyama. Artificial neural variability for deep learning: On overfitting, noise memorization, and catastrophic forgetting. *arXiv preprint arXiv:2011.06220*, 2020.

Dong Yin, Mehrdad Farajtabar, and Ang Li. Sola: Continual learning with second-order loss approximation. *arXiv preprint arXiv:2006.10974*, 2020.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

Peizhen Zhu and Andrew V Knyazev. Angles between subspaces and their tangents. *Journal of Numerical Mathematics*, 21(4):325–340, 2013.