ON THE DIFFICULTY OF WARM-STARTING NEURAL NETWORK TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

In many real-world deployments of machine learning systems, data arrive piecemeal. These learning scenarios may be passive, where data arrive incrementally due to structural properties of the problem (e.g., daily financial data) or active, where samples are selected according to a measure of their quality (e.g., experimental design). In both of these cases, we are building a sequence of models that incorporate an increasing amount of data. We would like each of these models in the sequence to be performant and take advantage of all the data that are available to that point. Conventional intuition suggests that when solving a sequence of related optimization problems of this form, it should be possible to initialize using the solution of the previous iterate—to "warm start" the optimization rather than initialize from scratch—and see reductions in wall-clock time. However, in practice this warm-starting seems to yield poorer generalization performance than models that have fresh random initializations, even though the final training losses are similar. While it appears that some hyperparameter settings allow a practitioner to close this generalization gap, they seem to only do so in regimes that damage the wall-clock gains of the warm start. Nevertheless, it is highly desirable to be able to warm-start neural network training, as it would dramatically reduce the resource usage associated with the construction of performant deep learning systems. In this work, we take a closer look at this empirical phenomenon and try to understand when and how it occurs. Although the present investigation did not lead to a solution, we hope that a thorough articulation of the problem will spur new research that may lead to improved methods that consume fewer resources during training.

1 INTRODUCTION

Although machine learning research generally assumes the existence of a fixed set of training data, real-life machine learning systems face more complicated situations. One particularly common scenario is where a production machine learning system must be constantly updated with new data. This situation occurs in finance, online advertising, recommendation systems, fraud detection, and many other domains where machine learning systems are used for prediction and decision making in the real world (He et al., 2014; Chandramouli et al., 2011; Kuncheva, 2008). When the new data arrive, the model needs to be updated so that it can be as accurate as possible and to also account for any domain shift that is occurring.

As a concrete example, consider a large-scale social media website, to which users are constantly uploading images and text. The company would like to have up-to-the-minute predictive models in order to recommend content, filter out inappropriate media, and determine what advertisements to show. There might be millions of new data arriving every day, the structure of which needs to be rapidly incorporated into the production machine learning systems.

It is natural in this scenario to imagine maintaining, in effect, a single model that is updated with the latest data at a regular cadence. Every day, for example, new training might be performed on the model with the updated,

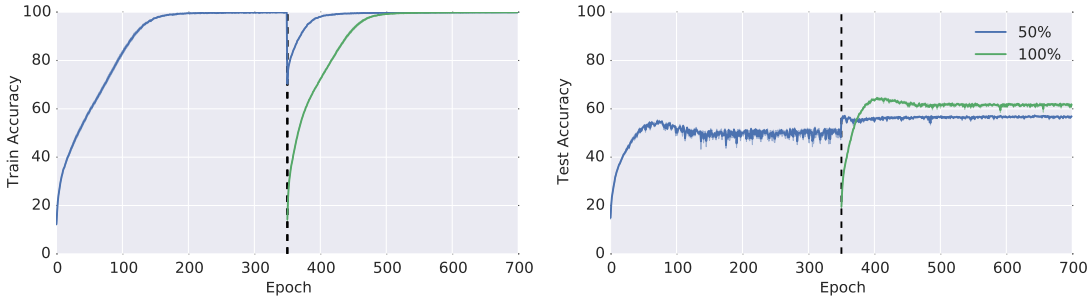

Figure 1: A comparison between ResNets trained using a warm start and a random initialization on the CIFAR-10 dataset. Blue lines correspond to models that were trained on 50% of CIFAR-10 for 350 epochs then trained using 100% of the data for an additional 350 epochs. Green lines correspond to models trained using 100% of the data from the start. The two training paradigms produce similar training performance but significantly and consistently differing test performance.

larger data set. Ideally, this new training procedure is initialized from the parameters of yesterday's model, i.e., it is "warm-started" from those parameters rather than a fresh initialization. Such an initialization makes intuitive sense: the data used yesterday are mostly the same as the data today, and it seems wasteful to throw away all the previous computation. For linear models and convex optimization problems, such warm starting is widely used and highly successful, e.g., He et al. (2014); the theoretical properties of online learning are well understood for such problems. **However, warm-starting seems to hurt generalization in deep neural networks.** This is particularly troubling, because warm-starting *does not* damage training accuracy.

Figure 1 shows a simple illustration of this phenomenon. Three 18-layer ResNets have been trained on the CIFAR-10 natural image classification task to create these figures. One was trained on 100% of the data, one was trained on 50% of the data, and a third warm-started model was trained on 100% of the data but initialized from the parameters found from the 50% training. All three achieve the upper bound on training accuracy. However, the warm-started network performs worse on test samples than the network trained on the same data but with a good random initialization. Problematically, **this phenomenon incentivizes performance-focused researchers and engineers to constantly retrain models from scratch**, at potentially enormous financial and environmental cost (Strubell et al., 2019); this is an example of "Red AI" as articulated in Schwartz et al. (2019) , which disregards resource consumption in the pursuit of raw predictive performance.

We observe that the warm-starting phenomenon has implications for other situations of interest as well. In active learning, for example, unlabeled samples are abundant but the labels themselves are expensive. The goal of the learner is to identify samples that would maximally improve its hypothesis, then send those to an oracle to be labeled and integrated into the existing training set. These decisions are best made using a model that has been fitted to every item in the training set; a model that is trained on samples that were chosen one at a time will almost always outperform a model that is trained on samples that were chosen in batches (Yang and Carbonell, 2013). It would be time efficient, then, to simply update the model each time a sample is appended to the training set, rather than retrain the model from scratch at every iteration of sample acquisition. However, this kind of update seems to damage generalization. In fact, although this phenomenon has not received much direct attention from the research community, some recent papers on deep active learning mention that their model is retrained at every step of data collection (Sener and Savarese, 2018; Ash et al., 2019), and popular deep active learning repositories on Github (Rostamiz, 2017–2019; Huang, 2018–2019) all retrain models from scratch. This caveat limits the feasibility of active learning for deep neural networks in the absence of the ability to effectively warm-start training.

The ineffectiveness of warm-starting has been observed anecdotally in the community, but this paper seeks to examine its empirical properties more closely in controlled settings. Note that the findings in this paper are not inconsistent with extensive work on unsupervised pre-training (Erhan et al., 2010; Bengio, 2011) and transfer learning in the small-data and "few shot" problem regimes (Vinyals et al., 2016; Snell et al.,

Table 1: Validation percent accuracies for various optimizers and models for both warm-started and randomly initialized models on various indicated datasets. We consider an 18-layer ResNet, three-layer multilayer perceptron (MLP), and logistic regression (LR) as our classifiers. Validation sets are a randomly-chosen third of the training data. Standard deviations are indicated parenthetically. Accuracies for the first-round of training warm-started models are in Apendix Table 6.

	RESNET	RESNET	MLP	MLP	LR	LR
CIFAR-10	SGD	ADAM	SGD	ADAM	SGD	ADAM
RANDOM INIT	56.2 (1.0)	78.0 (0.6)	39.0 (0.2)	39.4 (0.1)	40.5 (0.6)	33.8 (0.6)
WARM START	51.7 (0.9)	74.4 (0.9)	37.4 (0.2)	36.1 (0.3)	39.6 (0.2)	33.3 (0.2)
SVHN						
RANDOM INIT	89.4 (0.1)	93.6 (0.2)	76.5 (0.3)	76.7 (0.4)	28.0 (0.2)	22.4 (1.3)
WARM START	87.5 (0.7)	93.5 (0.4)	75.4 (0.1)	69.4 (0.6)	28.0 (0.3)	22.2 (0.9)
CIFAR-100						
RANDOM INIT	18.2 (0.3)	41.4 (0.2)	10.3 (0.2)	11.6 (0.2)	16.9 (0.18)	10.2 (0.4)
WARM START	15.5 (0.3)	35.0 (1.2)	9.4 (0.0)	9.9 (0.1)	16.3 (0.28)	9.9 (0.3)

2017; Finn et al., 2017). Rather here we are examining how to accelerate training in the large-data supervised regime in a way that is consistent with expectations from convex problems. The ultimate goal is to achieve an understanding that can lead to procedures for successful warm starting and reduced resource usage in common real-world retraining scenarios. This paper is fundamentally about investigating and reporting on a phenomenon that should be of broad interest to researchers and practitioners. Our investigation has not yet revealed a solution, but we hope it will spur a broader discussion about techniques for reducing the resource consumption of deep neural network training.

2 WARM STARTING DAMAGES GENERALIZATION

In this section we provide empirical evidence that warm starting consistently damages generalization performance in neural networks. To that end, we conduct a series of experiments across a several different architectures, optimizers, and image datasets. Our goal is to create simple, reproducible settings that are reflective of real-world situations, in which the warm-starting phenomenon is observable.

2.1 BASIC BATCH UPDATING

In order to clearly highlight the basic problem, we consider the simplest case of warm-starting, in which a single training dataset is partitioned into two subsets that are presented sequentially. In each series of experiments, we randomly segment the training data into two batches. The model is trained to convergence on the first half, then is trained on the union of the two batches, i.e., 100% of the data. This is repeated for modern deep learning classifiers (ResNet-18 of He et al. (2016), multilayer perceptrons (MLPs) consisting of three layers and tanh activations, and logistic regression. Models are optimized using either stochastic gradient descent or the Adam variant of stochastic gradient descent (Kingma and Ba, 2014), and are fitted to the standard CIFAR-10, CIFAR-100, and SVHN classification datasets. All models are trained using a mini-batch size of 128 and a learning rate of 0.001, the smallest learning rate used in the learning schedule for fitting state-of-the-art ResNet models (He et al., 2016). We further investigate the effect of these parameters in Section 3.

Our results (shown in Table 1) indicate that generalization performance is damaged consistently and significantly for both ResNets and MLPs. This effect is more dramatic when data are more challenging to model, as in CIFAR-10, for which obtaining competitive validation performance generally requires data augmentation and a sophisticated architecture. When the classification problem is believed to be relatively easy, as in SVHN, the warm-start generalization gap is less pronounced. Conversely, logistic regression, which enjoys a convex loss surface, is not significantly damaged by warm starting. Note that magnitude of the generalization gap is inversely proportional to the fraction of data available in the first-round of training (Appendix Figure 8).

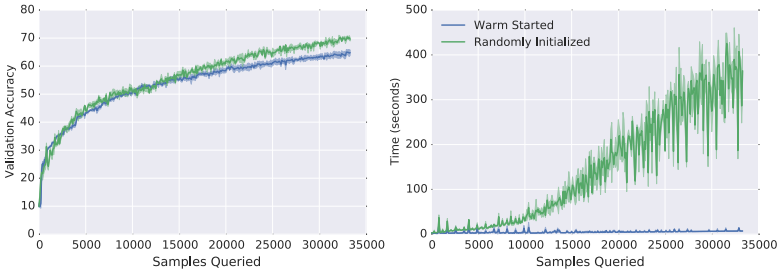

Figure 2: A passive online learning experiment for CIFAR-10 data using a ResNet. The horizontal axis shows the total number of samples in the training set available to the learner. Notice the growing generalization gap between warm-started and randomly-initialized models.

This result is surprising. Even though MLP and ResNet optimization is non-convex, conventional intuition suggests that the warm-started solution should be close to the full-data solution and therefore a good initialization. One view on pre-training is that the initialization is a "prior" on weights; we often view prior distributions as arising from inference on old (or hypothetical) data and so this sort of pre-training should always be helpful. The generalization gap shown here creates a computational burden for real-life machine learning systems that must be retrained from scratch to perform well, rather than initialized from previous models.

2.2 Online Learning

Passive Online Learning. A common real-world setting involves data that are being provided to the machine learning system in a stream. At every step, the learner is given k new samples to append to its training data, and it updates its hypothesis to reflect the larger dataset. Financial data, social media, and recommendations systems are common examples of scenarios where new data are constantly arriving. This paradigm is simulated in Figure 2, where we supply CIFAR-10 data, selected randomly without replacement, in groups of 100 to an 18-layer ResNet. We examine two cases: 1) where the model is retrained from scratch after each batch, starting from a random initialization, and 2) where the model is trained to convergence starting from the parameters learned in the previous iteration. In both cases, the models are optimized with Adam, using an initial learning rate of 0.001. Each was run five times with different random seeds and validation sets composed of a random third of the training data, reinitializing Adam's parameters at each step of learning (see Section 3.6) . Figure 2 shows the trade-off between the two approaches. The mean and standard deviations across the five runs are shown. On the right are the training times: clearly, starting from the previous model is preferable and has the potential to vastly reduce computational costs and wall-clock time. However, as can be seen on the left, generalization performance is worse in the warm-started situation. As more data arrive, the gap in validation accuracy increases substantially.

Active Online Learning. Instead of absorbing a stream of data, it may be possible to perform *active learning* (Mitchell et al., 1990) and choose what data to add to the dataset. When the data are expensive to acquire, it's sometimes advantageous to select samples carefully. In this situation, the data are often chosen according to a metric that prefers data which are highly informative about the predictive hypothesis or which are likely to maximally reduce expected risk.

Active learning is closely related to optimal experimental design in statistics. Generally, the setup is to allow the model to choose k new data samples for which it will be provided labels. These newly labeled data are then added to the training set and the model is updated. As in the passive scenario described in Section 2.2, we must choose whether to retrain the model from scratch with the new data or warm-start from the previous iteration.

While there are many algorithms for active learning, we study this scenario using margin-based sampling (Wang and Shang, 2014). This approach quantifies predictive uncertainty as inversely proportional to the difference between the model's largest and second largest prediction. As in Section 2.2, the algorithm receives 100 new training data at each round of learning, selected according to the margin criterion.

Domain Shift. Figure 10 in the Appendix shows that active selection of data does not significantly change the generalization gap between fresh initializations and warm starting. Relative to the passive scenario, more data are required before the warm-started model starts to underperform, but a gap nevertheless appears. As before, this is over five random restarts and the right-hand subfigure shows the dramatic difference in computational cost.

Many online learning scenarios involve *domain shift*, in which the distribution is shifting as data stream in. Such a shift could arise, for example, due to macroecomic changes effecting financial markets, or seasonal changes influencing people's tastes in a recommendation system. In general, one would expect domain shift to make the warm-starting phenomenon worse, as the initialization is now even farther from the final solution and indeed what the experiments show.

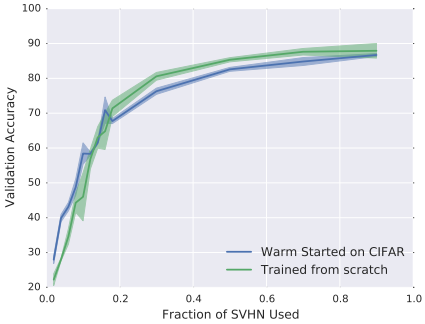

Figure 3: An SVHN transfer learning experiment that compares training from scratch to pretraining on CIFAR for various fractions of the SVHN dataset. Notice that warm starting is helpful when there is not much labeled data available but it becomes damaging as more labeled data are supplied.

We simulate domain shift by sorting the data according to their first principal component and supplying them to the learner in order. This induces nonstationarity across batches, reflecting large-scale shifts in, e.g., illumination. Figure 9 in the Appendix shows these deleterious effects.

2.3 Transfer Learning

Despite successes on a variety of machine learning tasks, deep neural networks are still data hungry and generally require large training sets to generalize well. For problems where only limited data are available, it has become popular to warm-start learning using the parameters from training on a different but related problem (Finn et al., 2017; Nichol and Schulman, 2018). Transfer and "few-shot" learning in this form has seen success in computer vision and NLP (Mou et al., 2016).

The experiments we perform here, however, imply that when the second problem is not data-limited we would expect to see this transfer learning approach deteriorate in quality. That is, at some point, the pre-training transfer learning approach is essentially the same as warm-starting under domain shift, and the generalization performance should suffer.

We demonstrate this phenomenon by first training a ResNet-18 to convergence on CIFAR-10, then using that solution to warm start a model trained on a varying fraction of the SVHN dataset. When only a small portion of SVHN is used, this is essentially the same as the pre-training transfer learning approach. As the proportion increases, the problem turns into what we have described here as warm-starting. Figure 3 shows the result of this experiment, and it appears to support our intuition. When the second dataset is small, warm-starting is helpful, but there is a crossover point where better generalization would be achieved from training from scratch on that fraction of SVHN. All models are optimized using SGD with a learning rate of 0.001.

3 Overcoming the Warm-Start Problem

The design space for initializing and training deep neural network models is very large, and so it is important to evaluate whether there is some trick that could be used to help warm-started training find good solutions. Put another way, a reasonable response to this problem is "Did you see whether X helped?" where X might be anything from batch normalization (Ioffe and Szegedy, 2015) to increasing the mini-batch size (Keskar et al., 2016). This section tries to answer some of these questions and further empirically probe the warm-start phenomenon. Unless otherwise stated, experiments in this section use a ResNet-18 model trained using SGD with a learning rate of 0.001 on CIFAR-10 data. All experiments were run five times to report means and variances.

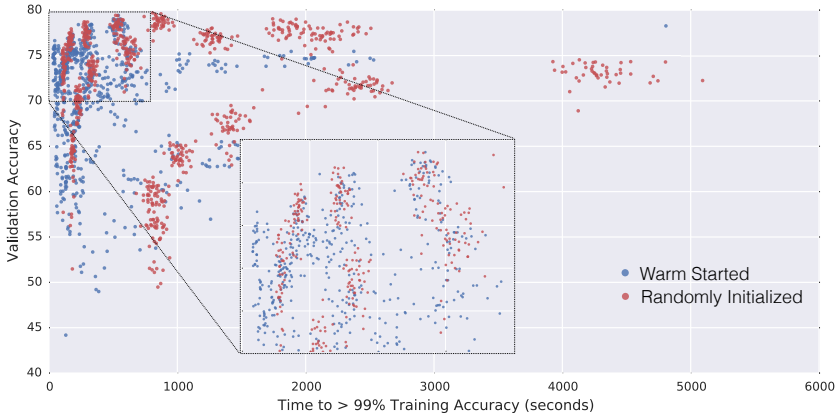

Figure 4: A comparison between ResNets trained from both a warm start and random initialization on CIFAR-10 for various hyperparameters. Red dots are randomly-initialized models and blue dots are warm-started models. Warm-started models that perform roughly as well as randomly-initialized models offer no benefit in terms of training time.

3.1 Is this an effect of batch size or learning rate?

One might reasonably ask whether or not there exist *any* hyperparameters that close the generalization gap between warm started and randomly initialized models. In particular, can setting a larger learning rate at either the first or second round of learning help the model escape to regions that generalize better? Can shrinking the batch size inject stochasticity that might improve generalization (Li et al., 2018b; Gunasekar et al., 2017)?

Here we again consider a warm-started experiment of training on 50% of CIFAR-10 until convergence, then training on 100% of CIFAR-10, using the initial round of training as an initialization. We explore all combinations of batch sizes $\{16, 32, 64, 128\}$, and learning rates $\{0.001, 0.01, 0.1\}$, varying them across the three rounds of training. This allows for the possibility that there exist different hyperparameters for the first stage of training that are better when used with a different set after warm-starting. Each of these combinations is run with three random initializations.

Figure 4 visualizes these results. Every resulting 100% model is shown from all three initializations and all combinations, with color indicating whether it was a random initialization or a warm-start. The horizontal axis shows the time to completion, excluding the pre-training time, and the vertical axis shows the resulting validation performance.

Interestingly, we do find warm-started models that perform as well as randomly-initialized models, but they are unable to do so while benefiting from their warm-started initialization. The training time for warm-started ResNet models that generalize as well as randomly-initialized models is roughly the same as those randomly-initialized models. That is, there is no computational benefit to using these warm-started initializations. It is worth noting that this plot does not capture the time or energy required to identify hyperparameters that close the generalization gap; such hyperparameter searches are often the culprit in the resource footprint of deep learning (Schwartz et al., 2019). Wall-clock time is measured by assigning every model identical resources, which includes 50GB of RAM and an NVIDIA Tesla P100 GPU.

This increased fitting time occurs because warm-started models, when using hyperparameters that generalize relatively well, seem to "forget" what was learned in the first round of training. Figure 5 demonstrates this phenomenon by computing the Pearson correlation between the weights of converged warm-started models and their initialization weights, again across various choices for learning rate and batch size, and comparing it to validation accuracy. Models that generalize well have little correlation with their initialization—there is a trend downward in accuracy with increasing correlation—suggesting that they have forgotten what was learned in the first round of training. Conversely, a similar plot for logistic regression shows no such relationship, and some of the best models have large correlations.

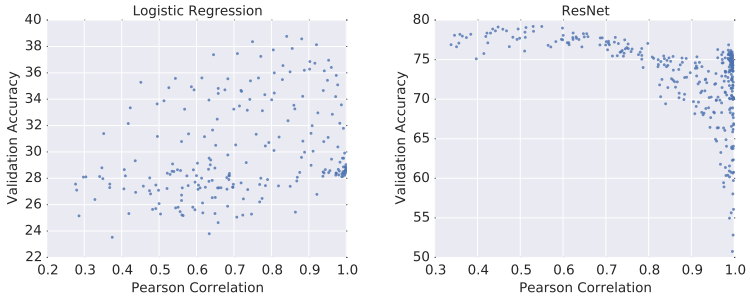

Figure 5: Validation accuracy as a function of the correlation between the warm-start initialization and the solution found after training for a large number of hyperparameter settings. **Left**: Warm-started logistic regressors often remember their initialization. **Right**: Warm-started ResNets that perform well do not retain much information from the initial round of training.

3.2 How quickly is generalization damaged?

One of the surprising results of our investigation is how little training is necessary to damage the validation performance of the warm-started model. Our hope was that warm-starting success might be achieved by switching from the 50% to 100% phase *before the first phase of training was completed*. We did a search over switching times to try to identify whether there might be a "sweet spot" in which a partially-trained checkpoint might provide a good initialization. We fit a ResNet-18 model on 50% of the training data, as before, and checkpointed its parameters every five epochs. We then took each of these checkpointed models and used them as an initialization for training on 100% of those data. As shown in Figure 7, generalization is damaged even when initializing from parameters obtained by training on incomplete data for only a few epochs.

3.3 Are there differences in the gradients of warm-started models?

We notice some interesting phenomena when examining the L_2 norm of the gradients as training progresses for a warm-started problem. Figure 6 shows a visualization of average gradients during training for a ResNet on CIFAR-10. On the left is the sequence of gradients during the first phase of training. On the right is the second phase of training, where the magnitudes are shown separately for the two halves of the data set. It is unsurprising that the "old" data have zero gradients in the beginning, but it is surprising that the gradient magnitudes of the "new" data are higher even than at the random initialization. This suggests they induce large changes to the model even though it has already been fitted on samples from the same distribution. Moreover, the average magnitudes are very slow to converge; the new data have consistently larger gradients.

3.4 Is regularization helpful?

The phenomenon of larger gradients induced by the addition of new data suggests that warm-started models might benefit from regularization. Here we investigate three different approaches to regularization: 1) basic L_2 weight penalties (Krogh and Hertz, 1992), 2) confidence-penalized training (Pereyra et al., 2017), and 3) adversarial training (Szegedy et al., 2014). We again take a ResNet fitted to 50% of available training data and use its parameters to warm-start learning on 100% of those data. Regularization is applied in both rounds of training. Table 2 shows the results of these experiments. Regularization often helps, but it does not resolve the generalization gap created by warm-starting. We regularize both stages using the regularization scheme and penalty size indicated in Table 2. Still, applying the same regularization to randomly-initialized models always produces a better-generalizing classifier.

Table 2: Average validation percent accuracies for various regularizers and regularization penalties with both warm-started (WS) and randomly-initialized (RI) models on CIFAR-10 data.

L2	1×10^{-1}	1×10^{-2}	1×10^{-3}	1×10^{-4}
RI	72.7 (4.2)	55.4 (2.7)	54.6 (2.4)	55.1 (3.4)
WS	63.9 (6.4)	51.2 (2.7)	50.5 (1.8)	50.4 (1.3)
Adversarial				
RI	54.8 (1.3)	55.1 (1.5)	55.3 (1.4)	55.6 (0.9)
WS	52.4 (1.0)	52.6 (1.5)	52.7 (1.2)	50.4 (1.4)
Confidence				
RI	53.1 (1.9)	55.8 (1.3)	55.4 (1.2)	55.9 (1.4)
WS	50.3 (0.7)	50.0 (3.8)	51.2 (1.2)	49.3 (1.2)

Figure 6: The L2 norm of gradients during training. **Left**: The gradients corresponding to 50% of data in the first round of training. **Right**: Gradients corresponding to both the same 50% of data used on the left and the new 50% of data. The first round of training uses a learning rate of 0.0001, the smallest in the default learning rate schedule for ResNets, and the second uses a learning rate of 0.00001, chosen to zoom in on this effect.

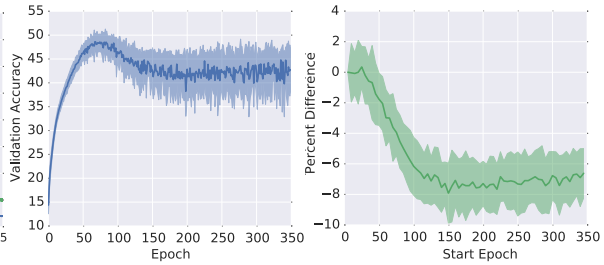

Figure 7: **Left:** CIFAR-10 validation accuracy of a ResNet as training progresses on 50% of the dataset. **Right:** The amount of validation accuracy damage, in terms of percentage difference from random initialization, after training on 100% of the data. Each warm-started model was initialized by training on 50% of CIFAR data for the indicated number of epochs.

3.5 Can we warm-start some layers but not others?

A common practice in deep learning is to train on one task, then continue training only the last network layer when new data become available (Mou et al., 2016; Karpathy and Johnson, 2019). This subsection investigates how performance is affected when we train a model on 50% of data, then use that initialization to retrain only the last layer of the network. As the gradient of the last layer affects all earlier layers during training, it is one possible culprit for the vast gradient magnitude differences in Section 3.3.

We examine this hypothesis in two ways, as shown in Table 4. First we ran experiments that fixed all parameters but the last layer to their pre-trained values and then only trained the last layer in the second phase (LL). Second we extend this experiment to a third phase in which the rest of the network was trained after allowing the last layer to converge to something reasonable (LL+WS). While training only the last layer from the warm-started initialization is typically worse than training all parameters (WS), some gains can be had by training the entire network after having only trained the last layer. That is, the last layer alone does not seem to be sufficient to explain this generalization gap.

3.6 Is this a problem with momentum or other persistent quantities during training?

One choice we have when using Adam or other adaptive methods involves whether or not to reset the optimization parameters after the first round of training. Can any part of this generalization gap be explained by a need to reinitialize hyperparameters or to use those that Adam has identified after training for a while? The answer appears to be *no*: allowing hyperparameter settings to follow from the first round of training to the second does not appear to affect out-of-sample performance of the final solution. This procedure produces results that are very similar to Adam in Table 1, which reinitializes the optimizer before warm-start training.

3.7 Can we just add noise?

One idea for overcoming the warm-start problem is to add noise to the warm-started parameters, potentially jostling the network out of a poor initialization. The motivation for this is to have a "partially random" initialization. While adding noise of increased variance does help generalization, warm-started models are still lower-performing than randomly-initialized networks. Even worse, the train time for noised, warm-started models is proportional to the variance of the added noise. The best-generalizing noised warm-started networks actually take longer to fit than freshly-initialized models (Table 3).

Table 3: Validation accuracies and warm-started model train times (minutes). Adding noise at the indicated standard deviations improves generalization, but not to the point of performing as well as randomly-initialized models. Better-generalizing warm-started models take even more time to train than their randomly-initialized peers, which on average achieve **55.2% accuracy in 34.0 minutes**.

	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	0
Accuracy	54.4 (0.9)	53.5 (1.0)	52.9 (1.0)	49.9 (1.6)	50.8 (1.8)
Train Time	165.3 (3.9)	38.0 (1.33)	16.5 (1.3)	14.6 (91.0)	13.6 (0.4)

Table 4: Validation percent accuracies for various datasets for last layer only warm-starting (LL), last layer warm starting followed by full network training (LL+WS), warm started (WS) and randomly initialized (RI) models on various indicated datasets.

	LL	LL+WS	WS	RI
CIFAR-10	48.8 (1.8)	50.9 (1.5)	52.5 (0.3)	56.0 (1.2)
SVHN	86.0 (0.6)	88.2 (0.2)	87.5 (0.7)	89.4 (0.1)
CIFAR-100	16.4 (0.5)	16.5 (0.6)	15.5 (0.3)	18.2 (0.3)

3.8 Is this a case of catastrophic forgetting?

Warm starting is conceptually similar to continual learning. In that framework, an agent is given learning tasks sequentially, and the goal is to become good at the current task while avoiding "catastrophic forgetting," i.e., losing performance on previous tasks. One hypothesis to consider is whether the warm-start phenomenon is simply a case of such catastrophic forgetting. We can examine this hypothesis by trying to fix the warm-start problem using a technique commonly used to prevent catastrophic forgetting. One such approach is Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), which adds a regularization penalty that encourages avoiding updating weights that are important for previously-learned tasks.

However, in the warm-start problem, each round of training adds data to what was available in the previous round. As described here, these data are often from the same distribution as before. Because a model that works well on the second task of the warm-start problem also works well on the first, there is no reason to avoid updating important parameters. Accordingly, as shown in Table 5, including the EWC penalty in warm-started network training actually damages performance more than not including it at all.

Table 5: CIFAR-10 Validation percent accuracies for warm-started ResNets using different degrees of EWC. Standard deviations are indicated parenthetically.

1×10^{-1}	1×10^{-2}	1×10^{-3}	1×10^{-4}
48.8 (2.16)	50.8 (1.0)	51.2 (1.3)	51.9 (0.4)

4 Discussion and Research Surrounding the Warm Start Problem

Warm-starting and online learning are well understood for convex models like linear classifiers (Chu et al., 2015) and SVMs (DeCoste and Wagstaff, 2000; Wen et al., 2017). However, it does not appear that generally applicable techniques exist for deep neural networks that do not damage generalization, and so models are typically retrained from scratch, e.g., Sener and Savarese (2018); Shyam et al. (2018).

There has been a variety of work in closely related areas, however. For example, in analyzing "critical learning periods," researchers show that a network initially trained on blurry images then on sharp images is unable to perform as well as one trained from scratch on sharp images, drawing a parallel between human vision and computer vision (Achille et al., 2018). In this article, we show that this phenomenon occurs more generally and that test performance is damaged even when first and second datasets are drawn from identical distributions.

Initialization. The problem of warm starting is closely related to the rich literature on initialization of neural network training "from scratch". Indeed, new insights into what makes an effective initialization have been critical to the revival of neural networks as machine learning models. While there have been several proposed methods for initialization (Sutskever et al., 2013; Srivastava et al., 2015; Erhan et al., 2010; Glorot and Bengio, 2010; He et al., 2015), this body of literature primarily concerns itself with initializations that are high-quality in the sense that they allow quick and reliable training error minimization.

Work relating initialization to generalization suggests that networks whose weights have moved far from their initialization are less likely to generalize well compared with ones that have remained relatively nearby (Nagarajan and Kolter, 2017). While this makes sense when learning a new problem from scratch, warm-started networks that have *less* in common with their initializations seem to generalize better than those that have more (Figure 5). So while it is not surprising that there exist initializations that generalize poorly, it is surprising that the initializations discussed in Section 1 are in that class.

Generalization. The warm start problem is fundamentally about generalization performance, which has been extensively studied both theoretically and empirically within the context of deep learning. These articles have investigated generalization by studying classifier margin (Bartlett et al., 2017; Wei et al., 2018), loss geometry (Hochreiter and Schmidhuber, 1997; Keskar et al., 2016; Li et al., 2018a), and measurements of complexity (Zhang et al., 2017; Liang et al., 2017), sensitivity (Novak et al., 2018), or compressiblity (Zhou et al., 2019).

These approaches can be seen as attempting to measure the intricacy of the hypothesis learned by the network. If two models are both consistent for the same training data, the one with the less complicated concept is more likely to generalize well. We know that networks trained with SGD are implicitly regularized (Li et al., 2018b; Gunasekar et al., 2017), suggesting that standard training of neural networks incidentally finds low-complexity solutions. It's possible, then, that the initial round of training disqualifies solutions that would most naturally explain the general problem of interest. If so, it would imply that the warm-start problem is a pathology of SGD.

Pre-training. As previously discussed, the warm-start problem is very similar to the idea of unsupervised and supervised pre-training (Li et al., 2019; Bengio, 2011; Erhan et al., 2010; Bengio et al., 2007). Under that paradigm, learning where limited labeled data are available is aided by first training on related data. The warm start problem, however, is not about limited labeled data in the second round of training. Instead, the goal of warm starting is to hasten the time required to fit a neural network by initializing using a similar supervised problem without damaging generalization. Our results suggest that while warm-starting is beneficial when labeled data are limited, it actually damages generalization to warm-start in rich-data situations. This is somewhat opposed to conventional intuition, which suggests that pre-training is helpful when there are limited data available, but that it is not harmful when data are not limited (Karpathy and Johnson, 2019).

Concluding thoughts. This article presented challenges of warm-starting neural network training. While this is a problem that the community seems somewhat aware of anecdotally, it does not seem to have been directly studied. We believe that this is a major problem in important real-life tasks for which neural networks are used, and it speaks directly to the resources consumed by training such models.

We also observe that while the presented experiments show strong evidence that the warm start problem is not something that can be easily overcome with current optimization tools, there are still many avenues to investigate. For example, Polyak averaging (Polyak and Juditsky, 1992), AdaGrad (Duchi et al., 2011), and KFAC (Martens and Grosse, 2015) are techniques that find favor on certain problems and architectures and might make warm-starting possible. Additionally, we have focused on widely-studied image data sets; we leave similar analyses of other tasks such as natural language processing and molecular function prediction to future work.

REFERENCES

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep networks. In *International Conference on Learning Representations*, 2018.

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249, 2017.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *International Conference on Unsupervised and Transfer Learning Workshop*, pages 17–37. JMLR. org, 2011.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.

Badrish Chandramouli, Justin J Levandoski, Ahmed Eldawy, and Mohamed F Mokbel. StreamRec: a real-time recommender system. In *International Conference on Management of Data*, pages 1243–1246. ACM, 2011.

Bo-Yu Chu, Chia-Hua Ho, Cheng-Hao Tsai, Chieh-Yen Lin, and Chih-Jen Lin. Warm start for parameter selection of linear classifiers. In *International Conference on Knowledge Discovery and Data Mining*, pages 149–158. ACM, 2015.

Dennis DeCoste and Kiri Wagstaff. Alpha seeding for support vector machines. In *International Conference on Knowledge Discovery and Data Mining*, pages 345–349. ACM, 2000.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. JMLR. org, 2017.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pages 6151–6159, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. Practical lessons from predicting clicks on ads at Facebook. In *Workshop on Data Mining for Online Advertising*, pages 1–9. ACM, 2014.

Sepp Hochreiter and J Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Kuan-Hao Huang. `https://github.com/ej0cl6/deep-active-learning`, 2018–2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Andrej Karpathy and Justin Johnson. Course notes: Transfer learning, CS231n convolutional neural networks for visual recognition. Course Notes, 2019.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *International Conference on Learning Representations*, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, pages 950–957, 1992.

Ludmila I Kuncheva. Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. In *2nd Workshop SUEMA*, volume 2008, pages 5–10, 2008.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pages 6389–6399, 2018a.

Hengduo Li, Bharat Singh, Mahyar Najibi, Zuxuan Wu, and Larry S Davis. An analysis of pre-training on object detection. *arXiv preprint arXiv:1904.05871*, 2019.

Yuanzhi Li, Tengyu Ma, and Hongyang Zhang. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. pages 75:2–47, 2018b.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. pages 888–896, 2017.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning*, pages 2408–2417, 2015.

Tom Mitchell, Bruce Buchanan, Gerald DeJong, Thomas Dietterich, Paul Rosenbloom, and Alex Waibel. Machine learning. *Annual Review of Computer Science*, 4(1):417–433, 1990.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in NLP applications? In *Empirical Methods in Natural Language Processing*, pages 478–489, 2016.

Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. In *Neural Information Processing Systems*, 2017.

Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2, 2018.

Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Rostamiz. `https://github.com/ej0cl6/deep-active-learning`, 2017–2019.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. Green AI. *arXiv preprint arXiv:1907.10597*, 2019.

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

Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International Conference on Machine Learning*, pages 5779–5788, 2018.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in Neural Information Processing Systems*, pages 2377–2385, 2015.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Annual Meeting of the Association for Computational Linguistics*, 2019.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147, 2013.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

Dan Wang and Yi Shang. A new active labeling method for deep learning. In *International Joint Conference on Neural Networks*, pages 112–119. IEEE, 2014.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. On the margin theory of feedforward neural networks. *arXiv preprint arXiv:1810.05369*, 2018.

Zeyi Wen, Bin Li, Ramamohanarao Kotagiri, Jian Chen, Yawen Chen, and Rui Zhang. Improving efficiency of SVM k-fold cross-validation by alpha seeding. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Liu Yang and Jaime Carbonell. Buy-in-bulk active learning. In *Advances in Neural Information Processing Systems*, pages 2229–2237, 2013.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P Adams, and Peter Orbanz. Non-vacuous generalization bounds at the ImageNet scale: a PAC-Bayesian compression approach. In *International Conference on Learning Representations*, 2019.

5 APPENDIX TABLES

Table 6: Validation percent accuracies for various optimizers and models for the first round of warm-started training, i.e. training on half of the training data available in Table 1. We consider an 18-layer ResNet, three-layer multilayer perceptron (MLP), and logistic regression (LR) as our classifiers. Validation sets are a randomly-chosen third of the training data. Standard deviations are indicated parenthetically.

	RESNET SGD	RESNET ADAM	MLP SGD	MLP ADAM	LR SGD	LR ADAM
CIFAR-10	41.7 (7.9)	70.5 (1.6)	37.2 (0.2)	36.0 (0.2)	37.9 (0.2)	31.8 (0.7)
SVHN	85.9 (0.3)	92.3 (0.2)	72.5 (0.4)	67.5 (0.3)	27.1 (0.3)	22.2 (0.7)
CIFAR-100	10.6 (1.6)	31.5 (0.7)	10.3 (0.2)	10.5 (0.3)	15.4 (0.21)	9.3 (0.3)

Table 7: Validation percent accuracies for both warm-started and randomly-initialized ResNets of verying depths on the CIFAR-10 dataset. In the absence of data augmentation, the additional parameters seem to hurt performance for even randomly-initialized models.

Layers	18	34	50
Random Init	56.5 (1.5)	52.9 (1.3)	54.3 (0.3)
Warm Start	51.7 (0.9)	50.2 (1.5)	52.0 (0.5)

Table 8: Training losses for both warm-started and randomly-initialized ResNets of verying depths on the CIFAR-10 dataset. Notice that warm-started models actually achieve lower training error despite worse validation performance.

Layers	18	34	50
Random Init	.017 (.002)	.009 (002)	.019 (.005)
Warm Start	.005 (.001)	.006 (002)	.008 (.001)

6 APPENDIX FIGURES

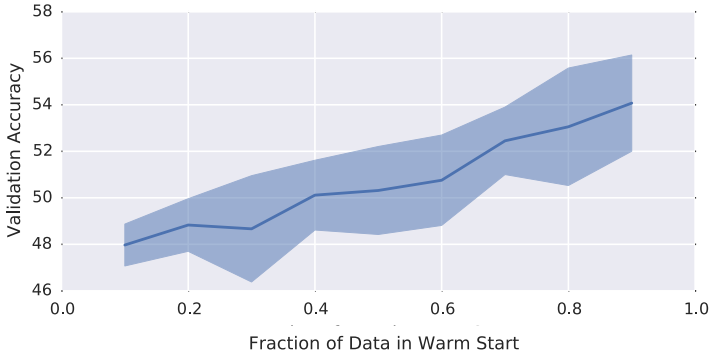

Figure 8: Warm-started ResNet-18 generalization as a function of the fraction of total data available in the first round of training. Models are trained on the indicated fraction of CIFAR-10 training data until convergence, then trained again on 100% of CIFAR-10 training data to produce this figure. When the initial data used to warm-start training more overlaps with the second round of training data, the generalization gap is less severe.

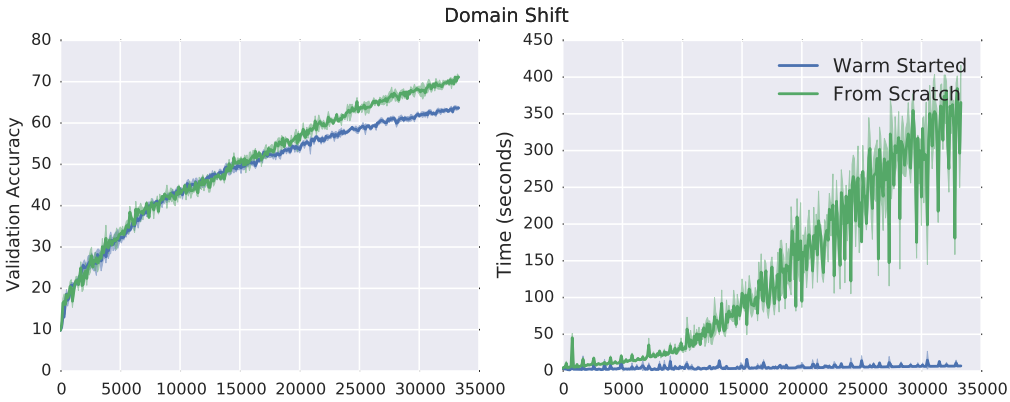

Figure 9: A domain shift online learning experiment for CIFAR-10 data using a ResNet. The horizontal axis displays the number of total samples in the training set available to the learner.

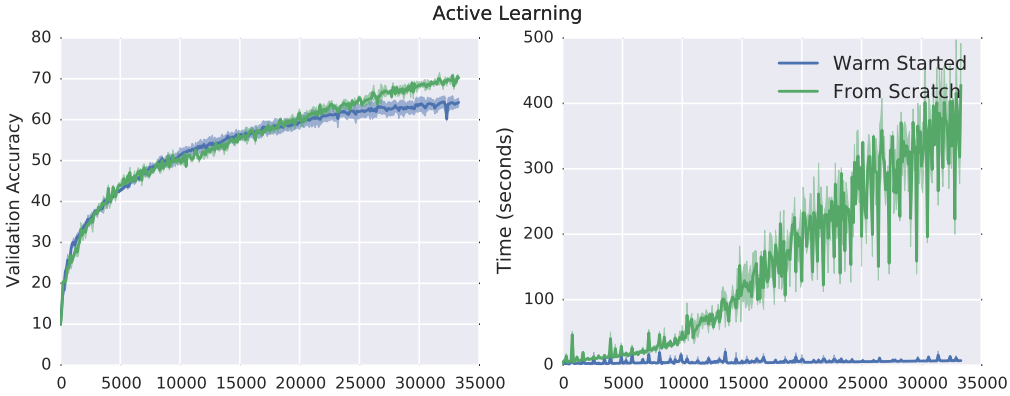

Figure 10: An active online learning experiment for CIFAR data using a ResNet. The horizontal axis displays the number of total samples in the training set available to the learner.

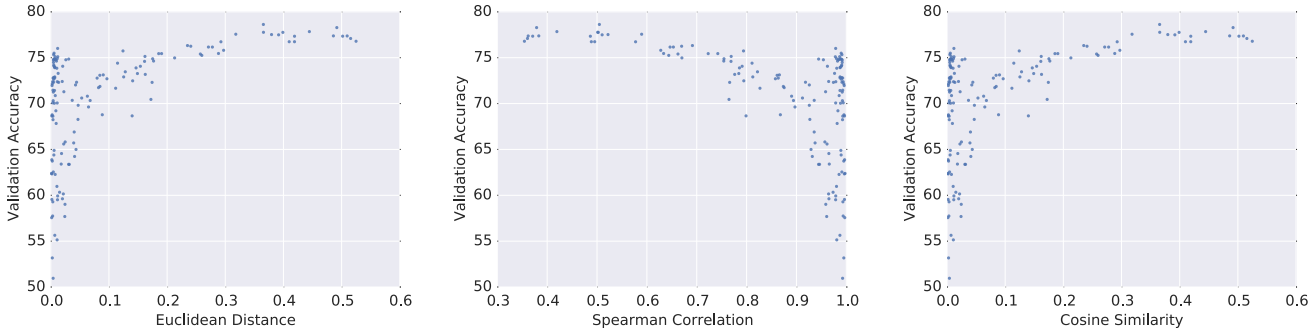

Figure 11: Companion plots to Figure 5, showing the correlation between warm-started models and their initialization for various hyperparameter settings. Each plot shows validation accuracy as a function of different measurements of correlation.