RetroKD : Leveraging Past States for Regularizing Targets in Teacher-Student Learning

Surgan Jandial*
jandialsurgan@gmail.com
MDSR Labs, Adobe
India

Yash Khasbage*†
yashkhasbage25@gmail.com
Microsoft, India
India

Arghya Pal
arghyapal5@gmail.com
Harvard Medical School
USA

Balaji Krishnamurthy
kbalaji@adobe.com
MDSR Labs, Adobe
India

Vineeth Balasubramanian
vineethnb@iith.ac.in
IIT Hyderabad
India

ABSTRACT

Several recent works show that higher accuracy models may not be better teachers for every student, and hence, refer this problem as student-teacher "knowledge gap". Further, they propose techniques, which, in this paper, we discuss are constrained to certain pre-conditions: 1). Access to Teacher Model/Architecture 2). Retraining Teacher Model 3). Models in Addition to Teacher Model. Being well known that for a lot of settings, these conditions may not hold true challenges the applicability of such approaches.
In this work, we propose RetroKD , which smoothes out the logits of a student network by leveraging students' past state logits with the ones from the teacher. By doing so, we hypothesize that the present target will no longer be as hard as the teacher target and not as more uncomplicated as the past student target. Such regularization on learning the parameters alleviates the needs as required by other methods. Our extensive set of experiments comparing against the baselines for CIFAR 10, CIFAR 100, and TinyImageNet datasets and a theoretical study further help in supporting our claim. We performed crucial ablation studies such as hyperparameter sensitivity, the generalization study by showing the flatness on loss landscape and feature similarly with teacher network.

CCS CONCEPTS

• **Computing Methodologies** → **Machine Learning**; • **Computing Methodologies-Artificial Intelligence**;

KEYWORDS

Knowledge Distillation, Regularization, Past States

*Both authors contributed equally to this research.
†Work done while the author was student at IIT Hyderabad

CODS-COMAD 2023, January 4–7, 2023, Mumbai, India

ACM ISBN 978-1-4503-9797-1/23/01…$15.00
https://doi.org/10.1145/3570991.3571014

ACM Reference Format:
Surgan Jandial, Yash Khasbage, Arghya Pal, Balaji Krishnamurthy, and Vineeth Balasubramanian. 2023. RetroKD : Leveraging Past States for Regularizing Targets in Teacher-Student Learning. In *6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD) (CODS-COMAD 2023), January 4–7, 2023, Mumbai, India.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3570991.3571014

1 INTRODUCTION

From its inception in Hinton's seminal paper [15], Knowledge Distillation (KD) helps to compress, miniaturize, and transfer the model parameters of deeper and wider deep learning models; those otherwise require huge computational resources and time, which severely poses challenges to their deployment. The information within the high-capacity teacher network, known as "dark knowledge" [15] in common parlance, is distilled to a low-capacity student network with a view that the student network will perform similar to the teacher network but with a low-capacity. To this end, the student is trained with a compound loss that comprised of the student's logit on the main task and a KD loss to emulate the behaviour of the teacher. Choosing the appropriate hyperparameters that balance the student's logit and the KD loss seems simple, but is a crucial active area of research in KD.

The classical methods provide more emphasis to the KD loss with the belief that a well-trained teacher network having high performance on a task will eventually help to improve the performance of the student network. To some extent, this intuitively led to a belief that teacher accuracy increases with their increasing size and so is their distilling ability. Interestingly, works such as [5, 27] provide evidence that such a hypothesis, when it comes to generalization, does not always hold true. What they show is that the student network does not benefit much from the high performing teacher as (i) labels from the teacher start becoming too complex and the student being a smaller network cannot absorb knowledge from the same; and (ii) the softmax output of the teacher class logits is very high class-probability for the correct class and almost zero for other classes. This does not add much information to the student network from the class labels of the student network. This capacity difference is however referred to as the "knowledge gap" between student and teacher by [27].

To mitigate the knowledge gap, the more recent work [5] proposed to retrain the teacher model while optimizing the student network. The [20] leverages KD loss from checkpoints at different

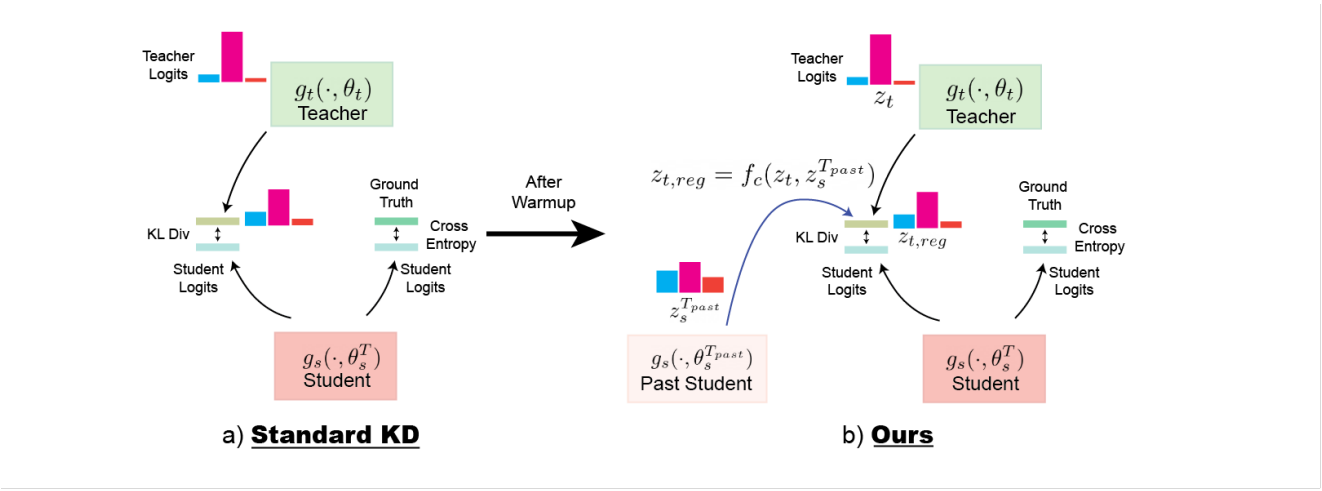

Figure 1: Visual representation for RetroKD . a)Standard KD is the framework mentioned in [15] The logits from Teacher (green box) are matched with logits of Student (red box). After T_{warmup} number of steps, we add the b) Ours (RetroKD) regularization to standard KD. The logits from Past Student (faded red box) and Teacher are composed using OCF f_c, to obtain regularized teacher logits $z_{t,reg}$.

Method	OTL	NAT	NAAM
Teacher Assistant KD [27]	✗	✓	✗
Early Stopped KD [5]	✗	✗	✓
Route Constrained Optimization [20]	✗	✗	✓
Base KD [15]	✓	✓	✓
Noisy Teacher [36]	✓	✓	✓
RetroKD (ours)	✓	✓	✓

Table 1: Comparison of proposed method with recent methods w.r.t required settings. OTL = method works with 'Only converged Teacher's Logits'; NAT = method does 'Not require Access to Teacher' (any of teacher architecture, teacher model snapshots at multiple training iterations, or requiring teacher model to retrain entirely); NAAM = method does 'Not require Access to Additional trained teacher Models'. We compare extensively against other methods that satisfy the same desiderata.

time-steps of teacher training and hence requires to save weights of different checkpoints. While [27] requires additional intermediate models to train the student leaving the crucial question of what would be the appropriate intermediate model that is capable of regularizing student. Such requirements, however, may seem unrealistic for a huge span of real-world applications - especially in the case of MLaaS (Machine-Learning-as-a-Service) [34] setting or other privacy-preserving settings [38]. In addition to these limitations, all the above methods are found to suffer from additional resource/time overhead of retraining. The mentioned setting contrast is clearly illustrated in Table 1 where we are showing three settings such as (i) methods work only when if there is access to the converged teacher logits (OTL); (ii) methods those do not require teacher access (NAT); and (iii) methods do not require additional intermediate models (NAAM). Analyzing these cons from the above discussion, one may conclude that for the settings where all that is accessible or feasible is only converged teacher's logits. Here, the aforementioned techniques [5, 20, 27] cease to work. Hence, calling for the emergence of methods to reduce the teacher-student knowledge gap with the given teacher logits only.

Following a similar motivation, we propose RetroKD, a novel yet simple method that leverages past student logits for regularizing the training with teacher logits. We hypothesize that composing the complex teacher logits with the ones from the past time steps of student can be intuitively seen as relaxing the complex training target by making it relatively similar to the student's logits while preserving the semantics from teacher targets. In other words, we can say that since the above targets contain knowledge from the past states, they are no longer as hard as the complex teacher targets. At the same time, since they contain knowledge from the teacher, they are no longer as easier as the past student. Further, we also periodically update the model state generating past state logits to allow an increase in hardness of training targets over the time steps, thereby insinuating curriculum progress in the targets while training. All the above components can be said to play their role in ensuring that the training targets are never too hard relative to the student and at the same time are periodically progressing as the student progresses. To compose previous logits with that of the teacher, we propose the use of two functions in our approach that are collectively referred to in the paper as Output Composition Functions (OCFs). Hence, RetroKD, by regularizing just with the presence of teacher's logits, alleviates any such limitations, mentioned in Table 1.

To the best of our knowledge, this is the first such work that leverages the past states of the student network in the way we do to regularize the difficulty of student in learning from teacher targets. Our contributions can be summarized as follows:

- We present a novel formulation, which we call RetroKD, that demonstrates the usefulness of leveraging the past states

of the student network in addition to the learning from the teacher targets in KD training.
- We present two strategies which we refer as the Output Composition Function (OCF) to compose teacher logits with the logits from student's past states.
- We perform detailed set of experiments on datasets such as CIFAR 10, CIFAR 100 and TinyImageNet to demonstrate the efficacy of our approach.
- We perform a comprehensive set of ablation studies including hyperparameter sensitivity studies and analysis of loss surface to study the usefulness of the proposed method.

2 RELATED WORK

Knowledge Distillation: Knowledge Distillation (KD) is a technique of transferring knowledge from one neural network to another, usually from a larger teacher network to a smaller student. First displayed by [2], and then glorified by [15], in recent years, several approaches with the aim of improving and rethinking KD have been proposed. With the initial methods predominantly focusing on distilling knowledge from logits, [35] realized that features can also be distilled, hence, proposed a two-stage training procedure to achieve the same. Succeeding on the feature distillation idea, several other variants were also proposed: [43] proposed to transfer attention maps, [42] defined the distilled knowledge from a teacher network as the flow of the solution process (FSP), [31] proposed "Relational Knowledge Distillation" that takes angle based and distance-based mutual relations between the convolutional activations to further optimize knowledge transfer. In all the above methods, the authors often use teacher networks with larger capacity than the student. Different from these, efforts such as [9, 11] utilize the same teacher-student architectures for distillation.
The methods described above perform well most of the time, however some works demonstrating the scenarios and reasons for their failures also do exist. Building on similar lines, our work proposes a regularization which, when added to the existing techniques, relaxes the difficulty of the student to learn.
The Knowledge Gap: Knowledge Distillation has often been a method in which a larger (or higher accuracy) teacher distills knowledge to a relatively smaller (or a lower accuracy) student. Having said the above notion, this also intuitively led to a belief to some extent that higher accuracy models make better teachers. Holding true for most cases, the observation has been greatly contradicted in the recent findings of [5, 27]. In some cases, due to the teacher's high certainty, the outputs may be highly devoid of class-correlations, thus becoming as hard as one-hot labels. In its own way, [25, 27] regard this phenomenon as the "knowledge gap" between student and teacher. To mitigate, [5, 20, 27] proposed their respective approaches, each of them focusing on the ways to transmit the knowledge via an alternate (usually low complexity/knowledge gap) teacher. The [27] introduced an intermediate model, which they call TA ("Teacher Assistant") model in-between converged teacher and student reducing the knowledge. Instead of introducing another model for training, [5] proposed to retrain the teacher model and stop its training early when its outputs are yet not converged. The above methods, by using relatively less complex labels demonstrate the effect of hardness on student training. With a similar direction, [20] proposed a curriculum training strategy wherein instead of using a single converged teacher, they use several intermediate checkpoints of the teacher. The detailed contrast of the same is illustrated in Table 1. Our work, on the other hand, albeit simple, alleviates the need for aforementioned constraints and performs well in the same too.

Using Information from Past States: Neural network training has always been thought of as progressive in nature, with parameters continuously evolving over the time steps. The relative difference obtained in the outputs over the time steps can be attributed to model making different mistakes at each of them. If used adequately, this information can help effectively in regularizing further training. The same has already been shown to help in several cases. [1, 6, 18] utilize labels from the previous time steps in supervised setting, [23, 39] utilize the temporal averages for consistency in semi-supervised setting whereas [4, 13] leverage the past states for visual representation learning. In the Knowledge Distillation setting, certain works like [9, 19, 41] have been found to use the model from the previous epoch/generation to train the later student. However, all the above methods use the past models as the only source of training information and hence the problem is completely different from what we or any of the previous methods ([5, 27]) describe. Hence, making RetroKD first such approach to leverage past states and regularize training with complex teacher targets.

3 METHODOLOGY

3.1 Background: Knowledge distillation

An ordinary training of Neural Network involves matching the output logits of a network z with the truth label $\hat{y}$, using the Cross-Entropy loss function.

$$\mathcal{L}_{CE} = H(softmax(z), \hat{y}) \tag{1}$$

Knowledge distillation is used to train a smaller Student Network (f_s) with the output of a large Pre-Trained Teacher Network (f_t) along with the truth labels. The outputs from the teacher (teacher labels) are seen to contain a lot of information in terms of class correlations and uncertainty, therefore forcing the student to mimic these distributions helps significantly in immersing these relationships into them. Formally, given an input image x, the output logits from student and teacher can be written as $z_s = f_s(x)$ and $z_t = f_t(x)$ respectively. The above logits are further softened via a temperature parameter (τ) and passed through softmax to obtain outputs y_s and y_t respectively:

$$y_s = softmax(z_s/\tau), y_t = softmax(z_t/\tau) \tag{2}$$

KD framework proposed adding a KD-loss to Eq. 1 for matching teacher and student logits. The KD objective L_{KD} is further written as:

$$\mathcal{L}_{KD} = \tau^2 KL(y_s, y_t) \tag{3}$$

Here, KL refers to the Kullback-Leibler Divergence and τ is the temperature parameter, as mentioned above. Combining, Eq. 1 and Eq. 3, the complete training objective can be written as

$$\mathcal{L} = \alpha\mathcal{L}_{KD} + (1-\alpha)\mathcal{L}_{CE} \tag{4}$$

Here α is the weight balancing parameter combining the individual training objectives.

3.2 RetroKD

Here, we formally describe our method. For clarity and ease of understanding, we begin by defining the notations.
We consider a teacher network f_t parameterized by θ_t as $f_t(\cdot;\theta_t)$ and a student network f_s parameterized by θ_s^T at time T as $f_s(\cdot;\theta_s^T)$. As a general notation, we use subscripts to distinguish between teacher (t) and student (s), while the superscript will denote the time-step (T) . Their outputs at time step T are taken as $z_s^T \in R^C$ and $z_t \in R^C$ respectively (R is set of real numbers and C is the number of classes):

$$z_s^T = f_s(x;\theta_s^T), z_t = f_t(x;\theta_t) \tag{5}$$

We take the past state of student at time step ($T_{past} < T$) as $f_s(\cdot;\theta_s^{T_{past}})$ and obtain the logits $z_s^{T_{past}}$:

$$z_s^{T_{past}} = f_s(x;\theta_s^{T_{past}}) \tag{6}$$

The past state logits are then combined with teacher logits via Output Composition Function (i.e., OCF, O_c) to obtain the Student-Regularized Teacher Outputs $z_{t,reg}$ as:

$$z_{t,reg} = O_c(z_t, z_s^{T_{past}};\lambda) \tag{7}$$

where λ is a hyperparameter. During training, it is very likely that outputs from very early stages of training may not be meaningful enough, thus, maligning student targets. Hence, we propose using a warmup period, T_{warmup} to introduce RetroKD after a certain number of steps of standard KD training. Now, the resultant teacher supervision (a_t) for time step T can be written as:

$$a_t = \begin{cases} z_t & T < T_{warmup} \\ z_{t,reg} & \text{otherwise} \end{cases} \tag{8}$$

Following Eq. 4, the entire training objective $\mathcal{L}$ can be written as:

$$\mathcal{L} = \alpha\mathcal{L}_{KD}(z_s^T/\tau, a_t/\tau) + (1-\alpha)\mathcal{L}_{CE}(z_s^T, \hat{y}) \tag{9}$$

Here $\hat{y}$ is the one-hot ground truth label and α is the loss balancing parameter between the two loss terms.

Composing Teacher-Past Student Logits. We suggest two kinds of OCF: 1) Interpolation and 2) Random Logit Switch [8]. Here, Interpolation refers to the standard mathematical operation of linear interpolation, while Random Switch involves choosing either of the student or teacher logits with a given probability.
Interpolation: OCF in this case is defined as

$$O_c(\mathbf{a},\mathbf{b};\lambda) = \lambda\mathbf{a} + (1-\lambda)\mathbf{b}$$

where λ is a parameter. Thus, given student's past state logits, $z_s^{T_{past}}(x)$ and teacher logits, $z_t(x)$, we take a linear interpolating factor (λ) and obtain student-regularized teacher outputs $z_{t,reg}$ as:

$$z_{t,reg}(x) = \lambda z_s^{T_{past}}(x) + (1-\lambda)z_t(x) \tag{10}$$

Random Logit Switch: OCF in this case is defined as

$$O(\mathbf{a},\mathbf{b};p_{th}) = \begin{cases} \mathbf{a}, & \beta < p_{th}, \beta \sim \mathcal{U}(0,1) \\ \mathbf{b} & otherwise \end{cases}$$

Thus, given student's past state logits $z_s^{T_{past}}(x)$ and teacher logits $z_t(x)$, we sample a number β randomly from Uniform Distribution $\mathcal{U}(0,1)$ and then based on the threshold probability β, the student-regularized teacher outputs $z_{t,reg}$ are taken as:

$$z_{t,reg} = \begin{cases} z_s^{T_{past}} & \beta < p_{th}, \beta \sim \mathcal{U}(0,1) \\ z_t & otherwise \end{cases} \tag{11}$$

Updating Past Student States: After a certain number of iterations, the student can be thought to progressively outgrow the past knowledge or be much better than the past state ($\theta_s^{T_{past}}$). Hence, we update the past state to a much recent one to subsequently advance the relative hardness of training targets. The aforementioned can also be viewed to follow a curriculum in the training target: a technique that is widely adopted to enhance network training. In our case, we follow an update frequency (f_{update}) to update the past state. However, intelligent methods to mine for the same can also be explored.

We provide a visual representation of our approach in Fig 1. Alongside this, we also present Algorithm 1 as a culmination of all the above mentioned components.

Algorithm 1 RetroKD Algorithm

Input: Current State Student Parameters θ_s^T, Teacher Parameters θ_t, Update frequency f_{update}, # of warm-up iterations T_{warmup}, learning rate η, loss scaling parameter λ, # of training iterations N.
$\theta_t^{Tpast} = NULL$
for step T=1 to N **do**
  Sample $(x, y)_{i=1}^B$, from train data
  $z_{s,i}^T = f_s(x_i;\theta_s^T)$
  $z_{t,i} = f_t(x_i;\theta_t)$
  $\mathcal{L} = \mathcal{L}_{CE}(z_{s,i}^T, y_i)$
  $a_{t,i} = z_{t,i}$
  if $step > T_{warmup}$ **then**
    $z_{s,i}^{Tpast} = f_s(x_i;\theta_s^{Tpast})$
    Using Eq 7 to get $z_{t,reg,i}$
    $a_{t,i} = z_{t,reg,i}$
  end if
  $\mathcal{L} = \alpha\mathcal{L}_{KD}(z_{s,i}^T/\tau, a_{t,i}/\tau) + (1-\alpha)\mathcal{L}_{CE}(z_{s,i}^T, \hat{y}_i)$
  $\theta_s^{T+1} \leftarrow \theta_s^T - \eta\nabla\mathcal{L}_{\theta_s^T}(x_i, y_i;\theta_s^T)$
  if T % f_{update} == 0 **then**
    $\theta_s^{Tpast} \leftarrow \theta_s^T$
  end if
end for

4 EXPERIMENTS AND RESULTS

Baselines: We compare against Base Knowledge Distillation (**BKD**) [15] and Distillation with Noisy Teacher (**NT**) on CIFAR-10, CIFAR-100 and TinyImageNet datasets. Each of the baseline caters to similar limitations and setting as ours (described in Table 1) . **BKD** [15] is the standard knowledge distillation approach as described in Sec. 2. **NT** [36] adds random noise to the logits and seeks to produce an

Dataset	Teacher	# params	BKD	NT	Ours (interpol)	Ours (switch)
CIFAR-10	CNN-4	37k	70.94	71.22	**72.08**	71.83
	CNN-8	328k	72.50	72.46	72.67	**72.83**
	CNN-10	2.48M	72.51	72.62	**73.17**	72.84
	ResNet-20	270k	86.58	86.48	86.70	**86.71**
	ResNet-32	464k	86.53	86.57	**86.74**	86.71
	ResNet-56	853k	86.43	86.49	86.55	**86.63**
CIFAR-100	CNN-4	476k	51.50	51.60	51.70	**51.77**
	CNN-8	1.24M	51.30	51.56	51.50	**51.57**
	CNN-10	2.93M	51.39	51.70	51.67	**51.83**
	ResNet-20	276k	56.86	56.35	**57.37**	57.33
	ResNet-32	470k	57.05	**57.24**	56.98	57.22
	ResNet-56	859k	56.45	56.67	**57.19**	57.04
TinyImageNet	ResNet-20	282k	37.44	37.59	37.74	**37.94**
	ResNet-32	477k	37.28	37.49	**38.02**	37.61
	ResNet-56	865k	37.61	37.46	37.60	**37.76**

Table 2: Accuracy (higher values are better) comparison against the baselines on CIFAR-10/100 and TinyImageNet dataset. Here, bold font denotes the values for best performing method while underline denotes the second-best.

ensemble effect to regularize the training.
In contrast to adding random noise, our method seeks to compose the teacher logits with a semantically consistent disturbance taken from the model's past steps. For this, we employ two techniques as described in Sec. 3.2 and in the rest of the paper we refer Interpolation as **Ours (Interpol)** whereas Random Logit Switch as **Ours (Switch)**.

Network Architectures: Following [27], we consider similar models for experimentation, i.e., VGG-like architectures, referred to as Plain CNNs and the ResNets. Plain CNNs consist of a series of standard convolutional layers (usually followed by max-pooling and/or batchnorm) and finally end in a fully connected layer. ResNets, on the other hand, follow the standard architecture as described in [14]. In each of the experiments, the choice of the student is always kept as CNN-2 in the case of Plain CNNs, whereas ResNet-8 in the case of ResNets. Correspondingly, for teachers one of CNN-{4,8,10} or ResNet-{20,32,56} is chosen.

Evaluation Metrics: We use Accuracy **(Acc)** as our evaluation metric. Table 2 shows a detailed comparison against the baselines. For a consistent comparison, we report mean over 3 runs for each method.

Implementation: We used PyTorch [32] for implementing the models/training routines and use a single single-precision GPU (NVIDIA GTX 1080Ti) having 12 GB RAM for experimentation. The input images are pre-processed with normalization and augmented with a random crop of padding 4 followed by random horizontal flips. In each of the experiments, parameter update happens with a standard SGD optimizer having a Nesterov momentum of 0.9. We now describe the dataset/model specific training settings for each. For ResNets on CIFAR-10/100, we train for total 100 epochs with an initial learning rate $1e-2$ that is divided by factor of 10 at 50^{th} and 80^{th} epochs whereas Plain-CNNs on CIFAR-10/100 are trained for 80 epochs with an initial learning rate $1e-1$ being divided by a factor of 10 at the 40^{th} epoch. For ResNets on TinyImageNet, we train for a total of 50 epochs with an initial learning rate $1e-2$ divided by factor of 10 at 25^{th} and 40^{th} epochs respectively. We set τ (as described in Eq. 4) for all the experiments as 4 and α as 0.9 for CIFAR-10 and 0.2 for CIFAR-100/TinyImageNet ([10, 30]). In each of the experiments, the hyperparameters $f_{update} \in \{1, 2, 3, 4\}$, $T_{warmup} \in \{25, 30, 40, 50, 60\}$, $\lambda \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$ and $p_{th} \in \{0.2, 0.45, 0.5, 0.7\}$. Further, to understand these choices better, we study the following parameters in Sec. 5.
Result Discussion: Table 2 summarizes the results of our experiments. Firstly, consistent to [27], for all of our datasets (CIFAR-10/100 and TinyImageNet), we also observe that increasing teacher size does not necessarily increase the student performance. Now, following the baselines as mentioned in Table 1 (see lower half) , we notice that adding random noise to the logits and producing an ensemble-like effect [36] does help a lot of times over standard-BKD, however, cases pertaining to under performance w.r.t BKD still do exist. This can be accorded to the maligned and uncertain behaviour of targets obtained as the result of adding noise while training. Further coming to our method, we can observe that either by achieving a higher accuracy, one of the configurations of RetroKD consistently outperforms the compared baselines.

5 ABLATIONS AND ANALYSIS

5.1 Analysis of the Student Regularized KD

In Sec. 1 we hypothesize that, composing the complex teacher logits with the ones from the past time steps of student can be intuitively seen as relaxing the complex training target by making it relatively similar to the student's logits while preserving the semantics from the teacher targets. In this section, we study this in a more theoretical way. First, we revisit the vanilla knowledge distillation Eqn. 4.

In NOKD situation where there is no teacher network, i.e. $\alpha = 1$ in Eqn. 4, and the student network $f_s \in \mathcal{F}_s$ with capacity $|\mathcal{F}_s|_C$ is learning the real target function $f \in \mathcal{F}$ using cross entropy loss. We use the VC theory [40] in the NOKD framework [15] and show

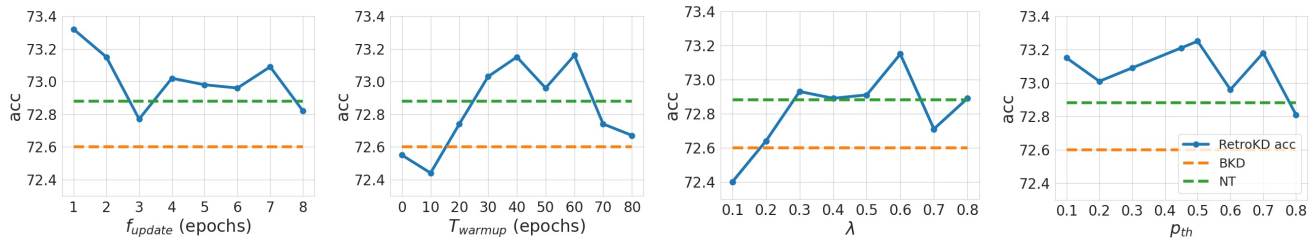

Figure 2: Hyperparameter sensitivity for f_{update}, T_{warmup}, λ and p_{th}. Student=CNN2, Teacher=CNN10 trained on CIFAR-10.

Dataset	RKD	RKD+RetroKD	AT	AT+RetroKD	NST	NST+RetroKD
CIFAR-10	86.42	**87.17**	86.61	**86.80**	86.79	**87.13**
CIFAR-100	56.95	**57.01**	57.78	**58.05**	57.05	**57.52**
TinyImageNet	37.9	**38.11**	39.05	**39.36**	37.97	**38.10**

Table 3: Accuracies before and after applying RetroKD to the feature distillation methods: Relational KD (RKD), Attention Transfer (AT) and Neuron Selectivity Transfer (NST). Bold font denotes the best accuracy.

the generalization bound of a student network, i.e.;

$$R(f_s) - R(f) \leq O\Big(\frac{|\mathcal{F}_s|_C}{n^{\zeta^s}}\Big) + \epsilon_s \tag{12}$$

where, n is the number of data point and the $\frac{1}{2} \leq \zeta^s \leq 1$ is the rate of learning [1] by which the student f_s learns the true function f. The $O(\cdot)$ is the estimation error [2] and the ϵ_s is the approximation error [3] of the student function class $\mathcal{F}_s$ w.r.t real class $f \in \mathcal{F}$. Similar to [26, 27, 37] we only discuss estimation error in terms of upper bound. We shall consider the approximation error to understand the generalization of any hypothesis function f.

The BaselineKD, i.e. BKD, leverages knowledge from *both* the teacher network $f_t \in \mathcal{F}_t$ and the learning from cross entropy and it is easy to note that;

$$\begin{aligned} R(f_s) - R(f) = & \underbrace{R(f_s) - R(f_t)}_{\text{Distillation from Teacher}} + \underbrace{R(f_t) - R(f)}_{\text{Teacher Error}} \\ \leq & \Big[O\Big(\frac{|\mathcal{F}_t|_C}{n^{\zeta^t}} + \frac{|\mathcal{F}_s|_C}{n^{\zeta^t}}\Big) + \epsilon_t + \epsilon_l\Big]\Big] \leq \Big[O\Big(\frac{|\mathcal{F}_s|_C}{\sqrt{n}}\Big) + \epsilon_s\Big] \end{aligned} \tag{13}$$

As the student that has a low capacity, i.e. $|\mathcal{F}_s|_C \ll |\mathcal{F}_t|_C$, that learns the real target function $f \in \mathcal{F}$ at a slow rate of learning (i.e. $\frac{1}{2} \leq \zeta^s$). Similar to the argument in [26], the teacher is a high capacity network with a near 1 rate of learning, i.e. $\zeta_t = 1$, and the ζ_s is in between $\frac{1}{2}$ and the 1, i.e. $\frac{1}{2} \leq \zeta_s \leq 1$, as it is easy to approximate the teacher f_t with a student f_s than the real function f. However, the $\zeta_s = \frac{1}{2}$ if the student is to approximate the real function f and hence we see the $n^{\zeta_s} = \sqrt{n}$ in the R.H.S of the inequality. To conclude, the inequality highlights the benefits of learning a low capacity student network with a teacher, that is, it *helps to generalize a student network* better than learning the student network alone, i.e. $(\epsilon_t + \epsilon_l) \ll \epsilon_s$ from Eqn. 13.

In RetroKD, without the loss of generality, we leverage the student regularization (temporal regularization) and show how it improves the generalization bound. Assuming the past student is $f_{\hat{s}} \in \mathcal{F}_s$, we can write the following inequality;

$$R(f_s) - R(f_{\hat{s}}) \leq O\Big(\frac{|\mathcal{F}_s|_C}{n^{\zeta^{\hat{s}}}}\Big) + \epsilon_{\hat{s}} \tag{14}$$

One may ask, *how $\epsilon_{\hat{s}}$ is helping to minimize approximation error*? To answer this, we refer to the theoretical result of Bartlett *et al.* [28] that assumes to learn;

$$f_s^* \equiv \arg\min_{f \in \mathcal{F}} R(f) \quad \text{s.t.} \frac{1}{K}\sum_k (f(x_k) - y_k)^2 \leq \epsilon \tag{15}$$

where, $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ is the space of all admissible functions from where we learn f_s^*. The finite dataset $\mathcal{D} \equiv \{x_k, y_k\}$ has K number of training points $k = \{1, 2, \cdots, K\}$ and $\epsilon > 0$ is a desired loss tolerance. Without the loss of generality the Eqn. 15 can be written as;

$$\begin{aligned} f_s^* = & \arg\min_{f \in \mathcal{F}} \frac{1}{K}\sum_k (f(x_k) - y_k)^2 \\ & + c\int_{\mathcal{X}}\int_{\mathcal{X}} u(x, x^\dagger) f(x) f(x^\dagger) dx dx^\dagger \end{aligned} \tag{16}$$

with $u(\cdot)$ implying that $\forall f_s \in \mathcal{F}$ the $R(f) > 0$ with equality when $f_s(x) = 0$ and the the $c > 0$ [28, 44]. Further, the Eqn. 16 can be written as a closed form;

$$f_s^*(x) = \mathbf{g}_x^T (cI + G)^{-1} y \tag{17}$$

where, $\mathbf{g}_x[k] \equiv \frac{1}{K} g(x, x_k)$, $G[j, k] \equiv \frac{1}{K} g(x_j, x_k)$, and $g(\cdot)$ is called Green's function [7]. The matrix G is positive definite and can be represented as $G = V^T D V$, the diagonal matrix D contains the eigenvalues and V comprised of eigenvectores. From the proof of [28] we can show that at time t of the student network f_s, i.e.;

[1] **Rate of Learning:** The rate at which the function $\hat{f}$ learns the true function f, where $\hat{f}$ can be a teacher network or a student network. For non-separable problems (*a.k.a* difficult problems) the exponent $\zeta = \frac{1}{2}$, meaning that, we need more data points to approximate true function f with some accuracy. On the other hand, for separable problems (*a.k.a* easy problems) the $\zeta = 1$ and we may require few data points to approximate true function.

[2] **Estimation Error**: Given $\mathcal{F}$ the estimation error is the minimum generalization error by f

[3] **Approximation Error**: Typically, the difference between the approximation error and the error achieved by the predictor in the hypothesis class minimizing the training error.

$f_{s,t} = \mathbf{g}_x^T(cI+G)^{-1}y_t = \mathbf{g}_x^T V^T D(c_t I + D)^{-1} V y_{t-1}$ can benefit from the previous round's $t-1$ knowledge distillation. Such self distillation sparsifies $(cI+G)^{-1}$ at a given rate, thus ensuring the progressively limiting the number of basis function that acts as a good regularizer [28].

As a consequence, we can write similar to Eqn. 13;

$$\begin{aligned} R(f_s) - R(f) \\ &= \underbrace{R(f_s) - R(f_{\hat{s}})}_{\text{Distillation from Past State}} + \underbrace{R(f_{\hat{s}}) - R(f_t)}_{\text{Distill from Teacher}} + \underbrace{R(f_t) - R(f)}_{\text{Teacher Error}} \\ &\leq O\Big(\frac{|\mathcal{F}_s|_C}{n^{\zeta^{\hat{s}}}} + \frac{|\mathcal{F}_t|_C}{n^{\zeta^t}} + \frac{|\mathcal{F}_s|_C}{n^{\zeta^l}}\Big) + \epsilon_t + \epsilon_l + \epsilon_{\hat{s}} \end{aligned} \tag{18}$$

Please note that, in Eqn. 18, the risk associated with past state $R(f_{\hat{s}})$ is asymptotically equivalent to the present state student $R(f_s)$. Therefore, we note that;

$$\begin{aligned} &O\Big(\frac{|\mathcal{F}_s|_C}{n^{\zeta^{\hat{s}}}} + \frac{|\mathcal{F}_t|_C}{n^{\zeta^t}} + \frac{|\mathcal{F}_s|_C}{n^{\zeta^l}}\Big) + \epsilon_t + \epsilon_l + \epsilon_{\hat{s}} \\ &\leq O\Big(\frac{|\mathcal{F}_t|_C}{n^{\zeta^t}} + \frac{|\mathcal{F}_s|_C}{n^{\zeta^l}}\Big) + \epsilon_t + \epsilon_l \leq O\Big(\frac{|\mathcal{F}_s|_C}{\sqrt{n}}\Big) + \epsilon_s \end{aligned} \tag{19}$$

According to Bartlett *et al.* [28], the approximation error $\epsilon_{\hat{s}}$ helps to reduce the training error in conjunction with the $\epsilon_t + \epsilon_l$, and hence we can say, $\epsilon_t + \epsilon_l + \epsilon_{\hat{s}} \leq \epsilon_t + \epsilon_l \ll \epsilon_s$, which means that the upper bound of error in RetroKD is smaller than its upper bound in BKD and NOKD, when $n \to \infty$. Even in the finite range, when the capacity of $|\mathcal{F}_t|_c$ is larger than $|\mathcal{F}_s|_c$ and the student network is distilling from its past state the RetroKD still works, pls. see Table 4 similarity column.

5.2 Hyperparameter Sensitivity

Here, we explore the hyperparameter sensitivity for RetroKD , when CNN-2 is distilled from CNN-10, on CIFAR-10. Figure 2 displays variations with respect to different hyperparameters choices. For T_{warmup}, it can be seen that introducing RetroKD too early or too late can hurt the performance. For too early, we believe it may be due to lesser semantic information in the labels. Whereas for the too late, we believe it may be due to lesser training time available for RetroKD to regularize. For λ and p_{th}, we can observe that the best performance is achieved somewhere in the middle rather than any of the extremes, which can be intuitively explained as logits exhibiting either behaviour of past state or teacher on the extremes. For f_{update}, the performance consistently decreases on increasing its value, thereby suggesting that updating the past state too late affects the performance in our case.

5.3 Similarity with Teacher Features

RetroKD explicitly modifies the logits, and its improvement is evident from Table 2. However, how RetroKD affects the internal representations is not trivial to guess. Since every KD method aims at obtaining knowledge from the teacher, we can expect a better student to be more similar to the teacher because it was able to acquire better knowledge from the same. Looking at the superiority of RetroKD from Table 2, one can expect that teacher features should be more similar to RetroKD features, rather than BKD features. There have been recent works like [22] and [29] that aim at providing metrics to evaluate feature similarity of neural network representations. Such metrics can be used in computing feature similarity between teacher and student networks. Intuitively, we expect that a better distillation method should yield more feature similarity between teacher and student. For our analysis, we use the Linear-CKA metric proposed by [22] for comparing the similarity of representations and compute Linear-CKA similarity between student models trained using BKD and RetroKD for the convolutional features. We compute CKA on 20K samples (40% of total) from the train set of CIFAR-10. The results are shown in Table 4 (see left half).

Student	Teacher	Similarity (↑)		Sharpness (↓)	
		BKD	RetroKD	BKD	RetroKD
CNN-2	CNN-4	0.1656	**0.1728**	**338.82**	380.36
	CNN-8	0.2469	**0.2658**	733.68	**681.33**
	CNN-10	**0.2228**	0.2217	763.38	**722.82**
ResNet-8	ResNet-20	0.7334	**0.7355**	611.73	**551.83**
	ResNet-32	0.6771	**0.6823**	**620.69**	696.60
	ResNet-56	0.6461	**0.6615**	751.54	**613.12**

Table 4: Similarity and Sharpness computed for CNN-2 student trained with CNN-4/8/10 teacher and ResNet-8 student trained with ResNet-20/32/56 teacher, on CIFAR-10 dataset. Similarity is higher for RetroKD , indicating better knowledge transfer from teacher. Sharpness is smaller for RetroKD , justifying the better generalization with RetroKD . Bold font indicates better values.

Almost all students trained using RetroKD , were found to have features more similar to teacher, as compared to student trained with BKD. This clearly indicates that RetroKD helps in learning better representations.

5.4 Flatness of Solution

In several recent works [3, 12, 16, 21, 24], it has been empirically observed that Neural Networks with better generalization have flatter converged solutions. However, this concept was heavily verified on a wide range of deep networks, with the help of visualization proposed by [24]. Interestingly in [45], the authors reason the success of their method can be related to entropy-regularization based approaches, where the goal is to reach a flatter minima. The visualization provided by [24] plots the loss along with two random orthogonal directions in the parameter space, given a point in parameter space, e.g., the converged weights. Considering the better generalizing solution of RetroKD , the landscape of RetroKD should be flatter. We plot the landscape when ResNet is trained with KD [15] (BKD) and RetroKD . The plots in Fig 3 clearly indicate that RetroKD solutions posses flatter minima.

Contrastive to the plot based visualization, a point estimate to the flatness was provided by [33]. The measure was termed as *sharpness* and can be considered as opposite to flatness. We compute sharpness over 2000 random train samples (4% of the total dataset) from the CIFAR-10 dataset for the students CNN-2 and ResNet-8 and report the numbers in Table 4 (see right half). In most of the

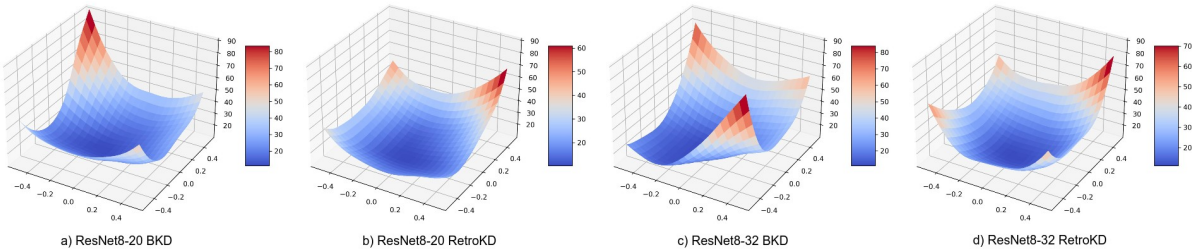

Figure 3: Loss landscape for a) student=ResNet-8, teacher= ResNet-20 (BKD), b) student=ResNet-8, teacher=ResNet-20 (RetroKD) c) student=ResNet-8, teacher=ResNet-32 (BKD), d) student=ResNet-8, teacher=ResNet-32 (RetroKD)

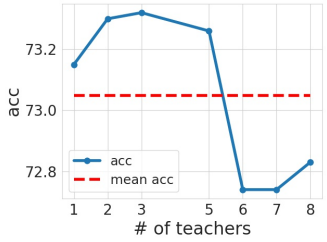

Figure 4: Performance vs. number of past models used

cases, RetroKD was found to be more flatter than BKD, justifying the flatter convergence.

5.5 Adding RetroKD to Other Feature-KD

As described in Sec. 2, the KD framework has evolved over time and one peculiar set of these methods is feature distillation. Broadly, feature distillation differs from the standard KD framework in using extra knowledge from teacher features of the teacher. In most of these methods, distillation happens using features alongside teacher logits rather than logits alone. Thus, it is possible to question if RetroKD can be used to improve the logit-distillation component in feature distillation. By virtue of its simplicity, we believe RetroKD can be easily applied to any such feature distillation method. Therefore, as an exploratory analysis, here, we consider 3 popular feature distillation methods i.e., **RKD**[31], **NST**[17], **AT**[43] and apply RetroKD to the logit component of the same. To describe the above methods briefly: **RKD**[31] maintains the angle and distance-based similarity relations between the convolutional features across examples. **NST**[17] poses knowledge transfer as a distribution matching problem and match the neuron selectivity pattern via Maximum Mean Discrepancy loss, **AT**[43] on the other hand shows that activation based attention transfer is better than full-activation transfer. For the aforementioned experiments, we use ResNet-20 as the teacher and ResNet-8 as the student, each of which follows the same number of epochs, learning rate schedule as described in Sec. 4. We report the maximum accuracy obtained from both kinds of OCFs. The results reported in Table 3 show encouraging improvements, thereby motivating further study of RetroKD with such approaches.

6 CONCLUSIONS AND FUTURE WORK

In this work, we specifically focus on regularizing student learning from complex teacher targets. We discussed several recent approaches and pointed out the limitations constraining each of them, mentioned in Table 1. Alleviating the above limitations, we then focus on the methods that help regularize the student training in the presence of teacher logits only. Motivated by the same, we then proposed RetroKD, a novel technique that utilizes the past student for relatively relaxing the hardness of complex teacher targets while training the student in KD. Our extensive experiments demonstrate the effectiveness of RetroKD. With the objective of understanding deeper into our method, we conduct ablation studies on several hyperparameter choices and diverse analysis focusing on different aspects of the performance.

REFERENCES

[1] Armen Aghajanyan. 2016. SoftTarget Regularization: An Effective Technique to Reduce Over-Fitting in Neural Networks. arXiv:arXiv:1609.06693
[2] Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model Compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*. Association for Computing Machinery, 535–541.
[3] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. [n. d.]. Entropy-SGD: Biasing Gradient Descent Into Wide Valleys. In *International Conference on Learning Representations, ICLR 2017*.
[4] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020. *Improved Baselines with Momentum Contrastive Learning*. Technical Report.
[5] J. H. Cho and B. Hariharan. 2019. On the Efficacy of Knowledge Distillation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 4793–4801.
[6] Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Shu-Tao Xia. 2019. Adaptive Regularization of Labels. arXiv:arXiv:1908.05474
[7] Hubert Ebert, Diemo Koedderitzsch, and Jan Minar. 2011. Calculating condensed matter properties using the KKR-Green's function method—recent developments and applications. *Reports on Progress in Physics* 74, 9 (2011), 096501.
[8] T. Fukuda, Masayuki Suzuki, Gakuto Kurata, S. Thomas, Jia Cui, and B. Ramabhadran. 2017. Efficient Knowledge Distillation from an Ensemble of Teachers. In *INTERSPEECH*.
[9] Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born-Again Neural Networks. In *International Conference on Machine Learning, ICML 2018 (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 1602–1611.
[10] Mengya Gao, Yujun Shen, Quanquan Li, Junjie Yan, Liang Wan, Dahua Lin, Chen Change Loy, and Xiaoou Tang. 2018. An Embarrassingly Simple Approach for Knowledge Distillation. arXiv:arXiv:1812.01819
[11] Sangchul Hahn and Heeyoul Choi. 2019. Self-Knowledge Distillation in Natural Language Processing. In *RANLP*.
[12] Haowei He, Gao Huang, and Yang Yuan. 2019. Asymmetric Valleys: Beyond Sharp and Flat Local Minima. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc., 2553–2564. https://proceedings.neurips.cc/paper/2019/file/01d8bae291b1e4724443375634ccfa0e-Paper.pdf

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[15] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. In *NIPS Deep Learning and Representation Learning Workshop*. http://arxiv.org/abs/1503.02531

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Flat Minima. *Neural Comput.* 9, 1 (Jan. 1997), 1–42.

[17] Z. Huang and Naiyan Wang. 2017. *Like What You Like: Knowledge Distill via Neuron Selectivity Transfer.* Technical Report.

[18] Surgan Jandial, Ayush Chopra, Mausoom Sarkar, Piyush Gupta, Balaji Krishnamurthy, and Vineeth Balasubramanian. 2020. Retrospective Loss: Looking Back to Improve Training of Deep Neural Networks *(KDD '20)*. Association for Computing Machinery, 9 pages. https://doi.org/10.1145/3394486.3403165

[19] Liang Jiang, Zujie Wen, Zhongping Liang, Yafang Wang, Gerard de Melo, Zhe Li, Liangzhuang Ma, Jiaxing Zhang, Xiaolong Li, and Yuan Qi. 2020. Long Short-Term Sample Distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. arXiv:arXiv:2003.00739

[20] Xiao Jin, Baoyun Peng, Yichao Wu, Yu Liu, Jiaheng Liu, Ding Liang, Junjie Yan, and Xiaolin Hu. 2019. Knowledge Distillation via Route Constrained Optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[21] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[22] Simon Kornblith, Mohammad Norouzi, H. Lee, and Geoffrey E. Hinton. 2019. Similarity of Neural Network Representations Revisited. In *International Conference on Machine Learning, ICML 2019*.

[23] Samuli Laine and Timo Aila. [n. d.]. Temporal Ensembling for Semi-Supervised Learning. In *International Conference on Learning Representations, ICLR 2017*. arXiv:arXiv:1610.02242

[24] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the Loss Landscape of Neural Nets. In *Advances in Neural Information Processing Systems*.

[25] Xuewei Li, Songyuan Li, Bourahla Omar, Fei Wu, and Xi Li. 2021. ResKD: Residual-Guided Knowledge Distillation. *IEEE Transactions on Image Processing* (2021).

[26] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. 2015. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643* (2015).

[27] Seyed Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2020. Improved Knowledge Distillation via Teacher Assistant. *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (04 2020), 5191–5198. https://doi.org/10.1609/aaai.v34i04.5963

[28] Hossein Mobahi, Mehrdad Farajtabar, and Peter Bartlett. 2020. Self-Distillation Amplifies Regularization in Hilbert Space. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 3351–3361. https://proceedings.neurips.cc/paper/2020/file/2288f691b58edecadcc9a8691762b4fd-Paper.pdf

[29] Ari S. Morcos, M. Raghu, and S. Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems*.

[30] Hideki Oki, Motoshi Abe, Junichi Miyao, and Takio Kurita. 2020. Triplet Loss for Knowledge Distillation. In *International Joint Conference on Neural Networks (IJCNN), 2020*. arXiv:arXiv:2004.08116

[31] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. 2019. Relational Knowledge Distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3967–3976.

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*. 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[33] Akshay Rangamani, Nam H. Nguyen, Abhishek Kumar, D. Phan, S. H. Chin, and Trac D. Tran. 2019. A Scale Invariant Flatness Measure for Deep Network Minima. *ArXiv* abs/1902.02434 (2019).

[34] M. Ribeiro, K. Grolinger, and M. A. M. Capretz. 2015. MLaaS: Machine Learning as a Service. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. 896–902. https://doi.org/10.1109/ICMLA.2015.152

[35] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. [n. d.]. FitNets: Hints for Thin Deep Nets. In *International Conference on Learning Representations, ICLR 2015*, Yoshua Bengio and Yann LeCun (Eds.).

[36] Bharat Bhusan Sau and Vineeth N Balasubramanian. 2016. Deep model compression: Distilling knowledge from noisy teachers. *arXiv preprint arXiv:1610.09650* (2016).

[37] Shai Shalev-Shwartz and Nathan Srebro. 2008. SVM optimization: inverse dependence on training set size. In *Proceedings of the 25th international conference on Machine learning*. 928–935.

[38] R. Shokri and V. Shmatikov. 2015. Privacy-preserving deep learning. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 909–910.

[39] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*. arXiv:arXiv:1703.01780

[40] V Vapnik. 1998. Statistical learning theory new york. *NY: Wiley* (1998).

[41] Chenglin Yang, Lingxi Xie, Chi Su, and A. Yuille. 2019. Snapshot Distillation: Teacher-Student Optimization in One Generation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 2854–2863.

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

[43] Sergey Zagoruyko and Nikos Komodakis. [n. d.]. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. In *International Conference on Learning Representations, ICLR 2017*. https://arxiv.org/abs/1612.03928

[44] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM* 64, 3 (2021), 107–115.

[45] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. 2018. Deep Mutual Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4320–4328. https://doi.org/10.1109/CVPR.2018.00454