# How Important is Importance Sampling for Deep Budgeted Training?

Eric Arazo
eric.arazo@insight-centre.org
Diego Ortego
diego.ortego@insight-centre.org
Paul Albert
paul.albert@insight-centre.org
Noel E. O'Connor
noel.oconnor@insight-centre.org
Kevin McGuinness
kevin.mcguinness@insight-centre.org

School of Electronic Engineering,
Insight SFI Centre for Data Analytics,
Dublin City University (DCU),
Dublin, Ireland

**Abstract**

Long iterative training processes for Deep Neural Networks (DNNs) are commonly required to achieve state-of-the-art performance in many computer vision tasks. Importance sampling approaches might play a key role in budgeted training regimes, i.e. when limiting the number of training iterations. These approaches aim at dynamically estimating the importance of each sample to focus on the most relevant and speed up convergence. This work explores this paradigm and how a budget constraint interacts with importance sampling approaches and data augmentation techniques. We show that under budget restrictions, importance sampling approaches do not provide a consistent improvement over uniform sampling. We suggest that, given a specific budget, the best course of action is to disregard the importance and introduce adequate data augmentation; e.g. when reducing the budget to a 30% in CIFAR-10/100, RICAP data augmentation maintains accuracy, while importance sampling does not. We conclude from our work that DNNs under budget restrictions benefit greatly from variety in the training set and that finding the right samples to train on is not the most effective strategy when balancing high performance with low computational requirements. Source code available at: https://git.io/JKHa3

## 1 Introduction

The availability of vast amounts of labeled data is crucial when training deep neural networks (DNNs) [39, 57]. Despite prompting considerable advances in many computer vision tasks [50, 58], this dependence poses two challenges: the generation of the datasets and the large computation requirements that arise as a result. Research addressing the former has experienced great progress in recent years via novel techniques that reduce the strong supervision required to achieve top results [52, 54] by, e.g. improving semi-supervised learning [5, 8],

self-supervised learning [23, 41], or training with noisy web labels [4, 33]. The latter challenge has also experienced many advances from the side of network efficiency via DNN compression [16, 35], neural architecture search [10, 52], or parameter quantization [25, 45]. All these approaches are designed with a common constraint: a large dataset is needed to achieve top results [57]. This conditions the success of the training process on the available computational resources. Conversely, a smart reduction of the amount of samples used during training can alleviate this constraint [30, 40].

The selection of samples plays an important role in the optimization of DNN parameters during training, where Stochastic Gradient Descent (SGD) [9, 17] is often used. SGD guides the parameter updates using the estimation of model error gradients over sets of samples (mini-batches) that are uniformly randomly selected in an iterative fashion. This strategy assumes equal importance across samples, whereas other works suggest that alternative strategies for revisiting samples are more effective in achieving better performance [11, 31] and faster convergence [26, 30]. Similarly, the selection of a unique and informative subset of samples (core-set) [14, 53] can reduce the computation requirements during training, while reducing the performance drop with respect to training on all data. However, although removing data samples speeds up training, precise sample selection often requires a pretraining stage that acts counter computational reduction [40, 47].

A possible solution to this limitation might be to dynamically change the important subset during training, as is done by importance sampling methods [3, 62], which select the samples based on a sampling distribution that evolves with the model and often depends on the loss or network logits [28, 37]. An up-to-date sample importance estimation is key for current methods to succeed but, in practice, is infeasible to compute [30]. The importance of a sample changes after each iteration and estimations become out-dated, yielding considerable performance drops [11, 62]. Importance sampling methods focus on training with the most relevant samples and achieve a convergence speed-up as a side effect. They do not, however, strictly study the benefits on DNN training when restricting the number of training iterations, i.e. the budget.

Budgeted training [29, 34, 42] imposes an additional constraint on the optimization of a DNN: a maximum number of iterations. Defining this budget provides a concise notion of the limited training resources. Li et al. [34] propose to address the budget limitation using specific learning rate schedules that better suit this scenario. Despite the standardized scenario that budgeted training poses to evaluate methods when reducing the computation requirements, there are few works to date in this direction [30, 34]. As mentioned, importance sampling methods are closely related, but the lack of exploration of different budget restrictions makes these approaches less applicable: the sensitivity to hyperparamenters that they often exhibit limits their generalization [11, 37].

In this paper, we overcome the limitations outlined above by analyzing the effectiveness of importance sampling methods when a budget restriction is imposed [34]. Given a budget restriction, we study synergies among importance sampling and data augmentation [15, 51, 61]. We find the improvements of importance sampling approaches over uniform random sampling are not always consistent across budgets and datasets. We argue and experimentally confirm (see Section 4.4) that when using certain data augmentation strategies [15, 51, 61], existing importance sampling techniques do not provide further benefits, making data augmentation the most effective strategy to exploit a given budget.

## 2 Related work

Few works exploit a budgeted training paradigm [34]. Instead, many aim to speed up convergence to a given performance using a better sampling strategy or carefully organizing the samples to allow the model to learn faster and generalize better. Others explore how to improve model performance by labeling the most important samples from an unlabeled set [6, 46, 59] or how to better train DNNs when limited samples per class are available [2, 12, 64]. None of these works, however, explore the efficiency of these approaches when trained under constraints in the number of iterations allowed, i.e. budgeted training. This section reviews relevant works that aim to improve the computational efficiency of training DNNs.

**Curriculum learning (CL)** aims to improve model performance by ordering the samples from easy to difficult [7, 13, 21, 56]. Like importance sampling approaches, CL leverages different samples at different stages of training. However, while CL prioritizes easy samples at the beginning of training and includes all of them at the end, importance sampling prioritizes the most difficult subset of samples at each stage of the training. The main drawback of CL is that, in most cases, the order of the samples (curriculum) has to be defined before training, which is already a costly task that requires manually assessing the sample difficulty or transferring knowledge from a pre-trained model. Some approaches remedy this with a simple curriculum [36] or by learning it during training [27]; these methods, however, do not aim to speed up training by ordering the samples, but to improve convergence by weighting the sample contribution to the loss.

**Core-set selection approaches** aim to find the subset of samples that is most useful [14, 40, 53] and maintain accuracy despite training on a fraction of the data. The ability of these methods to reduce training cost relies on using smaller training sets, but the benefit is limited since they require a pre-training stage with the full dataset. They do, however, demonstrate that DNNs can achieve peak performance with a fraction of the full dataset. Some approaches to core-set selection use the most often forgotten samples by the network [53], the nearest samples to cluster centroids built from model features [40], or use a smaller pretrained model to select the most informative samples [14].

**Importance sampling** approaches lie in the middle ground between the previous two: they aim to speed up training convergence by leveraging the most useful samples at each training stage [26, 30, 62], which correspond to those with highest loss gradient magnitude [1, 43, 63]. Johnson and Guestrin [28] have shown that the last layer gradients are a good approximation and are easier to obtain in deep learning frameworks. Alternative importance measures include the loss [26], the probability predicted for the true class [11], or the rank order of these probabilities [37].

The approximation of the optimal distribution by importance sampling approaches avoids the cost of computing the importance of each sample at every iteration. However, this distribution changes very rapidly between iterations, leading to outdated estimations. Initial attempts at addressing this included using several hyper-parameters to smooth the estimated distribution [11], more frequent distribution updates via additional forward passes [37], or alternative measures to estimate the sampling distribution [3]. Several works added complex support techniques to the training to estimate a better distribution: using robust optimization [28], introducing repulsive point techniques [60], or adding a second network [62].

More recent methods leverage the random-then-greedy technique [38], where the probabilities of an initial random batch of samples are computed and then used to select a batch for training. Within this scheme, [30] define a theoretical bound for the magnitude of the

gradients that allows for faster computation of the sampling probabilities, and [24, 26] use the loss as a measure of sample importance to keep the sampling distribution updated through the training. Finally, Kawaguchi and Lu [31] introduce the top-$k$ loss [19] to perform the back-propagation step using the samples with highest losses only. Note that these methods do a full forward pass every epoch to update the sampling probabilities.

**Learning rate schedules** have proven to be useful alternatives for faster convergence. In particular, [28, 29] propose a cyclic learning rate schedule to reach faster convergence by using larger learning rates at intermediate training stages and very low rates at final stages. Similarly, Li et al. [34] explore budgeted training and propose a linearly decaying learning rate schedule that approaches zero at the end of the training, which without additional hyper-parameters, provides better convergence than the standard learning rate schedulers. These approaches, however, do not explore sample selection techniques to further increase convergence speed.

# 3 Budgeted training

This section formally introduces budgeted training and the different importance sampling methods used through the paper to explore the efficiency of these approaches under budget restrictions. The standard way of training DNNs is by gradient based minimization of cross-entropy

$$\ell(\theta) = -\frac{1}{N}\sum_{i=1}^{N} y_i^T \log h_\theta(y|x_i), \tag{1}$$

where $N$ is the number of samples in the dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ and $y_i \in \{0,1\}^C$ is the one-hot encoding ground-truth label for sample $x_i$, $C$ is the number of classes, $h_\theta(y|x_i)$ is the predicted posterior probability of a DNN model given $x_i$ (i.e. the prediction after softmax normalization), and $\theta$ are the parameters of the model. Convergence to a reasonable level of performance usually determines the end of the training, whereas in budgeted training there is a fixed iteration budget. We adopt the setting defined by [34], where the budget is defined as a percentage of the full training setup. Formally, we define the budget $B \in [0,1]$ as the fraction of forward and backward passes used for training the model $h_\theta(x)$ with respect to a standard full training. As we aim at analyzing importance sampling, the budget restriction will be mainly applied to the amount of data $N \times B$ seen every epoch. However, a reduction on the number of epochs $T$ to $T \times B$ (where an epoch $T$ is considered a pass over all samples) is also considered as truncated training for budgeted training.

**Truncated training** is the simplest approach to budgeted training: keep the standard SGD optimization and reduce the number of epochs trained by the model to $T \times B$. We call this strategy, where the model sees all the samples every epoch, *scan-SGD*. While seeing all the samples is common practice, we remove this constraint and draw the samples from a uniform probability distribution at every iteration and call this strategy *unif-SGD*. In this approach the budget is defined by randomly selecting $N \times B$ samples every epoch (and still training for $T$ epochs).

**Importance sampling** aims to accelerate the convergence of SGD by sampling the most difficult samples $D_S = \{(x_i, y_i)\}_{i=1}^{N_S}$ more often, where $N_S = N \times B$ (the number of samples selected given a certain budget). Loshchilov and Hutter [37] proposed a simple approach for importance sampling that uses the loss of every sample as a measure of the sample importance. Chang et al. [11] adapts this approach to avoid additional forward passes by

using as importance:

$$p_i^t = \frac{1}{t}\sum_{k=1}^{t}\Big(1 - y_i^T h_\theta^k(y|x_i)\Big) + \varepsilon^t, \tag{2}$$

where $h_\theta^k(y|x_i)$ is the prediction of the model given the sample $x_i$ in epoch $k$, and $t$ is the current epoch. Therefore, the average predicted probability across previous epochs associated to the ground-truth class of each sample defines the importance of sample $x_i$. The smoothing constant $\varepsilon^t$ is defined as the mean per sample importance up to the current epoch: $\frac{1}{N}\sum_{i=1}^{N} p_i^t$. The sampling distribution $P^t$ at a particular epoch $t$ is then given by:

$$P_i^t = \frac{p_i^t}{\sum_{j=1}^{N} p_j}. \tag{3}$$

By drawing samples from the distribution $P^t$ this approach biases the training towards the most difficult samples, and selects samples with highest loss value; we name this method *p-SGD*. Similarly, Chang et al. [11] propose to select those samples that are closer to the decision boundaries and favor samples with higher uncertainty by defining the importance measure as $c_i^t = p_i^t \times (1 - p_i^t)$; we name this approach *c-SGD*.

Both *p-SGD* and *c-SGD* are very computationally efficient as the importance estimation only requires information available during training. Conversely, Jiang et al. [26] propose to perform forward passes on all the samples to determine the most important ones and later reduce the amount of backward passes; they name this method selective backpropagation (*SB*). At every forward pass, *SB* stores the sample $x_i$ with probability:

$$s_i^t = \Big[F_R(\ell(h_\theta^t(x_i), y_i))\Big]^b, \tag{4}$$

where $F_R$ is the cumulative distribution function from a history of the loss values of the last $R$ samples seen by the model and $b > 0$ is a constant that determines the selectivity of the method, i.e. the budget used during the training. In practice, *SB* does as many forward passes as needed until it has enough samples to form a full a mini-batch. It then performs the training forward and backward passes with the selected samples to update the model.

Finally, as an alternative training paradigm to prioritize the most important samples, Kawaguchi and Lu [31] propose to use only the $q$ samples with highest loss from a mini-batch in the backward pass. As the training accuracy increases, $q$ decreases until only $1/16$ of the images in the mini-batch are used in the backward pass. The authors name this approach *ordered* SGD (*OSGD*) and provide a default setting for the adaptive values of $q$.

**Importance sampling methods under budgeted training** give a precise notion of the training budget. For *unif-SGD*, *p-SGD*, and *c-SGD* the adaptation needed consists of selecting a fixed number of samples $N \times B$ per epoch based on the corresponding sampling probability distribution $P_t$ and still train the full $T$ epochs. For *SB*, the parameter $b$ determines the selectivity of the algorithm: higher values will reject more samples. Note that this method requires additional forward passes that we exclude from the budget as they do not induce the backward passes used for training. By assuming that each DNN backward pass is twice as computationally expensive as a forward pass [30] we could approximate the budget used by *SB* as $B_{SB} = B + 1/3$, e.g. the results under $B = 0.2$ for *SB* correspond to $B \approx 0.5$ for the other approaches. We adapt *OSGD* by truncating the training as in *scan-SGD*: all the parameters are kept constant but the total number of epochs is reduced to $T \times B$. We also consider the wall-clock time with respect to a full budget training as a metric to evaluate the approaches.

# 4 Experiments and Results

## 4.1 Experimental framework

**Datasets.** We experiment on image classification tasks using CIFAR-10/100 [32], SVHN [44], and mini-ImageNet [55] datasets. CIFAR-10/100 consist of 50K samples for training and 10K for testing; each divided into 10(100) classes for CIFAR-10(100). The samples are images extracted from ImageNet [18] and down-sampled to 32×32. SVHN contains 32×32 RGB images of real-world house numbers divided into 10 classes, 73257 for training and 26032 for testing. Mini-ImageNet is a subset of ImageNet with 50K samples for training and 10K for testing divided into 100 classes and down-sampled to 84×84. Unless otherwise stated, all the experiments use standard data augmentation: random cropping with padding of 4 pixels per side and random horizontal flip (except in SVHN, where horizontal flip is omitted).

**Training details.** We train a ResNet-18 architecture [22] for 200 epochs with SGD with momentum of 0.9 and a batch size of 128. We use two learning rate schedules: step-wise and linear decay. For both schedules we adopt the budget-aware version proposed by Li et al. [34] and use an initial learning rate of 0.1. In the step-wise case, the learning rate is divided by 10 at 1/3 (epoch 66) and 2/3 (epoch 133) of the training. The linear schedule decreases the learning rate value at every iteration linearly from the initial value to approximately zero ($10^{-6}$) at the end of the training. We always report the average accuracy and standard deviation of the model across 3 independent runs trained on a GeForce GTX 1080Ti GPU using the Pytorch library. For each budget, we report best results in bold and best results in each section – data augmentation or learning rate schedule – in blue (baseline SGD is excluded).

## 4.2 Budget-free training for importance sampling

Current importance sampling methods from the state-of-the-art are optimized with no restriction in the number of training iterations. While this allows the methods to better exploit the training process, it makes it difficult to evaluate their computational benefit. Therefore, Table 1 presents the performance, wall-clock time, and speed-up relative to a full training of the methods presented in Section 3. All methods train with a step-wise linear learning rate schedule. SGD corresponds to a standard training as described in Subsection 4.1. *p-SGD* and *c-SGD* correspond to the methods described in Section 3 introduced by Chang et al. [11] that for the experiments in Table 1 train for 200 epochs where the first 70 epochs consist of a warm-up stage with a uniform sampling strategy as done in the original paper. For CIFAR-10 we use a budget of 0.8 for *p-SGD* and 0.7 for *c-SGD*, and for CIFAR-100 a budget of 0.9 for both approaches (budgets retaining most accuracy were selected). Finally, *SB* and *OSGD* follow the setups described in the corresponding papers, [26] and [31], and run on the official code.

Table 1: Test accuracy (%), time (min) and speed-up (%) with respect SGD under a budget-free training (A, T, and S respectively). * denotes that we have used the official code.

| | CIFAR-10 | | | CIFAR-100 | | |
|---|---|---|---|---|---|---|
| Method | A | T | S | A | T | S |
| SGD | 94.58 ± 0.33 | 141 | 0.0 | 74.56 ± 0.06 | 141 | 0.0 |
| *p-SGD* | 94.41 ± 0.19 | 113 | 19.9 | 74.44 ± 0.06 | 127 | 9.9 |
| *c-SGD* | 94.17 ± 0.11 | 100 | 29.1 | 74.40 ± 0.06 | 127 | 9.9 |
| *SB* (*) | 93.90 ± 0.16 | 85 | 39.7 | 73.39 ± 0.37 | 119 | 15.6 |
| *OSGD* (*) | 94.34 ± 0.07 | 139 | 0.1 | 74.22 ± 0.21 | 141 | 0.0 |

While the simpler approaches to importance sampling, *p-SGD* and *c-SGD*, achieve similar performance to SGD and reduce computation up to 29.08% (9.93%) in CIFAR-10 (CIFAR-100), *SB* reduces the training time 39.72% (15.60%) in CIFAR-10 (CIFAR-100) with very small drops in accuracy. This supports importance sampling observations where particular

configurations effectively reduce computational requirements and maintain accuracy.

## 4.3 Budgeted training for importance sampling

We adapt importance sampling approaches as described in Section 3 and configure each method to constrain its computation to the given budget. Table 2 shows the analyzed methods performance under the same budget for a step-wise learning rate (SLR) decay and the linear decay (LLR) proposed by Li et al. [34] for budgeted training (described in Section 4.1). Surprisingly, this setup shows that most methods achieve very similar performance given a predefined budget, thus not observing faster convergence when using importance sampling. Both *p-SGD* and *c-SGD* provide marginal or no improvements: *p-SGD* marginally improves *unif-SGD* in CIFAR-10, but fails to do so in CIFAR-100. Similar behaviour is observed in *c-SGD*. Conversely, *SB* surpasses the other approaches consistently for SLR and in most cases in the LLR setup. However, *SB* introduces additional forward passes not considered as budget, while the other methods do not (see Section 3 for an estimation of the budget used by SB).

Table 2: Test accuracy with a step-wise and a linear learning rate decay under different budgets. Note that *SB* requires additional computation (forward passes).

| | CIFAR-10 | | | CIFAR-100 | | |
|---|---|---|---|---|---|---|
| *SGD - SLR* | 94.58 ± 0.33 | | | 74.56 ± 0.06 | | |
| *SGD - LLR* | 94.80 ± 0.08 | | | 75.44 ± 0.16 | | |
| Budget: | 0.2 | 0.3 | 0.5 | 0.2 | 0.3 | 0.5 |
| Step-wise decay of the learning rate (SLR) | | | | | | |
| *scan-SGD* | 92.03 ± 0.24 | 93.06 ± 0.15 | 93.80 ± 0.15 | 70.89 ± 0.23 | 72.31 ± 0.22 | 73.49 ± 0.20 |
| *unif-SGD* | 91.82 ± 0.05 | 92.69 ± 0.07 | 93.71 ± 0.07 | 70.36 ± 0.30 | 72.03 ± 0.47 | 73.36 ± 0.20 |
| *p-SGD* | 92.28 ± 0.05 | 92.91 ± 0.18 | 93.85 ± 0.07 | 70.24 ± 0.28 | 72.11 ± 0.39 | 72.94 ± 0.36 |
| *c-SGD* | 91.70 ± 0.25 | 92.83 ± 0.30 | 93.71 ± 0.15 | 69.86 ± 0.36 | 71.56 ± 0.27 | 73.02 ± 0.34 |
| *SB* | 93.37 ± 0.11 | 93.86 ± 0.27 | 94.21 ± 0.13 | 70.94 ± 0.38 | 72.25 ± 0.68 | 73.39 ± 0.37 |
| *OSGD* | 90.61 ± 0.31 | 91.78 ± 0.30 | 93.45 ± 0.10 | 70.09 ± 0.25 | 72.18 ± 0.35 | 73.39 ± 0.22 |
| Linear decay of the learning rate (LLR) | | | | | | |
| *scan-SGD* | 92.95 ± 0.07 | 93.55 ± 0.21 | 94.22 ± 0.16 | **72.04 ± 0.42** | 72.97 ± 0.07 | 73.90 ± 0.43 |
| *unif-SGD* | 92.83 ± 0.14 | 93.48 ± 0.05 | 93.98 ± 0.11 | 72.02 ± 0.24 | 72.74 ± 0.57 | 73.93 ± 0.16 |
| *p-SGD* | 93.23 ± 0.14 | 93.63 ± 0.04 | 94.14 ± 0.11 | 71.72 ± 0.37 | 72.94 ± 0.37 | 74.06 ± 0.10 |
| *c-SGD* | 92.95 ± 0.17 | 93.54 ± 0.07 | 94.11 ± 0.24 | 71.37 ± 0.49 | 72.33 ± 0.18 | 73.93 ± 0.35 |
| *SB* | **93.78 ± 0.11** | **94.06 ± 0.37** | **94.57 ± 0.18** | 71.96 ± 0.67 | **73.11 ± 0.42** | **74.35 ± 0.34** |
| *OSGD* | 91.87 ± 0.36 | 93.00 ± 0.08 | 93.93 ± 0.22 | 71.25 ± 0.11 | 72.56 ± 0.36 | 73.40 ± 0.14 |

We consider *scan-SGD* and *unif-SGD*, as two naive baselines for budgeted training. Despite having similar results (*scan-SGD* seems to be marginally better than *unif-SGD*), we use *unif-SGD* for further experimentation in the following subsections as it adopts a uniform random sampling distribution, which allows contrasting with the importance sampling methods. Additionally, Table 2 confirms the effectiveness of a linear learning rate schedule as proposed in [34]: all methods consistently improve with this schedule and, in most cases, *unif-SGD* and LLR perform on par with *SB* and SLR, and surpasses all the other methods when using SLR.

This failure of the sampling strategies to consistently outperform *unif-SGD* could be explained by importance sampling breaking the assumption that samples are i.i.d: SGD assumes that a set of randomly selected samples represents the whole dataset and provides an unbiased estimation of the gradients. Importance sampling explicitly breaks this assumption and biases the gradient estimates. While this might produce gradient estimates that have a bigger impact on the loss, breaking the i.i.d. assumption leads SGD to biased solutions [11, 37, 62], which offsets the possible benefits of training with the most relevant samples. As a result, importance sampling does not bring a consistent speed-up in training. Note that approaches that weight the contribution of each sample with the inverse sampling probability to generate an unbiased gradient estimate obtain similar results [1, 11, 20, 28, 62].

## 4.4 Data variability importance during training

Core-set selection approaches [14, 53] aim to find the most representative samples in the dataset to make training more efficient, while keeping accuracy as high as possible. Fig-

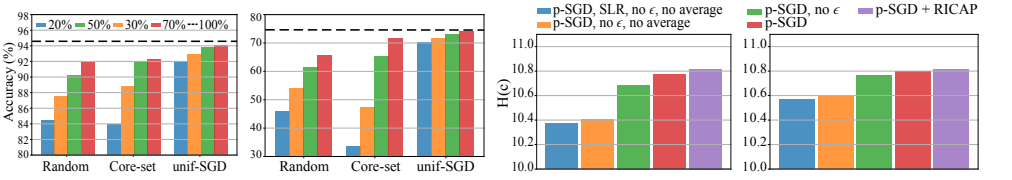


Figure 1: Importance of data variability in CIFAR-10, (a) and (c), and CIFAR-100, (b) and (d). (a) and (b) compare different training set selection strategies: randomly selecting samples at every epoch (*unif-SGD*) outperforms fixed core-set or random subsets. (c) and (d) compares the data variability of different training strategies: the entropy of sample counts during training (0.3 budget) demonstrates that importance sampling, linear learning rate, and data augmentation contribute to higher data variability (entropy).

ure 1, (a) and (b), presents how core-set selection and a randomly chosen subset (Random) both under-perform uniform random sampling of a subset each epoch (*unif-SGD*), which approaches standard training performance (black dashed line). This shows that randomly selecting a different subset every epoch (*unif-SGD*), which is equally computationally efficient, achieves substantially better accuracy. This result supports the widely adopted assumption that data variability is key and suggests that it might be more important than sample quality.

We also find data variability to play an important role within importance sampling. Figure 1 (c) and (d) shows data variability measured using the entropy $H(c)$ of the number of times that a sample is seen by the network during training, with $c$ being the $N$-D distribution of sample counts. These results show how increases in variability (higher entropy) follow accuracy improvements in *p-SGD* when introducing the LLR, the smoothing constant to the $P^t$ sampling distribution, the average of the predictions across epochs, and data augmentation.

## 4.5 Data augmentation for importance sampling

Importance sampling approaches usually do not explore the interaction of sampling strategies with data augmentation techniques [26, 30, 37]. To better understand this interaction, we explore interpolation-based augmentations via RICAP [51] and mixup [61]; and non-interpolation augmentations using RandAugment [15]. We implemented these data augmentation policies as reported in the original papers (see Table 3 for the hyperparameters used in our experiments). Note that for mixup and RICAP we combine 2 and 4 images respectively within each mini-batch, which results in the same number of samples being shown to the network ($T \times B$).

Table 3: Data augmentation for budgeted importance sampling in CIFAR-10/100. N and M are the number and strength of RandAugment augmentations, and $\alpha$ controls the interpolation in mixup and RICAP. Note that SGD corresponds to the full training.

| | CIFAR-10 | | | CIFAR-100 | | |
|---|---|---|---|---|---|---|
| Budget: | 0.2 | 0.3 | 0.5 | 0.2 | 0.3 | 0.5 |
| Standard data augmentation | | | | | | |
| SGD ($B=1$) | | 94.80 ± 0.08 | | | 75.44 ± 0.16 | |
| *unif-SGD* | 92.83 ± 0.14 | 93.48 ± 0.05 | 93.98 ± 0.11 | 72.02 ± 0.24 | 72.74 ± 0.57 | 73.93 ± 0.16 |
| *p-SGD* | 93.23 ± 0.14 | 93.63 ± 0.04 | 94.14 ± 0.11 | 71.72 ± 0.37 | 72.94 ± 0.37 | 74.06 ± 0.10 |
| *SB* | 93.78 ± 0.11 | 94.06 ± 0.37 | 94.57 ± 0.18 | 71.96 ± 0.67 | 73.11 ± 0.42 | 74.35 ± 0.34 |
| RandAugment data augmentation (N = 2, M = 4) | | | | | | |
| SGD ($B=1$) | | 95.56 ± 0.12 | | | 75.52 ± 0.17 | |
| *unif-SGD* | 92.76 ± 0.16 | 93.78 ± 0.11 | 94.64 ± 0.08 | 71.44 ± 0.37 | 73.23 ± 0.29 | 74.78 ± 0.45 |
| *p-SGD* | 92.95 ± 0.31 | 93.99 ± 0.28 | 94.91 ± 0.18 | 71.63 ± 0.27 | 72.91 ± 0.13 | 74.30 ± 0.04 |
| *SB* | 93.27 ± 0.38 | 94.64 ± 0.07 | 95.27 ± 0.26 | 66.84 ± 1.15 | 73.79 ± 0.40 | 74.87 ± 0.18 |
| mixup data augmentation ($\alpha = 0.3$) | | | | | | |
| SGD ($B=1$) | | 95.82 ± 0.17 | | | 77.62 ± 0.40 | |
| *unif-SGD* | 93.64 ± 0.27 | 94.49 ± 0.04 | 95.18 ± 0.05 | 73.28 ± 0.51 | 75.13 ± 0.52 | 75.80 ± 0.34 |
| *p-SGD* | 93.78 ± 0.04 | 94.41 ± 0.16 | 95.26 ± 0.06 | 73.35 ± 0.29 | 75.05 ± 0.15 | 75.87 ± 0.15 |
| *SB* | 93.62 ± 0.36 | 93.92 ± 0.08 | 94.51 ± 0.17 | 73.38 ± 0.13 | 74.88 ± 0.31 | 75.57 ± 0.23 |
| RICAP data augmentation ($\alpha = 0.3$) | | | | | | |
| SGD ($B=1$) | | 96.17 ± 0.09 | | | 78.91 ± 0.07 | |
| *unif-SGD* | 93.85 ± 0.10 | **94.93 ± 0.29** | 95.47 ± 0.18 | **74.87 ± 0.28** | 76.27 ± 0.32 | **77.83 ± 0.15** |
| *p-SGD* | **94.02 ± 0.18** | 94.79 ± 0.18 | **95.63 ± 0.15** | 74.59 ± 0.15 | **76.50 ± 0.22** | 77.58 ± 0.49 |
| *SB* | 89.93 ± 0.84 | 93.64 ± 0.42 | 94.76 ± 0.02 | 56.66 ± 0.65 | 72.24 ± 0.58 | 76.26 ± 0.22 |

Table 3 and 4 show that data augmentation is beneficial in a budgeted training scenario: in most cases all strategies increase performance compared to standard data augmentation. The main exception is for the lowest budget for *SB* where in some cases data augmentation hurts performance. In particular, with RICAP and mixup, the improvements from importance sampling approaches are marginal and the naive *unif-SGD* provides results close to full training with standard augmentation. In some cases *unif-SGD* surpasses full-training with standard augmentations, e.g. RICAP with 0.3 and 0.5 budget and both mixup and RICAP with 0.3 budget in CIFAR-10/100. This is even more evident in SVHN where all the budgets in Table 4 for *unif-SGD* with RICAP surpass full training (SGD) with standard augmentation.

Table 4: Data augmentation for budgeted importance sampling in SVHN and mini-ImageNet. N and M are the number and strength of RandAugment augmentations, and $\alpha$ controls the interpolation in mixup and RICAP.

| | SVHN | | | mini-ImageNet | | |
|---|---|---|---|---|---|---|
| Budget: | 0.2 | 0.3 | 0.5 | 0.2 | 0.3 | 0.5 |
| Standard data augmentation | | | | | | |
| SGD ($B=1$) | | 97.02 ± 0.05 | | | 75.19 ± 0.16 | |
| *unif-SGD* | 96.56 ± 0.12 | 96.78 ± 0.13 | 96.95 ± 0.07 | 70.87 ± 0.56 | 72.19 ± 0.43 | 73.88 ± 0.42 |
| *p-SGD* | 96.52 ± 0.03 | 96.75 ± 0.03 | 96.84 ± 0.06 | 71.05 ± 0.29 | 72.39 ± 0.45 | 73.66 ± 0.39 |
| *SB* | 96.93 ± 0.07 | 96.85 ± 0.01 | 96.97 ± 0.06 | 69.68 ± 0.09 | 71.46 ± 0.15 | 73.51 ± 0.30 |
| RandAugment data augmentation (N = 2, M = 4) | | | | | | |
| SGD ($B=1$) | | 97.59 ± 0.14 | | | 74.15 ± 0.22 | |
| *unif-SGD* | 97.38 ± 0.05 | 97.50 ± 0.07 | 97.60 ± 0.05 | 71.29 ± 0.25 | 73.04 ± 0.34 | 73.21 ± 0.52 |
| *p-SGD* | 97.25 ± 0.03 | 97.44 ± 0.02 | 97.52 ± 0.03 | 71.43 ± 0.25 | 72.36 ± 0.15 | 73.21 ± 0.38 |
| *SB* | 97.42 ± 0.09 | 97.43 ± 0.19 | 97.56 ± 0.05 | 67.17 ± 2.51 | 71.69 ± 0.31 | 73.28 ± 0.03 |
| mixup data augmentation ($\alpha = 0.3$) | | | | | | |
| SGD ($B=1$) | | 97.24 ± 0.03 | | | 76.28 ± 0.28 | |
| *unif-SGD* | 96.99 ± 0.09 | 97.04 ± 0.08 | 97.24 ± 0.07 | 72.50 ± 0.51 | 73.76 ± 0.26 | 75.05 ± 0.29 |
| *p-SGD* | 96.92 ± 0.08 | 97.34 ± 0.49 | 97.37 ± 0.49 | 72.21 ± 0.81 | 73.63 ± 0.13 | 74.54 ± 0.53 |
| *SB* | 96.80 ± 0.09 | 96.92 ± 0.09 | 96.96 ± 0.09 | 70.12 ± 0.51 | 72.01 ± 0.72 | 73.76 ± 0.36 |
| RICAP data augmentation ($\alpha = 0.3$) | | | | | | |
| SGD ($B=1$) | | 97.61 ± 0.06 | | | 78.75 ± 0.40 | |
| *unif-SGD* | 97.47 ± 0.04 | **97.62 ± 0.16** | 97.55 ± 0.04 | 73.56 ± 0.24 | 75.15 ± 0.45 | 77.20 ± 0.33 |
| *p-SGD* | **97.48 ± 0.08** | 97.45 ± 0.06 | **97.57 ± 0.05** | **73.67 ± 0.60** | **75.46 ± 0.27** | **77.25 ± 0.47** |
| *SB* | 97.34 ± 0.03 | 97.40 ± 0.06 | 97.45 ± 0.01 | 53.26 ± 0.71 | 71.75 ± 0.67 | 75.65 ± 0.40 |

Given that the cost of the data augmentation policies used is negligible (see Table 5 for the wall-clock times when $B = 0.3$), our results show that adequate data augmentation alone can reduce training time at no accuracy cost and in some cases with a considerable increase in accuracy. For example, a 70% reduction in training time (0.3 budget) corresponds to an increase in accuracy from 75.44% to 76.27% in CIFAR-100 and from 94.80% to 94.93% in CIFAR-10. Also, a 50% reduction (0.5 budget) corresponds to an increase in accuracy from 75.44% to 77.83% in CIFAR-100 and from 94.80% to 95.47% in CIFAR-10.

Table 5: Wall-clock time (minutes) in CIFAR-100 for a training of 0.3 of budget.

| Approaches: | *unif-SGD* | *p-SGD* | *SB* |
|---|---|---|---|
| Standard data augmentation | 47 | 48 | 91 |
| RandAugment | 48 | 48 | 93 |
| mixup | 48 | 48 | 93 |
| RICAP | 49 | 49 | 95 |

We also experimented with extremely low budgets (see Table 6) and found that importance sampling approaches (*p-SGD* and *SB*) still bring little improvement over uniform random sampling (*unif-SGD*). Here additional data augmentation does not bring a significant improvement in accuracy and in the most challenging cases, hinders convergence. For example, when introducing RICAP with $B = 0.05$, the accuracy drops approximately 2 points in CIFAR-10, 5 points in CIFAR-100, and 7 points

Table 6: Test accuracy for CIFAR-10/100 and mini-ImageNet under extreme budgets.

| | CIFAR-10 | | CIFAR-100 | | mini-ImageNet | |
|---|---|---|---|---|---|---|
| Budget: | 0.05 | 0.1 | 0.05 | 0.1 | 0.05 | 0.1 |
| Standard data augmentation | | | | | | |
| *unif-SGD* | 87.90 ± 0.40 | 91.46 ± 0.08 | **62.66 ± 0.65** | 69.34 ± 0.68 | 56.38 ± 0.11 | 67.61 ± 0.52 |
| *p-SGD* | **88.86 ± 0.17** | 91.66 ± 0.11 | 62.20 ± 0.56 | 69.32 ± 0.17 | **56.95 ± 0.43** | 67.67 ± 0.41 |
| *SB* | 79.45 ± 4.31 | **92.66 ± 0.14** | 50.53 ± 2.27 | 68.29 ± 0.68 | 11.19 ± 3.46 | 61.25 ± 1.76 |
| RandAugment data augmentation (N = 2, M = 4) | | | | | | |
| *unif-SGD* | 83.24 ± 0.06 | 88.95 ± 0.22 | 47.64 ± 3.34 | 64.48 ± 0.10 | 42.35 ± 1.54 | 64.98 ± 0.47 |
| *p-SGD* | 83.94 ± 0.26 | 89.77 ± 0.38 | 48.78 ± 1.48 | 65.05 ± 0.37 | 41.72 ± 0.77 | 65.88 ± 0.15 |
| *SB* | 32.21 ± 4.14 | 33.86 ± 5.02 | 5.05 ± 0.64 | 5.05 ± 0.64 | 5.61 ± 0.66 | 5.94 ± 0.13 |
| mixup data augmentation ($\alpha = 0.3$) | | | | | | |
| *unif-SGD* | 87.33 ± 0.42 | 91.74 ± 0.04 | 59.90 ± 0.71 | **70.43 ± 0.45** | 53.13 ± 0.83 | **68.54 ± 0.98** |
| *p-SGD* | 87.56 ± 0.67 | 91.59 ± 0.17 | 59.68 ± 0.71 | 70.31 ± 0.10 | 54.20 ± 0.95 | 68.39 ± 0.46 |
| *SB* | 77.72 ± 5.31 | 92.56 ± 0.15 | 43.27 ± 7.37 | 69.64 ± 0.24 | 12.10 ± 0.27 | 61.01 ± 0.64 |
| RICAP data augmentation ($\alpha = 0.3$) | | | | | | |
| *unif-SGD* | 85.61 ± 0.24 | 91.32 ± 0.28 | 55.85 ± 0.51 | 69.43 ± 0.33 | 48.95 ± 0.65 | 67.26 ± 0.63 |
| *p-SGD* | 85.57 ± 0.70 | 90.94 ± 0.16 | 56.09 ± 0.71 | 70.05 ± 0.07 | 49.35 ± 0.60 | 67.27 ± 0.85 |
| *SB* | 44.93 ± 2.67 | 54.76 ± 4.31 | 10.75 ± 0.72 | 13.33 ± 0.39 | 8.71 ± 0.45 | 10.84 ± 0.86 |

in mini-ImageNet with respect to 87.90%, 62.66%, and 56.38% for *unif-SGD* with standard data augmentation.

# 5 Conclusion

This paper studied DNN training for image classification when the number of iterations is fixed (i.e. budgeted training) and explores the interaction of importance sampling techniques and data augmentation in this setup. We empirically showed that, in budgeted training, DNNs prefer variability over selection of important samples: adequate data augmentation surpasses state-of-the-art importance sampling methods and allows for up to a 70% reduction of the training time (budget) with no loss (and sometimes an increase) in accuracy. In future work, we plan to explore the limitations found in extreme budgets and extend the study to large-scale datasets where training DNNs becomes a longer process. Additionally, we find particularly interesting as future work to study the generalization of the conclusions presented in this paper to different tasks, types of data, and model architectures. Finally, we encourage the use of data augmentation techniques rather than importance sampling approaches in scenarios where the iterations budget is restricted, and motivate research on these scenarios to better exploit computational resources.

# Acknowledgment

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under grant number SFI/15/SIRG/3283 and SFI/12/RC/2289_P2.

# References

[1] Guillaume Alain, Alex Lamb, Chinnadhurai Sankar, Aaron Courville, and Yoshua Bengio. Variance reduction in sgd by distributed importance sampling. In *International Conference on Learning Representations (ICLR)*, 2016.

[2] Paul Albert, Diego Ortego, Eric Arazo, Noel E. O'Connor, and Kevin McGuinness. Relab: Reliable label bootstrapping for semi-supervised learning. In *International Joint Conference on Neural Networks (IJCNN)*, 2021.

[3] Hadi Amiri, Timothy Miller, and Guergana Savova. Repeat before forgetting: Spaced repetition for efficient and effective training of neural networks. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[4] Eric Arazo, Diego Ortego, Paul. Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised Label Noise Modeling and Loss Correction. In *International Conference on Machine Learning (ICML)*, 2019.

[5] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *International Joint Conference on Neural Networks (IJCNN)*, 2020.

[6] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, 2020.

[7] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning (ICML)*, 2009.

[8] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. MixMatch: A Holistic Approach to Semi-Supervised Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[9] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 2018.

[10] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations (ICLR)*, 2019.

[11] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[12] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations (ICLR)*, 2019.

[13] Hao Cheng, Dongze Lian, Bowen Deng, Shenghua Gao, Tao Tan, and Yanlin Geng. Local to global learning: Gradually adding classes for training deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[14] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[15] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRw)*, 2020.

[16] Bin Dai, Chen Zhu, Baining Guo, and David Wipf. Compressing neural networks using the variational information bottleneck. In *International Conference on Machine Learning (ICML)*, 2018.

[17] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[19] Yanbo Fan, Siwei Lyu, Yiming Ying, and Baogang Hu. Learning with average top-k loss. In *Advances in Neural Information Processing systems (NeurIPS)*, 2017.

[20] Siddharth Gopal. Adaptive sampling for sgd by exploiting side information. In *International Conference on Machine Learning*, 2016.

[21] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning (ICML)*, 2019.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[24] George Ioannou, Thanos Tagaris, and Andreas Stafylopatis. Improving the convergence speed of deep neural networks with biased sampling. In *International Conference on Advances in Artificial Intelligence (ICAAI)*, 2019.

[25] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[26] Angela H Jiang, Daniel L-K Wong, Giulio Zhou, David G Andersen, Jeffrey Dean, Gregory R Ganger, Gauri Joshi, Michael Kaminksy, Michael Kozuch, Zachary C Lipton, et al. Accelerating deep learning by focusing on the biggest losers. *arXiv:1910.00762*, 2019.

[27] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning (ICML)*, pages 2304–2313. PMLR, 2018.

[28] Tyler B Johnson and Carlos Guestrin. Training deep models faster with robust, approximate importance sampling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[29] Mohammad Kachuee, Orpaz Goldstein, Kimmo Karkkainen, Sajad Darabi, and Majid Sarrafzadeh. Opportunistic learning: Budgeted cost-sensitive learning from data streams. In *International Conference on Machine Learning (ICML)*, 2019.

[30] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning (ICML)*, 2018.

[31] Kenji Kawaguchi and Haihao Lu. Ordered sgd: A new stochastic optimization framework for empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

[32] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[33] Junnan Li, Richard Socher, and Steven CH Hoi. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[34] Mengtian Li, Ersin Yumer, and Deva Ramanan. Budgeted training: Rethinking deep neural network training under resource constraints. In *International Conference on Learning Representations (ICLR)*, 2020.

[35] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[37] Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv:1511.06343*, 2015.

[38] Haihao Lu and Rahul Mazumder. Randomized gradient boosting machine. *arXiv:1810.10158*, 2018.

[39] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *European Conference on Computer Vision (ECCV)*, September 2018.

[40] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning (ICML)*, 2020.

[41] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[42] Feng Nan and Venkatesh Saligrama. Adaptive classification for prediction under a budget. In *Advances in neural information processing systems (NeurIPS)*, 2017.

[43] Deanna Needell, Rachel Ward, and Nati Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in neural information processing systems (NeurIPS)*, 2014.

[44] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A.Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[45] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[46] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *arXiv: 2009.00236*, 2020.

[47] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.

[48] Leslie N Smith. Cyclical learning rates for training neural networks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[49] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 2019.

[50] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[51] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *Asian Conference on Machine Learning (ACML)*, 2018.

[52] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[53] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations (ICLR)*, 2018.

[54] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[55] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[56] Daphna Weinshall, Gad Cohen, and Dan Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *International Conference on Machine Learning (ICML)*, 2018.

[57] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[58] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *European Conference on Computer Vision (ECCV)*, 2018.

[59] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[60] Cheng Zhang, Cengiz Öztireli, Stephan Mandt, and Giampiero Salvi. Active mini-batch sampling using repulsive point processes. In *AAAI Conference on Artificial Intelligence*, 2019.

[61] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[62] Jiong Zhang, Hsiang-Fu Yu, and Inderjit S Dhillon. Autoassist: A framework to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[63] Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *International Conference on Machine Learning (ICML)*, 2015.

[64] Linjun Zhou, Peng Cui, Xu Jia, Shiqiang Yang, and Qi Tian. Learning to select base classes for few-shot classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.