Are Labels Necessary for Neural Architecture Search?

Chenxi Liu[1], Piotr Dollár[2], Kaiming He[2], Ross Girshick[2], Alan Yuille[1], and Saining Xie[2]

[1] Johns Hopkins University
[2] Facebook AI Research

Abstract Existing neural network architectures in computer vision — whether designed by humans or by machines — were typically found using both images and their associated labels. In this paper, we ask the question: can we find high-quality neural architectures using only images, but no human-annotated labels? To answer this question, we first define a new setup called Unsupervised Neural Architecture Search (UnNAS). We then conduct two sets of experiments. In sample-based experiments, we train a large number (500) of diverse architectures with either supervised or unsupervised objectives, and find that the architecture rankings produced with and without labels are highly correlated. In search-based experiments, we run a well-established NAS algorithm (DARTS) using various unsupervised objectives, and report that the architectures searched without labels can be competitive to their counterparts searched with labels. Together, these results reveal the potentially surprising finding that labels are not necessary, and the image statistics alone may be sufficient to identify good neural architectures.[3]

Keywords: Neural Architecture Search; Unsupervised Learning

1 Introduction

Neural architecture search (NAS) has emerged as a research problem of searching for architectures that perform well on target data and tasks. A key mystery surrounding NAS is what factors contribute to the success of the search. Intuitively, using the target data and tasks during the search will result in the least domain gap, and this is indeed the strategy adopted in early NAS attempts [35,26]. Later, researchers [36] started to utilize the transferability of architectures, which enabled the search to be performed on different data and labels (*e.g.*, CIFAR-10) than the target (*e.g.*, ImageNet). However, what has not changed is that *both the images and the (semantic) labels* provided in the dataset need to be used in order to search for an architecture. In other words, existing NAS approaches perform search in the *supervised* learning regime.

In this paper, we take a step towards understanding what role supervision plays in the success of NAS. We ask the question: How indispensable are labels in

---

[3] Code release: https://github.com/facebookresearch/unnas

neural architecture search? Is it possible to find high-quality architectures using images only? This corresponds to the important yet underexplored *unsupervised setup* of neural architecture search, which we formalize in Section 3.

With the absence of labels, the quality of the architecture needs to be estimated in an unsupervised fashion during the search phase. In the present work, we conduct two sets of experiments using three unsupervised training methods [11,34,21] from the recent self-supervised learning literature.[4] These two sets of experiments approach the question from complementary perspectives. In *sample-based experiments*, we randomly sample 500 architectures from a search space, train and evaluate them using supervised *vs.* self-supervised objectives, and then examine the rank correlation (when sorting models by accuracy) between the two training methodologies. In *search-based experiments*, we take a well-established NAS algorithm, replace the supervised search objective with a self-supervised one, and examine the quality of the searched architecture on tasks such as ImageNet classification and Cityscapes semantic segmentation. Our findings include:

- The architecture rankings produced by supervised and self-supervised pretext tasks are *highly correlated*. This finding is consistent across two datasets, two search spaces, and three pretext tasks.
- The architectures searched without human annotations are *comparable in performance* to their supervised counterparts. This result is consistent across three pretext tasks, three pretext datasets, and two target tasks. There are even cases where unsupervised search outperforms supervised search.
- Existing NAS approaches typically use *labeled* images from a *smaller* dataset to learn transferable architectures. We present evidence that using *unlabeled* images from a *large* dataset may be a more promising approach.

We conclude that labels are not necessary for neural architecture search, and the deciding factor for architecture quality may hide within the image pixels.

2   Related Work

Neural Architecture Search. Research on the NAS problem involves designing the search space [36,33] and the search algorithm [35,25]. There are special focuses on reducing the overall time cost of the search process [18,23,19], or on extending to a larger variety of tasks [4,17,10,27]. Existing works on NAS all use human-annotated labels during the search phase. Our work is orthogonal to existing NAS research, in that we explore the unsupervised setup.

Architecture Transferability. In early NAS attempts [35,26], the search phase and the evaluation phase typically operate on the same dataset and task. Later, researchers realized that it is possible to relax this constraint. In these

---

[4] Self-supervised learning is a form of unsupervised learning. The term "unsupervised learning" in general emphasizes "without *human*-annotated labels", while the term "self-supervised learning" emphasizes "producing labels from data".

situations, the dataset and task used in the search phase are typically referred as the *proxy* to the target dataset and task, reflecting a notion of architecture transferability. [36] demonstrated that CIFAR-10 classification is a good proxy for ImageNet classification. [18] measured the rank correlation between these two tasks using a small number of architectures. [15] studied the transferability of 16 architectures (together with trained weights) between more *supervised* tasks. [14] studied the role of architecture in several self-supervised tasks, but with a small number of architectures and a different evaluation method (*i.e.* linear probe). Part of our work studies architecture transferability at a larger scale, *across* supervised and unsupervised tasks.

Unsupervised Learning. There is a large literature on unsupervised learning, *e.g.*, [9,8,30,34,21,11,22,31,29,12]. In the existing literature, methods are generally developed to learn the *weights* (parameters) of a fixed architecture without using labels, and these weights are evaluated by transferring to a target *supervised* task. In our study, we explore the possibility of using such methods to learn the *architecture* without using labels, rather than the weights. Therefore, our subject of study is simultaneously the unsupervised generalization of NAS, and the architecture level generalization of unsupervised learning.

3   Unsupervised Neural Architecture Search

The goal of this paper is to provide an answer to the question asked in the title: are labels necessary for neural architecture search? To formalize this question, in this section, we define a new setup called Unsupervised Neural Architecture Search (UnNAS). Note that UnNAS represents a general problem setup instead of any specific algorithm for solving this problem. We instantiate UnNAS with specific algorithms and experiments to explore the importance of labels in neural architecture search.

3.1   Search Phase

The traditional NAS problem includes a search phase: given a pre-defined search space, the search algorithm explores this space and estimates the performance (*e.g.* accuracy) of the architectures sampled from the space. The accuracy estimation can involve full or partial training of an architecture. Estimating the accuracy requires access to the labels of the dataset. So the traditional NAS problem is essentially a *supervised learning* problem.

We define UnNAS as the counterpart *unsupervised learning* problem. It follows the definition of the NAS problem, only except that there are no human-annotated labels provided for estimating the performance of the architectures. An algorithm for the UnNAS problem still explores a pre-defined search space,[5] but it requires other criteria to estimate how good a sampled architecture is.

---

[5] Admittedly, the search space design is typically heavily influenced by years of manual search, usually with supervision.
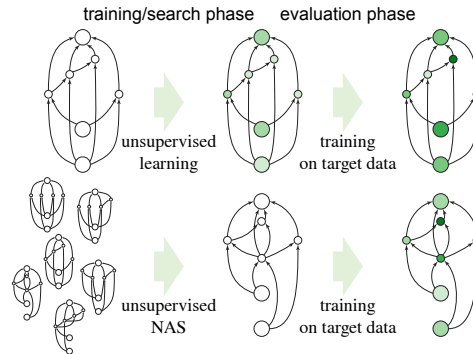
Figure 1: **Unsupervised neural architecture search**, or UnNAS, is a new problem setup that helps answer the question: are labels necessary for neural architecture search? In traditional unsupervised learning (top panel), the *training phase* learns the weights of a fixed architecture; then the *evaluation phase* measures the quality of the weights by training a classifier (either by fine-tuning the weights or using them as a fixed feature extractor) using supervision from the target dataset. Analogously, in UnNAS (bottom panel), the *search phase* searches for an architecture without using labels; and the *evaluation phase* measures the quality of the architecture found by an UnNAS algorithm by training the architecture's weights using supervision from the target dataset.

3.2   Evaluation Phase

Generally speaking, the goal of the NAS problem is to find an architecture. The weights of the found architecture are *not* necessarily the output of a NAS algorithm. Instead, the weights are optimized after the search phase in an evaluation phase of the NAS problem: it includes training the found architecture on a target dataset's training split, and validating the accuracy on the target dataset's validation split. We note that "*training* the architecture weights" is part of the NAS *evaluation* phase—the labels in the target dataset, both training and validation splits, play a role of evaluating an architecture.

Based on this context, we define *the evaluation phase of UnNAS* in the same way: training the weights of the architecture (found by an UnNAS algorithm) on a target dataset's training split, and validating the accuracy on the target dataset's validation split, both *using* the labels of the target dataset. We remark that using labels during the evaluation phase does *not* conflict with the definition of UnNAS: the search phase is unsupervised, while the evaluation phase requires labels to examine how good the architecture is.

3.3   Analogy to Unsupervised Learning

Our definition of UnNAS is analogous to unsupervised *weight* learning in existing literature [11,34,21]. While the unsupervised learning phase has no labels

for training weights, the quality of the learned weights is evaluated by transferring them to a target task, supervised by labels. We emphasize that in the UnNAS setup, labels play an analogous role during evaluation. See Figure 1 for an illustration and elaboration on this analogy.

Similar to unsupervised weight learning, in principle, the search dataset should be different than the evaluation (target) dataset in UnNAS in order to more accurately reflect real application scenarios.

4 Experiments Overview

As Section 3 describes, an architecture discovered in an *unsupervised* fashion will be evaluated by its performance in a *supervised* setting. Therefore, we are essentially looking for some type of architecture level *correlation* that can reach across the unsupervised *vs.* supervised boundary, so that the unsupervisedly discovered architecture could be reliably *transferred* to the supervised target task. We investigate whether several existing self-supervised pretext tasks (described in Section 4.1) can serve this purpose, through two sets of experiments of complementary nature: *sample-based* (Section 5) and *search-based* (Section 6). In *sample-based*, each network is trained and evaluated individually, but the downside is that we can only consider a small, random subset of the search space. In *search-based*, the focus is to find a top architecture from the entire search space, but the downside is that the training dynamics during the search phase does not exactly match that of the evaluation phase.

4.1 Pretext Tasks

We explore three unsupervised training methods (typically referred to as *pretext tasks* in self-supervised learning literature): rotation prediction, colorization, and solving jigsaw puzzles. We briefly describe them for completeness.

- *Rotation prediction* [11] (`Rot`): the input image undergoes one of four preset rotations (0, 90, 180, and 270 degrees), and the pretext task is formulated as a 4-way classification problem that predicts the rotation.
- *Colorization* [34] (`Color`): the input is a grayscale image, and the pretext task is formulated as a pixel-wise classification problem with a set of predefined color classes (313 in [34]).
- *Solving jigsaw puzzles* [21] (`Jigsaw`): the input image is divided into patches and randomly shuffled. The pretext task is formulated as an image-wise classification problem that chooses from one out of K preset permutations.[6]

All three pretext tasks are image or pixel-wise classification problems. Therefore, we can compute the classification accuracy of the pretext task (on a validation set). Based on this pretext task accuracy, we can analyze its correlation

---

[6] On ImageNet, each image is divided into $3{\times}3{=}9$ patches and K is 1000 selected from $9!{=}362{,}880$ permutations [21]; on CIFAR-10 in which images are smaller, we use $2{\times}2{=}4$ patches and K is $4!{=}24$ permutations.
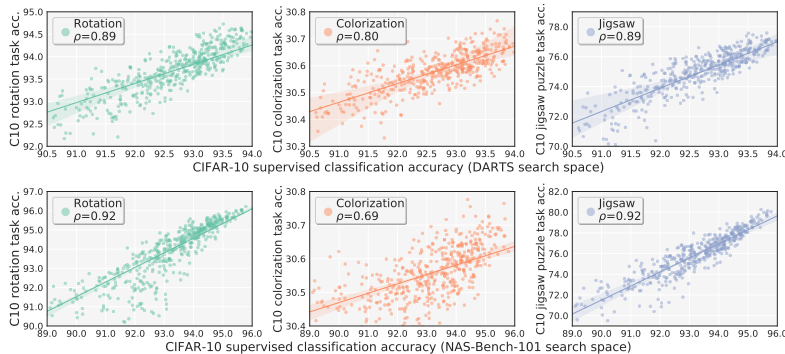
Figure 2: **Correlation between supervised classification accuracy *vs*. pretext task accuracy on CIFAR-10 ("C10")**. Top panel: DARTS search space. Bottom panel: NAS-Bench-101 search space. The straight lines are fit with robust linear regression [13] (same for Figure 3 and Figure 4).

with the supervised classification accuracy (also on a validation set), as in our sample-based experiments (Section 5). Also, since these pretext tasks all use cross entropy loss, it is also straightforward to use them as the training objective in standard NAS algorithms, as done in our search-based experiments (Section 6).

5   Sample-Based Experiments

5.1   Experimental Design

In *sample-based experiments*, we first randomly sample 500 architectures from a certain search space. We train each architecture from scratch on the pretext task and get its pretext task accuracy (*e.g.*, the 4-way classification accuracy in the rotation task), and also train the same architecture from scratch on the supervised classification task (*e.g.*, 1000-way classification on ImageNet).[7]

With these data collected, we perform two types of analysis. For the *rank correlation* analysis, we empirically study the statistical rank correlations between the pretext task accuracy and the supervised accuracy, by measuring the Spearman's Rank Correlation [28], denoted as ρ. For the *random experiment* analysis, we follow the setup proposed in [2] and recently adopted in [24]. Specifically, for each experiment size m, we sample m architectures from our pool of $n = 500$ architectures. For each pretext task, we select the architecture with the highest pretext task accuracy among the m. This process is repeated $\lceil n/m \rceil$ times, to compute the mean and error bands ($\pm$ 2 standard deviations) of the top-1 accuracy of these $\lceil n/m \rceil$ architectures *on the target dataset/task*.

---

[7] These architectures only have small, necessary differences when trained towards these different tasks (*i.e.*, having 4 neurons or 1000 neurons at the output layer).
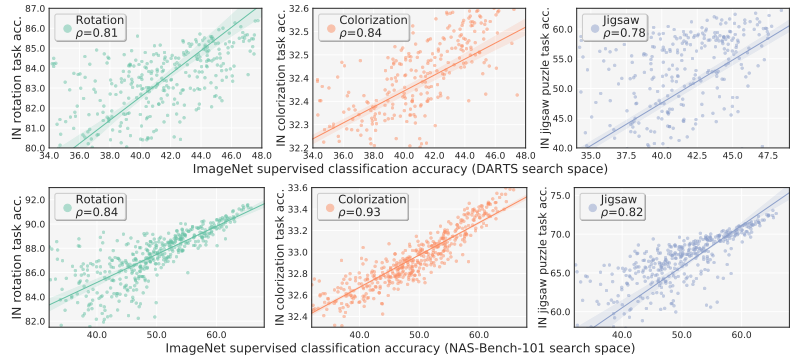
Figure 3: **Correlation between supervised classification accuracy *vs*. pretext task accuracy on ImageNet ("IN")**. Top panel: DARTS search space. Bottom panel: NAS-Bench-101 search space.

These two studies provide complementary views. The *rank correlation* analysis aims to provide a global picture for *all* architectures sampled from a search space, while the *random experiment* analysis focuses on the *top* architectures in a random experiment of varying size.

We study two search spaces: the DARTS search space [19], and the NAS-Bench-101 search space [33]. The latter search space was built for benchmarking NAS algorithms, so we expect it to be less biased towards the search space for a certain algorithm. The experiments are conducted on two commonly used datasets: CIFAR-10 [16] and ImageNet [7].

5.2   Implementation Details

Details of sampling the 500 architectures are described in Appendix. In the DARTS search space, regardless of the task, each network is trained for 5 epochs with width 32 and depth 22 on ImageNet; 100 epochs with width 16 and depth 20 on CIFAR-10. In the NAS-Bench-101 search space, regardless of the task, each network is trained for 10 epochs with width 128 and depth 12 on ImageNet; 100 epochs with width 128 and depth 9 on CIFAR-10. Please refer to [19,33] for the respective definitions of *width* and *depth*. The performance of each network on CIFAR-10 is the average of 3 independent runs to reduce variance.

We remark that the small-scale of CIFAR-10 dataset and the short training epochs on ImageNet are the compromises we make to allow for more diverse architectures (*i.e.* 500).

5.3   Results

High rank correlation between supervised accuracy and pretext accuracy on the <u>same</u> dataset. In Figure 2 and Figure 3, we show the scatter plots of the 500 architectures' supervised classification accuracy (horizontal axis) and
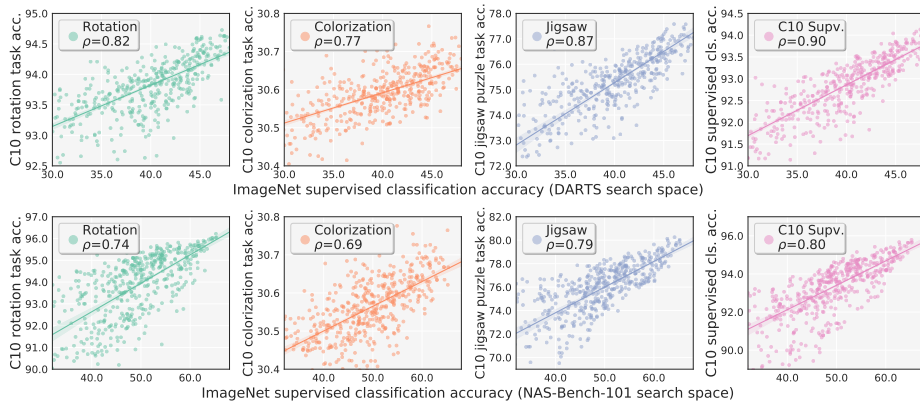
Figure 4: **Correlation between ImageNet supervised classification accuracy *vs*. CIFAR-10 ("C10") pretext task accuracy**. Rankings of architectures are highly correlated between supervised classification and three unsupervised tasks, as measured by Spearman's rank correlation (ρ). We also show rank correlation using CIFAR-10 supervised proxy in the rightmost panel. Top panel: DARTS search space. Bottom panel: NAS-Bench-101 search space.

pretext task accuracy (vertical axis) on CIFAR-10 and ImageNet, respectively. We see that this rank correlation is typically higher than 0.8, regardless of the dataset, the search space, and the pretext task. This type of consistency and robustness indicates that this phenomenon is general, as opposed to dataset/search space specific.

The same experiment is performed on both the DARTS and the NAS-Bench-101 search spaces. The rank correlations on the NAS-Bench-101 search space are generally higher than those on the DARTS search space. A possible explanation is that the architectures in NAS-Bench-101 are more diverse, and consequently their accuracies have larger gaps, *i.e.*, are less affected by training noise.

Interestingly, we observe that among the three pretext tasks, colorization consistently has the lowest correlation on CIFAR-10, but the highest correlation on ImageNet. We suspect this is because the small images in CIFAR-10 make the learning of per-pixel colorization difficult, and consequently the performance after training is noisy.

High rank correlation between supervised accuracy and pretext accuracy <u>across</u> datasets. In Figure 4 we show the *across dataset* rank correlation analysis, where the pretext task accuracy is measured on CIFAR-10, but the supervised classification accuracy is measured on ImageNet.

On the DARTS search space (top panel), despite the image distribution shift brought by different datasets, for each of the three pretext tasks (left three plots), the correlation remains consistently high (~0.8). This shows that across the entire search space, the relative ranking of an architecture is likely to be
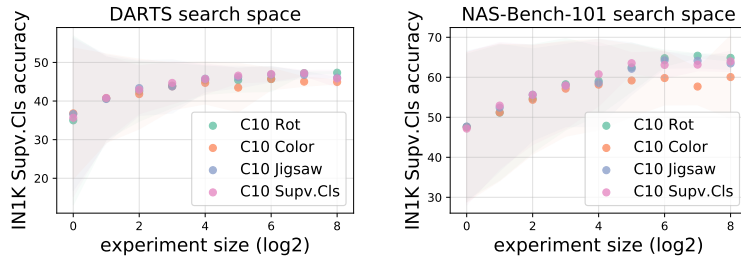
Figure 5: **Random experiment efficiency curves**. Left panel: DARTS search space. Right panel: NAS-Bench-101 search space. We show the range of ImageNet classification accuracies of top architectures identified by the three pretext tasks and the supervised task under various experiment sizes. See text for more details.

similar, whether under the unsupervised pretext task accuracy or under the supervised target task accuracy. In the rightmost panel, we compare with a scenario where instead of using an unsupervised pretext task, we use a proxy task of CIFAR-10 supervised classification. The correlation is $\rho = 0.90$ in this case. As the CIFAR-10 supervised classification is a commonly used proxy task in existing NAS literature [36], it gives a reference on ρ's value.

In the bottom panel we show more analysis of this kind by replacing the search space with NAS-Bench-101. Although the architectures are quite different, the observations are similar. In all cases, the self-supervised pretext task accuracy is highly correlated to the supervised classification accuracy.

Better pretext accuracy translates to better supervised accuracy. In addition to the rank correlation analysis, we also perform the random experiment analysis. Figure 5 shows the *random experiment efficiency* curve for DARTS and NAS-Bench-101 search spaces. Again, the pretext accuracies are obtained on CIFAR-10, and the target accuracies are from ImageNet. By design of this experiment, as the experiment size m increases, the *pretext* accuracies of the $\lceil n/m \rceil$ architectures should increase. Figure 5 shows that the *target* accuracies of these $\lceil n/m \rceil$ architectures also increase with m. In addition, at each experiment size, most unsupervised pretext objectives perform similarly compared to the commonly used supervised CIFAR-10 proxy. The overall trends are also comparable. This shows that the architecture rankings produced with and without labels are not only correlated across the entire search space, but also towards the top of the search space, which is closer to the goal of UnNAS.

6 Search-Based Experiments

6.1 Experimental Design

In *search-based experiments*, the idea is to run a well-established NAS algorithm, except that we make the minimal modification of replacing its supervised search

objective with an unsupervised one. Following the UnNAS setup, we then examine (by training from scratch) how well these unsupervisedly discovered architectures perform on supervised target tasks. Since all other variables are controlled to be the same, search-based experiments can easily compare between the supervised and unsupervised counterparts of a NAS algorithm, which can help reveal the importance of labels.

The NAS algorithm we adopt is DARTS (short for "differentiable architecture search") [19] for its simplicity. DARTS formulates the activation tensor selection and operation selection as a categorical choice, implemented as a softmax function on a set of continuous parameters (named α). These parameters are trained in a similar fashion as the architecture weights, by backpropagation from a loss function. After this training, the softmax outputs are discretized and produce an architecture.

The self-supervised objectives that we consider are, still, those described in Section 4.1: rotation prediction (`Rot`), colorization (`Color`), and solving jigsaw puzzles (`Jigsaw`). For comparison, we also perform NAS search with supervised objectives, *e.g.* classification (`Supv.Cls`) or semantic segmentation (`Supv.Seg`). To help distinguish, we name the method NAS-`DARTS` if the search objective is supervised, and UnNAS-`DARTS` if the search objective is unsupervised.

6.2   Implementation Details

Search phase. We use three different datasets for architecture search: ImageNet-1K (IN1K) [7], ImageNet-22K (IN22K) [7], and Cityscapes [6]. IN1K is the standard ImageNet benchmark dataset with 1.2M images from 1K categories. IN22K is the full ImageNet dataset that has ~14M images from 22K categories. Cityscapes is a dataset of street scenes that has drastically different image statistics. Note that UnNAS-`DARTS` will only access the images provided in the dataset, while NAS-`DARTS` will additionally access the (semantic) labels provided in the respective dataset. The search phase will operate only within the training split, without accessing the true validation or test split.

We report in Appendix the hyper-parameters we used. One major difference between our experiments and DARTS [19] is that the images in the search datasets that we consider are much larger in size. We use $224{\times}224$ random crops for search on IN1K/IN22K, and $312{\times}312$ for search on Cityscapes following [17]. To enable DARTS training with large input images, we use 3 stride-2 convolution layers at the beginning of the network to reduce spatial resolution. This design, together with appropriately chosen number of search epochs (see Appendix), allows UnNAS search to be efficient (~2 GPU days on IN1K/Cityscapes, ~10 GPU days on IN22K, regardless of task) despite running on larger images.

Evaluation phase. We use two distinct datasets and tasks for UnNAS evaluation: (1) ImageNet-1K (IN1K) for image classification. The performance metric is top-1 accuracy on the IN1K validation set. (2) Cityscapes for semantic segmentation. We use the `train_fine` set (2975 images) for training. The performance metric is mean Intersection-over-Union (mIoU) evaluated on the `val` set (500 images).

method	search dataset & task	top-1 acc.	FLOPs (M)	params (M)
NAS-DARTS [19]	CIFAR-10 Supv.Cls	73.3	574	4.7
NAS-P-DARTS [5]	CIFAR-10 Supv.Cls	75.6	557	4.9
NAS-PC-DARTS [32]	CIFAR-10 Supv.Cls	74.9	586	5.3
NAS-PC-DARTS [32]	IN1K Supv.Cls	75.8	597	5.3
NAS-DARTS†	CIFAR-10 Supv.Cls	$74.9_{\pm 0.08}$	538	4.7
NAS-DARTS	IN1K Supv.Cls	$76.3_{\pm 0.06}$	590	5.3
UnNAS-DARTS	IN1K Rot	$75.8_{\pm 0.18}$	558	5.1
UnNAS-DARTS	IN1K Color	$75.7_{\pm 0.12}$	547	4.9
UnNAS-DARTS	IN1K Jigsaw	$75.9_{\pm 0.15}$	567	5.2
NAS-DARTS	IN22K Supv.Cls	$75.9_{\pm 0.09}$	585	5.2
UnNAS-DARTS	IN22K Rot	$75.7_{\pm 0.23}$	549	5.0
UnNAS-DARTS	IN22K Color	$75.9_{\pm 0.21}$	547	5.0
UnNAS-DARTS	IN22K Jigsaw	$75.9_{\pm 0.31}$	559	5.1
NAS-DARTS	Cityscapes Supv.Seg	$75.8_{\pm 0.13}$	566	5.1
UnNAS-DARTS	Cityscapes Rot	$75.9_{\pm 0.19}$	554	5.1
UnNAS-DARTS	Cityscapes Color	$75.2_{\pm 0.15}$	594	5.1
UnNAS-DARTS	Cityscapes Jigsaw	$75.5_{\pm 0.06}$	566	5.0

Table 1: **ImageNet-1K classification results of the architectures searched by NAS and UnNAS algorithms**. Rows in gray correspond to invalid UnNAS configurations where the search and evaluation datasets are the same. † is our training result of the DARTS architecture released in [19].

For IN1K evaluation, we fix depth to 14 and adjust the width to have #FLOPs $\in$ [500,600]M. Models are trained for 250 epochs with an auxiliary loss weighted by 0.4, batch size 1024 across 8 GPUs, cosine learning rate schedule [20] with initial value 0.5, and 5 epochs of warmup. For Cityscapes evaluation, we fix depth to 12 and adjust the width to have #Params $\in$ [9.5,10.5]M. We train the network for 2700 epochs, with batch size 64 across 8 GPUs, cosine learning rate schedule with initial value 0.1. For both ImageNet and Cityscapes evaluations, we report the mean and standard deviation of 3 independent trainings of the same architecture. More implementation details are described in Appendix.

We note that under our definition of the UnNAS setup, the same dataset should not be used for both search and evaluation (because this scenario is unrealistic); We provide the IN1K→IN1K and Cityscapes→Cityscapes results purely as a reference. Those settings are analogous to the linear classifier probe for IN1K in conventional unsupervised learning research.

6.3   Results

In search-based experiments, the architectures are evaluated on both ImageNet classification, summarized in Table 1, and Cityscapes semantic segmentation, summarized in Table 2. We provide visualization of all NAS-DARTS and UnNAS-DARTS cell architectures in Appendix.

method	search dataset & task	mIoU	FLOPs (B)	params (M)
NAS-DARTS[†]	CIFAR-10 `Supv.Cls`	$72.6_{\pm 0.55}$	121	9.6
NAS-DARTS	IN1K `Supv.Cls`	$73.6_{\pm 0.31}$	127	10.2
UnNAS-DARTS	IN1K `Rot`	$73.6_{\pm 0.29}$	129	10.4
UnNAS-DARTS	IN1K `Color`	$72.2_{\pm 0.56}$	122	9.7
UnNAS-DARTS	IN1K `Jigsaw`	$73.1_{\pm 0.17}$	129	10.4
NAS-DARTS	IN22K `Supv.Cls`	$72.4_{\pm 0.29}$	126	10.1
UnNAS-DARTS	IN22K `Rot`	$72.9_{\pm 0.23}$	128	10.3
UnNAS-DARTS	IN22K `Color`	$73.6_{\pm 0.41}$	128	10.3
UnNAS-DARTS	IN22K `Jigsaw`	$73.1_{\pm 0.59}$	129	10.4
NAS-DARTS	Cityscapes `Supv.Seg`	$72.4_{\pm 0.15}$	128	10.3
UnNAS-DARTS	Cityscapes `Rot`	$73.0_{\pm 0.25}$	128	10.3
UnNAS-DARTS	Cityscapes `Color`	$72.5_{\pm 0.31}$	122	9.5
UnNAS-DARTS	**Cityscapes `Jigsaw`**	$\mathbf{74.1_{\pm 0.39}}$	**128**	**10.2**

Table 2: **Cityscapes semantic segmentation results of the architectures searched by NAS and UnNAS algorithms**. These are trained from scratch: there is no fine-tuning from ImageNet checkpoint. Rows in gray correspond to an illegitimate setup where the search dataset is the same as the evaluation dataset. † is our training result of the DARTS architecture released in [19].

UnNAS architectures perform competitively to supervised counterparts. We begin by comparing NAS-DARTS and UnNAS-DARTS when they are performed on the same search dataset. This would correspond to every four consecutive rows in Table 1 and Table 2, grouped together by horizontal lines.

As discussed earlier, strictly speaking the IN1K→IN1K experiment is not valid under our definition of UnNAS. For reference, we gray out these results in Table 1. NAS-DARTS on IN1K dataset has the highest performance among our experiments, achieving a top-1 accuracy of 76.3%. However, the UnNAS algorithm variants with `Rot`, `Color`, `Jigsaw` objectives all perform very well (achieving 75.8%, 75.7% and 75.9% top-1 accuracy, respectively), closely approaching the results obtained by the supervised counterpart. This suggests it might be desirable to perform architecture search on the target dataset directly, as also observed in other work [3].

Two valid UnNAS settings include IN22K→IN1K and Cityscapes→IN1K for architecture search and evaluation. For IN22K→IN1K experiments, NAS and UnNAS results across the board are comparable. For Cityscapes→IN1K experiments, among the UnNAS architectures, `Rot` and `Jigsaw` perform well, once again achieving results comparable to the supervised search. However, there is a drop for UnNAS-DARTS search with `Color` objective (with a 75.2% top-1 accuracy). We hypothesize that this might be owing to the fact that the color distribution in Cityscapes images is not as diverse: the majority of the pixels are from *road* and *ground* categories of gray colors.

In general, the variances are higher for Cityscapes semantic segmentation (Table 2), but overall UnNAS-DARTS architectures still perform competitively to

NAS-`DARTS` architectures, measured by mIoU. For the Cityscapes→Cityscapes experiment, we observe that searching with segmentation objective directly leads to inferior result (mean 72.4% mIoU), compared to the architectures searched for ImageNet classification tasks. This is different from what has been observed in Table 1. However, under this setting, our UnNAS algorithm, in particular the one with the `Jigsaw` objective shows very promising results (mean 74.1% mIoU). In fact, when the search dataset is IN22K or Cityscapes, all UnNAS-`DARTS` architectures perform *better* than the NAS-`DARTS` architecture. This is the opposite of what was observed in IN1K→IN1K. Results for Cityscapes→Cityscapes are grayed out for the same reason as before (invalid under UnNAS definition).

NAS and UnNAS results are robust across a large variety of datasets and tasks. The three search datasets that we consider are of different nature. For example, IN22K is 10 times larger than IN1K, and Cityscapes images have a markedly different distribution than those in ImageNet. In our experiments, NAS-`DARTS`/UnNAS-`DARTS` architectures searched on IN22K do not significantly outperform those searched on IN1K, meaning that they do not seem to be able to enjoy the benefit of having more abundant images. This reveals new opportunities in designing better algorithms to exploit bigger datasets for neural architecture search. For Cityscapes→IN1K experiments, it is interesting to see that after switching to a dataset with markedly distinct search images (urban street scenes), we are still able to observe decent performance. The same goes for the reverse direction IN1K/IN22K→Cityscapes, which implies that the search does not severely overfit to the images from the dataset.

In addition to this robustness to the *search dataset* distribution, NAS and UnNAS also exhibit robustness to *target dataset and task*. Classification on ImageNet and segmentation on Cityscapes are different in many ways, but among different combinations of search dataset and task (whether supervised or unsupervised), we do not observe a case where the same architecture performs well on one but poorly on the other.

UnNAS outperforms previous methods. Finally, we compare our UnNAS-`DARTS` results against existing works. We first note that we are able to achieve a better baseline number with the NAS-`DARTS` architecture (searched with the CIFAR-10 proxy) compared to what was reported in [19]. This is mainly due to better hyper-parameter tuning we adopt from [5] for the *evaluation phase* model training. This baseline sets up a fair ground for all the UnNAS experiments; we use the same evaluation phase hyper-parameters across different settings.

On ImageNet, our UnNAS-`DARTS` architectures can comfortably outperform this baseline by up to 1% classification accuracy. In fact, the extremely competitive UnNAS-`DARTS` results also outperform the previous best result (75.8%) on this search space, achieved with a more sophisticated NAS algorithm [32].

On Cityscapes, there have not been many works that use DARTS variants as the backbone. The closest is Auto-DeepLab [17], but we use a lighter architecture (in that we do not have the Decoder) and shorter training iterations, so the

results are not directly comparable. Nonetheless, according to our evaluation, the UnNAS architectures perform favorably against the DARTS architecture released in [19] (discovered with the CIFAR-10 proxy). The best UnNAS-`DARTS` variant (Cityscapes `Jigsaw`) achieves 74.1% mIoU, which outperforms this baseline by 1.5% on average. Overall, our experiments demonstrate that exploring neural architecture search with unsupervised/self-supervised objectives to improve target task performance might be a fruitful direction.

Outperforming previous methods was far from the original goal of our study. Nonetheless, the promising results of UnNAS suggest that in addition to developing new *algorithms* and finding new *tasks*, the role of *data* (in our case, more/larger images) and *paradigm* (in our case, no human annotations) is also worth attention in future work on neural architecture search.

7   Discussion

In this paper, we challenge the common practice in neural architecture search and ask the question: do we really need labels to successfully perform NAS? We approach this question with two sets of experiments. In sample-based experiments, we discover the phenomenon that the architecture rankings produced with and without labels are highly correlated. In search-based experiments, we show that the architectures learned without accessing labels perform competitively, not only relative to their supervised counterpart, but also in terms of absolute performance. In both experiments, the observations are consistent and robust across various datasets, tasks, and/or search spaces. Overall, the findings in this paper indicate that labels are *not* necessary for neural architecture search.

How to learn and transfer useful representations to subsequent tasks in an unsupervised fashion has been a research topic of extensive interest, but the discovery of neural network architectures has been driven solely by supervised tasks. As a result, current NAS products or AutoML APIs typically have the strict prerequisite for users to *"put together a training dataset of labeled images"* [1]. An immediate implication of our study is that the job of the user could potentially be made easier by dropping the labeling effort. In this sense, UnNAS could be especially beneficial to the many applications where data constantly comes in at large volume but labeling is costly.

At the same time, we should still ask: if not labels, then what factors are needed to reveal a good architecture? A meaningful unsupervised task seems to be important, though the several pretext tasks considered in our paper do not exhibit significant difference in either of the two experiments. In the future we plan to investigate even more and even simpler unsupervised tasks. Another possibility is that the architecture quality is mainly decided by the image statistics, and since the datasets that we consider are all natural images, the correlations are high and the results are comparable. This hypothesis would also suggest an interesting, alternative direction: that instead of performing NAS again and again for every specific labeled task, it may be more sensible to perform NAS once on large amounts of unlabeled images that capture the image distribution.

References

1. Google AutoML Vision API Tutorial (2019 (accessed Nov 14, 2019)), https://cloud.google.com/vision/automl/docs/tutorial
2. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of Machine Learning Research **13**(Feb), 281–305 (2012)
3. Cai, H., Zhu, L., Han, S.: Proxylessnas: Direct neural architecture search on target task and hardware. In: ICLR (2019)
4. Chen, L.C., Collins, M., Zhu, Y., Papandreou, G., Zoph, B., Schroff, F., Adam, H., Shlens, J.: Searching for efficient multi-scale architectures for dense image prediction. In: NeurIPS (2018)
5. Chen, X., Xie, L., Wu, J., Tian, Q.: Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In: ICCV (2019)
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The Cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR (2009)
8. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV (2015)
9. Dosovitskiy, A., Springenberg, J.T., Riedmiller, M., Brox, T.: Discriminative unsupervised feature learning with convolutional neural networks. In: NeurIPS (2014)
10. Ghiasi, G., Lin, T.Y., Le, Q.V.: Nas-fpn: Learning scalable feature pyramid architecture for object detection. In: CVPR (2019)
11. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: ICLR (2018)
12. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR (2020)
13. Huber, P.J.: Robust statistics. Springer (2011)
14. Kolesnikov, A., Zhai, X., Beyer, L.: Revisiting self-supervised visual representation learning. In: CVPR (2019)
15. Kornblith, S., Shlens, J., Le, Q.V.: Do better imagenet models transfer better? In: CVPR (2019)
16. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009)
17. Liu, C., Chen, L.C., Schroff, F., Adam, H., Hua, W., Yuille, A.L., Fei-Fei, L.: Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In: CVPR (2019)
18. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: ECCV (2018)
19. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. In: ICLR (2019)
20. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. In: ICLR (2017)
21. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV (2016)
22. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv:1807.03748 (2018)
23. Pham, H., Guan, M.Y., Zoph, B., Le, Q.V., Dean, J.: Efficient neural architecture search via parameter sharing. In: ICML (2018)

24. Radosavovic, I., Johnson, J., Xie, S., Lo, W.Y., Dollár, P.: On network design spaces for visual recognition. In: ICCV (2019)
25. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized evolution for image classifier architecture search. In: AAAI (2019)
26. Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y.L., Tan, J., Le, Q.V., Kurakin, A.: Large-scale evolution of image classifiers. In: ICML (2017)
27. So, D.R., Liang, C., Le, Q.V.: The evolved transformer. In: ICML (2019)
28. Spearman, C.: The proof and measurement of association between two things. The American Journal of Psychology (1904)
29. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. arXiv:1906.05849 (2019)
30. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: ICCV (2015)
31. Wu, Z., Xiong, Y., Yu, S., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
32. Xu, Y., Xie, L., Zhang, X., Chen, X., Qi, G.J., Tian, Q., Xiong, H.: Pc-darts: Partial channel connections for memory-efficient differentiable architecture search. In: ICLR (2020)
33. Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K., Hutter, F.: Nas-bench-101: Towards reproducible neural architecture search. In: ICML (2019)
34. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016)
35. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. In: ICLR (2017)
36. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: CVPR (2018)