DistilNAS: Neural Architecture Search With Distilled Data

SWAROOP N. PRABHAKAR1, ANKUR DESHWAL1, RAHUL MISHRA1, AND HYEONSU KIM2
1Samsung Semiconductor India Research (SSIR), Bangalore 560048, India
2Samsung Electronics, Hwaseong 443-743, South Korea
Corresponding author: Ankur Deshwal (a.deshwal@samsung.com)

ABSTRACT Can we perform Neural Architecture Search (NAS) with a smaller subset of target dataset and still fair better in terms of performance with significant reduction in search cost? In this work, we propose a method, called DistilNAS, which utilizes a curriculum learning based approach to distill the target dataset into a very efficient smaller dataset to perform NAS. We hypothesize that only the data samples containing features highly relevant to a given class should be used in the search phase of the NAS. We perform NAS with a distilled version of dataset and the searched model achieves a better performance with a much reduced search cost in comparison with various baselines. For instance, on Imagenet dataset, the DistilNAS uses only 10% of the training data and produces a model in $\approx 1$ GPU-day (includes the time needed for clustering) that achieves near SOTA accuracy of 75.75% (PC-DARTS had achieved SOTA with an accuracy of 75.8% but needed 3.8 GPU-days for architecture search). We also demonstrate and discuss the efficacy of DistilNAS on several other publicly available datasets.

INDEX TERMS Curriculum learning, dataset distillation, neural architecture search.

I. INTRODUCTION
Image classification is one of the fundamental tasks in computer vision field. Since the success of Alexnet [1], the deep learning models that achieved state-of-the-art performance in image classification were all manually designed.

Recently, with the advent of Neural Architecture Search (NAS) [2], [3], the focus has shifted to automating the design of neural networks. NAS has received considerable attention due to its remarkable success in image classification [4] and other related tasks.

Due to the success of deep neural network architectures in various application domains [5], [6], [7], [8] specifically in field of vision such as Resnet [9], NAS in the recent past has focussed on finding optimal building blocks called cells. The idea is to first find an efficient cell and then stack these cells together in an appropriate manner tailored for a specific task. Some of the earliest works in this regard employed reinforcement learning [4], [10] and evolutionary algorithms [11], [12] to design the cells. As these approaches had to evaluate architectures over a vast search space, both these approaches suffers from the inordinate amount of required computational resources (measured in terms of GPU-days). This implies that the architecture search can be generally performed on a simpler proxy dataset and then evaluation is done with the target dataset.

In order to alleviate some of the aforementioned problems with RL and EA based NAS methods, Liu et al., came up with a more efficient strategy of doing NAS called Differentiable Architecture Search (DARTS) [13]. DARTS followed the one-shot approach where a super network contains every candidate architecture in the search space as a sub-network and in each step, a sub-network or combination of sub-networks is trained and evaluated. In DARTS, instead of searching over a discrete set of candidate architectures, the search space was relaxed to be continuous. This enabled them to perform gradient descent with respect to architectural parameters based on validation loss. Through this approach, DARTS achieved a comparable performance with the RL and EA based methods at a small fraction of the search cost needed by them.

In spite of this, DARTS is limited by its severe memory requirement. Due to this, DARTS is forced to search the architecture in a shallow network where as the architecture being evaluated later is deep. This is referred to as ‘depth
Given an arbitrary dataset, can we perform architecture search on it within a reasonable amount of time?

This question is important in order to address the performance gap that may arise due to transfer learning through a model searched on simple datasets. In this work, we propose a data preprocessing method based on a curriculum learning method that enables us to sample a smaller effective subset of the dataset which we call “Distilled Data”. Using this distilled data, we demonstrate that NAS can be performed on large datasets in a reasonable amount of time without compromising on the accuracy of the model. The intuition behind our approach is that, since the training budget in the search phase is limited, we must focus only on finding a reasonably adequate model that can detect features of interest for a given class. Hence, we propose that only “informative” images, i.e., those images that contain only the relevant features corresponding to the class of interest, must be used in the search phase of NAS. The “uninformative” images are typically used for increasing the generalization ability of the model and hence do not add value in the model search phase. The generalization ability of the model can be tackled on in the evaluation phase where we train the searched model on the full dataset.

An overview of our method is as follows: Given a dataset, we perform density clustering on each class in the feature space and split them into three clusters - ‘CREL’ (RELEVANT, very informative about the relevant features), ‘CLREL’ (LESS-RELEVANT, less informative about the relevant features) and ‘CIRREL’ (IRRELEVANT, uninformative about the relevant features). The criterion which we use for this clustering is proposed in [15]. After performing the clustering process, we sample a small fraction of the data from the ‘CREL’ clusters and denote these samples as our ‘distilled data’. For instance, on ImageNet [16] which contains 1000 classes, we randomly sample only 10% of the training data from the ‘CREL’ clusters corresponding to the 1000 classes. Using this distilled data for ImageNet, we perform NAS using the P-DARTS method (P-DARTS will be our standard baseline for all the experiments except for those in the ablation study section).

The search process takes up only about 0.6 GPU-days on a Nvidia Tesla V100 GPU (≈ 1 GPU-day if we include the time for clustering) and achieves a top-1 error of 24.25% under the mobile setting in the evaluation phase. To the best of our knowledge, SOTA on Imagenet was achieved by PC-DARTS [17] with a top-1 error of 24.2% but needed 3.6 GPU-days and 12.5% of randomly sampled images from the training data for the search process. So our distilled data with P-DARTS method achieves a near SOTA performance at a much lower computational cost.

To summarize, our contributions are as follows:

1) We propose a novel method called DistilNAS, which uses a data distillation method based on curriculum learning to accelerate the neural architecture search process.

2) DistilNAS achieves near SOTA performance on ImageNet at a much lower search cost in experiments.

3) We perform extensive experiments to establish the efficacy of the proposed model DistilNAS with various community standard computer vision datasets such as CIFAR10 [18], CIFAR100 [18], Food-101 [19], Oxford-IIIT Pets [20], Stanford Cars [21], Caltech-101 [22].

4) We conduct an ablation study to show the significance of selecting samples from informative vs uninformative clusters and effect of selection of different architectures for feature extraction.

The remainder of this paper is organized as follows. Next, in Section II, we list some of the relevant related works along with their limitations and discuss how DistilNAS alleviates them. Section III is dedicated to the necessary background and details of the proposed techniques. In Section IV, we present the evaluation of the proposed model by comparing the results with adequate baseline and state-of-the-art methods. Finally, we conclude with major takeaways, potential limitations, and future research directions in Section V.

II. RELATED WORKS

The purpose of NAS is to automatically learn the best possible neural architectures and reduce the human drudgery involved in this process [23], [24]. Neural Architecture Search (NAS) has significantly impacted the various important research areas such as image classification [4], [10], object detection [25], [26], semantic segmentation [27], and speech recognition [28] etc.,. In particular, NAS has gained a great deal of traction in the research community because of it’s breakthrough in image classification [4]. Some of the popular initial works in NAS, represent the neural architectures in form of a variable length string and use a combination of recurrent neural net and reinforcement learning to optimize and control it [29]. In general in a typical neural network, similar modules are stacked together to make it more intricate and effective architecture, in case of NAS, we use cells to achieve the same [30]. Efficacy of stacking similar smaller cells and a combination normal cells and reduction cells is also reported in some other important works [31]. A line a recent works [32] propose and advocate for stacking of diversified cell blocks instead of identical blocks and show

---

1. Used single Nvidia Tesla V100 GPU with 16GB graphics RAM.
the effectiveness of such an arrangement in compared to stacking homogeneous cell blocks.

In literature, it is well analysed fact that NAS methods suffer from very high computational cost [33]. Several solutions are proposed in the literature to alleviate this limitation such as the authors in [34] transform the neural architecture search space into a differentiable form and use gradient optimization. The limitation of these models is that they rely on fine tuning a specific structure of the network. Some recent models such as [13] utilizes cell-based search space based search strategy which uses each cell as a direct acyclic graph and relaxes the discrete search strategy. The limitation of [13] is very high memory requirements. A recent work [17] improves over [13] reducing memory usage during search and by improving the search efficiency.

In this work, we propose a data distillation method based on a curriculum learning approach that deals with the limitations mentioned earlier i.e. large memory requirement and humongous search cost of architecture search phase. We extract a very small, efficient, and coherent dataset from the original larger dataset, which contains only samples which are very informative about the corresponding class, we call this dataset as distilled dataset. Using this distilled data, we demonstrate that NAS can be performed on large datasets in a reasonable amount of time without compromising on the accuracy of the model. The intuition behind using distilled dataset during the search phase is that it is wise to only focus on samples which are very informative (containing relevant features) with respect to the underlying class.

III. BACKGROUND AND PROPOSED MODEL

In this section, Firstly, we describe the necessary background details pertaining to intuition and cues related to the proposed approach. Subsequently, we explain the overall methodology and underlying steps involved in DistilNAS.

A. BACKGROUND

In this work, we leverage concepts from curriculum learning [35] in order to accelerate the search stage of NAS. In curriculum learning, a curriculum is devised so that the network starts by learning on a simple task and proceeds progressively to learn from more complex ones. In [15], one such learning curriculum was devised to deal with the problem of noisy labels as follows: Given a dataset, a deep neural network is used to project the images in a given category on to a deep feature space. This is done via the features of the fully connected layer. Naturally, images that are visually similar will be close to each other in the feature space. Then, by formulating a density metric based on the Euclidean distance between the images, density clustering is performed in this feature space.

This clustering step partitions the image dataset into clusters named ‘CREL’, ‘CLRREL’ and ‘CIRREL’. The cluster labelled ‘CREL’ has the highest density whereas the one labelled ‘CIRREL’ has the least density in feature space. Intuitively, an ‘CREL’ cluster corresponds to those images that are visually similar whereas the ‘CIRREL’ cluster corresponds to those images that are visually dissimilar. After creating these clusters for every category, the network is trained sequentially by starting with the ‘CREL’ images and then proceeding to ‘CLRREL’ and ‘CIRREL’ ones.

For doing NAS, we use P-DARTS as the baseline method. So, we will briefly describe the DARTS framework on which P-DARTS is built. As in most NAS approaches, in DARTS also, we search for a robust and efficient building block called cell. Then, we stack several cells together to complete the network architecture for the given task. A cell is a directed acyclic graph (DAG) consisting of ‘n’ nodes (x₀, x₁, ..., xₙ₋₁) where x₀ and x₁ are the input nodes. Every node represents a network layer and the edges between the nodes within a cell are labelled by a fixed operation (e.g., 3 × 3 convolution, identity, maxpool etc) that are chosen from a predefined space of operations, O. The main idea of DARTS is to formulate the information flow from node ‘i’ to node ‘j’ as a weighted sum of |O| operations. This can be mathematically expressed as

\[ f_{i,j}(x_i) = \sum_{o \in O} \frac{\exp(\alpha_{i,j}^o)}{\sum_{o' \in O} \exp(\alpha_{i,j}^{o'})} \cdot o(x_i) \tag{1} \]

where \( x_i \) is the output of the \( i \)-th node and \( \alpha_{i,j}^o \) is the weight parameter for operation \( o(x_i) \). The output of a node \( j \) is given by \( x_j = \sum_{i<j} f_{i,j}(x_i) \) and the output of a cell is obtained by concatenating the output of all the nodes from \( x_2, x_3, ..., x_{n-1} \). Due to the differentiable nature of the function above, we can do gradient descent with respect to both the layer weights and architectural parameters, \( \alpha_{i,j} \)’s. Once the search phase ends, the operation with the highest value of \( \alpha_{i,j} \) on the edge \((i, j)\) is kept and the rest of the operations on \((i, j)\) are discarded. Due to some limitations of DARTS such as severe memory requirement and ‘depth-gap’ in search and evaluation phases, an improved method called P-DARTS is proposed in the literature [14]. In P-DARTS, techniques such as ‘search space approximation’ and ‘search space regularization’ are introduced. The aim of these techniques is to progressively prune the operation search space thereby addressing the issue of depth gap in search and evaluation phases. For more details, please refer [14].

B. PROPOSED METHOD

Given an arbitrary dataset, we hypothesize that:

- Only ‘informative’ images, i.e., images that contain only the features corresponding to the relevant class, must be used in the search phase of NAS.
- Furthermore, ‘uninformative’ images, i.e., those images that also contain features corresponding to the objects that are irrelevant for classification, might have a detrimental effect on the searched architecture.

An intuitive explanation for the first part of the hypothesis is as follows: Firstly, the ‘informative’ images within a class by and large correspond to those images that belong to the \( \text{CREL} \) cluster created via CurriculumNet [15]. The images that belong to the \( \text{CREL} \) cluster are visually very similar.
Also, those images in the \( C_{REL} \) cluster are very important and useful for the corresponding class as they contain only the relevant features that are necessary for the classification task. Moreover, since the clustering criterion ensures that \( C_{REL} \) clusters are sufficiently dense, this means that there are enough samples of a similar type from which the network can learn. For all these reasons, we expect that the samples from the \( C_{REL} \) cluster should be given more weightage when doing NAS.

As for the second part of the hypothesis, ‘uninformative’ images are the samples that belong to the not so relevant clusters (i.e., \( C_{LREL} \) and \( C_{IRREL} \)). The images belonging to these clusters are visually dissimilar. In most of the cases, these samples contain lots of ‘irrelevant’ information also, for an instance they contain features of the concerned class and also features pertaining to objects that are not of interest for classification. So, we conclude that these samples are not of much use in the search phase and in some cases might affect the search process adversely. This is because, we must update the parameters of the architecture primarily based on the ability of the model to detect features of the class of interest. Moreover, the uninformative images are typically used to increase the ability of the model to generalize well. But, in the search phase our training budget is limited and hence we must focus only on detecting those features that are of interest for the classification task. The generalization ability of the model can be dealt with during the evaluation phase where we train it with the full dataset.

We use the CurriculumNet algorithm to create intra-category clusters for a given dataset, i.e., we create the three clusters \( C_{REL} \), \( C_{LREL} \) and \( C_{IRREL} \) for every class respectively. Here, it is important to note that only samples in the training set are clustered and the test data is left untouched. In using CurriculumNet, we employ a pre-trained Resnet-152 for projecting the samples on to a deep feature space (FC layer features) in the first stage. The reason for choosing Resnets is that they have a similar architecture compared to the NAS models: a basic block stacked together appropriately to form the full network. In the ablation study section, we will explain the effect of choosing different architectures for feature extraction. Once the clusters are created (\( C_{REL} \), \( C_{LREL} \), and \( C_{IRREL} \)), we sample a fraction of the images from the \( C_{REL} \) cluster and call this our “distilled data” for the given dataset. The size of the distilled set is constrained by the size of the original dataset. For instance, on Imagenet, we only need 10% of the training set (all belonging to the \( C_{REL} \) cluster) to construct our distilled dataset. In the next section, we will provide experimental justification for our hypothesis.

### IV. EXPERIMENTAL EVALUATION

#### A. RESEARCH QUESTIONS

We design and conduct the experiments to find the answers to the following research questions.

- **RQ1** Are the class informative samples (containing relevant features for corresponding class) are specifically useful during search phase of NAS?
- **RQ2** Can we perform NAS with a smaller, highly efficient distilled dataset, extracted from original dataset and still perform better in terms of search cost and performance?
- **RQ3** Does the performance degrades with the introduction of uninformative samples?
- **RQ4** What is the effect of different architectures of feature extraction before clustering on overall performance of the model?

#### B. DATASETS

In this section, we show the efficacy of the distilled dataset on any arbitrary dataset. In order to do so in the best manner possible, we use the following criterion while choosing the datasets:

- **Size** - We evaluate our model with datasets of varying sizes, ranging from a few thousand images to a million or more.
- **Resolution** - We experiment with both low as well as high resolution images.
- **Class labels** - To evaluate the robustness of our model, we use datasets with number of labels ranging from only ten labels to hundreds or even thousands of labels.

Based on the criterion above, we report experimental results on the following datasets: Imagenet, Food-101, Stanford Cars, Oxford-IIIT Pets, Caltech-101, CIFAR-10 and CIFAR-100. All the experiments were performed on a single Nvidia Tesla V100 gpu. Before we present the results, we mention some notation for our baseline models which will be used from now on:

- **DIST** - denotes the model that is obtained by doing model search on the distilled data of the given dataset.
- **RAND** - represents the model that is obtained by doing model search on a random sample of the given dataset. The size of the sample is the same as the size of the distilled data.
- **MIX** - refers the model that is obtained by doing model search on the data which consists of samples from all the clusters, i.e., \( C_{REL} \), \( C_{LREL} \) and \( C_{IRREL} \). For instance, if the distilled data consists of 10% of the total training data (all of which are sampled from the \( C_{REL} \) cluster), then we need to sample 10% from each of the three clusters for this baseline.
- **COMP** - denotes the model that is obtained by doing model search on the complete dataset.

The models obtained from the search phase are then evaluated on the full dataset. The evaluation is repeated five times and then the average top-1 accuracies are reported. As previously

| Model   | Top-1 Acc. (%) | Params (M) |
|---------|---------------|------------|
| DIST    | 75.75         | 5.4        |
| RAND    | 74.15         | 4.9        |
| MIX     | 71.2          | 4.5        |
| COMP    | 74.63         | 5.32       |

#### TABLE 1. Comparison of DIST with other baselines on Imagenet.
TABLE 2. Comparison of various models on Imagenet in the mobile setting.

| Architecture    | Test Err. (%) | Params (M) | Search Cost (GPU-days) | ×+ | Search Method |
|-----------------|---------------|------------|------------------------|----|---------------|
| Nasnet-A [4]    | 26.0          | 8.4        | 5.3                    | 2000 | 564          | RL          |
| A moebian-C [11]| 24.3          | 7.6        | 6.4                    | 3150 | 570          | evolution   |
| PNAS [39]       | 25.8          | 8.1        | 5.1                    | 225  | 588          | SMBO        |
| MNASNET-92 [40]| 25.2          | 8.0        | 4.4                    | -    | 388          | RL          |
| DARTS (2nd) [13]| 26.7          | 8.7        | 4.7                    | 4.0  | 574          | gradient    |
| SNAS [41]       | 27.3          | 9.2        | 4.3                    | 4.7  | 574          | gradient    |
| GDAS [42]       | 26.0          | 8.5        | 5.3                    | 0.21 | -            | gradient    |
| BayesNAS [43]   | 26.5          | 8.9        | 3.9                    | 0.2  | -            | gradient    |
| P-DARTS (CIFAR-10) [14] | 24.4 | 7.4        | 4.9                    | 0.3  | 557          | gradient    |
| P-DARTS (CIFAR-100) [14] | 24.7 | 7.5        | 5.1                    | 0.3  | 577          | gradient    |
| PC-DARTS (CIFAR-10) [17] | 25.1 | 7.8        | 5.3                    | 0.1  | 586          | gradient    |
| PC-DARTS* (Imagenet) [17] | 24.2 | 7.3        | 5.3                    | 3.8  | 597          | gradient    |
| ProxlessNAS* (GPU) [36] | 24.9 | 7.5        | 7.1                    | 8.3  | -            | gradient    |
| TE-NAS* [37]    | 24.5          | 7.5        | 5.4                    | 0.17 | -            | training-free |
| DIST* (Ours)    | 24.5          | 7.2        | 5.4                    | 1    | 574          | gradient    |

* indicates that the architecture search was done directly on Imagenet.

TABLE 3. Comparison of DIST with other baselines on Food-101.

| Model         | Top-1 Acc. (%) | Params (M) |
|---------------|----------------|------------|
| DIST          | 84.05          | 5.27       |
| RAND          | 83.3           | 5.24       |
| MIX           | 83.02          | 5.05       |
| COMP          | 83.6           | 5.1        |

TABLE 4. Comparison of DIST with other baselines on Stanford Cars.

| Model         | Top-1 Acc. (%) | Params (M) |
|---------------|----------------|------------|
| DIST          | 86.4           | 4.93       |
| RAND          | 85.0           | 5.16       |
| MIX           | 84.8           | 5.24       |
| COMP          | 86.5           | 5.38       |

TABLE 5. Comparison of DIST with other baselines on Caltech-101.

| Model  | Mean per-class (%) | Params (M) |
|--------|---------------------|------------|
| DIST   | 65.8                | 5.1        |
| RAND   | 65.5                | 4.8        |
| MIX    | 64.3                | 5.02       |
| COMP   | 64.9                | 5.95       |

1) IMAGENET RESULTS

The Imagenet dataset is amongst the most popular datasets for image classification in computer vision. It consists of nearly 1.3 million images for training and 50k images for testing, distributed over a 1000 classes. Most NAS approaches with the exception of PC-DARTS [17], Proxless-NAS [36] and TE-NAS [37], perform architecture search on the CIFAR datasets and then do transfer learning on the Imagenet. In this work, we perform model search directly via the distilled dataset extracted from original Imagenet. As is the norm in the literature on Imagenet, we work in the mobile setting where the resolution of the input images are $224 \times 224$ and flops limited to 600 million.

The distilled data for Imagenet is constructed by randomly sampling 10% of the training data (roughly 130,000 images distributed equally over the 1000 classes) from the CREL clusters. We run the P-DARTS search on this distilled data. The architecture for search is modified slightly by adding three convolutional layers of stride 2 to reduce the resolution of the image from $224 \times 224$ to $28 \times 28$. The search process of P-DARTS is done in three stages. In the first stage, the architecture consists of five cells which is increased to eleven and seventeen in the second and the third stage respectively. The Dropout [38] for skip-connect are set to 0.1, 0.2 and 0.3 for the three stages respectively. The rest of the search settings remain the same as in P-DARTS where in every stage, the network is trained with a batch size of 128 for 25 epochs. An Adam optimizer with a learning rate of $6 \times 10^{-4}$, a weight decay of 0.001 and momentum set to 0.5 and 0.999 is used for architecture search. The search phase takes about 1 GPU-day approximately including the time needed for clustering the data (0.6 GPU-day for just the search). To achieve a reasonable trade-off between computational complexity and accuracy, we choose the model with one skip-connection and this model is labelled DIST.

In the evaluation phase, we follow the same approach as in the DARTS framework where a network of 14 cells and 48 initial channels is evaluated. We train the network for 350 epochs with a batch-size of 128 on a single Nvidia Tesla V100 GPU. The training process takes about 13 days.
TABLE 7. Comparison of DIST with various baselines on CIFAR-10 and CIFAR-100.

| Model   | CIFAR-10 Top-1 Acc. (%) | Parameters (million) | CIFAR-100 Top-1 Acc. (%) | Parameters (million) |
|---------|-------------------------|----------------------|--------------------------|----------------------|
| DIST    | 96.02                   | 4.47                 | 83                       | 4.27                 |
| RAND    | 94.9                    | 4.48                 | 82.4                     | 4.13                 |
| MIX     | 95.8                    | 4.37                 | 82.8                     | 4.19                 |
| COMP    | 95.5                    | 4.14                 | 83.3                     | 4.07                 |

TABLE 8. Effect of noisy samples on NAS.

| Dataset   | UNINFORMATIVE | DIST | UNINFORMATIVE | DIST |
|-----------|---------------|------|---------------|------|
|           | P (M)         | Top-1| P (M)         | Top-1|
| Imagenet  | 4.86          | 73.8 | 5.4           | 75.75|
| Food-101  | 5.3           | 81.9 | 5.27          | 84.05|

The network is optimized via SGD with a learning rate of 0.08 which is decayed linearly after every epoch, a momentum of 0.9 and a weight decay of $3 \times 10^{-5}$. As before, we apply auxiliary tower and label smoothing while training. The model achieves a top-1 accuracy of 75.75%, averaged over five runs. This is very close to the SOTA achieved by PC-DARTS with an accuracy of 75.8% but needs 3.8 GPU-days for architecture search and 12.5% of the training data. The performance of the various NAS algorithms on the Imagenet is compared with our proposed method in table 2.

2) FOOD-101 RESULTS

Food-101 dataset for image classification is comprised of 101 classes and 75,750 images for training and 25,250 images for testing purposes. To the best of our knowledge, NAS has not been performed directly on Food-101 in any of the previous works. As in the case of Imagenet, we resize the images from a resolution of $224 \times 224$ to $28 \times 28$ by adding three convolutional layers of stride 2. For Food-101, the distilled data is constructed by randomly sampling 20% of the training data from the $C_{REL}$ clusters. This increase in the size of the distilled data as compared to the distilled data of Imagenet can be attributed to the fact that Food-101 is much smaller than Imagenet. Using only 10% of training data as before lead to unstable architectures with lower accuracy in our experiments. Therefore, we had to increase the size of distilled data to 20%. The search process is the same as the one described for Imagenet in the previous section and is performed with a single NVIDIA Tesla V100 GPU. The search process takes about 0.25 GPU-days (0.1 GPU-days for just model search).

In the evaluation phase, we again consider a network comprising of 14 cells and 48 initial channels. The network is trained for 400 epochs with a batch size of 128 on a single Nvidia Tesla V100 GPU. Again, we use the SGD optimizer with a learning rate of 0.1 decayed linearly after every epoch, a momentum of 0.9 and a weight decay of $6 \times 10^{-5}$. We also apply the auxiliary tower and label smoothing in the training process. The model achieves a top-1 accuracy of 84.05% averaged over five runs. The results of our method along with the baselines are given in table 3.

3) STANFORD CARS RESULTS

Stanford Cars is also a very useful smaller dataset in computer vision. It consists of 196 categories with 8,144 images for training and 8,041 images for testing. The input images of resolution $224 \times 224$ are reduced to $28 \times 28$. Since the dataset is small, we randomly sample 50% of the training data from the $C_{REL}$ clusters and use it for architecture search. The search settings are the same as that of Imagenet except for the batch size which is reduced to 64. The search process takes 0.15 GPU-days including the time for clustering.

The network configuration for the evaluation phase is the same as that for Imagenet. We train it for 600 epochs with a batch size of 128. The optimizer used is the SGD with a learning rate of 0.15 decayed linearly every epoch, a momentum of 0.9 and a weight decay of $6 \times 10^{-5}$. As before, we apply auxiliary tower and label smoothing while training. The results from our approach is documented in table 4.

4) CALTECH-101 RESULTS

Caltech-101 is a publicly available dataset that is well studied in computer vision. It comprises of 102 categories with 3,060 images for training and 6,084 images for testing. Here also, we randomly sample 50% of the training images from the $C_{REL}$ cluster and run the P-DARTS search algorithm on it. The search settings are the same as that of Imagenet with the batch size set to 96. The search phase takes 0.2 GPU-days including the time for clustering.

The searched architecture is then evaluated with the same network configuration as that of Imagenet. We train it for 400 epochs with the SGD optimizer whose parameters are the same as the ones used in Imagenet evaluation. Since the Caltech-101 dataset contains class imbalances, we report the mean per-class accuracy. We refer to table 5 for results.

C. OXFORD-IIIT PETS RESULTS

Oxford-IIIT pets dataset consists of 37 classes with 3,680 images for training and 3,369 images for testing. The distilled data for Oxford-Pets is obtained by sampling 50% of the training images from the $C_{REL}$ clusters. Then, we run the P-DARTS algorithm with batch size set to 64. The search phase takes about 0.2 GPU-days.

In the evaluation phase, we train the network for 500 epochs with a batch size of 128. The network is optimized
TABLE 9. Comparison of various baselines when VGG-19 and Densenet-169 are used as feature extractors.

| Model | Food-101 Top-1 Accuracy | Parameters (million) | Caltech-101 Top-1 Accuracy | Parameters (million) |
|-------|-------------------------|----------------------|----------------------------|----------------------|
| DIST-V | 83.5                    | 5.16                 | 81.7                       | 5.2                  |
| DIST-D | 83                      | 5.33                 | 81.6                       | 5.24                 |
| RAND   | 82.9                    | 4.9                  | 81.5                       | 4.8                  |
| MIX-V  | 82.6                    | 5.33                 | 80.6                       | 5.1                  |
| MIX-D  | 81.1                    | 5.39                 | 80.4                       | 5.23                 |
| COMP   | 83.6                    | 5.32                 | 80.9                       | 5.9                  |

Note that DIST-V and MIX-V baselines here are created from the clusters in the feature space of VGG-19. Similarly DIST-D and MIX-D correspond to Densenet-169.

via the SGD with a learning rate of 0.1 decayed linearly after every epoch, a momentum of 0.9 and a weight decay of $8 \times 10^{-5}$. The performance of our approach is given in table 6.

D. CIFAR-10 AND CIFAR-100 RESULTS

CIFAR-10 is one of the most well known datasets for image classification. It consists of 50,000 images for training and 10,000 images for testing purposes, distributed equally over 10 classes. The resolution of the images is $32 \times 32$. For CIFAR-10, we randomly sample 20% of the training images from the $C_{REL}$ clusters as the distilled data. Then, we perform NAS on the distilled data using the P-DARTS method with the search parameters set to the same values as mentioned in the P-DARTS paper. The search process takes about 0.2 GPU-days including the time needed for clustering. For evaluation, we follow the same process as mentioned in the P-DARTS paper.

CIFAR-100 is a similar dataset with the same size as that of CIFAR-10, but comprising of 100 categories. Here also, we randomly sample 20% of the training images from the $C_{REL}$ clusters and use it as the distilled data for doing NAS. The search settings and the parameters are the same as mentioned in the P-DARTS paper. Here we note that the reported top-1 accuracies on CIFAR-10 and CIFAR-100 in the P-DARTS paper were 97.5% and 84.08% respectively. However, we were not able to achieve the same performance when we ran P-DARTS on the full dataset with the same settings on a Nvidia Tesla V100 GPU. Our results on CIFAR-10 and CIFAR-100 are shown in table 7.

E. ABLATION STUDY

1) EFFECT OF NOISY SAMPLES

In this section, we will provide experimental evidence and justification for our hypothesis and expectation that uninformative samples can be detrimental in the architecture search phase. We create two subsets of data, one for Imagenet and one for Food-101 and perform ablation study on them. The reason for choosing these datasets is that they contain substantial number of uninformative samples for each class. The size of the subsets used here is the same as the size of the distilled data used for Imagenet (10%) and Food-101 (20%). The proportion of samples from the $C_{REL}$ and the $C_{IRREL}$ clusters are kept in the ratio of 1:1 for each class. The model search is done via P-DARTS with the same settings as that of Imagenet and Food-101 respectively.

The model evaluation is also done in the same settings as that of Imagenet and Food-101. We start by using the same hyperparameter values for learning rate, weight decay etc. Then, we also do a small grid search around the hyperparameter values previously used and repeat the evaluation process. This is done to ensure that our evaluation of these models is fair. The accuracy reported is the best amongst all the repeated evaluations (more precisely, we repeat the evaluation phase six times on each dataset). We refer to table 8 for results.

2) DIFFERENT ARCHITECTURES AS FEATURE EXTRACTORS

We now discuss the effect of using different architectures for feature extraction before clustering. The architectures we consider here are VGG [44] and Densenet [45]. Specifically, we have used VGG-19 with batch normalization and Densenet-169. Using these two models we create representative subsets for Food-101 and Caltech-101 (Four subsets in total, two each corresponding to the two models). The experiments are conducted with P-DARTS and the settings used are the same as the ones used in Section IV-B2 and IV-B4. As seen from the table 9, our hypothesis that informative samples lead to better architectures still holds. We also note that the individual accuracies are lesser as compared to the model searched on the distilled data created via Resnet-152.

3) EFFECT OF RANDOM SAMPLING

Since we randomly sample from the $C_{REL}$ clusters to create distilled data, it is important to verify that the results are not biased/dependent on sampling itself. To do so, we repeat the sampling process (with replacement) five times and create
five different sets of distilled data on Food-101 and Stanford Cars. Then, we perform the search and the evaluation process of P-DARTS on all of them. From the table, it can be seen that the model complexity and accuracy has negligible variance across the five different subsets. The results are documented in table 10.

V. CONCLUSION
To speed up the search phase and avoid the need to search on a simple proxy dataset (which is not the true representation of the target dataset) are the two major challenges in neural architecture search. In this work, we introduce a very intuitive and effective solution to address these challenges in the context of NAS for image classification. The crux of our approach is to curtail down time/cost incurred in search phase of NAS via focusing on detecting the relevant features only for classification and we do so by using a very intricate smaller distilled dataset to perform the NAS, which is extracted from the original target dataset.

A possible limitation of the proposed approach might be that the model may suffer from overfitting in some cases due to over optimization, this can be addressed in the evaluation phase where we train with the full dataset. We demonstrate the effectiveness of this approach with a series of well-known benchmark datasets, ranging in size from a few thousand to a million or more. We also showcase by experiments that it is futile to utilize uninformative samples at the time of architecture search. Since we focus extensively on image classification, in future it is worth to extend this approach to problems such as object detection, segmentation etc.

REFERENCES
[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 25, 2012.
[2] B. Baker, O. Gupta, N. Naik, and R. Raskar, “Designing neural network architectures using reinforcement learning,” in Proc. 5th Int. Conf. Learn. Represent., Toulon, France, Apr. 2017.
[3] M. Suganuma, S. Shirakawa, and T. Nagao, “A genetic programming approach to designing convolutional neural network architectures,” in Proc. Genet. Evol. Comput. Conf., Jul. 2017, pp. 5369–5373.
[4] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning transferable architectures for scalable image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Salt Lake City, UT, USA, Jun. 2018, pp. 8697–8710.
[5] A. K.-F. Lui, Y.-H. Chan, and M.-F. Leung, “Modelling of pedestrian movements near an amenity in walkways of public buildings,” in Proc. 8th Int. Conf. Control, Automat. Robot., 2022, pp. 394–400.
[6] R. Zhang, L. Xu, Z. Yu, Y. Shi, C. Mu, and M. Xu, “Deep-IRTarget: An automatic target detector in infrared imagery using dual-domain feature extraction and allocation,” IEEE Trans. Multimedia, vol. 24, pp. 1735–1749, 2022.
[7] R. Zhang, L. Wu, Y. Yang, W. Wu, Y. Chen, and M. Xu, “Multi-camera multi-player tracking with deep player identification in sports video,” Pattern Recognit., vol. 102, Jun. 2020, Art. no. 107260. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320320300650
[8] A. K.-F. Lui, Y.-H. Chan, and M.-F. Leung, “Modelling of destinations for data-driven pedestrian trajectory prediction in public buildings,” in Proc. IEEE Int. Conf. Big Data, Dec. 2021, pp. 1709–1717.
[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jan. 2016, pp. 770–777.
[10] Z. Zhong, J. Yan, W. Wu, J. Shao, and C.-L. Liu, “Practical block-wise neural network architecture generation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018.
[11] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, “Regularized evolution for image classifier architecture search,” in Proc. AAAI Conf. Artif. Intell., vol. 33, 2019, pp. 4780–4789.
[12] L. Xie and A. Yuille, “Genetic CNN,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017.
[13] H. Liu, K. Simonyan, and Y. Yang, “DARTS: Differentiable architecture search,” in Proc. 7th Int. Conf. Learn. Represent., New Orleans, LA, USA, May 2019.
[14] X. Chen, L. Xie, J. Wu, and Q. Tian, “Progressive differentiable architecture search: Bridging the depth gap between search and evaluation,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019.
[15] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott, and D. Huang, “CurriculumNet: Weakly supervised learning from large-scale web images,” in Proc. 15th Eur. Conf. Comput. Vis. (ECCV), Munich, Germany, vol. 12141, Sep. 2018, pp. 139–154.
[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet large scale visual recognition challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, 2015.
[17] Y. Xu, L. Xie, X. Zhang, X. Chen, G. Qi, Q. Tian, and H. Xiong, “PC-DARTS: Partial channel connections for memory-efficient architecture search,” in Proc. 8th Int. Conf. Learn. Represent., Addis Ababa, Ethiopia, Apr. 2020.
[18] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Tech. Rep., 2009.
[19] L. Bossard, M. Guillaumin, and L. V. Gool, “Food-101—Mining discriminative components with random forests,” in Proc. ECCV, vol. 8694, 2014, pp. 446–461.
[20] O. M Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, “Cats and dogs,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Jun. 2012, pp. 3498–3505.
[21] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, “3D object representations for fine-grained categorization,” in Proc. IEEE Int. Conf. Comput. Vis. Workshops, Dec. 2013.
[22] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories,” in Proc. Conf. Comput. Vis. Pattern Recognit. Workshop, 2004.
[23] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, X. Chen, and X. Wang, “A comprehensive survey of neural architecture search: Challenges and solutions,” 2020, arXiv:2006.02903.
[24] X. Cheng, Y. Zhong, M. Harandi, Y. Dai, X. Chang, T. Drummond, H. Li, and Z. Ge, “Hierarchical neural architecture search for deep stereo matching,” in Proc. NIPS, 2020.
[25] Y. Chen, T. Yang, X. Zhang, G. MENG, X. Xiao, and J. Sun, “DetNAS: Backbone search for object detection,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/2288255874792c67570428e88c0bcad-Paper.pdf
[26] G. Ghiassi, T.-Y. Lin, and Q. V. Le, “NAS-FPN: Learning scalable feature pyramid architecture for object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 7029–7038.
[27] C. Liu, L.-C. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei, “Auto-DeepLab: Hierarchical neural architecture search for semantic image segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 82–92.
[28] S. Ding, T. Chen, X. Gong, W. Zha, and Z. Wang, “AutoSpeech: Neural architecture search for speaker recognition,” 2020, arXiv:2005.03215.
[29] B. Zoph and Q. Le, “Neural architecture search with reinforcement learning,” 2016, arXiv:1611.01578.
[30] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning transferable architectures for scalable image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 6867–8710.
[31] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, “Efficient neural architecture search via parameters sharing,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 4095–4104.
[32] J. Cui, P. Chen, R. Li, S. Liu, X. Shen, and J. Jia, “Fast and practical neural architecture search,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 6509–6518.
S. N. Prabhakar et al.: DistilNAS: Neural Architecture Search With Distilled Data

[33] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, X. Chen, and X. Wang, “A comprehensive survey of neural architecture search: Challenges and solutions,” 2020, arXiv:2006.02903.

[34] R. Shin, C. Packer, and D. Song, “Differentiable neural network architecture search,” Tech. Rep., 2018.

[35] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in Proc. ICML, vol. 382, 2009, pp. 41–48.

[36] H. Cai, L. Zhu, and S. Han, “ProxylessNAS: Direct neural architecture search on target task and hardware,” in Proc. Int. Conf. Learn. Represent., 2019.

[37] W. Chen, X. Gong, and Z. Wang, “Neural architecture search on imagenet in four hours: A theoretically inspired perspective,” in Proc. Int. Conf. Learn. Represent., 2021.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” J. Mach. Learn. Res., vol. 15, pp. 1929–1958, Jun. 2014.

[39] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, “Progressive neural architecture search,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018.

[40] M. Tan, B. Chen, R. Pang, V. Vasudevan, and Q. V. Le, “MnasNet: Platform-aware neural architecture search for mobile,” CoRR, vol. abs/1807.11626, pp. 1–9, Jun. 2018.

[41] S. Xie, H. Zheng, C. Liu, and L. Lin, “SNAS: Stochastic neural architecture search,” in Proc. Int. Conf. Learn. Represent., 2019.

[42] X. Dong and Y. Yang, “Searching for a robust neural architecture in four GPU hours,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 1761–1770.

[43] H. Zhou, M. Yang, J. Wang, and W. Pan, “BayesNAS: A Bayesian approach for neural architecture search,” in Proc. 36th Int. Conf. Mach. Learn. (ICML), vol. 97, Jun. 2019, pp. 7603–7613.

[44] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. Int. Conf. Learn. Represent., 2015.

[45] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2261–2269.

ANKUR DESHWAL received the master’s degree in computer science and engineering from IIT Kanpur, in 2010. Currently, he is a Staff Engineer with the AI-Computing Group, Samsung Semiconductor India Research, Bangalore. He has authored publications in conferences, such as ISCAS, ICML, and SIGGraph, and granted with several patents in area of compiler, deep learning, and AutoML. His research interests include intersection of deep learning and compiler.

RAHUL MISHRA received the master’s degree in computer science and engineering with specialization in data engineering from IIT-Delhi, in 2013, and the Ph.D. degree in computer science and engineering with specialization in NLP from the University of Stavanger, Norway, in 2021. Currently, he is working as a Senior Staff Engineer with the AI-Computing Group, Samsung Semiconductor India Research, Bangalore. He is a Frequent Invited Reviewer for various top-notch conferences, such as CIKM, CoNLL, IJCAI, and KDD, and journals such as, Expert Systems with Applications, ACM Computing Surveys, and IEEE Access. His research interests include intersection of deep learning, and natural language processing with specific interest in representation learning, self-supervision, and multi-modal learning.

HYEONSU KIM received the bachelor’s degree in computer science and engineering from the Hankuk University of Foreign Studies, Seoul, South Korea. He is a Staff Researcher with Samsung Electronics, Hwaseong, South Korea. His research interests include AI compiler and in-memory processor programming model.

Swaroop N. Prabhakar received the Ph.D. degree in theoretical computer science from The Institute of Mathematical Sciences, Chennai. Currently, he is working on industrial applications at the intersection of NLP and computer vision. His research interest includes algorithmic problems with a mathematical flavor to them.

**ACKNOWLEDGMENTS**

This work was supported by the Samsung Research Funding Center, Samsung Electronics Co., Ltd. under Project Number SRFC-DY-28-18.

**REFERENCES**

[1] S. N. Prabhakar et al., "DistilNAS: Neural Architecture Search With Distilled Data," IEEE Access, vol. 10, no. 1, pp. 124998–125012, 2022.

**AUTHORS**

ANKUR DESHWAL received the master’s degree in computer science and engineering from IIT Kanpur, in 2010. Currently, he is a Staff Engineer with the AI-Computing Group, Samsung Semiconductor India Research, Bangalore. He has authored publications in conferences, such as ISCAS, ICML, and SIGGraph, and granted with several patents in area of compiler, deep learning, and AutoML. His research interests include intersection of deep learning and compilers.

RAHUL MISHRA received the master’s degree in computer science and engineering with specialization in data engineering from IIT-Delhi, in 2013, and the Ph.D. degree in computer science and engineering with specialization in NLP from the University of Stavanger, Norway, in 2021. Currently, he is working as a Senior Staff Engineer with the AI-Computing Group, Samsung Semiconductor India Research, Bangalore. He is a Frequent Invited Reviewer for various top-notch conferences, such as CIKM, CoNLL, IJCAI, and KDD, and journals such as, Expert Systems with Applications, ACM Computing Surveys, and IEEE Access. His research interests include intersection of deep learning, and natural language processing with specific interest in representation learning, self-supervision, and multi-modal learning.

HYEONSU KIM received the bachelor’s degree in computer science and engineering from the Hankuk University of Foreign Studies, Seoul, South Korea. He is a Staff Researcher with Samsung Electronics, Hwaseong, South Korea. His research interests include AI compiler and in-memory processor programming model.

***

Swaroop N. Prabhakar received the Ph.D. degree in theoretical computer science from The Institute of Mathematical Sciences, Chennai. Currently, he is working on industrial applications at the intersection of NLP and computer vision. His research interest includes algorithmic problems with a mathematical flavor to them.