EVEN YOUR TEACHER NEEDS GUIDANCE: GROUND-TRUTH TARGETS DAMPEN REGULARIZATION IMPOSED BY SELF-DISTILLATION

Kenneth Borup
Department of Mathematics
Aarhus University
kennethborup@math.au.dk

Lars N. Andersen
Department of Mathematics
Aarhus University
larsa@math.au.dk

ABSTRACT

Knowledge distillation is classically a procedure where a neural network is trained on the output of another network along with the original targets in order to transfer knowledge between the architectures. The special case of self-distillation, where the network architectures are identical, has been observed to improve generalization accuracy. In this paper, we consider an iterative variant of self-distillation in a kernel regression setting, in which successive steps incorporate both model outputs and the ground-truth targets. This allows us to provide the first theoretical results on the importance of using the weighted ground-truth targets in self-distillation. Our focus is on fitting nonlinear functions to training data with a weighted mean square error objective function suitable for distillation, subject to ℓ_2 regularization of the model parameters. We show that any such function obtained with self-distillation can be calculated directly as a function of the initial fit, and that infinite distillation steps yields the same optimization problem as the original with amplified regularization. Finally, we examine empirically, both in a regression setting and with ResNet networks, how the choice of weighting parameter influences the generalization performance after self-distillation[1].

1 Introduction

Knowledge distillation, most commonly known from Hinton et al. (2015), is a procedure to transfer *knowledge* from one neural network (teacher) to another neural network (student)[2]. Often the student has fewer parameters than the teacher, and the procedure can be seen as a model compression technique. Originally, the distillation procedure achieves the knowledge transfer by training the student network using the original training targets, denoted as ground-truth targets, as well as a softened distribution of logits from the (already trained and fixed) teacher network[3]. Since the popularization of knowledge distillation by Hinton et al. (2015), the idea of knowledge distillation has been extended to a variety of settings[4]. This paper will focus on the special case where the teacher and student are of identical architecture, called self-distillation, and where the aim is to improve predictive performance, rather than compressing the model.

The idea of self-distillation is to use outputs from a trained model together with the original targets as new targets for retraining the same model from scratch. We refer to this as one step of self-distillation, and one can iterate this procedure for multiple distillation steps (see Figure 1). Empirically, it has been shown that this procedure often generalizes better than the model trained merely on the original targets, and achieves higher predictive performance

[1] Code to reproduce the experiments is publicly available at github.com/Kennethborup/self_distillation
[2] Often knowledge distillation is also referred to under the name *Teacher-Student learning*.
[3] We will refer to the weighted outputs of the penultimate layer, i.e. pre-activation of the last layer, as logits.
[4] See Section 2 for a brief overview, or see Wang and Yoon (2020) for a more exhaustive survey

on validation data, despite no additional information is provided during training (Furlanello et al., 2018; Ahn et al., 2019; Yang et al., 2018).

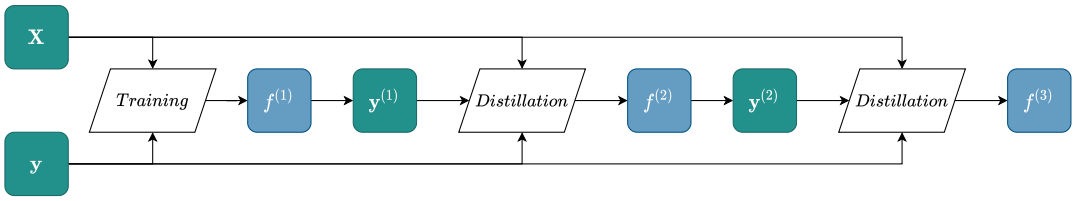


Figure 1: Illustration of self-distillation for two steps after the initial training. See Section 3 for details.

Modern deep neural networks are often trained in the over-parameterized regime, where the amount of trainable parameters highly exceed the amount of training samples. Under simple first-order methods such as gradient descent, such large networks can fit any target, but in order to generalize well, such overfitting is usually undesirable (Zhang et al., 2017; Nakkiran et al., 2020). Thus, some type of regularization is typically imposed during training, in order to avoid overfitting. A common choice is to add an ℓ_2-regularization[5] term to our objective function, which has been shown to perform comparably to early-stopping gradient descent training (Yao et al., 2007). However, in the theoretical study of the over-parameterized regime, regularization is often overlooked. Though, recent results has shown a connection between wide neural networks and kernel ridge regression through the Neural Tangent Kernel (NTK) (Lee et al., 2019, 2020; Hu et al., 2019). We elaborate on this connection, which motivates our problem setup, in Section 6.

2 Related Work

The idea of knowledge distillation dates back to Bucila et al. (2006), and was later brought to the deep learning setting by Ba and Caruana (2014) and more recently popularized by Hinton et al. (2015) in the context of compressing neural networks. Since the original formulation, various extensions has been proposed. Some approaches focus on matching the teacher and student models on other statistics than the distribution of the logits, such as intermediate representations (Romero et al., 2015), spacial attention maps (Zagoruyko and Komodakis, 2019), Jacobians (Srinivas and Fleuret, 2018), Gram matrices (Yim et al., 2017), or relational information between teacher outputs (Park et al., 2019). Other extensions focus on developing the transfer procedure, such as self-distillation (Furlanello et al., 2018), data-free distillation (Lopes et al., 2017; Nayak et al., 2019; Micaelli and Storkey, 2019; Chen et al., 2019; Fang et al., 2019), data distillation (Radosavovic et al., 2018), residual knowledge distillation (Gao et al., 2020), online distillation (Anil et al., 2018) or contrastive distillation (Ahn et al., 2019; Tian et al., 2020a).

The practical benefits of knowledge distillation has been proven countless of times in a variety of settings, but the theoretical justification for knowledge distillation is still highly absent. Hinton et al. (2015) conjecture that the success of knowledge distillation should be attributed to the transfer of *dark knowledge* (e.g. inter-class relationships revealed in the soft labels). Müller et al. (2019); Tang et al. (2020) support this conjecture, and argue that knowledge distillation is similar to performing adaptive label smoothing weighted by the teacher's confidence in the predictions. Dong et al. (2019) show the importance of early stopping when training over-parameterized neural networks for distillation purposes by arguing that neural networks tend to fit informative and simple patterns faster than noisy signals, and knowledge distillation utilizes these simple patterns for knowledge transfer. Abnar et al. (2020) empirically investigates how knowledge distillation can transfer inductive biases between student and teacher models, and Gotmare et al. (2019) empirically shows how the dark knowledge shared by the teacher mainly is disbursed to some of the deepest layers of the teacher.

To the best of our knowledge, few papers investigate knowledge distillation from a rigorous theoretical point of view, and those that do, do so with strong assumptions on the setting. Phuong and Lampert (2019) ignore the ground-truth targets during distillation and furthermore assume linear models. Mobahi et al. (2020) investigate self-distillation in a Hilbert space setting with kernel ridge regression models where the teacher is trained on the ground-truth targets, and

[5]With slight differences, ℓ_2 regularization is often referred to as weight decay and ridge regularization in deep learning and statistical learning literature, respectively.

the student (and subsequent iterations) is only trained on the predictions from the teacher without access to the ground-truth targets. They show that self-distillation progressively limits the number of basis functions used to represent the solutions, thus eventually causing the solutions to underfit. In this paper, we build on the theoretical results of Mobahi et al. (2020), but we include the weighted ground-truth targets in the self-distillation procedure and show how this drastically affect the behavior and effect of self-distillation[6].

Summary of Our Contributions We summarize our contributions as the following; we show that

- the solution at any distillation step can easily be calculated as a function of the initial fit, and infinite steps of self-distillation corresponds to solving the usual kernel ridge regression problem with a specific amplified regularization parameter for non-zero weights,
- self-distillation (strictly) amplifies the regularization at each distillation step,
- the ground-truth targets dampens the sparsification of the self-distilled solutions, ensuring non-zero solutions for any number of distillation steps,
- including the ground-truth targets in the distillation procedure empirically affect the generalization performance after self-distillation.

The proofs of all our main results can be found in Supplementary Material A, and code to reproduce both our illustrative example in Section 4.4 and results in Section 5 can be found at github.com/Kennethborup/self_distillation.

Notation Vectors and matrices are denoted by bold-faced letters; vectors are column vectors by default, and for a vector $\mathbf{a}$ let $[\mathbf{a}]_i$ be the i-th entry and for a matrix $\mathbf{A}$ let $[\mathbf{A}]_{i,j}$ be the (i,j)-th entry. Let $\mathbf{I}_n$ denote the identity matrix of dimension n, and $[k] = \{1, 2, \dots, k\}$, and for a vector $\mathbf{x}$ let $\|\mathbf{x}\|_2$ denote the ℓ_2 norm of $\mathbf{x}$. Finally, for some function $h : \mathbb{R}^n \to \mathbb{R}^d$ and $\mathbf{X} \in \mathbb{R}^{m \times n}$, we denote by $h(\mathbf{X})$ the $\mathbb{R}^{m \times d}$ matrix of outcomes, where the i'th row of $h(\mathbf{X})$ is the function applied to the i'th row of $\mathbf{X}$, i.e. $[h(\mathbf{X})]_{i,\cdot} = h(\mathbf{x}_i)$.

3 Problem Setup

Consider the training dataset $\mathcal{D} \subseteq \mathbb{R}^d \times \mathbb{R}$, and let $\mathcal{X} = \{\mathbf{x} \mid (\mathbf{x}, y) \in \mathcal{D}\}$ and $\mathcal{Y} = \{y \mid (\mathbf{x}, y) \in \mathcal{D}\}$ denote the inputs and targets, respectively. Let $\mathbf{X} \overset{\text{def}}{=} [\mathbf{x}_i]_{i \in [n]} \in \mathbb{R}^{n \times d}$ be the matrix of inputs and $\mathbf{y} \overset{\text{def}}{=} [y_i]_{i \in [n]}$ the vector of targets. Given a feature map $\varphi : \mathbb{R}^d \to \mathcal{V}$, where $\mathcal{V}$ has dimension D, we denote by $\mathbf{K} = \kappa(\mathbf{X}, \mathbf{X}) = [\kappa(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^n$, where $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$, the symmetric kernel (Gram) matrix associated with the feature map φ[7]. In order to avoid overfitting our training data, we will impose a regularization term on our weights, and thus investigate the kernel ridge regression functions $f \in \mathcal{F}$ mapping $f : \mathcal{X} \to \mathcal{Y}$, to construct a solution which best approximates the true underlying data generating map and generalize well to new unseen data from this underlying map.

3.1 Self-Distillation of Kernel Ridge Regressions

We consider self-distillation in the kernel ridge regression setup; i.e. consider the (self-distillation) objective function

$$\mathcal{L}^{\text{distill}}(f(\mathbf{X}, \boldsymbol{\beta}), \mathbf{y}_1, \mathbf{y}_2) = \frac{\alpha}{2} \|f(\mathbf{X}, \boldsymbol{\beta}) - \mathbf{y}_1\|_2^2 + \frac{1-\alpha}{2} \|f(\mathbf{X}, \boldsymbol{\beta}) - \mathbf{y}_2\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2^2, \tag{1}$$

where $\alpha \in [0, 1]$, $\lambda > 0$, $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{R}^n$ and $f(\mathbf{X}, \boldsymbol{\beta}) = \varphi(\mathbf{X})\boldsymbol{\beta}$. The objective in (1) is a weighted sum of two Mean Square Error (MSE) objective functions with different targets[8] and an ℓ_2-regularization on the weights. Minimization

[6] In Supplementary Material C we relate our problem setup to Mobahi et al. (2020) and extend some of our results to a constrained optimization setting with a regularization functional in Hilbert space.

[7] Since the kernel trick makes the predictions depend only on inner products in the feature space, it is not a restriction if D is infinite. However, for ease of exposition we assume D is finite.

[8] It is easy to verify that minimizing (1) and the classic MSE objective with a weighted target, i.e. $\tilde{\mathcal{L}}^{\text{distill}}(f(\mathbf{X}, \boldsymbol{\beta}), \mathbf{y}_1, \mathbf{y}_2) = \frac{1}{2} \|f(\mathbf{X}, \boldsymbol{\beta}) - (\alpha \mathbf{y}_1 + (1-\alpha)\mathbf{y}_2)\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2^2$, are equivalent and that the objective functions are equal up to the additive constant $\alpha(\alpha - 1) \|\mathbf{y}_1 - \mathbf{y}_2\|_2^2$.

of $\mathcal{L}^{\text{distill}}$ w.r.t. the parameters $\boldsymbol{\beta}$ is straightforward and yield the minimizer

$$\begin{aligned}\hat{\boldsymbol{\beta}} &\overset{\text{def}}{=} \underset{\beta}{\operatorname{argmin}}\, \mathcal{L}^{\text{distill}}(f(\mathbf{X}, \boldsymbol{\beta}), \mathbf{y}_1, \mathbf{y}_2) \\ &= \left(\varphi(\mathbf{X})^\intercal \varphi(\mathbf{X}) + \lambda \mathbf{I}_D\right)^{-1} \varphi(\mathbf{X})^\intercal \left(\alpha \mathbf{y}_1 + (1-\alpha)\mathbf{y}_2\right) \\ &= \varphi(\mathbf{X})^\intercal \left(\mathbf{K} + \lambda \mathbf{I}_n\right)^{-1} \left(\alpha \mathbf{y}_1 + (1-\alpha)\mathbf{y}_2\right)\end{aligned} \tag{2}$$

where the last equality follows by Woodbury's matrix identity and definition of $\mathbf{K}$. This solution can also be seen as a direct application of the Representer Theorem (Schölkopf et al., 2001). Let $\mathbf{y}^{(0)} \overset{\text{def}}{=} \mathbf{y}$, i.e. the original targets, and recursively define for the steps $\tau \geq 1$,

$$\begin{aligned}\hat{\boldsymbol{\beta}}^{(\tau)} &\overset{\text{def}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}}\, \mathcal{L}^{\text{distill}}(\boldsymbol{\beta}, \mathbf{y}, \mathbf{y}^{(\tau-1)}) \\ &= \varphi(\mathbf{X})^\intercal \left(\mathbf{K} + \lambda \mathbf{I}_n\right)^{-1} \left(\alpha \mathbf{y} + (1-\alpha)\mathbf{y}^{(\tau-1)}\right),\end{aligned} \tag{3}$$

$$\begin{aligned}f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) &\overset{\text{def}}{=} \varphi(\mathbf{x})^\intercal \hat{\boldsymbol{\beta}}^{(\tau)} \\ &= \kappa(\mathbf{x}, \mathbf{X})^\intercal \left(\mathbf{K} + \lambda \mathbf{I}_n\right)^{-1} \left(\alpha \mathbf{y} + (1-\alpha)\mathbf{y}^{(\tau-1)}\right),\end{aligned} \tag{4}$$

$$\mathbf{y}^{(\tau)} \overset{\text{def}}{=} f(\mathbf{X}, \hat{\boldsymbol{\beta}}^{(\tau)}). \tag{5}$$

Notice, the initial step ($\tau = 1$) corresponds to standard training by definition. Self-distillation treats the weighted average of the predictions, $\mathbf{y}^{(1)}$, from this initial model on $\mathcal{X}$, and the ground-truth targets, $\mathbf{y}$ as targets. This procedure is repeated as defined in (3)-(5) and we obtain the self-distillation procedure as illustrated in Figure 1. Note, the extreme cases $\alpha = 0$ and $\alpha = 1$ corresponds to merely training on the predictions from the previous model, and only training on the original targets, respectively. Thus, $\alpha = 1$ is usually not of interest, as the same solution is obtained from all distillation steps, and self-distillation plays no role in this scenario.

4 Main Results

In this section we present our main results for finite and infinite distillation steps along with an illustrative example highlighting the effect of α on the solutions. We show that the solution at each finite distillation step can be expressed as a function of the initial fit and the training data alone, obviating the need to explicitly fit a kernel regression at each distillation step. We then show that the degree at which the basis of the solutions sparsify depend on α. For $\alpha > 0$ infinite steps of self-distillation yields a simple form.

4.1 Finite Self-Distillation Steps

In the following theorem we show that the predictions obtained after any finite number of distillation steps can be expressed directly as a function of $\mathbf{y}$ and the kernel matrix $\mathbf{K}$ calculated at the initial fit ($\tau = 1$).

Theorem 4.1. *Let $\mathbf{y}^{(\tau)}, \hat{\boldsymbol{\beta}}^{(\tau)}$, and $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$ be defined as above and fix $\alpha \in [0, 1)$, then for $\tau \geq 1$, we have that*

$$\mathbf{y}^{(\tau)} = \left(\frac{\alpha}{1-\alpha} \sum_{i=1}^{\tau-1} \left((1-\alpha)\mathbf{K}(\mathbf{K} + \lambda \mathbf{I}_n)^{-1}\right)^i + (1-\alpha)^{\tau-1} \left(\mathbf{K}(\mathbf{K} + \lambda \mathbf{I}_n)^{-1}\right)^\tau\right) \mathbf{y}, \tag{6}$$

$$f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) = \alpha f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}_{\alpha=0}) \tag{7}$$

for any $\mathbf{x} \in \mathbb{R}^d$, where $\hat{\boldsymbol{\beta}}^{(\tau)}_{\alpha=0}$ is the minimizer in (3) with $\alpha = 0$.

Since (6) and (7) are expressed only in terms of $\mathbf{K}$, $(\mathbf{K} + \lambda \mathbf{I}_n)^{-1}$, and $\kappa(\mathbf{x}, \mathbf{X})$ we are able to calculate the predictions for the training data as well as for any $\mathbf{x} \in \mathbb{R}^d$ based merely on the initial fit ($\tau = 1$) without the need for any additional fits. Hence, despite the calculations of $\mathbf{K}$, $\kappa(\mathbf{x}, \mathbf{X})$ and especially $(\mathbf{K} + \lambda \mathbf{I}_n)^{-1}$ being (potentially) highly computationally demanding, when obtained, we can calculate any distillation step directly by the equations in Theorem 4.1. Furthermore, predictions at step τ can be seen as a weighted combination of two classical ridge regression solutions, based on the original targets and the predicted targets from step $\tau - 1$, respectively. However, since $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)}_{\alpha=0})$ depend

on α through $\mathbf{y}^{(\tau-1)}$, choosing α is non-trivial. We explore these dynamics in Section 4.3. Finally, as will be shown in the subsequent sections, Theorem 4.1 also allow us to analyse the regularization that self-distillation progressively impose on the solutions.

4.2 Sparsification of Self-Distillation Solutions

We now show that we can represent the solutions as a weighted sum of basis functions, and that this basis sparsifies when we increase τ, but also that the amount of sparsification highly depends on the choice of α.

Using the spectral decomposition of the symmetric matrix $\mathbf{K}$ we write $\mathbf{K} = \mathbf{V}\mathbf{D}\mathbf{V}^\intercal$, where $\mathbf{V} \in \mathbb{R}^{n\times n}$ is an orthogonal square matrix with the eigenvectors of $\mathbf{K}$ as rows and $\mathbf{D} \in \mathbb{R}^{n\times n}$ is a positive diagonal matrix with the associated eigenvalues in the diagonal. Replacing $\mathbf{K}$ by the diagonalization yields

$$\begin{aligned}\mathbf{K}(\mathbf{K}+\lambda\mathbf{I}_n)^{-1} &= \mathbf{V}\mathbf{D}\mathbf{V}^\intercal(\mathbf{V}\mathbf{D}\mathbf{V}^\intercal+\lambda\mathbf{I}_n)^{-1}\\ &= \mathbf{V}\mathbf{D}\mathbf{V}^\intercal\left(\mathbf{V}\left(\mathbf{D}+\lambda\mathbf{I}_n\right)\mathbf{V}^\intercal\right)^{-1}\\ &= \mathbf{V}\mathbf{D}\left(\mathbf{D}+\lambda\mathbf{I}_n\right)^{-1}\mathbf{V}^\intercal, \end{aligned} \tag{8}$$

where $\lambda > 0$, $\mathbf{D}$ is positive diagonal and $\mathbf{V}$ is orthogonal. By straightforward calculations on (6) and (8) we have

$$\mathbf{y}^{(\tau)} = \mathbf{V}\mathbf{B}^{(\tau)}\mathbf{V}^\intercal\mathbf{y}, \quad \text{where} \tag{9}$$

$$\mathbf{B}^{(\tau)} \overset{\text{def}}{=} \frac{\alpha}{1-\alpha}\sum_{i=1}^{\tau-1}(1-\alpha)^i\mathbf{A}^i + (1-\alpha)^{\tau-1}\mathbf{A}^\tau, \tag{10}$$

$$\mathbf{A} \overset{\text{def}}{=} \mathbf{D}(\mathbf{D}+\lambda\mathbf{I}_n)^{-1}, \tag{11}$$

and $\mathbf{A}, \mathbf{B}^{(\tau)} \in \mathbb{R}^{n\times n}$ are diagonal matrices for any τ. Furthermore, by (9) the only step-depending part of the solution is the diagonal matrix, $\mathbf{B}^{(\tau)}$, and in the following we show how $\mathbf{B}^{(\tau)}$ determines the sparsification of the solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$.

Lemma 4.2. *Let $\mathbf{B}^{(\tau)}$, and $\mathbf{A}$ be defined as above, and let $\mathbf{B}^{(0)} \overset{\text{def}}{=} \mathbf{I}$. Then we can express $\mathbf{B}^{(\tau)}$ recursively as*

$$\mathbf{B}^{(\tau)} = \mathbf{A}\left((1-\alpha)\mathbf{B}^{(\tau-1)} + \alpha\mathbf{I}_n\right), \tag{12}$$

and $[\mathbf{B}^{(\tau)}]_{k,k} \in [0,1]$ is (strictly) decreasing in τ for all $k \in [n]$ and $\tau \geq 1$ (if and only if $\lambda > 0$).

Similarly to (9), if we use Lemma 4.2 and Theorem 4.1, we can show that for any $\mathbf{x} \in \mathbb{R}^p$

$$\begin{aligned} f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) &= \kappa(\mathbf{x}, \mathbf{X})^\intercal\mathbf{V}\mathbf{D}^{-1}\mathbf{B}^{(\tau)}\mathbf{V}^\intercal\mathbf{y}\\ &= \mathbf{P}(\mathbf{x})^\intercal\mathbf{B}^{(\tau)}\mathbf{Z}, \quad \text{where}\\ \mathbf{P}(\mathbf{x}) &\overset{\text{def}}{=} \mathbf{D}^{-1}\mathbf{V}^\intercal\kappa(\mathbf{x}, \mathbf{X}),\\ \mathbf{Z} &\overset{\text{def}}{=} \mathbf{V}^\intercal\mathbf{y}. \end{aligned}$$

Thus, the solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$ can be represented as a weighted sum of some basis functions, where the basis functions are the components of the orthogonally transformed and scaled basis $\mathbf{P}(\mathbf{x})$, and $\mathbf{Z}$ is a orthogonally transformed vector of targets. In the following we show how $\mathbf{B}^{(\tau)}$ sparsifies with each distillation step when $\alpha < 1$, and thus also effectively sparsifies the solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$. Note, Lemma 4.2 not only provides us with a recursive formula for $\mathbf{B}^{(\tau)}$, but also show that each diagonal element of $\mathbf{B}^{(\tau)}$ is in $[0,1]$ and is strictly decreasing in τ, which in turn implies that the self-distillation procedure progressively shrinks the coefficients of the basis functions. Using Lemma 4.2 we can now show, that not only does $\mathbf{B}^{(\tau)}$ decrease in τ, smaller elements of $\mathbf{B}^{(\tau)}$ shrink faster than larger elements.

Theorem 4.3. *For any pair of diagonals of $\mathbf{D}$, i.e. d_k and d_j, where $d_k > d_j$, we have that for all $\tau \geq 1$,*

$$\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}} = \begin{cases} \frac{1+\frac{\lambda}{d_j}}{1+\frac{\lambda}{d_k}}, & \text{for } \alpha = 1,\\ \left(\frac{1+\frac{\lambda}{d_j}}{1+\frac{\lambda}{d_k}}\right)^\tau, & \text{for } \alpha = 0, \end{cases} \tag{13}$$

and for $\alpha \in (0, 1)$ *we have that*

$$\operatorname{sgn}\left(\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}} - \frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}}\right) = \operatorname{sgn}\left(\left(\left(\frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}} - \frac{[\mathbf{A}]_{k,k}}{[\mathbf{A}]_{j,j}}\right)\frac{[\mathbf{A}]_{j,j}}{[\mathbf{B}^{(\tau-1)}]_{k,k}([\mathbf{A}]_{k,k} - [\mathbf{A}]_{j,j})} + 1\right)^{-1} - \alpha\right), \tag{14}$$

where $\operatorname{sgn}(\cdot)$ *is the sign function*[9].

If we consider a pair of diagonals of $\mathbf{D}$, where $d_k > d_j$, then for $\alpha = 0$, the fraction $\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}}$ is strictly increasing in τ, due to the r.h.s. of (13) inside the parenthesis being strictly larger than 1. Hence, the diagonals corresponding to smaller eigenvalues shrinks faster than the larger ones as τ increases. However, for $\alpha \in (0, 1)$ we can not ensure this behaviour, but at step τ we are able to predict the behaviour at step $\tau + 1$, by using (14). Thus, when we include the ground-truth targets in our distillation procedure we do not consistently increase the regularization with each distillation step, but can potentially obtain a solution which does not sparsify any further. In the following we show that we can perform an infinite amount of distillation steps and still obtain non-zero solutions with a simple and explicit form.

4.3 Infinite Self-Distillation Steps

We now prove that if we were to perform an infinite amount of distillations steps ($\tau \to \infty$) the solution would solve the classical kernel ridge regression problem, with an amplified regularization parameter (by α^{-1}) if $\alpha > 0$. Observe, when $\alpha = 0$ and $\tau \to \infty$, (6) and (7) yield that the predictions $\mathbf{y}^{(\infty)}$ and $f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\infty)})$ collapse to the zero-solution for any $\mathbf{x} \in \mathbb{R}^p$.

Theorem 4.4. *Let* $\mathbf{y}^{(\tau)}, \hat{\boldsymbol{\beta}}^{(\tau)}$, *and* $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$ *be defined as above, and* $\alpha \in (0, 1]$, *then the following limits hold*

$$\begin{aligned} \mathbf{y}^{(\infty)} &\overset{\text{def}}{=} \lim_{\tau\to\infty} \mathbf{y}^{(\tau)} = \mathbf{K}\left(\mathbf{K} + \frac{\lambda}{\alpha}\mathbf{I}_n\right)^{-1}\mathbf{y} \\ f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\infty)}) &\overset{\text{def}}{=} \lim_{\tau\to\infty} f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) \\ &= \alpha f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\gamma}}^{(\infty)}) \end{aligned} \tag{15}$$

where (15) *corresponds to* classical *kernel ridge regression with amplified regularization parameter* $\frac{\lambda}{\alpha}$, *and we let* $\hat{\boldsymbol{\gamma}}^{(\infty)}$ *denote the kernel ridge regression parameter associated with this solution. Furthermore, the convergence* $\lim_{\tau\to\infty} \mathbf{y}^{(\tau)}$ *is of linear rate.*

Thus, not only are we able to calculate the predictions of a finite set of distillation steps based on the initial fit, we are also able to calculate the predictions of an infinite amount of steps.

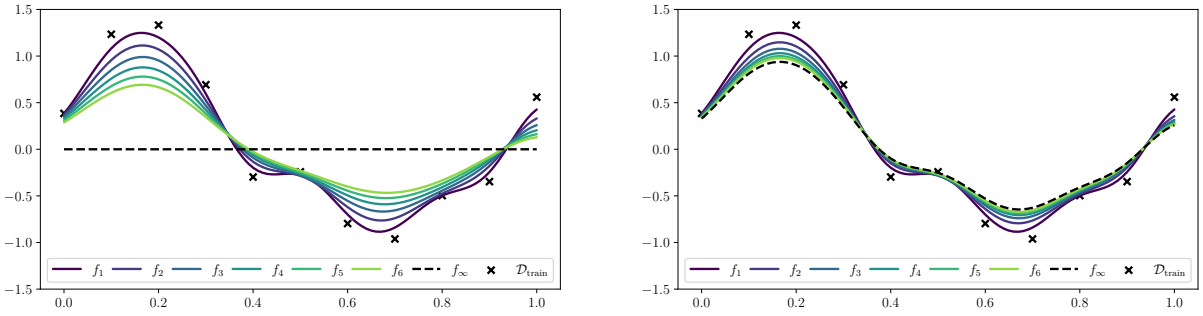
If $\alpha > 0$, then by (8) and Theorem 4.4, we have that $\mathbf{y}^{(\infty)} = \sum_{j=1}^{p} \mathbf{v}_j \frac{d_j}{d_j + \frac{\lambda}{\alpha}} \mathbf{v}_j^\intercal \mathbf{y}$ and we shrink the eigenvectors with smallest eigenvalues, corresponding to the directions with least variance, the most. Furthermore, if $\alpha > 0$ the limiting solution is a non-zero kernel ridge regression with regularization parameter $\frac{\lambda}{\alpha} \geq \lambda$, causing the eigenvectors associated with the smallest eigenvalues to shrink even more than in the original solution.

The solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(\infty)})$ is a weighted combination of the initial solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(1)})$ and the limiting solution $f(\cdot, \hat{\boldsymbol{\gamma}}^{(\infty)})$, both solving the same classical kernel ridge regression problem, but with regularization parameter λ and $\frac{\lambda}{\alpha}$, respectively. However, since $f(\cdot, \hat{\boldsymbol{\gamma}}^{(\infty)})$ depend on the choice of α, the dynamics of changing α is non-trivial. Increasing α reduces the regularization in $f(\cdot, \hat{\boldsymbol{\gamma}}^{(\infty)})$, but also reduce the weight on this part of the solution. Similarly, decreasing α causes the regularization of $f(\cdot, \hat{\boldsymbol{\gamma}}^{(\infty)})$ to increase, but it also increases the weight associated with this part of the solution. Thus, too large α causes the solution to mainly focus on the original targets, and gain little regularization from the distillation procedure. On the other hand, too small α causes the solution to mainly focus on the limiting solution, but also regularize this solution more. Hence, despite commonly practiced in empirical deep learning, our results gives a theoretical explanation for why one should treat α as an adjustable hyperparameter to fine-tune the amount of regularization that self-distillation should impose for a particular problem.

[9] Note, we use the definition of $\operatorname{sgn}(\cdot)$ where $\operatorname{sgn}(0) \overset{\text{def}}{=} 0$.

4.4 Illustrative example

Consider the training dataset $\mathcal{D}_{\text{train}}$ where $\mathcal{X}_{\text{train}} = \{0, 0.1, \ldots, 0.9, 1\}$ and $\mathcal{Y}_{\text{train}} = \{\sin(2\pi x) + \varepsilon \mid x \in \mathcal{X}_{\text{train}}\}$, and ε is sampled from a zero-mean Gaussian random variable with standard deviation 0.5. Let φ be the Radial Basis Function (RBF) kernel, i.e. $\kappa(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}$, where we choose $\gamma = \frac{1}{80}$, and let $\lambda = 0.2$ and consider the two cases; (a) $\alpha = 0$ and (b) $\alpha = 0.35$.

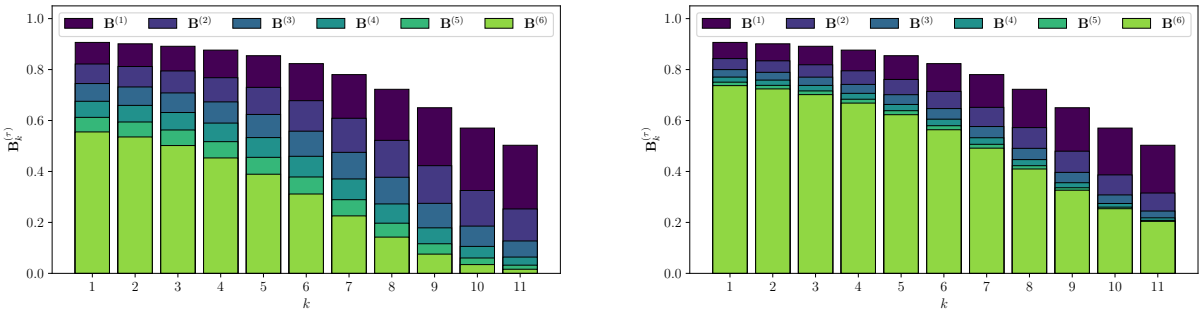


(a) $\alpha = 0$ (b) $\alpha = 0.35$

Figure 2: Six steps of self-distillation with (a) zero limiting solution (dashed) and (b) non-zero limiting solution (dashed). Training examples are represented with $\times$.

As illustrated in Figure 2a for case (a), the regularization imposed by self-distillation initially improves the quality of the solution, but eventually overregularize and the solutions underfit the data, and will eventually converge to the zero-solution. However, as shown in Figure 2b, using $\alpha > 0$, and more specifically $\alpha = 0.35$, reduce the imposed regularization and increases the stability of the distillation procedure; i.e. the solutions differ much less between each distillation step. This allows for a more dense exploration of solutions during iterated distillation steps, where increasing α reduces the difference between solutions from two consecutive steps, but also reduces the space of possible solutions as the limit, $f(\cdot, \hat{\boldsymbol{\beta}}^{(\infty)})$, approaches the initial solution $f(\cdot, \hat{\boldsymbol{\beta}}^{(1)})$. Finally, as expected by Theorem 4.4, we experience a fast convergence to the limit; usually less than 10 iterations are sufficient to converge.

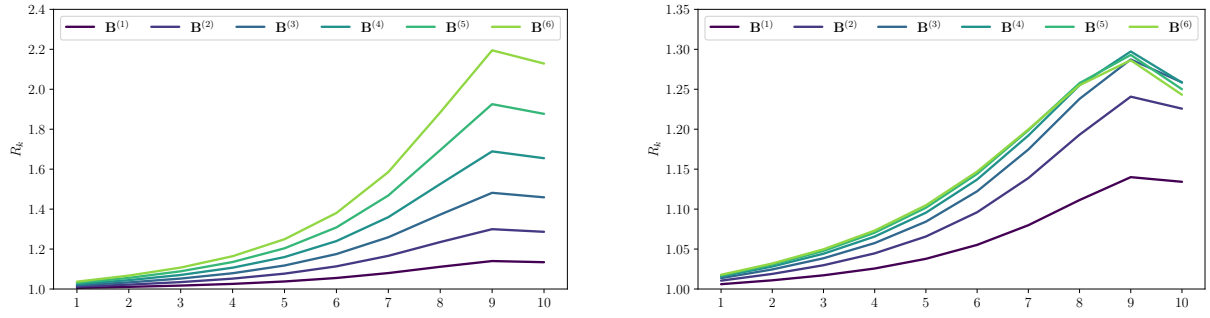
As expected from Lemma 4.2 and Theorem 4.3, Figure 3 verifies that both in case (a) and (b), the diagonal of $\mathbf{B}^{(\tau)}$ is decreasing in τ and the diagonal coordinates corresponding to smaller eigenvalues shrink faster than those corresponding to larger eigenvalues. Without loss of generality we can assume $d_1 < d_2 < \cdots < d_n$, and for $k = 1, \ldots n-1$ and any $\tau \geq 1$ define $R_k^{(\tau)} \overset{\text{def}}{=} \frac{[\mathbf{B}^{(\tau)}]_{k+1,k+1}}{[\mathbf{B}^{(\tau)}]_{k,k}}$. We expect $R_k^{(\tau)}$ to be strictly increasing in τ for all k in case (a), but for case (b) we can make no such guarantee. Both of these properties are verified in Figure 4.



(a) $\alpha = 0$ (b) $\alpha = 0.35$

Figure 3: Diagonal of $\mathbf{B}^{(\tau)}$ for $\tau = 1, \ldots, 6$ associated with Figure 2. Note, the plots are overlaid, but since the diagonal of $\mathbf{B}^{(\tau)}$ decrease in τ, all values until convergence are visible. In (a) we expect and observe strictly decreasing values in τ for all indices, until collapsing at 0, but in (b) the values converge to a non-zero solution.

Finally, we observe that in case (*a*), the values of $\mathbf{B}^{(\tau)}$ shrink much faster than in case (*b*), and eventually collapse to all zeros, whereas the latter is nearly converged after six iterations. Furthermore, case (*a*) appear to obtain a more sparsified solution, as the smallest coordinates effectively diminishes, which is not true for case (*b*). Furthermore, when directly comparing solutions from both cases with similar quality of fit, the solutions obtained with $\alpha = 0$ usually has smaller coordinates in $\mathbf{B}^{(\tau)}$ than those obtained with larger values of α.



(a) $\alpha = 0$ (b) $\alpha = 0.35$

Figure 4: Ratios, $R_k^{(\tau)}$ of the ordered diagonal of $\mathbf{B}^{(\tau)}$ for all τ. In (a) we expect and observe strictly increasing values in τ for all k, but have no such guarantee in (b). The x-axis corresponds to indices $k = 1, \ldots, n-1$.

5 Experiments

In the following section we show empirical results of performing a simple self-distillation procedure with deep neural networks with varying choices of α to investigate the large scale effects. The experiments are adapted from Mobahi et al. (2020) with the additional introduction of the α-parameter. For stronger baselines of the possible performance gains from self-distillation see Furlanello et al. (2018); Tian et al. (2020b); Ahn et al. (2019); Yang et al. (2018).

5.1 Experiment Setup

We perform self-distillation with ResNet-50 (He et al., 2016) networks on CIFAR-10 (Krizhevsky and Hinton, 2009), with minor pre-processing and augmentations[10]. The model is initialized randomly at each step[11] and trained with MSE loss and weighted targets[12] as described in Section 4. We use Adam optimizer with a learning rate of 10^{-4}, ℓ_2 regularization with regularization coefficient 10^{-4}, and train on the full 50000 training images and validate our generalization performance on the 10000 test images. We use the weights from the last step of optimization at each distillation step for the next step, irrespective of whether a better model occurred earlier in the training. Our models are trained for a fixed 75 epochs, which does not allow our models to overfit the training data, which is important for our models to be suitable for distillation procedures (Dong et al., 2019). The experiments are performed on a single Nvidia Tesla V100 16GB GPU with the PyTorch Lightning framework (Falcon, 2019).

5.2 Results

We perform our experiment 5 times and illustrate the mean, minimum and maximum at each distillation step in Figure 5. Each experiment is 10 chains of distillation steps, corresponding to $\alpha \in \{0.0, 0.1, \ldots, 0.9\}$, with the first model initialized identically across all 10 chains. The accuracy reported at the τ'th step is based on comparing the predictions $\mathbf{y}^{(\tau)}$ and $f(\mathbf{X}_{\text{valid}}, \boldsymbol{\beta}^{(\tau)})$ with the original targets; $\mathbf{y}_{\text{train}}$ and $\mathbf{y}_{\text{valid}}$, where $\mathbf{y}^{(\tau)}$ is the best performing model at the τ'th step.

[10] Training: We randomly flip an image horizontally with probability $\frac{1}{2}$, followed by a random 32×32 crop of the 40×40 zero padded image. Finally we normalize the image to have mean 0 and standard deviation 1. Validation: We normalize the image with the empirical mean and standard deviation used in the training pre-processing.

[11] Note, we initialize the models equally across all α for one experiment, but alter the seed for initialization between experiments.

[12] We treat the class labels as one-hot encoded vectors and use MSE between the predicted class probabilities and the target vector as well as the predicted class probabilities by the model from the previous distillation step.

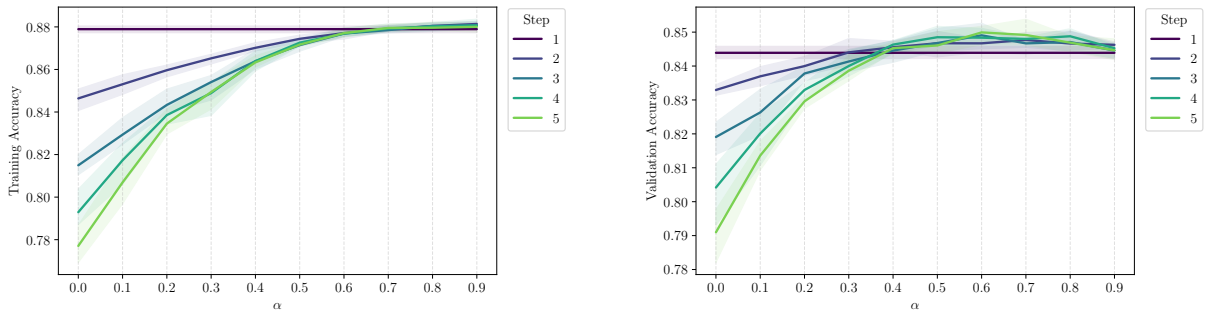


Figure 5: Training and validation accuracy for five distillation steps with ResNet-50 models on CIFAR-10 and varying α. The shaded area is bounded by the minimum and maximum accuracy obtained across all five experiments at the specific step and α-value.

The theory introduced in Section 4 suggests that self-distillation corresponds to a progressively amplified regularization of the solution, and larger α dampens the amount of regularization imposed by the procedure more than small values of α. Thus, for small α we expect the training accuracy to decrease with each distillation, but for larger α we might experience an increase in training accuracy, due to additional training iterations and a sufficient amount of ground-truth target information being kept in the optimization problem, which could prove beneficial. However, depending on the need for increased regularization, we expect the validation accuracy to increase for some α, and possibly decrease for α values either too small or too large. The above properties are observed in Figure 5, where $\alpha \leq 0.3$ generally overregularize the solution, and performance drops with distillation steps. For $\alpha > 0.3$ the performance generally improve with distillation, but for α close to one, the gains reduces, and a suitably balanced α for this experiment would be in $[0.5, 0.7]$. Thus, α should be considered an adjustable handle for the amount of regularization one need from self-distillation, and it should be tuned.

We conjecture that the choice of α should be chosen in relation to the degree of overfitting. If each distillation step is trained long enough to overfit the data, low values of α would regularize the solutions the most, and likely yield the best generalization error. If the models are not overfit to the training data, e.g. trained for fewer epochs, larger values of α would yield a better trade-off between continued training on the ground-truth targets and regularization imposed by self-distillation. Figure 6 support this conjecture; here we depict five distillation steps with ResNet-18 models using weights pre-trained on ImageNet. We observe an initial improvement in generalization performance for $\alpha = 0.3$, which drops off after a few distillation steps, leading to worse performance, likely due to overregularization, whereas $\alpha = 0.6$ and $\alpha = 0.7$ appear to be suitably balanced choices. See Supplementary Material B for additional experimental details and empirical results.

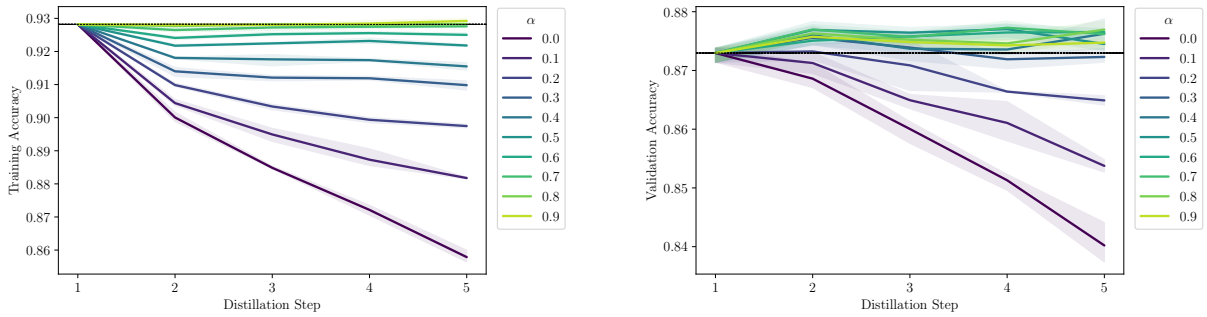


Figure 6: Training and validation accuracy for five distillation steps with ResNet-18 models (using weights pre-trained on ImageNet) on CIFAR-10 and varying α. The experiment is repeated three times.

6 Connection to Neural Networks

This paper theoretically investigates self-distillation of kernel ridge regression models, but distillation procedures are more commonly used in a deep learning setting. However, recent research in the over-parameterized regime has shown great progress and connected wide neural networks with kernel ridge regression using the Neural Tangent Kernel (NTK) (Lee et al., 2019, 2020; Hu et al., 2019). The following is a brief and informal connection between kernel ridge regression and wide[13] neural networks, motivating our problem setup.

Consider a neural network with scalar output $f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}) \in \mathbb{R}$, where $\boldsymbol{\theta}(t) \in \mathbb{R}^D$ is the vector of all network parameters at training iteration $t \geq 0$, and $\mathbf{x} \in \mathbb{R}^d$ some input. We consider the case where we use gradient descent on the MSE objective, $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(f_{\text{nn}}(\mathbf{x}_i, \boldsymbol{\theta}) - y_i)^2$ over some training dataset $\mathcal{D}_{\text{train}} \subseteq \mathbb{R}^d \times \mathbb{R}$. Consider the first-order Taylor-expansion of $f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta})$ w.r.t its parameters at initialization, $\boldsymbol{\theta}(0)$,

$$f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}) \approx f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}(0)) + \langle \nabla_{\boldsymbol{\theta}} f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}(0)), \boldsymbol{\theta} - \boldsymbol{\theta}(0) \rangle \tag{16}$$

where $f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}(0))$ and $\nabla_{\boldsymbol{\theta}} f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}(0))$ are constants w.r.t. $\boldsymbol{\theta}$. For sufficiently wide networks, (16) holds, and we say that we are in the *(NTK) regime* (Arora et al., 2019; Lee et al., 2019). Now, let $\varphi(\mathbf{x}) \overset{\text{def}}{=} \nabla_{\boldsymbol{\theta}} f_{\text{nn}}(\mathbf{x}, \boldsymbol{\theta}(0))$ for any $\mathbf{x} \in \mathbb{R}^d$, and denote the random kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) \overset{\text{def}}{=} \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$ for any $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ (Jacot et al., 2018). For sufficiently wide networks, the random kernel converges to a deterministic kernel, and since the r.h.s. of (16) is linear, one can show that minimizing $\mathcal{L}$ with gradient descent leads to the solution of the kernel regression problem, with the NTK; $\mathbf{x} \mapsto \kappa(\mathbf{x}, \mathbf{X})^\intercal \kappa(\mathbf{X}, \mathbf{X})^{-1}\mathbf{y}$, where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the matrix of training inputs, and $\mathbf{y} \in \mathbb{R}^n$ the vector of training targets (Arora et al., 2019; Lee et al., 2019). It has been shown that when minimizing the ℓ_2-regularized MSE loss, the solution becomes the kernel ridge regression solution (Lee et al., 2020).

The connections between neural networks and kernel ridge regressions in knowledge distillation settings have, to the best of out knowledge, not been explicitly investigated yet, but we hope that the results of this paper, will improve the understanding of self-distillation of neural networks once such connection is made rigorously.

7 Conclusion

In this paper, we provide theoretical arguments for the importance of weighting the teacher outputs with the ground-truth targets when performing self-distillation with kernel ridge regressions and empirically supported our results in a deep learning setting. We proved how the solution at any (possibly infinite) distillation step can be calculated directly from the initial distillation step, and that self-distillation for infinite steps corresponds to a classical kernel ridge regression solution with amplified regularization parameter. We showed both empirically and theoretically that the weighting parameter α determines the amount of regularization imposed by self-distillation, and thereby also the effectiveness of self-distillation.

7.1 Future Research Directions

Interesting directions of future research are on rigorously connecting neural networks and kernel methods in a knowledge distillation setting, extend to other objective functions than MSE as well as including intermediate model statistics in the distillation procedure. Finally, a larger systematic empirical study of the connection between the choice of α and the degree of overfitting is interesting as well.

Acknowledgement

We would like to thank GenomeDK and Aarhus University for providing computational resources that contributed to these research results. Furthermore, we would like to thank Daniel Borup and Ragnhild Ø. Laursen for comments and discussion, as well as Google Researchers Hossein Mobahi and Mehrdad Farajtabar (Deepmind) for clarifications on their experimental setup.

[13] Note, we refer to width as the number of hidden nodes in a fully connected neural network or channels in a convolutional neural network.

References

Abnar, S., Dehghani, M., and Zuidema, W. (2020). Transferring Inductive Biases through Knowledge Distillation. *arXiv preprint arXiv:2006.00555*.

Ahn, S., Hu, S. X., Damianou, A., Lawrence, N. D., and Dai, Z. (2019). Variational information distillation for knowledge transfer. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:9155–9163.

Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G. E., and Hinton, G. E. (2018). Large scale distributed neural network training through online distillation. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, pages 1–12.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. (2019). On exact computation with an infinitely wide neural net. *Advances in Neural Information Processing Systems*, 32(NeurIPS).

Ba, L. J. and Caruana, R. (2014). Do Deep Nets Really Need to be Deep? *Advances in Neural Information Processing Systems*, 3(January):2654–2662.

Bucila, C., Caruana, R., and Niculescu-Mizil, A. (2006). Model Compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 535–541, New York, NY, USA. ACM.

Chen, H., Wang, Y., Xu, C. C. C., Yang, Z., Liu, C., Shi, B., and Tian, Q. (2019). Data-free learning of student networks. In *Proceedings of the IEEE International Conference on Computer Vision*.

Dong, B., Hou, J., Lu, Y., and Zhang, Z. (2019). Distillation approximately early stopping? harvesting dark knowledge utilizing anisotropic information retrieval for overparameterized neural network.

Falcon, W. (2019). PyTorch Lightning.

Fang, G., Song, J., Shen, C., Wang, X., Chen, D., and Song, M. (2019). Data-Free Adversarial Distillation. *arXiv preprint arXiv:1912.11006*.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., and Anandkumar, A. (2018). Born Again Neural Networks. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1607–1616, Stockholmsmässan, Stockholm Sweden. PMLR.

Gao, M., Shen, Y., Li, Q., and Loy, C. C. (2020). Residual Knowledge Distillation. *arXiv preprint arXiv:2002.09168*.

Gotmare, A., Shirish Keskar, N., Xiong, C., and Socher, R. (2019). A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In *7th International Conference on Learning Representations, ICLR 2019*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:770–778.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Hu, W., Li, Z., and Yu, D. (2019). Simple and Effective Regularization Methods for Training on Noisily Labeled Data with Generalization Guarantee. *8th International Conference on Learning Representations, ICLR 2020*.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. *Advances in Neural Information Processing Systems*, 2018-Decem(5):8571–8580.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images.

Lee, J., Xiao, L., Schoenholz, S. S., Novak, Y. B. R., Sohl-Dickstein, J., Pennington, J., Bahri, Y., Dec, M. L., and Brain, G. (2019). Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. *Advances in Neural Information Processing Systems*.

Lee, J. D., Shen, R., Song, Z., Wang, M., and Yu, Z. (2020). Generalized Leverage Score Sampling for Neural Networks. *arXiv preprint arXiv:2009.09829*.

Lopes, R. G., Fenu, S., and Starner, T. (2017). Data-Free Knowledge Distillation for Deep Neural Networks. *arXiv preprint arXiv:1710.07535*.

Micaelli, P. and Storkey, A. J. (2019). Zero-shot Knowledge Transfer via Adversarial Belief Matching. In Wallach, H., Larochelle, H., Beygelzimer, A., d\textquotesingle Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 9551–9561. Curran Associates, Inc.

Mobahi, H., Farajtabar, M., and Bartlett, P. L. (2020). Self-Distillation Amplifies Regularization in Hilbert Space. *Advances in Neural Information Processing Systems*.

Müller, R., Kornblith, S., and Hinton, G. (2019). When Does Label Smoothing Help? In *Advances in Neural Information Processing Systems*.

Nakkiran, P., Barak, B., Kaplun, G., Sutskever, I., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2020). Deep Double Descent: Where Bigger Models and More Data Hurt. In *International Conference on Learning Representations*.

Nayak, G. K., Mopuri, K. R., Shaj, V., Babu, R. V., Chakraborty, A., Venkatesh Babu, R., and Chakraborty, A. (2019). Zero-shot knowledge distillation in deep networks. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:8317–8325.

Park, W., Kim, D., Lu, Y., and Cho, M. (2019). Relational Knowledge Distillation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Phuong, M. and Lampert, C. H. (2019). Towards understanding knowledge distillation. *36th International Conference on Machine Learning, ICML 2019*, 2019-June(2014):8993–9007.

Radosavovic, I., Dollár, P., Girshick, R., Gkioxari, G., He, K., Dollar, P., Girshick, R., Gkioxari, G., and He, K. (2018). Data Distillation: Towards Omni-Supervised Learning. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 4119–4128.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. (2015). FitNets: Hints for thin deep nets. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pages 1–13.

Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). A generalized representer theorem. In *International conference on computational learning theory*, volume 2111, pages 416–426. Springer.

Srinivas, S. and Fleuret, F. F. (2018). Knowledge transfer with jacobian matching. *35th International Conference on Machine Learning, ICML 2018*, 11:7515–7523.

Tang, J., Shivanna, R., Zhao, Z., Lin, D., Singh, A., Chi, E. H., and Jain, S. (2020). Understanding and Improving Knowledge Distillation. *arXiv preprint arXiv:2002.03532*.

Tian, Y., Krishnan, D., and Isola, P. (2020a). Contrastive Representation Distillation. In *International Conference on Learning Representations*, pages 1–19.

Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J. B., and Isola, P. (2020b). Rethinking Few-Shot Image Classification: a Good Embedding Is All You Need?

Wang, L. and Yoon, K.-J. (2020). Knowledge Distillation and Student-Teacher Learning for Visual Intelligence: A Review and New Outlooks. *arXiv preprint arXiv:2004.05937*.

Yang, C., Xie, L., Qiao, S., and Yuille, A. (2018). Knowledge Distillation in Generations: More Tolerant Teachers Educate Better Students. *arXiv preprint arXiv:1805.05551*.

Yao, Y., Rosasco, L., and Caponnetto, A. (2007). On Early Stopping in Gradient Descent Learning. *Constructive Approximation*, 26(2):289–315.

Yim, J., Joo, D., Bae, J., and Kim, J. (2017). A Gift from Knowledge Distillation: Fast Optimization, Network Minimization and Transfer Learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7130–7138.

Zagoruyko, S. and Komodakis, N. (2019). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–13.

Zhang, C., Recht, B., Bengio, S., Hardt, M., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*.

A Proofs

This section includes all proofs referenced in the main part of the paper, along with the associated theorems and lemmas for completeness.

Theorem A.1. *Let $\mathbf{y}^{(\tau)}, \hat{\boldsymbol{\beta}}^{(\tau)}$, and $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$ be defined as above and fix $\alpha \in [0,1)$, then for $\tau \geq 1$, we have that*

$$\mathbf{y}^{(\tau)} = \left(\frac{\alpha}{1-\alpha} \sum_{i=1}^{\tau-1} \left((1-\alpha)\mathbf{K}(\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \right)^i + (1-\alpha)^{\tau-1} \left(\mathbf{K}(\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \right)^\tau \right) \mathbf{y}, \tag{17}$$

$$f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) = \alpha f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}_{\alpha=0}) \tag{18}$$

for any $\mathbf{x} \in \mathbb{R}^d$, where $\hat{\boldsymbol{\beta}}^{(\tau)}_{\alpha=0}$ is the minimizer (3) with $\alpha = 0$.

Proof. We prove the theorem by induction, where we let $\tilde{\mathbf{K}} \overset{\text{def}}{=} \mathbf{K}(\mathbf{K}+\lambda \mathbf{I}_n)^{-1}$. For $\tau = 1$, the result hold trivially, and thus, assume it hold for $\tau = t$. Since $\boldsymbol{\beta}^{(t+1)} = \varphi(\mathbf{X})^\intercal (\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \left(\alpha \mathbf{y} + (1-\alpha)\mathbf{y}^{(t)} \right)$ we have that

$$\begin{aligned}
\mathbf{y}^{(t+1)} &= \varphi(\mathbf{X})\varphi(\mathbf{X})^\intercal (\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \left(\alpha \mathbf{y} + (1-\alpha)\mathbf{y}^{(t)} \right) \\
&= \alpha \tilde{\mathbf{K}} \mathbf{y} + (1-\alpha)\tilde{\mathbf{K}} \left(\frac{\alpha}{1-\alpha} \sum_{i=1}^{t-1} ((1-\alpha)\tilde{\mathbf{K}})^i + (1-\alpha)^{t-1} \tilde{\mathbf{K}}^t \right) \mathbf{y} \\
&= \alpha \tilde{\mathbf{K}} \mathbf{y} + \left(\frac{\alpha}{1-\alpha} \sum_{i=2}^{t} ((1-\alpha)\tilde{\mathbf{K}})^i + (1-\alpha)^{t} \tilde{\mathbf{K}}^{t+1} \right) \mathbf{y} \\
&= \left(\frac{\alpha}{1-\alpha} \sum_{i=1}^{t} ((1-\alpha)\tilde{\mathbf{K}})^i + (1-\alpha)^{t} \tilde{\mathbf{K}}^{t+1} \right) \mathbf{y},
\end{aligned}$$

which finalizes our induction proof for the first part. For the second part, note that it also holds trivially for $\tau = 1$. Thus assume, it holds for $\tau = t$, then by direct manipulations

$$\begin{aligned}
f(\mathbf{x}, \boldsymbol{\beta}^{(t+1)}) &= \kappa(\mathbf{x}, \mathbf{X})^\intercal (\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \left(\alpha \mathbf{y} + (1-\alpha)\mathbf{y}^{(t)} \right) \\
&= \alpha f(\mathbf{x}, \boldsymbol{\beta}^{(1)}) + (1-\alpha)\kappa(\mathbf{x}, \mathbf{X})^\intercal (\mathbf{K}+\lambda \mathbf{I}_n)^{-1} \mathbf{y}^{(t)} \\
&= \alpha f(\mathbf{x}, \boldsymbol{\beta}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(t+1)}_{\alpha=0}),
\end{aligned}$$

where we let $\hat{\boldsymbol{\beta}}^{(t+1)}_{\alpha=0}$ denote the minimizer (3) with $\alpha = 0$; i.e. minimizing the classical kernel ridge regression problem with targets $\mathbf{y}^{(t)}$. □

Lemma A.2. *Let $\mathbf{B}^{(0)} \overset{\text{def}}{=} \mathbf{I}$, and $\mathbf{B}^{(\tau)}$, $\mathbf{A}$ be defined as above, then we can express $\mathbf{B}^{(\tau)}$ recursively as*

$$\mathbf{B}^{(\tau)} = \mathbf{A} \left((1-\alpha)\mathbf{B}^{(\tau-1)} + \alpha \mathbf{I}_n \right),$$

and $[\mathbf{B}^{(\tau)}]_{k,k} \in [0,1]$ is (strictly) decreasing in τ for all $k \in [n]$ and $\tau \geq 1$ (if and only if $\lambda > 0$).

Proof. The case, $\tau = 1$, is easy to verify, and we assume the claim holds for $\tau = t$. Then note that

$$\begin{aligned}
\mathbf{A} \left((1-\alpha)\mathbf{B}^{(t)} + \alpha \mathbf{I}_n \right) &= \frac{\alpha}{1-\alpha} \sum_{i=2}^{t} (1-\alpha)^i \mathbf{A}^i + (1-\alpha)^t \mathbf{A}^{t+1} + \alpha \mathbf{A} \\
&= \frac{\alpha}{1-\alpha} \sum_{i=1}^{t} (1-\alpha)^i \mathbf{A}^i + (1-\alpha)^t \mathbf{A}^{t+1} \\
&= \mathbf{B}^{(t+1)},
\end{aligned}$$

finalizing the induction proof. Now, note that for any k and $\tau \geq 1$, then

$$[\mathbf{A}]_k \left((1-\alpha)[\mathbf{B}^{(\tau-1)}]_{k,k} + \alpha \right) = [\mathbf{B}^{(\tau)}]_{k,k} \leq [\mathbf{B}^{(\tau-1)}]_{k,k} = [\mathbf{A}]_k \left((1-\alpha)[\mathbf{B}^{(\tau-2)}]_{k,k} + \alpha \right),$$

if and only if $[\mathbf{B}^{(\tau-1)}]_{k,k} \leq [\mathbf{B}^{(\tau-2)}]_{k,k}$, and iteratively, if and only if $[\mathbf{B}^{(1)}]_{k,k} \leq [\mathbf{B}^{(0)}]_{k,k}$. The latter is indeed true, since $\mathbf{B}^{(1)} = \mathbf{A}$ and $\mathbf{B}^{(0)} = \mathbf{I}_n$ by definition. Finally, $\mathbf{A} = \mathbf{I}_n$ if and only if $\lambda = 0$.

□

Theorem A.3. *For any pair of diagonals of* $\mathbf{D}$*, i.e.* d_k *and* d_j*, where* $d_k > d_j$*, we have that for all* $\tau \geq 1$*,*

$$\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}} = \begin{cases} \frac{1+\frac{\lambda}{d_j}}{1+\frac{\lambda}{d_k}}, & \text{for } \alpha = 1, \\ \left(\frac{1+\frac{\lambda}{d_j}}{1+\frac{\lambda}{d_k}}\right)^{\tau}, & \text{for } \alpha = 0, \end{cases}$$

and for $\alpha \in (0,1)$ *we have that*

$$\operatorname{sgn}\left(\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}} - \frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}}\right) = \operatorname{sgn}\left(\left(\left(\frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}} - \frac{[\mathbf{A}]_{k,k}}{[\mathbf{A}]_{j,j}}\right)\frac{[\mathbf{A}]_{j,j}}{[\mathbf{B}^{(\tau-1)}]_{k,k}([\mathbf{A}]_{k,k} - [\mathbf{A}]_{j,j})} + 1\right)^{-1} - \alpha\right),$$

where $\operatorname{sgn}(\cdot)$ *is the sign function, i.e.*

$$\operatorname{sgn}(x) \overset{\text{def}}{=} \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases}.$$

Proof. First note that

$$\frac{[\mathbf{A}]_{k,k}}{[\mathbf{A}]_{j,j}} = \frac{\frac{d_k}{d_k+\lambda}}{\frac{d_j}{d_j+\lambda}} = \frac{1+\frac{\lambda}{d_j}}{1+\frac{\lambda}{d_k}},$$

and for $\alpha = 1$, (12) amounts to $\mathbf{B}^{(\tau)} = \mathbf{A}$, which gives the first result. For $\alpha = 0$, (12) amounts to $\mathbf{B}^{(\tau)} = \mathbf{A}^{\tau}$, and the second result follows. For the remainder we denote $[\mathbf{B}^{(\tau-1)}]_{k,k}$ by $\mathbf{B}_k$ and $[\mathbf{A}]_{k,k}$ by $\mathbf{A}_k$ to simplify notation. We investigate the case where both r.h.s. and l.h.s. equals zero. Thus, for $\alpha \in (0,1)$, we observe that if

$$\frac{\mathbf{B}_k}{\mathbf{B}_j} = \frac{\mathbf{A}_k}{\mathbf{A}_j}\frac{(1-\alpha)\mathbf{B}_k + \alpha}{(1-\alpha)\mathbf{B}_j + \alpha} = \frac{\mathbf{A}_k}{\mathbf{A}_j}\frac{\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1}{\frac{1-\alpha}{\alpha}\mathbf{B}_j + 1},$$

then we have that

$$\mathbf{B}_j = \frac{1}{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{\frac{1-\alpha}{\alpha}\mathbf{B}_k+1}{\mathbf{B}_k}\right) - \frac{1-\alpha}{\alpha}} = \frac{\mathbf{B}_k}{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k}$$

$$\frac{1-\alpha}{\alpha}\mathbf{B}_j + 1 = \frac{\frac{1-\alpha}{\alpha}\mathbf{B}_k}{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k} + 1 = \frac{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right)}{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k},$$

which in turn yield that

$$\begin{aligned}\frac{\mathbf{B}_k}{\mathbf{B}_j} &= \frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right)\left(\frac{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k}{\frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right)}\right) \\ &= \frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k.\end{aligned}$$

Now, observe that $0 = \alpha - \left(\left(\frac{\mathbf{B}_k}{\mathbf{B}_j} - \frac{\mathbf{A}_k}{\mathbf{A}_j}\right)\frac{\mathbf{A}_j}{\mathbf{B}_k(\mathbf{A}_k - \mathbf{A}_j)} + 1\right)^{-1}$ yield that

$$\frac{\mathbf{B}_k}{\mathbf{B}_j} = \frac{1-\alpha}{\alpha}\frac{\mathbf{B}_k(\mathbf{A}_k - \mathbf{A}_j)}{\mathbf{A}_j} + \frac{\mathbf{A}_k}{\mathbf{A}_j} = \frac{\mathbf{A}_k}{\mathbf{A}_j}\left(\frac{1-\alpha}{\alpha}\mathbf{B}_k + 1\right) - \frac{1-\alpha}{\alpha}\mathbf{B}_k.$$

Thus, similar calculations with $>$ and $<$ instead of $=$, completes the claim. □

Note, in the following we state and prove a slightly more general result than Theorem 4.4.

Theorem A.4. *Let $\mathbf{y}^{(\tau)}, \hat{\boldsymbol{\beta}}^{(\tau)}$, and $f(\cdot, \hat{\boldsymbol{\beta}}^{(\tau)})$ be defined as above, and $\alpha \in [0,1]$, then the following limits hold*

$$\mathbf{y}^{(\infty)} \stackrel{\text{def}}{=} \lim_{\tau\to\infty} \mathbf{y}^{(\tau)} = \alpha\mathbf{K}\left(\alpha\mathbf{K} + \lambda\mathbf{I}_n\right)^{-1}\mathbf{y}$$
$$f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\infty)}) \stackrel{\text{def}}{=} \lim_{\tau\to\infty} f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\tau)}) = \alpha\kappa(\mathbf{x}, \mathbf{X})^\intercal(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\left(\mathbf{I}_n + (1-\alpha)\mathbf{K}(\alpha\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\right)\mathbf{y}$$

and if $\alpha > 0$, then

$$\mathbf{y}^{(\infty)} = \mathbf{K}\left(\mathbf{K} + \frac{\lambda}{\alpha}\mathbf{I}_n\right)^{-1}\mathbf{y}$$
$$f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(\infty)}) = \alpha f(\mathbf{x}, \hat{\boldsymbol{\beta}}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\gamma}}^{(\infty)})$$

where the former corresponds to classical *kernel ridge regression with amplified regularization parameter $\frac{\lambda}{\alpha}$, and we let $\hat{\boldsymbol{\gamma}}^{(\infty)}$ denote the kernel ridge regression parameter associated with this solution. Furthermore, the convergence $\lim_{\tau\to\infty}\mathbf{y}^{(\tau)}$ is of linear rate.*

Proof. By (8) we have that $\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1} = \mathbf{V}\mathbf{D}\left(\mathbf{D} + \lambda\mathbf{I}_n\right)^{-1}\mathbf{V}^\intercal$ where $\lambda > 0$, $\mathbf{D}$ is positive diagonal and $\mathbf{V}$ is orthogonal, and hence, the eigenvalues of $\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}$ are all smaller than 1 in absolute value, and thus $(1-\alpha)^{\tau-1}\left(\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\right)^\tau$ converge to the zero-matrix when $\tau \to \infty$. Thus, using the limit for a geometric series of matrices we get that

$$\begin{aligned}
\lim_{\tau\to\infty}\mathbf{y}^{(\tau)} &= \left(\frac{\alpha}{1-\alpha}\sum_{i=1}^{\infty}\left((1-\alpha)\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\right)^i\right)\mathbf{y} \\
&= \frac{\alpha}{1-\alpha}(1-\alpha)\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}(\mathbf{I}_n - (1-\alpha)\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1})^{-1}\mathbf{y} \\
&= \alpha\mathbf{K}\left(\alpha\mathbf{K} + \lambda\mathbf{I}_n\right)^{-1}\mathbf{y}.
\end{aligned}$$

If $\alpha > 0$, the remaining result for $\lim_{\tau\to\infty}\mathbf{y}^{(\tau)}$ follows directly. Now, by inserting $\mathbf{y}^{(\infty)}$ and manipulating the result, we get that

$$\begin{aligned}
f(\mathbf{x}, \boldsymbol{\beta}^{(\infty)}) &= \kappa(\mathbf{x}, \mathbf{X})^\intercal(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\left(\alpha\mathbf{y} + (1-\alpha)\mathbf{y}^{(\infty)}\right) \\
&= \kappa(\mathbf{x}, \mathbf{X})^\intercal(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\left(\alpha\mathbf{I}_n + (1-\alpha)\alpha\mathbf{K}\left(\alpha\mathbf{K} + \lambda\mathbf{I}_n\right)^{-1}\right)\mathbf{y} \\
&= \alpha\kappa(\mathbf{x}, \mathbf{X})^\intercal(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\left(\mathbf{I}_n + (1-\alpha)\mathbf{K}\left(\alpha\mathbf{K} + \lambda\mathbf{I}_n\right)^{-1}\right)\mathbf{y},
\end{aligned}$$

and if $\alpha > 0$, then

$$\begin{aligned}
f(\mathbf{x}, \boldsymbol{\beta}^{(\infty)}) &= \alpha f(\mathbf{x}, \boldsymbol{\beta}^{(1)}) + (1-\alpha)\kappa(\mathbf{x}, \mathbf{X})^\intercal(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{K}\left(\mathbf{K} + \frac{\lambda}{\alpha}\mathbf{I}_n\right)^{-1}\mathbf{y} \\
&= \alpha f(\mathbf{x}, \boldsymbol{\beta}^{(1)}) + (1-\alpha) f(\mathbf{x}, \hat{\boldsymbol{\gamma}}^{(\infty)}),
\end{aligned}$$

where we let $\hat{\boldsymbol{\gamma}}^{(\infty)}$ denote the kernel ridge regression parameter associated with the classical kernel ridge regression problem on the original targets with regularization parameter $\frac{\lambda}{\alpha}$.

Finally, denote by $\mathbf{C} \stackrel{\text{def}}{=} (1-\alpha)\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}$, then we have that

$$\mathbf{E}(t) \stackrel{\text{def}}{=} \sum_{i=1}^{t}\mathbf{C}^i - \sum_{i=1}^{\infty}\mathbf{C}^i = \mathbf{C}^{t+1}\left(\mathbf{C} - \mathbf{I}_n\right)^{-1},$$

and thus for an additional s steps we have $\mathbf{E}(t+s) = \mathbf{C}^{t+s+1}(\mathbf{C} - \mathbf{I}_n)^{-1} = \mathbf{C}^s\mathbf{E}(t)$. Hence, the convergence is of linear rate as claimed. □

B Additional Empirical Results

In the following we include additional results for the experiments reported in the main part of the paper, along with an additional experiment with weights pre-trained on ImageNet.

B.1 ResNet-50 with Randomly Initialized Weights

In Section 5 we reported results for the ResNet-50 experiment on CIFAR-10, and showed how the choice of α greatly influences whether self-distillation improves generalization or not. In Figure 7 we show the training and validation accuracy at each distillation step, plotted with distillation steps on the x-axis. Note, the experiment is identical to that reported in the paper, but here we easily see the evolution of our solutions as we progressively distill it. The training accuracy are ordered according to α over all steps, with 0.0 the smallest and 0.9 the largest. However, for the validation accuracy, the performance generally increase with α from 0.0 to 0.6, and decrease again with α going from 0.7 to 0.9.

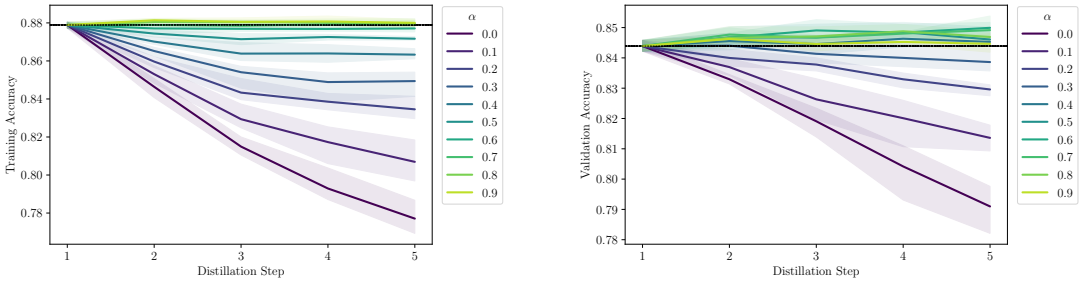


Figure 7: Training and validation accuracy for five distillation steps with ResNet-50 models on CIFAR-10 and varying α.

B.2 ResNet-18 with Pretrained Weights

The following experiment is similar to the ResNet-50 experiment above, but with the following changes: We run the experiment three times with 100 epochs, and we use a ResNet-18 with weights pre-trained on ImageNet to obtain higher training accuracy, and likely overfitting the training data a bit more than in the original experiment[14]. See Figure 8 for the results of this experiment, where self-distillation with $\alpha = 0.3$ improves the performance initially, but drop off after a few steps of distillations, indicating that the regularization initially improves generalization, but eventually lead to underfitting[15]. However, in the original experiment $\alpha = 0.3$ were performing worse than the non-distilled model, supporting our conjecture that α should be chosen smaller, when the remaining hyper-parameters are chosen such that the model is prone to overfitting the data. In these experiments, $\alpha = 0.6$ or $\alpha = 0.7$ would be suitably balanced choices, and once again, $\alpha = 0$ overregularizes the solutions leading to worse performance when distilled.

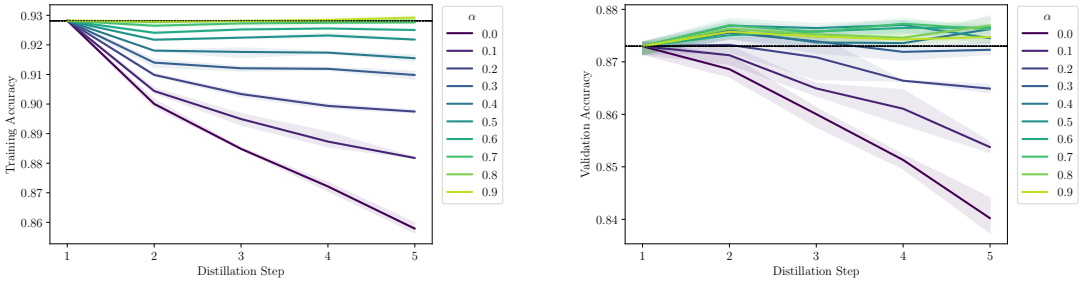


Figure 8: Training and validation accuracy for five distillation steps with ResNet-18 models (using weights pre-trained on ImageNet) on CIFAR-10 and varying α.

[14] Standard pre-trained weights obtained from the torchvision package.

[15] Note, Figure 8 is identical to Figure 6, but is included for completeness of this section.

C Connections to Constrained Optimization Problem

The setup investigated in this paper is the unconstrained optimization problem presented in (2), but many of the results can easily be extended to a constrained optimization problem, with a general regularization functional in Hilbert spaces, namely the natural extension of the setup proposed by Mobahi et al. (2020). Mobahi et al. (2020) propose to solve the problem

$$f^{(\tau)} \overset{\text{def}}{=} \underset{f\in\mathcal{F}}{\operatorname{argmin}} \int_{\mathcal{X}}\int_{\mathcal{X}} u(\boldsymbol{x},\boldsymbol{x}')f(\boldsymbol{x})f(\boldsymbol{x}')d\boldsymbol{x}d\boldsymbol{x}' \quad \text{s.t.} \quad \frac{1}{N}\sum_{n=1}^{N}\left(f(\boldsymbol{x}_n)-y_n\right)^2 \leq \varepsilon, \tag{19}$$

where $\varepsilon > 0$ is a desired loss tolerance, $\tau \geq 1$, $f^{(0)}(\boldsymbol{x}_n) = y_n$ for $n = 1,\dots,N$, and u being symmetric and such that $\forall f \in \mathcal{F}$[16] the double integral is greater than or equal to 0 with equality only when $f(x) = 0$. See Mobahi et al. (2020) for details.

The natural extension of this problem is to include ground-truth labels, and solve the weighted problem

$$f^{(\tau)} = \underset{f\in\mathcal{F}}{\operatorname{argmin}} \int_{\mathcal{X}}\int_{\mathcal{X}} u(\boldsymbol{x},\boldsymbol{x}')f(\boldsymbol{x})f(\boldsymbol{x}')d\boldsymbol{x}d\boldsymbol{x}' \tag{20}$$

$$\text{s.t.} \quad \frac{\alpha}{N}\sum_{n=1}^{N}\left(f(\boldsymbol{x}_n)-y_n\right)^2 + \frac{1-\alpha}{N}\sum_{n=1}^{N}\left(f(\boldsymbol{x}_n)-f^{(\tau-1)}(\boldsymbol{x}_n)\right)^2 \leq \varepsilon, \tag{21}$$

for $\tau \geq 1$, where $\alpha \in [0,1]$ and $f^{(0)}(\boldsymbol{x}_n) = y_n$ for $n = 1,\dots,N$. In Mobahi et al. (2020), $\alpha = 0$, and this problem completely ignores the ground truth data after the first model fit, and it is easy to see that consecutive self-fits will be penalized increasingly stronger, and eventually collapse to zero, whenever $\frac{1}{N}\sum_{n=1}^{N}\left(f(\boldsymbol{x}_n)-y_n\right)^2 \leq \varepsilon$. The case $\alpha = 1$, corresponds to fitting to the ground-truth at each iteration, and do not benefit from distillation, and thus is without interest here.

C.1 Collapsing and converging conditions

The regularization functional of (20), is clearly minimized by $f_t(\boldsymbol{x}) = 0$, but in order for this to be a solution for some $\tau \geq 1$, it must hold that

$$\frac{\alpha}{N}\left\|\boldsymbol{y}\right\|_2^2 + \frac{1-\alpha}{N}\left\|\boldsymbol{y}^{(\tau-1)}\right\|_2^2 \leq \varepsilon,$$

where we use the notation that $\boldsymbol{y}^{(\tau)} = (f^{(\tau)}(\boldsymbol{x}_1),\dots,f^{(\tau)}(\boldsymbol{x}_N))^T$ and $\boldsymbol{y} = (y_1,\dots,y_N)$. For $\tau = 1$, this amounts to $\frac{1}{N}\left\|\boldsymbol{y}\right\|_2^2 \leq \varepsilon$, and for $\tau > 1$

$$\frac{1-\alpha}{N}\left\|\boldsymbol{y}^{(\tau-1)}\right\|_2^2 \leq \varepsilon - \frac{\alpha}{N}\left\|\boldsymbol{y}\right\|_2^2. \tag{22}$$

But since the l.h.s. is non-negative, it is required that $\frac{\alpha}{N}\left\|\boldsymbol{y}\right\|_2^2 \leq \varepsilon$ in order for $f^{(\tau)}(\boldsymbol{x}) = 0$ to be a solution. Hence, we can construct the following settings, that determine the behavior of the solutions:

1. $\frac{1}{N}\left\|\boldsymbol{y}\right\|_2^2 \in [0,\varepsilon] \implies \left\|\boldsymbol{y}^{(\tau)}\right\|_2 = 0 \;\forall \tau \geq 1,$ (Collapsed solution)
2. $\frac{1}{N}\left\|\boldsymbol{y}\right\|_2^2 \in \left(\varepsilon,\frac{\varepsilon}{\alpha}\right] \implies \exists \underline{\tau} \geq 1$ such that $\begin{cases}\left\|\boldsymbol{y}^{(\tau)}\right\|_2 > 0 & \forall \tau < \underline{\tau},\\ \left\|\boldsymbol{y}^{(\tau)}\right\|_2 = 0 & \forall \tau \geq \underline{\tau},\end{cases}$ (Converging to collapsed solution)
3. $\frac{1}{N}\left\|\boldsymbol{y}\right\|_2^2 \in \left(\frac{\varepsilon}{\alpha},\infty\right) \implies \left\|\boldsymbol{y}^{(\tau)}\right\|_2 > 0 \;\forall \tau \geq 1.$ (Converging to non-collapsed solution)

If we let $\alpha \to 0$ the interval $\left(\varepsilon,\frac{\varepsilon}{\alpha}\right]$ effectively becomes (ε,∞), and any solution will collapse at some point (Mobahi et al., 2020). Analogously, if we let $\alpha \to 1$, the interval $(\varepsilon,\frac{\varepsilon}{\alpha}]$ effectively becomes empty, and all non-collapsed solutions will converge to a non-zero solution. Hence, if $\alpha > 0$, one can obtain non-collapsing convergence with infinite iterations. Furthermore, if we let $\varepsilon \to 0$, then $[0,\varepsilon]$ and $(\varepsilon,\frac{\varepsilon}{\alpha}]$ will practically collapse to empty intervals, and we will always obtain convergence to non-collapsing solutions, which will correspond to an interpolating solution.

[16] For a given u the function space $\mathcal{F}$ is the space of functions f for which the double integral in (19) is bounded.

For the remainder we assume $\alpha \in (0,1)$, since the boundary cases are covered in Mobahi et al. (2020) ($\alpha = 0$) or is without interest ($\alpha = 1$). Furthermore, we assume that $\|\mathbf{y}\|_2 > \sqrt{N\varepsilon}$ to avoid a collapsed solution from the beginning. Utilizing the Karush-Kuhn-Tucker (KKT) conditions for this problem, we can rephrase our optimization problem as

$$f^{(\tau)} = \operatorname*{argmin}_{f \in \mathcal{F}} \frac{\alpha}{N} \sum_{n=1}^{N} (f(\boldsymbol{x}_n) - y_n)^2 + \frac{1-\alpha}{N} \sum_{n=1}^{N} \left(f(\boldsymbol{x}_n) - f^{(\tau-1)}(\boldsymbol{x}_n) \right)^2 + \lambda_\tau \int_{\mathcal{X}} \int_{\mathcal{X}} u(\boldsymbol{x}, \boldsymbol{x}') f(\boldsymbol{x}) f(\boldsymbol{x}') d\boldsymbol{x} d\boldsymbol{x}', \tag{23}$$

where $\lambda_\tau \geq 0$. For suitably chosen λ_τ, one can show that $f^{(\tau)}$ is an optimal solution to our problem[17].

C.2 Extending our results

By direct calculations similar to those of Mobahi et al. (2020) one can obtain the closed form solution of (23), but first we will repeat some definitions from Mobahi et al. (2020). Let the Green's Function $g(x,t)$ be such that $\int_{\mathcal{X}} u(x,x')g(x',t)dx' = \delta(x-t)$, where δ is the Dirac delta, and let $[\mathbf{G}]_{j,k} = \frac{1}{N} g(\mathbf{x}_j, \mathbf{x}_k)$ and $[\mathbf{g}(\mathbf{x})]_k = \frac{1}{N} g(\mathbf{x}, \mathbf{x}_k)$, where $\mathbf{G}$ is a matrix and $\mathbf{g}(\mathbf{x})$ a vector dependent on $\mathbf{x}$. Now we can present the proposition.

Proposition C.1. *For any $\tau \geq 1$, the problem (23) has a solution of the form*

$$\mathbf{y}^{(\tau)} = \mathbf{g}(\mathbf{x})^\intercal \left(\mathbf{G} + \lambda_\tau \mathbf{I} \right)^{-1} \left(\alpha \mathbf{y} + (1-\alpha) \mathbf{y}^{(\tau-1)} \right),$$

where $\mathbf{y}^{(0)} \stackrel{\text{def}}{=} \mathbf{y}$.

Since $\mathbf{G}$ is positive semi-definite, we can decompose it as $\mathbf{G} = \mathbf{V}\mathbf{D}\mathbf{V}^\intercal$. Define $\mathbf{B}^{(0)} \stackrel{\text{def}}{=} \mathbf{I}$, then for $\tau \geq 1$, we have that $\mathbf{y}^{(\tau)} = \mathbf{V}\mathbf{B}^{(\tau)}\mathbf{V}^\intercal \mathbf{y}$, where we set

$$\mathbf{B}^{(\tau)} \stackrel{\text{def}}{=} \frac{\alpha}{1-\alpha} \sum_{i=1}^{\tau-1} (1-\alpha)^{\tau-i} \prod_{j=i}^{\tau-1} \mathbf{A}^{(j+1)} + (1-\alpha)^{\tau-1} \prod_{j=1}^{\tau} \mathbf{A}^{(j)},$$
$$\mathbf{A}^{(\tau)} \stackrel{\text{def}}{=} \mathbf{D}(\mathbf{D} + \lambda_\tau \mathbf{I})^{-1}.$$

By equivalent calculations as those of Lemma 4.2, we have that

$$\mathbf{B}^{(\tau)} = \mathbf{A}^{(\tau)}((1-\alpha)\mathbf{B}^{(\tau-1)} + \alpha \mathbf{I}_N) \quad \forall \tau \geq 1,$$

and we can now formulate the following theorem, similar to Theorem 4.3.

Theorem C.2. *For any pair of diagonals of $\mathbf{D}$, i.e. d_k and d_j, where $d_k > d_j$, we have that for all $\tau \geq 1$ and for $\alpha \in (0,1)$, then*

$$\operatorname{sign}\left(\frac{[\mathbf{B}^{(\tau)}]_{k,k}}{[\mathbf{B}^{(\tau)}]_{j,j}} - \frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}} \right) \tag{24}$$
$$= \operatorname{sign}\left(\left(\left(\frac{[\mathbf{B}^{(\tau-1)}]_{k,k}}{[\mathbf{B}^{(\tau-1)}]_{j,j}} - \frac{[\mathbf{A}^{(\tau)}]_{k,k}}{[\mathbf{A}^{(\tau)}]_{j,j}} \right) \frac{[\mathbf{A}^{(\tau)}]_{j,j}}{[\mathbf{B}^{(\tau-1)}]_{k,k}([\mathbf{A}^{(\tau)}]_{k,k} - [\mathbf{A}^{(\tau)}]_{j,j})} + 1 \right)^{-1} - \alpha \right). \tag{25}$$

Proof. The proof follows by analogous calculations to the proof of Theorem 4.3. □

For the case $\alpha = 1$, the results is identical to Theorem 4.3, since no distillation is actually performed. However, the case $\alpha = 0$ is more involved and we refer the reader to Mobahi et al. (2020) for the treatment of this case, since our setup is identical to that of Mobahi et al. (2020) when $\alpha = 0$.

Finally, due to the dependency of λ_τ on the solution from the previous step of distillation in this constrained optimization problem, we are unable to obtain a simple recurrent expression as (6) and a limiting solution as in Theorem 4.4. However, by the results in Section C.1, we can determine, from the norm of the targets, whether the solution collapses or not.

[17] See Mobahi et al. (2020) for a detailed argument.