Snapshot Distillation: Teacher-Student Optimization in One Generation

Chenglin Yang[1], Lingxi Xie[1,2](✉), Chi Su[3], Alan L. Yuille[1]
[1]Johns Hopkins University [2]Noah's Ark Lab, Huawei Inc. [3]Kingsoft Cloud
{chenglin.yangw,198808xc,alan.l.yuille}@gmail.com suchi@kingsoft.com

Abstract

Optimizing a deep neural network is a fundamental task in computer vision, yet direct training methods often suffer from over-fitting. Teacher-student optimization aims at providing complementary cues from a model trained previously, but these approaches are often considerably slow due to the pipeline of training a few generations in sequence, i.e., time complexity is increased by several times.

This paper presents snapshot distillation (SD), the first framework which enables teacher-student optimization in one generation. The idea of SD is very simple: instead of borrowing supervision signals from previous generations, we extract such information from earlier epochs in the same generation, meanwhile make sure that the difference between teacher and student is sufficiently large so as to prevent under-fitting. To achieve this goal, we implement SD in a cyclic learning rate policy, in which the last snapshot of each cycle is used as the teacher for all iterations in the next cycle, and the teacher signal is smoothed to provide richer information. In standard image classification benchmarks such as CIFAR100 and ILSVRC2012, SD achieves consistent accuracy gain without heavy computational overheads. We also verify that models pre-trained with SD transfers well to object detection and semantic segmentation in the PascalVOC dataset.

1. Introduction

A large portion of recent advances in computer vision have been built upon deep learning, in particular training very deep neural networks. With the depth increasing from tens [25, 37, 40] to hundreds [18, 22], the issue of the network optimization becomes more and more important yet challenging. As such, researchers proposed various approaches to deal with both under-fitting [30], over-fitting [39] and numerical instability [23].

As an alternative approach to assist training, teacher-student (T-S) optimization was originally designed for training a smaller network to approximate the behavior of a larger one, *i.e.*, model compression [19], but later researchers found its effectiveness in providing complementary cues to training the same network [11, 2]. These approaches require a teacher model which is often obtained from a standalone training process. Then, an extra loss term which measures the similarity between the teacher and the student is added to the existing cross-entropy loss term. It was believed that such an optimization process benefits from so-called *secondary information* [49], *i.e.*, class-level similarity that allows the student not to fit the one-hot class distribution. Despite their success in improving recognition accuracy, these approaches often suffer much heavier computational overheads, because a sequence of models need to be optimized one by one. A training process with one teacher and K students requires $K\times$ more training time compared to a single model.

	SA?	IN?	1G?
Knowledge Distillation (2015) [19]			
FitNet (2015) [35]			
Net2Net (2016) [5]		✓	
A Gift from KD (2017) [50]			
Label Refinery (2018) [2]	✓	✓	
Born-Again Network (2018) [11]	✓		
Tolerant Teacher (2018) [49]	✓	✓	
Snapshot Distillation (this work)	✓	✓	✓

Table 1. The attributes of different teacher-student optimization approaches, where SA indicates that teacher and student have the *same architecture*, IN indicates being evaluated on *ImageNet*, and 1G indicates that the entire process is done within *one generation*. See Section 2 for a detailed survey.

This paper presents an algorithm named **snapshot distillation** (SD) to perform T-S optimization *in one generation* which, to the best of our knowledge, was not achieved in prior research. The differences between SD and previous methods are summarized in Table 1. The key idea of SD is straightforward: taking extra supervision (*a.k.a.* the teacher signal) from the prior *iterations* (in the same generation) instead of the prior *generations*. Based on this framework, we investigate several factors that impact the performance of T-S optimization, and summarize three principles, namely, (i) the teacher model has been well optimized; (ii) the teacher and student models are sufficiently different from each other; and (iii) the teacher provides secondary information [49] for the student to learn. Summarizing these requirements leads to our solution that using a cyclic learning rate policy, in which the last snapshot of each cycle (which arrives at a high accuracy and thus satisfies (i)), serves as the teacher for all iterations in the next cycle (these iterations are pulled away from the teacher after a learning rate boost, which satisfies (ii)). We also introduce a novel method to smooth the teacher signal in order to provide mild and more effective supervision (which satisfies (iii)).

Experiments are performed in two standard benchmarks for image classification, namely, CIFAR100 [24] and ILSVRC2012 [36]. SD consistently outperforms the baseline (direct optimization) especially in deeper networks. In addition, SD requires merely less than $1/3$ extra training time beyond the baselines (see Section 3.3.4 for details), which is K times faster than the existing K-multi-generation approaches [11, 49, 2], theoretically and practically. We also fine-tune the models trained by SD for object detection and semantic segmentation in the PascalVOC dataset [10] and observe accuracy gain, implying that the improvement brought by SD is transferrable.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 describes snapshot distillation and provides practical guides for T-S optimization in one generation. After experiments are shown in Section 4, we conclude this work in Section 5.

2. Related Work

Recently, computer vision research has been largely boosted by deep learning [26]. With the availability of large-scale datasets [7] and powerful computational resources, researchers designed deep networks to replace traditional handcrafted features [32] for visual recognition. The fundamental idea is to build a hierarchical network structure containing multiple *layers*, each of which contains a number of *neurons* having the same or similar mathematical functions, *e.g.*, convolution, pooling, normalization, *etc*. The strong ability of deep networks at fitting complicated feature-space distributions is widely verified in the previous literature. In a fundamental task known as image classification, deep convolutional neural networks [25] have been dominating in the large-scale competitions [36]. To further improve classification accuracy, researchers designed even deeper networks [37, 40, 18, 22, 20], and also explored the possibility of discovering network architectures automatically [46, 57, 27].

The rapid progress of deep neural networks has helped a lot of visual recognition tasks. Features extracted from pre-trained classification networks can be transferred to small datasets for image classification [8], retrieval [33] or object detection [14]. To transfer knowledge to a wider range of tasks, researchers often adopt a technique named fine-tuning, which replaces the last few layers of a classification network with some specially-designed modules (*e.g.*, up-sampling for semantic segmentation [28, 3] and edge detection [48] or regional proposal extraction for object detection [13, 34]), so that the network can take advantage of the properties of the target problem while borrowing visual features from basic classification.

On the other hand, optimizing a deep neural network is a challenging problem. When the number of layers becomes very large (*e.g.*, more than 100 layers), vanilla gradient descent approaches often encounter stability issues and/or over-fitting. To deal with them, researchers designed various approaches such as ReLU activation [30], Dropout [39] and batch normalization [23]. However, as depth increases, the large number of parameters makes it easy for the neural networks to be over-confident [15], especially in the scenarios of limited training data. An effective way is to introduce extra *priors* or *biases* to constrain the training process. A popular example is to assume that some visual categories are more similar than others [6], so that a class-level similarity matrix is added to the loss function [43, 45]. However, this method still suffers the lack of modeling per-image class-level similarity (*e.g.*, a *cat* in one image may look like a *dog*, but in another image, it may be closer to a *rabbit*), which is observed in previous research [44, 1, 52].

Teacher-student optimization is an effective way to formulate per-image class-level similarity. In this flowchart, a teacher network is first trained, and then used to guide the student network, so that class-level similarities for each image are delivered by the teacher's output (*e.g.*, confidence scores). This idea was first proposed to distill knowledge from a larger teacher network and compress it to a smaller student network [19, 35], or initialize a deeper/wider network with pre-trained weights of a shallower/narrower network [5, 37]. Later, it was extended in various aspects, including using an adjusted way of teacher supervision [41, 31], using multiple teachers towards a better guidance [42], adding supervision to intermediate neural responses [50], and allowing two networks to provide supervision to each other [55]. Recently, researchers noted that this idea can be used to optimize deep networks in many *generations* [2, 11], namely, a few networks with *the same architecture* are optimized one by one, in which the next one borrows supervision from the previous one. It was argued that the *softness* of the teacher signal plays an important role in educating a good student [49]. Despite the success of these approaches in boosting recognition accuracy, they suffer from lower training efficiency, as in a K-generation process (one teacher and K students) requires $K\times$ more training time. An inspiring cue comes from the effort of training a few models for ensemble within the same time [21], in which the cost of training was largely reduced.

3. Snapshot Distillation

This section presents snapshot distillation (SD), the first approach that achieves teacher-student (T-S) optimization within one generation. We first briefly introduce a general flowchart of T-S optimization and build a notation system. Then, we analyze the main difficulties that limit its efficiency, based on which we formulate SD and discuss principles and techniques to improve its performance.

3.1. Teacher-Student Optimization

Let a deep neural network be $\mathbb{M} : \mathbf{y} = \mathbf{f}(\mathbf{x}; \boldsymbol{\theta})$, where $\mathbf{x}$ denotes the input image, $\mathbf{y}$ denotes the output data (*e.g.*, a G-dimensional vector for classification with G being the number of classes), and $\boldsymbol{\theta}$ denotes the learnable parameters. These parameters are often initialized as random noise, and then optimized using a training set with N data samples, $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$.

Conventional optimization algorithm works by sampling mini-batches or subsets from the training set. Each of them, denoted as $\mathcal{B}$, is fed into the current model to estimate the difference between prediction and ground-truth labels:

$$\mathcal{L}(\mathcal{B}; \boldsymbol{\theta}) = -\frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{B}} \mathbf{y}_n^\top \ln \mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta}). \qquad (1)$$

This process searches over the parameter space to find the approximately optimal $\boldsymbol{\theta}$ that interprets or fits $\mathcal{D}$. However, the model trained in this way often over-fits the training set, *i.e.*, $\boldsymbol{\theta}$ cannot be transferred to the testing set to achieve good performance as in the training set. As observed in prior work [15], this is partly because the supervision was provided in one-hot vectors, which forces the network to prefer the true class overwhelmingly to all other classes – this is often not the optimal choice because rich information of class-level similarity is simply discarded [45, 49].

To alleviate this issue, teacher-student (T-S) optimization was proposed, in which a pre-trained teacher network added an extra term to the loss function to measure the KL-divergence between teacher and student [11]:

$$\mathcal{L}^{\mathrm{S}}\left(\mathcal{B}; \boldsymbol{\theta}^{\mathrm{S}}\right) = -\frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{B}} \left\{ \lambda^{\mathrm{S}} \cdot \mathbf{y}_n^\top \ln \mathbf{f}\left(\mathbf{x}_n; \boldsymbol{\theta}^{\mathrm{S}}\right) + \lambda^{\mathrm{T}} \cdot \mathrm{KL}\left[\mathbf{f}\left(\mathbf{x}_n; \boldsymbol{\theta}^{\mathrm{T}}\right) \| \mathbf{f}\left(\mathbf{x}_n; \boldsymbol{\theta}^{\mathrm{S}}\right)\right] \right\}, \qquad (2)$$

where $\boldsymbol{\theta}^{\mathrm{S}}$ and $\boldsymbol{\theta}^{\mathrm{T}}$ denote the parameters in teacher and student models, respectively. This is to say, the fitting goal of the student is no longer the ground-truth one-hot vector which is too strict, but leans towards the teacher signal (a softened vector most often with correct prediction). This formulation can be applied in the form of *multiple generations*. Let K be the total number of generations [2, 11, 49]. These approaches started with a so-called patriarch model $\mathbb{M}^{(0)}$, and in the k-th generation, $\mathbb{M}^{(k-1)}$ was used to teach $\mathbb{M}^{(k)}$. [49] showed the necessity of setting *a tolerant teacher* so that the students can absorb richer information from class-level similarity and achieve higher accuracy.

Algorithm 1: Snapshot Distillation

```
Input : training set $\mathcal{D}$, number of iterations $L$,
        training configurations $\left\{\gamma_l, \lambda_l^{\mathrm{T}}, \lambda_l^{\mathrm{S}}, c_l\right\}_{l=1}^{L}$;
Initialize $\boldsymbol{\theta}_0$;
for $l = 1, 2, \ldots, L$ do
    Sample a mini-batch $\mathcal{B}_l$ from $\mathcal{D}$;
    Compute loss $\mathcal{L}(\mathcal{B}_l; \boldsymbol{\theta}_{l-1})$ using Eqn (3);
    $\boldsymbol{\theta}_l \leftarrow \boldsymbol{\theta}_{l-1} - \gamma_l \cdot \nabla_{\boldsymbol{\theta}_{l-1}} \mathcal{L}(\mathcal{B}_l; \boldsymbol{\theta}_{l-1})$
end
Return: $\mathbb{M} : \mathbf{y} = \mathbf{f}(\mathbf{x}; \boldsymbol{\theta} = \boldsymbol{\theta}_L)$.
```

Despite the ability of T-S optimization in improving recognition accuracy, it often suffers the weakness of being computationally expensive. Typically, a T-S process with one teacher and K students costs $K\times$ more time, yet this process is often difficult to parallelize[1]. This motivates us to propose an approach named **snapshot distillation** (SD), which is able to finish T-S optimization *in one generation*.

3.2. The Flowchart of Snapshot Distillation

The idea of SD is very simple. To finish T-S optimization in one generation, during the training process, we always extract the teacher signal from an earlier *iteration*, by which we refer to an intermediate status of the same model, rather than another model that was optimized individually.

Mathematically, let $\boldsymbol{\theta}_0$ be the randomly initialized parameters. The baseline training process contains a total of L iterations, the l-th of which samples a mini-batch $\mathcal{B}_l$, computes the gradient of Eqn (1), and updates the parameters from $\boldsymbol{\theta}_{l-1}$ to $\boldsymbol{\theta}_l$. SD works by assigning a number $c_l < l$ for the l-th iteration, indicating a previous snapshot $\mathbf{f}(\mathbf{x}; \boldsymbol{\theta}_{c_l})$ as the teacher to update $\boldsymbol{\theta}_{l-1}$. Thus, Eqn (2) becomes:

$$\mathcal{L}(\mathcal{B}_l; \boldsymbol{\theta}_{l-1}) = -\frac{1}{|\mathcal{B}_l|} \sum_{(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{B}_l} \left\{ \lambda_l^{\mathrm{S}} \cdot \mathbf{y}_n^\top \ln \mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta}_{l-1}) + \lambda_l^{\mathrm{T}} \cdot \mathrm{KL}[\mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta}_{c_l}) \| \mathbf{f}(\mathbf{x}_n; \boldsymbol{\theta}_{l-1})] \right\}. \qquad (3)$$

Here λ_l^{S} and λ_l^{T} are weights for one-hot and teacher supervisions. When $\lambda_l^{\mathrm{T}} = 0$, the teacher signal is ignored at the current iteration, and thus Eqn (3) degenerates to Eqn (1). The pseudo code of SD is provided in Algorithm 1. In what follows, we will discuss several principles required to improve the performance of SD.

[1] To make fair comparison, researchers often train deep networks using a fixed number of GPUs. T-S optimization trains $K + 1$ models serially, which is often difficult to accelerate even with a larger number of GPUs.

	$\mathbb{M}^{\mathrm{T}}_{\#0}$	$\mathbb{M}^{\mathrm{T}}_{\#75}$	$\mathbb{M}^{\mathrm{T}}_{\#150}$
$\mathbb{M}_{\#0}$	78.13	–	–
$\mathbb{M}_{\#75}$	78.18	78.02	–
$\mathbb{M}_{\#150}$	77.67	77.58	77.47

Table 2. Classification error rates (%) on CIFAR100 with different T-S similarities. All these models are trained for 300 epochs, and all numbers are the average of two individual runs. The first row (self) shows the accuracies of standard models (no T-S optimization), and in the following rows, when $\mathbb{M}^{\mathrm{T}}_{\#E_1}$ teaches $\mathbb{M}_{\#E_2}$, they share the first $\min\{E_1, E_2\}$ common epochs. Some T-S pairs that are probabilistically identical, so only one of them is tested (see Section 3.3.2 for details).

3.3. Principles of Snapshot Distillation

This subsection forms the core contribution of our work, which discusses the principles that should be satisfied to improve the performance of SD. In practice, this involves how to design the hyper-parameters $\{\gamma_l, \lambda_l, c_l\}_{l=1}^{L}$. We first describe three principles individually, and summarize them to give our solution in the final part.

3.3.1 Principle #1: The Quality of Teacher

In prior work, the importance of having a high-quality teacher model has been well studied. At the origin of T-S optimization [19, 35, 50], a more powerful teacher model was used to guide a smaller and thus weaker student model, so that the teacher knowledge is distilled and compressed into the student. This phenomenon persists in a multi-generation T-S optimization in which teacher and student share the same network architecture [2].

Mathematically, the teacher model determines the second term on the right-hand side of Eqn (3), *i.e.*, the KL-divergence between teacher and student. If the teacher is not well optimized and provides noisy supervision, the risk that two terms conflict with each other becomes high. As we shall see later, this principle is even more important in SD, as the number of iterations allowed for optimizing each student becomes smaller, and the efficiency (or the speed of convergence) impacts the final performance heavier.

3.3.2 Principle #2: Teacher-Student Difference

In the context of T-S optimization in one generation, one more challenge emerges. In each iteration, the teacher $\boldsymbol{\theta}_{c_l}$ and student $\boldsymbol{\theta}_{l-1}$ are two snapshots from the same training process, and so the similarity between them is higher than that in multi-generation T-S optimization. This makes the second term on the right-hand side of Eqn 3 degenerate and, consequently, its contribution to the gradient that $\boldsymbol{\theta}_{l-1}$ receives for updating itself is considerably changed.

We evaluate the impact of T-S similarity using the 100-layer DenseNet [22] on the CIFAR100 dataset [24]. All models are trained with the cosine annealing learning rate policy [29] for a total of 300 epochs. Detailed settings are elaborated in Section 4.1. To construct T-S pairs with different similarities, we first perform a complete training process containing 300 standard epochs and starting from scratch, and denote the final model by $\mathbb{M}_{\#300}$. Then, we take the snapshots at 150, 75 and 0 (scratch) epochs, and denote them by $\mathbb{M}_{\#150}$, $\mathbb{M}_{\#75}$ and $\mathbb{M}_{\#0}$, respectively, with the number after $\#$ indicating the number of elapsed epochs. Then, we continue training these snapshots with the same configurations (mini-batch size, learning rates, *etc.*) but different randomization which affects the sampled mini-batch in each iteration and the data augmentation performed at each training sample. These models are denoted by $\mathbb{M}^{\mathrm{T}}_{\#150}$, $\mathbb{M}^{\mathrm{T}}_{\#75}$ and $\mathbb{M}^{\mathrm{T}}_{\#0}$, respectively, where the superscript T implies being used as a teacher model, and each number after $\#$ indicates the number of common epochs shared with $\mathbb{M}_{\#300}$. All these teacher models have exactly 300 epochs.

Now, we use these models to teach the intermediate snapshots, *i.e.*, $\mathbb{M}_{\#150}$, $\mathbb{M}_{\#75}$ and $\mathbb{M}_{\#0}$. When $\mathbb{M}^{\mathrm{T}}_{\#E_1}$ is used to teach $\mathbb{M}_{\#E_2}$, their common part, *i.e.*, the first $E_0 = \min\{E_1, E_2\}$ epochs are preserved, *i.e.*, the first E_0 epochs used Eqn (1) and the remaining $300 - E_0$ epochs used Eqn (2). Results are summarized in Table 2. Note that from a probabilistic perspective, $\mathbb{M}^{\mathrm{T}}_{\#150}$, $\mathbb{M}^{\mathrm{T}}_{\#75}$ and $\mathbb{M}^{\mathrm{T}}_{\#0}$ are identical to each other in classification accuracy, and from the previous part we expect them to provide the same teaching ability. We start with observing their behavior when $\mathbb{M}_{\#0}$ is the student. This case degenerates to a two-generation T-S optimization. Since all teachers are probabilistically identical, we only evaluate one of these pairs, reporting a 78.13% accuracy which is higher than the baseline (the average of $\mathbb{M}^{\mathrm{T}}_{\#150}$, $\mathbb{M}^{\mathrm{T}}_{\#75}$ and $\mathbb{M}^{\mathrm{T}}_{\#0}$ is 77.65%). However, when $\mathbb{M}_{\#75}$ is the student, $\mathbb{M}^{\mathrm{T}}_{\#0}$ serves as a better teacher because it does not share the first 75 epochs with $\mathbb{M}_{\#75}$. This offers a larger difference between teacher and student and, consequently, produces better classification performance (78.18% vs. 78.02%). When $\mathbb{M}_{\#150}$ is chosen to be the student, this phenomenon preserves, *i.e.*, T-S optimization prefers a larger difference between teacher and student.

3.3.3 Principle #3: Secondary Information

The last factor, also being the one that was most studied before, is how knowledge is delivered from teacher to student. There are two arguments, both of which suggesting that a smoother teacher signal preserves richer information, but they differ from each other in the way of achieving this goal. The *distillation* algorithm [19] used a temperature term T to smooth both input and output scores, and the *tolerant teacher* algorithm [49] trained a less confident teacher by adding a regularization term in the first generation (*a.k.a.*

the patriarch), and this strategy was verified the advantageous over the non-regularized version [11].

In the context of snapshot distillation, we follow [19] to divide the teacher signal (in *logits*, the neural responses before the softmax layer) by a temperature coefficient $T > 1$. In the framework of knowledge distillation, the student signals should also be softened before the KL divergence is computed with the teacher signals. The reason is that, the student with a shallow architecture is not capable of completely mimicking the same outputs of the teacher with a deep architecture [2, 19], and thus matching the soft versions of their outputs is a more rational choice. The aim of knowledge distillation is to match the outputs, forcing the student to predict what the teacher predicts as much as possible. However, our goal is to generate secondary information in T-S optimization, instead of matching. As a result, we do not divide the student signal by T. This strategy also aligns with Eqn 1 used in the very first iterations (*i.e.*, no teacher signals are provided). In experiments, we observe a faster convergence as well as consistent accuracy gain – see Section 4.1 for detailed numbers. We name it as *asymmetric distillation*.

3.3.4 Summary and Solution

Summarizing the above three principles, we present our solution to improve the performance of SD. We partition the entire training process with L iterations into K mini-generations with $L_1, L_2 \ldots, L_K$ iterations, respectively, and $\sum_{k=1}^{K} L_k = L$. The last iteration in each mini-generation serves as the teacher of all iterations in the next mini-generation. This is to say, there are $K - 1$ teachers. The first teacher is the snapshot at $L_1' = L_1$ iterations, the second one at $L_2' = L_1 + L_2$ iterations, and the last one at $L_{K-1}' = L_1 + L_2 \ldots + L_{K-1}$ iterations. We have:

$$c_l = \max\left\{L_k', L_k' < l\right\}. \tag{4}$$

For $l \leqslant L_1'$, we define $c_l = 0$ for later convenience, and in this case $\lambda_l^S = 1$, $\lambda_l^T = 0$ and Eqn (3) degenerates to Eqn (1). In comparison to a normal training scheme, SD needs $\frac{1}{3} \times \frac{K-1}{K}$ extra computation: the last $K - 1$ mini-generations require additional teacher's forward propagation. This number is 25% for our CIFAR100 experiments (Section 4.1, $K = 4$) and 16.7% for our ILSVRC2012 experiments (Section 4.2, $K = 2$). In comparison to other K-generation methods [11, 49], SD is theoretically and practically K times faster, because all other methods require teacher inference except for the first generation. Following **Principle #2**, we shall assume that the iterations right after each teacher have large learning rates, in order to ensure the sufficient difference between the teacher and student models. Meanwhile, according to **Principle #1**, the teacher itself should be good, which implies that the iterations before each teacher have small learning rates, which makes the network converge to an acceptable state after sufficient training iterations with the large ones. To satisfy both conditions, we require the learning rates within each mini-generation to start from a large value and gradually decay. In practice, we use the cosine annealing strategy [29] which was verified to converge better:

$$\gamma_l = \frac{1}{2}\alpha_{k_l} \times \left[1 + \cos\left(\frac{l - L_{k_l-1}'}{L_{k_l}' - L_{k_l-1}'} \cdot \pi\right)\right]. \tag{5}$$

Here, k_l is the index of mini-generation of l, and α_{k_l} is the starting learning rate at the beginning of this mini-generation (often set to be large). Finally, we follow Section 3.3.3 to use asymmetric distillation in order to satisfy **Principle #3**.

3.4. Discussions

If we set $L_1 = L_2 = \ldots = L_K$ and switch off the teacher signal, the above solution degenerates to snapshot ensemble (SE) [21]. In experiments, we compare these two approaches under the same setting, and find that both approaches work well on CIFAR100 (SD reports better results), but on ILSVRC2012, SD achieves higher accuracy over the baseline while SE does not[2]. This is arguably because CIFAR100 is relatively simple, so that the original setting (L iterations) are over-sufficient for convergence, and thus reducing the number of iterations of each mini-generation does not cause significant accuracy drop. ILSVRC2012, however, is much more challenging and thus convergence becomes a major drawback of both SD and SE. SD, with the extra benefit brought by T-S optimization, bridges this gap and outperforms the baseline.

Note that the above solution is only one choice. Under Algorithm 1 and the three principles, other training strategies can be explored, *e.g.*, using super-convergence [38] to alleviate the drawback of weaker convergence. These options will be studied in the future.

4. Experiments

4.1. The CIFAR100 Dataset

- **Settings and Baselines**

We first evaluate SD on the CIFAR100 dataset [24], a low-resolution (32 × 32) dataset containing 60,000 RGB images. These images are split into a training set of 50,000 images and a testing set of 10,000 images, and in both of them, images are uniformly distributed over all 100 classes (20 superclasses each of which contains 5 fine-level

[2] The SE paper [21] reported a higher accuracy on ResNet50, but it was compared to the baseline with the stepwise learning rate policy, not the cosine annealing policy that should be the direct baseline. The latter baseline is more than 1% higher than the former, and also outperforms SE.

Backbone	Alg.	T	$\mathbb{M}_{\#L_1}$	$\mathbb{M}_{\#L_2}$	$\mathbb{M}_{\#L_3}$	$\mathbb{M}_{\#L_4}$	*best*	*ensemble*	SOTA	
ResNet20	**BL**	N/A	–	–	–	33.57	33.57	–	*Year*	——
	SE	N/A	36.17	33.36	32.98	32.66	32.54	30.86		
	SD	2	36.17	33.78	32.98	32.31	32.31	32.08	2016 [51]	19.25
	SD	3	36.17	33.69	32.24	**31.97**	**31.76**	**30.76**		
ResNet32	**BL**	N/A	–	–	–	31.61	31.61	–	2017 [54]	19.25
	SE	N/A	33.78	32.15	31.41	30.74	30.51	28.93		
	SD	2	33.78	32.07	31.05	30.67	30.57	29.80	2017 [56]	17.73
	SD	3	33.78	31.52	30.64	**30.32**	**30.16**	**28.71**		
ResNet56	**BL**	N/A	–	–	–	30.23	29.94	–	2017 [47]	17.31
	SE	N/A	32.85	31.60	30.45	29.68	29.55	27.93		
	SD	2	32.85	30.47	29.72	**29.29**	**29.22**	28.11	2017 [22]	17.18
	SD	3	32.85	30.82	29.55	29.37	29.28	**27.74**		
ResNet110	**BL**	N/A	–	–	–	28.77	28.53	–	2017 [16]	17.01
	SE	N/A	31.89	29.81	29.07	28.27	28.09	26.45		
	SD	2	31.89	29.84	28.71	**27.71**	**27.52**	27.19	2017 [53]	16.80
	SD	3	31.89	29.22	28.37	27.87	27.75	**26.19**		
DenseNet100	**BL**	N/A	–	–	–	22.49	22.00	–	2017 [9]	16.53
	SE	N/A	24.31	22.76	22.16	22.18	22.00	**19.63**		
	SD	2	24.31	23.10	22.06	21.78	21.59	20.27	2017 [12]	15.85
	SD	3	24.31	23.19	21.60	**21.17**	**21.17**	19.71		
DenseNet190	**BL**	N/A	–	–	–	16.82	16.69	–	2018 [11]	14.90*
	SE	N/A	18.98	18.12	16.95	**16.84**	16.70	**15.70**		
	SD	2	18.98	17.48	16.32	18.02	**16.06**	15.72	2018 [49]	14.47*
	SD	3	18.98	17.67	16.95	18.65	16.33	15.92		

Table 3. CIFAR100 classification errors (%) obtained by different network backbones. Regarding the algorithm option, **BL** indicate the *baseline* model trained with cosine annealing learning rates, **SE** indicates *snapshot ensemble* with the same learning rate policy as **SD** during the entire training process. T is the temperature term. We report the accuracy at the end of each mini-generation, at the *best* epoch, and obtained from model ensemble ($\mathbb{M}_{\#L_1}$ through $\mathbb{M}_{\#L_4}$), respectively. The logits of $\mathbb{M}_{\#L_k}$ are multiplied by T^{k-1} for ensemble of **SD**. Among the state-of-the-art (SOTA) methods, an asterisk indicates that model ensemble was used to achieve the corresponding error rate. In addition, [12] used complicated data augmentation to achieve an error rate of 15.85% – we just applied standard data augmentation.

classes). We do not perform experiments on the CIFAR10 dataset because it does not contain fine-level visual concepts, and thus the benefit brought by T-S optimization is not significant (as observed in [11] and analyzed in [49]).

We investigate two groups of baseline models. The first group contains standard deep ResNets [18] with 20, 32, 56 and 110 layers. Given a 32×32 input image, a 3×3 convolution is first performed without changing its spatial resolution. Three stages followed, each of which has a few residual blocks (two 3×3 convolutions summed up with an identity connection). Batch normalization [23] and ReLU activation [30] are applied after each convolutional layer. The spatial resolution changes in the three stages (32×32, 16×16 and 8×8), as well as the number of channels (16, 32 and 64). An average pooling layer is inserted after each of the first two stages. The network ends with global average-pooling followed by a fully-connected layer with 100 outputs. The second group has two DenseNets [22] with 100 and 190 layers, respectively. These networks share the similar architecture with the ResNets, but the building blocks in each stage are densely-connected, with the output of each block concatenated to the accumulated feature vector and fed into the next block. The base feature length and growth rate are 24 and 12 for DenseNet100, and 80 and 40 for DenseNet190.

Following the conventions, we train all these networks from scratch. We use the standard Stochastic Gradient Descent (SGD) with a weight decay of 0.0001 and a Nesterov momentum of 0.9. In ResNets, we train the network for 164 epochs with a mini-batch size of 128 and a base learning rate of 0.1. In DenseNets, we train the network for 300 epochs with a mini-batch size of 64 and a base learning rate of 0.1. The cosine annealing learning rate [29] is used, in order to make fair comparison between the baseline and SD. In the training process, standard data-augmentation is used, *i.e.*, each image is symmetrically-padded with a 4-pixel margin on each of the four sides. In the enlarged 40×40 image, a subregion with 32×32 pixels is randomly cropped and flipped with a probability of 0.5. We do not use any data augmentation in the testing stage.

To apply SD, we evenly partition the entire training process into 4 mini-generations, *i.e.*, $K = 4$. For ResNets, we have $L_1 = 41$, $L_2 = 82$ and $L_3 = 123$, and for DenseNets, $L_1 = 75$, $L_2 = 150$ and $L_3 = 225$. The same learning rate

$\alpha_k = 0.1$ is used at the beginning of each mini-generation, and decayed following Eqn (5). We use an asymmetric distillation strategy (Section 3.3.3) with $T = 2$ and $T = 3$, respectively. In Eqn (3), we set $\lambda_l^{\mathrm{S}} = 1 + 1/T$ and $\lambda_l^{\mathrm{T}} = 1$ to approximately balance two gradients in magnitudes [19].

- **Quantitative Results and Analysis**

Results are summarized in Table 3. Towards fair comparison, for different instances of the same backbone, network weights are initialized in the same way, although randomness during the training process (*e.g.*, data shuffle and augmentation) is not unified. In addition, the first mini-generation ($\mathbb{M}_{\#L_1}$, no T-S optimization) is shared between SE (snapshot ensemble) and SD.

We first observe that SD brings consistent accuracy gain for all models, regardless of network backbones, and surpassing both the baseline and SE. In DenseNet190, the most powerful baseline, SD with $T = 2$ achieves an error rate of 16.06% at the best epoch, which is competitive among the state-of-the-arts (all of which reported the best epoch). Moreover, in terms of model ensemble from $\mathbb{M}_{\#L_1}$ through $\mathbb{M}_{\#L_4}$, SD provides comparable numbers to SE, although we emphasize that SD focuses on optimizing a single model while SE, with weaker single models, requires ensemble to improve classification accuracy. Another explanation comes from the optimization policy of SD. By introducing a teacher signal to optimize each student, different snapshots in SD tend to share a higher similarity than SE, and this is the reason that SD reports a smaller accuracy gain from a single model to model ensemble.

Another important topic to discuss is how asymmetric distillation impacts T-S optimization, for which we show several evidences. With a temperature term $T > 1$, the student tends to become smoother, *i.e.*, the entropy of the class distribution is larger. However, as shown in [11] and [49], T-S optimization achieves satisfying performance via finding a balancing point between certainty and uncertainty, so, as the latter gradually increases, we can observe a *peak* in classification accuracy. In DenseNet190 with $T = 2$, this peak appears during the third mini-generation which achieves the lowest error rate at 16.06%, but the final error rate goes up 18.02%. A similar phenomenon also appears in DenseNet100 with $T = 4$, which also achieves the lowest error at the third mini-generation (the lowest error of 21.26% vs. the last error 21.86%), and in ResNets with $T > 5$. This reveals that the optimal temperature term is closely related to the network backbone. For a deeper backbone (*e.g.*, DenseNet190) which itself has a strong ability of fitting data, we use a smaller T to introduce less soft labels, decreasing the ambiguity.

4.2. The ILSVRC2012 Dataset

- **Settings and Baselines**

We now investigate a much more challenging dataset, ILSVRC2012 [36], which is a popular subset of the ImageNet database [7]. It contains 1.3M training images and 50K testing images, all of which are high-resolution, covering 1,000 object classes in total. The distribution over classes is approximately uniform in the training set and and strictly uniform in the testing set.

We use deep ResNets [18] with 101 and 152 layers. They share the same overall design with the ResNets used for CIFAR100, but in each residual block, there is a so-called bottleneck structure which compresses the number of channels by 3/4 and later recovers the original number. Each input image has a size of 224 × 224. After the first 7 × 7 convolutional layer with a stride of 2 and a 3 × 3 max-pooling layer, four main stages follow with different numbers of blocks (ResNet101: $3, 4, 23, 3$; ResNet152: $3, 8, 36, 3$). The spatial resolutions in these four stages are 56 × 56, 28 × 28, 14 × 14 and 7 × 7, and the number of channels are 256, 512, 1,024 and 2,048, respectively. Three max-pooling layers are inserted between these four stages. The network ends with global average-pooling followed by a fully-connected layer with 1,000 outputs.

We follow the conventions to configure the training parameters. The standard Stochastic Gradient Descent (SGD) with a weight decay of 0.0001 and a Nesterov momentum of 0.9 is used. In a total of 90 epochs, the mini-batch size is fixed to be 256. We still use the cosine annealing learning rate [29] starting with 0.1. A series of data-augmentation techniques [40] are applied in training to alleviate over-fitting, including rescaling and cropping the image, randomly mirroring and rotating (slightly) the image, changing its aspect ratio and performing pixel jittering. In the testing stage, the standard single-center-crop is used.

To apply SD, we set $K = 2$ which partitions the training process into two equal sections (each has 45 epochs). The reason of using a smaller K (compared to CIFAR experiments) is that on ILSVRC2012 with high-resolution images and more complex semantics, it is much more difficult to guarantee convergence with a fewer number of iterations within each mini-generation. Regarding the temperature term, we fix $T = 2$. Other settings are the same as in the CIFAR experiments.

- **Quantitative Results**

Experimental results are summarized in Table 4. SD achieves consistent accuracy gain over the baseline in terms of both top-1 and top-5 error rates. On ResNet101, the top-1 and top-5 errors drop by 0.37% and 0.25% absolutely, or 1.71% and 4.31% relatively; on ResNet152, the top-1 and top-5 errors drop by 0.26% and 0.11% absolutely, or 1.23% and 1.94% relatively. These improvement seems small, but we emphasize that (i) to the best of our knowledge, this is the first time that a model achieves higher accuracy on ILSVRC2012 with T-S optimization within one generation; (ii) SD also collaborates well with SENet [20], a powerful

Backbone	Alg.	$\mathbb{M}_{\#L_1}$		$\mathbb{M}_{\#L_2}$	
		Top-1	Top-5	Top-1	Top-5
ResNet101	**BL**	–	–	21.62	5.80
ResNet101	**SE**	22.94	6.51	22.14	6.07
ResNet101	**SD**	22.94	6.51	**21.25**	**5.55**
ResNet152	**BL**	–	–	21.17	5.66
ResNet152	**SE**	22.56	6.44	21.84	5.84
ResNet152	**SD**	22.56	6.44	**20.93**	**5.48**
ResNet101+S	**BL**	–	–	21.10	5.59
ResNet101+S	**SD**	22.41	6.10	**20.59**	**5.29**
ResNet152+S	**SD**	21.89	6.04	**20.21**	**5.17**

Table 4. ILSVRC2012 classification errors (%) obtained by different network backbones. Regarding the algorithm option, **BL** indicate the *baseline* model trained with cosine annealing learning rates, and **SD** snapshot distillation with $T = 2$. The error rates of **SE** [21] are 21.66% on ResNet101 and 21.19% on ResNet152 – even worse than **BL**. "+S" means equipping the network with squeeze-and-excitation modules (SENet [20]).

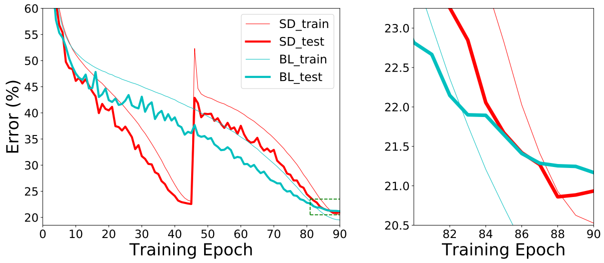

Figure 1. Training and testing curves of ResNet152. The right figure shows the details in the rectangular region of the left figure.

and generalized add-on to the backbones; and (iii) these accuracy gain transfers well to other visual recognition tasks, as shown in the next subsection.

We plot the curves of both the baseline and SD in the training process of ResNet152. We can see that, in the second mini-generation, SD achieves a higher training error but a lower testing error, *i.e.*, the gap between training and testing accuracies becomes smaller, which aligns with our motivation that T-S optimization alleviates over-fitting.

4.3. Transfer Experiments

Last but not least, we fine-tune the models pre-trained on ILSVRC2012 to the object detection and semantic segmentation tasks in the PascalVOC dataset [10], a widely used benchmark in computer vision. The most powerful models, *i.e.*, the baseline and SD versions of ResNet152, are transferred using a standard approach, which preserves the network backbone (all layers before the final pooling layer), and introduces a network head known as Faster R-CNN [34] for object detetion, and DeepLab-v3 [4] for semantic segmentation.

Backbone	mAP @ 2007	mIOU @ 2012
ResNet152-**BL**	73.49	77.53
ResNet152-**SD**	**74.93**	**77.97**

Table 5. PascalVOC object detection (2007, mAP, %) and semantic segmentation (2012, mIOU, %) results, both obtained by fine-tuning the pre-trained deep networks on ILSVRC2012 with Faster R-CNN [34] and DeepLab-v3 [4].

This model is fine-tuned in an end-to-end manner. For object detection on PascalVOC 2007, 5,011 training images are fed into the network through 10 epochs with a mini-batch size of 16. We start a learning rate of 0.01 and divide it by 10 after 8 epochs. For semantic segmentation on PascalVOC 2012, 10,582 training images [17] are fed into the network through 50 epochs with a mini-batch size of 8. We use the "poly" learning rate policy where the initial learning rate is 0.007 and the power is 0.9. Results in terms of mAP and mIOU are summarized in Table 4.3. One can see that, the model with a higher accuracy on ILSVRC2012 also works better in both tasks, *i.e.*, the benefit brought by SD preserves after fine-tuning. Also, we emphasize that SD, with the same network architecture, does not require any additional costs in transfer learning, which claims its potential applications in a wide range of vision problems.

5. Conclusions

In this paper, we present a framework named snapshot distillation (SD), which finishes teacher-student (T-S) optimization within one generation. To the best of our knowledge, this goal was never achieved before. The key contribution is to take teacher signals from the previous iterations of the same training process, and discuss on three principles that impact the performance of SD. The final solution is easy to implement yet efficient to carry out. With less than $1/3$ extra training time, SD boosts the classification accuracy of several baseline models on CIFAR100 and ILSVRC2012 consistently, and the performance gain persists after the trained model is fine-tuned on other vision tasks, *e.g.*, object detection, semantic segmentation.

Our research reduces the basic unit of T-S optimization from a complete generation to a mini-generation which is composed of a number of iterations. The essential difficulty that prevents us from further partitioning this unit is the requirement of T-S difference. We believe there exists, though not yet found, a way of eliminating this constraint so that the basic unit can be even smaller, *e.g.*, one single iteration. In this way, we can integrate supervision from the previous iteration into the current one, obtaining a new loss function in which the teacher signal appears as a term of higher-order gradients. We leave this for future research.

Acknowledgements This paper was supported by NSF award CCF-1317376 and ONR N00014-12-1-0883. We thank Siyuan Qiao, Huiyu Wang and Chenxi Liu who provided insights and expertises to improve this research.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1425–1438, 2016. 2

[2] H. Bagherinezhad, M. Horton, M. Rastegari, and A. Farhadi. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641*, 2018. 1, 2, 3, 4, 5

[3] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. In *International Conference on Learning Representations*, 2016. 2

[4] L. C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 8

[5] T. Chen, I. Goodfellow, and J. Shlens. Net2net: Accelerating learning via knowledge transfer. In *International Conference on Learning Representations*, 2016. 1, 2

[6] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *European Conference on Computer Vision*, 2010. 2

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, 2009. 2, 7

[8] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, 2014. 2

[9] X. Dong, G. K., K. Zhan, and Y. Yang. Eraserelu: a simple way to ease the training of deep convolution neural networks. *arXiv preprint arXiv:1709.07634*, 2017. 6

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 2, 8

[11] T. Furlanello, Z. C. Lipton, L. Itti, and A. Anandkumar. Born again neural networks. In *International Conference on Machine Learning*, 2018. 1, 2, 3, 5, 6, 7

[12] X. Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017. 6

[13] R. Girshick. Fast r-cnn. In *Computer Vision and Pattern Recognition*, 2015. 2

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014. 2

[15] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017. 2, 3

[16] D. Han, J. Kim, and J. Kim. Deep pyramidal residual networks. In *Computer Vision and Pattern Recognition*, 2017. 6

[17] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision*, 2011. 8

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016. 1, 2, 6, 7

[19] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 2, 4, 5, 7

[20] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Computer Vision and Pattern Recognition*, 2018. 2, 7, 8

[21] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger. Snapshot ensembles: Train 1, get m for free. In *International Conference on Learning Representations*, 2018. 2, 5, 8

[22] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *Computer Vision and Pattern Recognition*, 2017. 1, 2, 4, 6

[23] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 1, 2, 6

[24] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 2, 4, 5

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 1, 2

[26] Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nature*, 521(7553):436, 2015. 2

[27] C. Liu, B. Zoph, J. Shlens, W. Hua, L. J. Li, L. Fei-Fei, A. L. Yuille, J. Huang, and K. Murphy. Progressive neural architecture search. In *European Conference on Computer Vision*, 2018. 2

[28] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2015. 2

[29] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 4, 5, 6, 7

[30] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, 2010. 1, 2, 6

[31] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017. 2

[32] F. Perronnin, J. Sanchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, 2010. 2

[33] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition*, 2014. 2

[34] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 2, 8

[35] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2014. 1, 2, 4

[36] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 2, 7

[37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 1, 2

[38] L. N. Smith and N. Topin. Super-convergence: Very fast training of residual networks using large learning rates. *arXiv preprint arXiv:1708.07120*, 2017. 5

[39] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 1, 2

[40] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. In *Computer Vision and Pattern Recognition*, 2015. 1, 2, 7

[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Computer Vision and Pattern Recognition*, 2016. 2

[42] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, 2017. 2

[43] N. Verma, D. Mahajan, S. Sellamanickam, and V. Nair. Learning hierarchical similarity metrics. In *Computer Vision and Pattern Recognition*, 2012. 2

[44] J. Wang, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, Y. Wu, et al. Learning fine-grained image similarity with deep ranking. In *Computer Vision and Pattern Recognition*, 2014. 2

[45] C. Wu, M. Tygert, and Y. LeCun. Hierarchical loss for classification. *arXiv preprint arXiv:1709.01062*, 2017. 2, 3

[46] L. Xie and A. Yuille. Genetic cnn. In *International Conference on Computer Vision*, 2017. 2

[47] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition*, 2017. 6

[48] S. Xie and Z. Tu. Holistically-nested edge detection. In *International Conference on Computer Vision*, 2015. 2

[49] C. Yang, L. Xie, S. Qiao, and A. L. Yuille. Knowledge distillation in generations: More tolerant teachers educate better students. *arXiv preprint arXiv:1805.05551*, 2018. 1, 2, 3, 4, 5, 6, 7

[50] J. Yim, D. Joo, J. Bae, and J. Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Computer Vision and Pattern Recognition*, 2017. 1, 2, 4

[51] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 6

[52] C. Zhang, J. Cheng, and Q. Tian. Image-level classification by hierarchical structure learning with visual and semantic similarities. *Information Sciences*, 422:271–281, 2018. 2

[53] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 6

[54] T. Zhang, G. J. Qi, B. Xiao, and J. Wang. Interleaved group convolutions. In *Computer Vision and Pattern Recognition*, 2017. 6

[55] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. Deep mutual learning. *arXiv preprint arXiv:1706.00384*, 2017. 2

[56] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017. 6

[57] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017. 2