Committee of NAS-based models

Bruno S. Sette*†, Lucas C. Silva*†, Fernando R. Zagatti*†, Lucas N. S. Silva*†,
Daniel Lucrédio*, Helena M. Caseli*, Diego F. Silva*
*Federal University of São Carlos, São Carlos, SP, Brazil
†2W Digital – Digital Lab, São Paulo, SP, Brazil
Emails: bruno.sette@ufscar.br, {lucascardoso, fernando.zagatti, lucas.silva}@estudante.ufscar.br,
{daniel.lucredio, helenacaseli, diegofs}@ufscar.br

2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IJCNN52387.2021.9533446

***Abstract*—Network Architecture Search (NAS) has achieved impressive results and generated models comparable with humans' classifications. Automating the definition of a neural architecture reduces the need for expert work efforts and mitigates human bias from architecture design. NAS techniques usually consist of an algorithm to search for the best architecture in a predetermined space of parameters or functions. Due to the number of deep neural architectures' parameters, this search space includes millions of parameters, which makes NAS a cost procedure and may lead the search to overfit the training set. To reduce NAS search spaces' complexity and still obtain competitive results, we propose CoNAS, a committee of NAS-based models, by restricting the search spaces to perform Differentiable ARchiTecture Search (DARTS). Our results point to improved accuracy over DARTS on CIFAR-10, training the networks from scratch, and Imagenette, using a transfer learning approach.**

***Index Terms*—Neural Architecture Search, Ensemble, DARTS**

I. Introduction

Deep Learning (DL) has achieved great success in many applications [1], [2] such as image analysis [3], speech recognition [4], and text understanding [5]. DL can be performed in a supervised or unsupervised way and it is able to learn multiple-level representations and features in hierarchical architectures for the tasks of classification and pattern recognition [6].

In this context, Neural Architecture Search (NAS) aims to automatically find the best neural network architecture for a given learning task and dataset and promises to advance the field by removing human bias from architecture design. NAS can automatically suggest neural architectures that are comparable or outperform those manually designed [7]–[9].

Figure 1 illustrates the search process performed by NAS in which a search strategy selects an architecture a from a predefined search space A. The architecture is passed to a performance estimation strategy, which returns the estimated performance of A to the search strategy [10].

Elsken et al. [10] categorize the methods applied for NAS according to three dimensions:

- *Search Space*: Defines the set of potential candidate network architectures to explore. To reduce the size of the search space, NAS takes advantage of prior knowledge about common properties of architectures that are suitable for specific tasks.

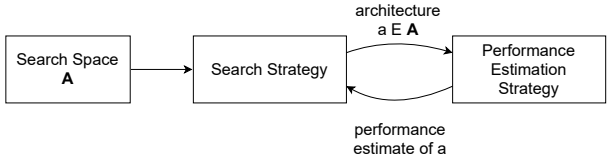


Fig. 1. Abstract illustration of Neural Architecture Search methods [10].

- *Search Strategy*: Guides the investigation through the search space, trying to find the best architecture by trading-off exploitation and exploration.
- *Performance Estimation Strategy*: Defines the process to estimate the quality of each candidate architecture.

Due to the extensive search space, NAS techniques suffer from high computational costs. Besides, current NAS algorithms usually look for the weights and architecture of single neural network architecture, taking risks such as under or overfitting.

We hypothesize that using combinations of architectures generated by NAS brings the possibility of using smaller search spaces and parallel searching, resulting in specific and different architectures. The advantages of this strategy are twofold. First, the reduced search space implicates in a reduced search runtime. Besides, combined (or ensemble) methods can improve classification results compared to individual models in a wide variety of applications [11].

Combined neural networks have shown an improvement of accuracy in classification problems [12], showing that the possibility of a NAS looking for smaller networks in different search spaces can be a good search strategy. However, to ensure this improvement, the combined models need to make independent decisions [13]. In other words, the ensemble requires the individual classifiers to be diverse, which requires a few additional steps to ensure such variety.

In this scenario, we propose using an ensemble of neural networks obtained with NAS, referred to as Committee of NAS-based models (CoNAS). To ensure the diversity of neural architectures in CoNAS, we impose different search space limitations to the Differentiable ARchiTecture Search (DARTS) [14] by fixing different sets of search spaces.

Figure 2 illustrates how CoNAS works. From left to right, the search subspaces are triggered by the training set, and, from there, DARTS works in the conventional way to search

978-0-7381-0366-9/21/$31.00 ©2021 IEEE
Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

for the best network architectures restricted to each search space. In the end, the combination of the predictions of each neural network defines the final decision to a given input data.

We evaluated the committee of three architectures obtained by varying search sub-spaces with DARTS in two distinct scenarios. First, we trained searched for neural architectures from scratch using the dataset CIFAR-10. Later, we used the found architectures to classify images from the Imagenette dataset in a transfer-learning fashion. In both cases, we achieved slightly better accuracy rates. Besides, the combined search takes the same time as searching in the entire search space when done in parallel.

Thus, the main contributions of this work are:

- We propose and evaluate for the first time the combination between neural models found by DARTS, favoring the diversity between these models by limiting their search spaces with different restrictions.
- The proposed method achieved better classification performance than DARTS on CIFAR-10 data when trained from scratch. Moreover, we demonstrate the capacity of reusing discovered network architectures to perform transfer learning, obtaining an increase of accuracy on the Imagenette dataset.
- Through our experimental evaluation, we discuss possible paths for future developments that may bring meaningful advances in NAS research.

II. Related Work

Currently, NAS methods can successfully handle some of the deep learning tasks such as image classification, object detection [14], [15], and semantic segmentation [16]. NAS has significant overlap with transfer learning [17], for example, DARTS uses transfer learning on the network generated in CIFAR-10 for the imagenet.

Next, we present fundamental concepts to define NAS techniques.

A. Search Space

In the context of a *convolutional neural network* (CNN), the search space is parameterized by (i) the maximum number of layers n; (ii) the type of operation every layer executes, e.g., convolution, pooling, or more advanced operations like depthwise separable convolutions [18]; and (iii) hyperparameters regarding operations, e.g., number of filters, kernel size and strides for a convolutional layer [19], or only the number of units for fully-connected networks, activation functions and regularization methods [20]. Figure 3 illustrates a simple search space of chain-structured neural networks.

In this figure, each node in the graphs corresponds to a layer in a neural network, e.g., a convolutional or pooling layer. Different layer types are visualized by different colors. An edge from layer L_i to layer L_j denotes that L_j receives the output of L_i as input. In the left it is illustrated an element of a chain-structured space; and in the right we can see a more complex search space with additional layer types and multiple branches and skip connections [10].

Many NAS approaches require a large number of layers to achieve good performance. Furthermore, in many approaches, each of the neural networks has to be trained from scratch taking a very long time to output the result [21].

B. Search Strategy

Many different search strategies can be used to explore the search space of neural architectures including Random Search (RS), Bayesian Optimization (BO), Evolutionary Methods (EM), Reinforcement learning (RL) and methods based on Gradient Descent (GD). The main search strategies are summarized in Table I.

TABLE I
Overview of different methods to search network architectures in a NAS search space.

Search Method	Results
Random Search	Randomly selects network architecture settings [10].
Reinforcement Learning	Highly competitive results with humans in CIFAR-10 and Treebank. [15]
Evolutionary Algorithms	Performance similar to **reinforcement learning**, however, generating smaller models. [22]
Bayesian Optimization	Some results of this approach may also overcome the **evolutionary algorithms** [23]
Gradient Descent	Four hours of GPU on CIFAR-10 with competitive results compared to **reinforcement learning** and **evolutionary algorithms**. [24]

NAS became one of the top research topics in the machine learning community after Zoph et al. [15] achieved competitive performance on the CIFAR-10 and Penn Treebank [25] benchmarks with a search strategy based on RL. Zoph et al. [15] used huge computational power to achieve this result (800 GPUs for four weeks). After their work, a wide variety of methods were published to reduce computational costs and obtain additional performance improvements.

Bayesian Optimization (BO) achieved several early successes on NAS by suggesting high-level architectures [26]. NAS algorithms using some search strategies obtained competitive architectures for the CIFAR-10 and generated the first automatically adjusted neural networks to win some competitions against human experts.

BO [27] is one of the most popular methods for optimizing hyperparameters, but it has been applied to NAS only in a few works, since typical BO toolboxes are based on Gaussian processes and focus on low-dimensional continuous optimization problems. On the other hand, several works use tree-based models (in particular, Parzen [26] tree estimators or random forests [28]) to effectively research high-dimensional conditional spaces and obtain cutting-edge performance over a wide range of problems, optimizing neural architectures and their hyperparameters together [26].

To use NAS as a reinforcement learning (RL) problem [7], [15], the generation of a neural architecture can be considered the action of the agent of RL. The agent's reward is based on

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

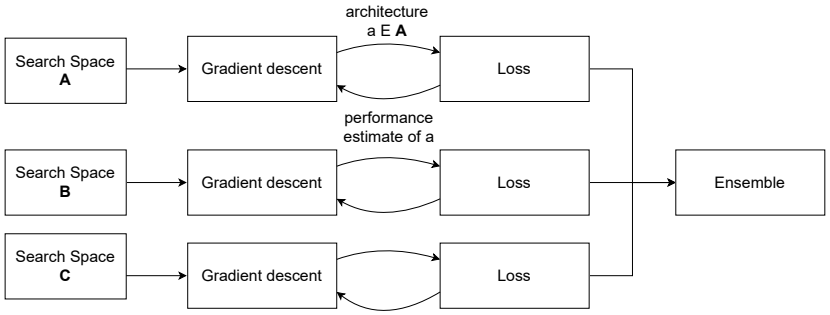


Fig. 2. Abstract illustration of CoNAS [10].

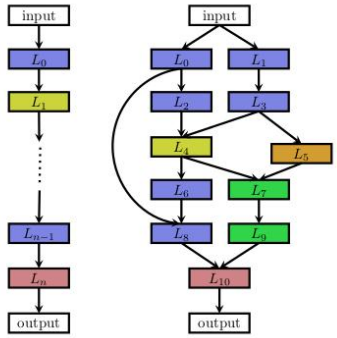


Fig. 3. An illustration of different architecture spaces [10].

an estimate of the performance of the architecture trained on unseen data. RL approaches differ in how they represent the agent's policy and how they optimize it.

An alternative to RL is the neuro-evolutionary approaches, which use evolutionary algorithms to optimize the neural architecture. For the best of our knowledge, the first neuro-evolutionary approach was proposed at least three decades ago (1989) by [29] and their genetic algorithms applied to propose architectures with backpropagation and to optimize their weights. Many neuro-evolutionary approaches have used genetic algorithms to optimize neural architecture and its weights since then; however, when scaling to contemporary neural architectures with millions of weights for supervised learning tasks, weight optimization methods based on Stochastic Gradient Descent (SGD) currently surpass evolutionary ones [10].

The newest neuro-evolutionary approaches [10], [22], [30], [31] use gradient-based methods to optimize weights and evolutionary algorithms only to optimize the neural architecture. Evolutionary algorithms give rise to a population of models, that is, a set of networks; and at each stage of evolution at least one model is picked up from the population to generate children by applying mutations on it. In the context of NAS, mutations are local operations, such as adding or removing a layer, changing the hyper-parameters of a layer, adding skip-connections and changing the training hyper-parameters. After training the children, their performance (for example, in a validation set) is assessed and they only the best ones are added to the population.

Neuro-evolutionary methods differ in the way they select architectures, update populations and generate children. For example, [9], [30], [31] use *tournament selection* to select parents, while [10] selects parents with *multi-objective Pareto*. [9] removes the worst individual from a population, while [22] considered removing the oldest individual and [31] does not remove individuals. To generate children, most approaches initialize a population at random, while [10] employs *Lamarckian inheritance* where the knowledge (in the form of learned weights) is passed from a parent network to its children using morphisms in the network. [9] also allows a child to inherit (from their parents) all parameters that are not affected by the applied mutation; even if this inheritance does not strictly preserve the architecture, it can speed up learning compared to random startup.

Another possible approach to research network architectures is by using gradient descent in neural network architectures [14], [24]. [14] shows that the search for network architecture through gradient descent achieves highly competitive results in CIFAR-10, in addition to surpassing the state of the art in Penn Treebank (PTB). This is a very interesting result, considering that until now the best architectural research methods used non-differentiable research techniques, for example, based on RL [15] or neuro-evolution [9], [31].

Another proposal is a NAS with gradient descent [24]. Such an approach represents the search space as a directed acyclic graph (DAG). This DAG can contains billions of subgraphs, each one representing a type of neural architecture. The results are surprising: the NAS completed a search procedure in four hours of GPU on the CIFAR-10, obtaining an error in the test set of only 2.82% with around 2.5 million parameters.

C. Performance Estimation Strategy

The search strategies presented in section II-B generate network architectures that need to be evaluated somehow to conduct the exploration through the search space. The simplest way to accomplish that is to integrally perform the training stage and then evaluate the generated architecture with the test data. However, considering the order of magnitude *O(nt)* [21], where n is the number of networks to be evaluated during the search and t is the average time spent to evaluate each of

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

these n architectures, performing the training stage leads to a high computational and time cost, spending several days in a GPU [9], [15], [17], [30].

III. Darts

DARTS (Differentiable ARchiTecture Search) is one of most popular NAS techniques. The new paradigm proposed by [14] uses gradient descent to search for a computation cell as the building block of the final architecture. To use gradient descent, Liu et al. [14] creates a continuous search space by relaxing the original search space for possible operations with softmax:

$$o^{-(i,j)}(x) = \sum_{o \epsilon O} \frac{exp(\alpha_o^{(i,j)})}{\sum_{o' \epsilon O} exp(\alpha_{o'}^{(i,j)})} o(x) \tag{1}$$

Darts starts the search with a complete network loaded with operations, as illustrated in Figure 4. The set of possible operations on each neural network node is individually parameterized by the weights represented by α. The gradient descent allows us to select the operations contributing to a lower loss. This way, after a fixed number of epochs (50 by default), we will have the chosen architecture.

DARTS can learn high-performance architecture building blocks with complex graph topologies in a rich search space with 10^{18} architectures.

The following definitions are assumed to define the architectures explored by DARTS. Each directed edge (i, j) is associated with some operation $o(i, j)$ that transforms $x(i)$. Each cell has two input nodes and a single output node. For convolutional cells, the input nodes are defined as the cell outputs in the previous two layers [15], where $x(i)$ is a latent representation (e.g. a feature map in convolutional networks).

Figure 1 shows an overview of DARTS: (a) Operations on the edges are initially unknown. (b) Continuous relaxation of the search space by placing a mixture of candidate operations on each edge. (c) Joint optimization of the mixing probabilities and the network weights by solving a bi-level optimization problem. (d) Inducing the final architecture from the learned mixing probabilities [14].

[14] conducts NAS experiments on CIFAR-10 and PTB and transferable architectures to ImageNet and WikiText-2, respectively.

In CIFAR-10, Liu et al. [14] include the following operations: 3 × 3, 5 × 5 and 7 x 7 separable convolutions, 3 × 3 and 5 × 5 dilated separable convolutions, 3 × 3 max pooling, 3 × 3 average pooling, identity, and zero. All operations are of stride one. Liu et al. [14] hold out half of the CIFAR-10 training data as the validation set. A small network of 8 cells is trained using DARTS for 50 epochs, with batch size 64 and the initial number of channels 16. The momentum SGD are used to optimize the weights w, with initial learning rate ηw = 0.025.

Unfortunately, researchers have pointed out that DARTS suffers from overfitting if executed for too many epochs. For this reason, we can find proposals to improve the chances of DARTS finding architectures that generalize better, such as applying early stopping [32].

IV. Ensemble

The term ensemble is used to identify a set of predictors whose individual decisions are combined or aggregated in some way to predict new examples [11]. This article focuses only on classification problems. However, the combination of techniques can be used for regression problems with only minor cost function changes.

According to the no-free lunch theorem [33], there is no optimal algorithm for all decision problems. For this reason, ensemble techniques betake the decisions of distinct classifiers and aggregate them in a single decision, like a decision made by a committee. Thereby, even if a small number of models make a mistake, the combined answer can still be correct..

The most straightforward technique to aggregate individual classifiers decisions is by majority voting. In this case, the final decision is for the class label with the individual highest number of votes. A model "votes" for a class label k if its output is k. Formally, the majority voting is defined by Equation 2.

$$argmax_{k \in C}(\sum_{i=1}^{m} \phi(i, k)) \tag{2}$$

where C is the set of class labels to assign to an example, m is the number of individual models, and $\phi(i, k)$ is defined by Equation 3.

$$\phi(i, k) = \begin{cases} 1, \text{ if } k = argmax_{j \in C} P_{ij} \\ 0, \text{ otherwise} \end{cases} \tag{3}$$

where P_{ij} is the output score of the i-th model to the j-th label in C.

Although simple, the majority voting disregards the confidence of each classifier in its decision. For instance, consider a scenario with three individual binary classification models. Besides, consider the following obtained scores for a given instance: $P_{10} = 0.51$, $P_{11} = 0.49$, $P_{20} = 0.51$, $P_{21} = 0.49$, $P_{30} = 0.01$, $P_{31} = 0.99$. The committee would decide for the class 0, despite the models who voted for this class have low confidence in their decisions.

An alternative to avoid this limitation of majority voting is considering the models output scores when aggregating their votes in a weighted voting manner. The ensemble technique uses a simple function of each class's scores, S_k, and decides for that with the highest obtained value. Two examples of aggregation rules are the sum (Equation 4) and product (Equation 5).

$$S_k = \sum_{i=1}^{m} P_{ik} \tag{4}$$

$$S_k = \prod_{i=1}^{m} P_{ik} \tag{5}$$

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

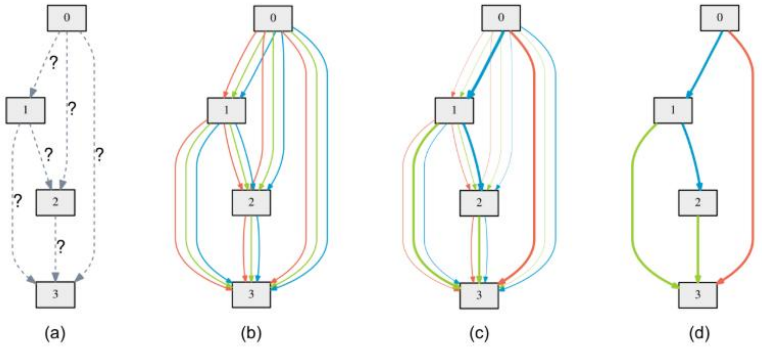


Fig. 4. DARTS starts with a Directed-Acyclic Graph where the edges define the operations. Then, it uses the Gradient Descent to choose the most adequate functions from a pre-defined group of functions.

[34] concludes that the sum rule is the most conservative and most used. The product may have superior results but they are risky.

A good observation regarding the ensemble is that combining similar models is useless. The variability of characteristics and domain spaces are crucial for good ensemble results. Thus, the models to be combined must have a certain level of disagreement.

V. Our Proposal

Our proposal is to create CoNAS. CoNAS is based on the diversity of search spaces, due to the need for diversity between the models to be combined. For this, we defined 3 search darts sub-spaces. The 3 sub-spaces have the following operations:

- **Space A**: 3 × 3, 5 × 5 separable convolutions, 3 × 3 and 5 × 5 dilated separable convolutions, 3 × 3 max pooling, 3 × 3 average pooling, identity, and zero.
- **Space B**: 5 × 5 and 7 x 7 separable convolutions, 5 × 5 dilated separable convolutions, 3 × 3 max pooling, 3 × 3 average pooling, identity, and zero.
- **Space C**: 3 × 3 and 7 x 7 separable convolutions, 3 × 3 dilated separable convolutions, 3 × 3 max pooling, 3 × 3 average pooling, identity, and zero.

Although we set specific parameters, our proposal is a framework. Therefore, changing fundamental elements of the algorithm is straightforward. We implemented our proposal so that these changes are made by execution parameters.

This paper will use the sum of scores to combine the models. The first reason for that choice is because DARTS convolutional networks return probabilistic values. Besides, the sum rule technique is the most conservative and widely applied.

Several previous works use an ensemble strategy based on neural networks of the most diverse types [35]–[40]. The most common approach found in these works is the use different subsets of data for training each network architecture and, then, making up the combination. In this paper, we use an alternative technique, changing architectures instead of data, using DARTS.

VI. Experiments

Liu et al. [14] determines the architecture for final evaluation by running DARTS four times with different random seeds and pick the best cell based on its validation performance obtained by training from scratch for a short period (100 epochs on CIFAR-10).

Our experiments[1] run DARTS a single time for each specific subspace, with the same seeds, and pick the best cell based on its validation performance to determine the final evaluation architecture. We run the DARTS algorithm in each search space and generate the models to combine them.

Figure 5 shows how the accuracy obtained by DARTS, with complete and the three proposed reduced search spaces, and CoNAS evolves through the training epochs on CIFAR-10. All experiments used the default cutout from DARTS.

The results show a difference between the models generated in different search spaces. Specifically, we can note that the accuracy decreases for a model in some epochs while it increases to another one. It seems a piece of evidence that these models are making independent decisions. This difference may have contributed to a superior result in the combination. The combination results were superior to the individual search

[1]The experiments were performed on a computer with Intel (R) Core (TM) i9-7900X CPU @ 3.30GHz, 128 GB of RAM, and a single NVIDIA 2080 TI 12GB RAM GPU.

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

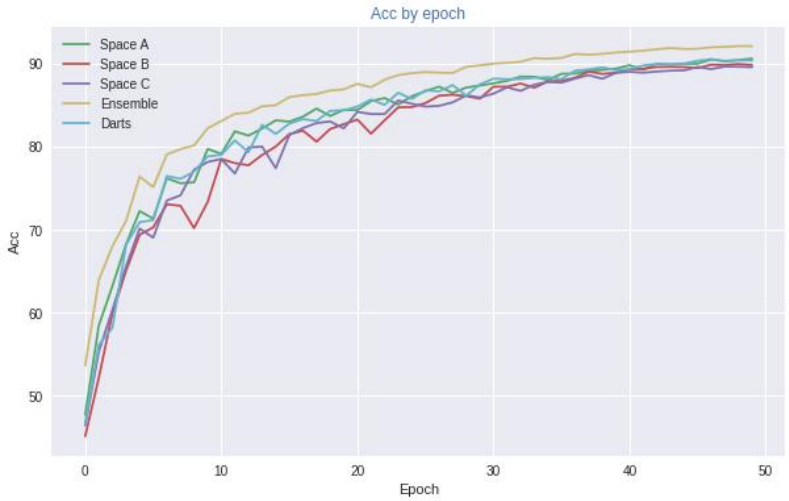


Fig. 5. Accuracy for each search spaces on CIFAR-10.

spaces and the full DARTS search space during the entire process.

Table II shows the test set's accuracy after 50 training epochs on CIFAR-10. We note that the ensemble of DART sub-spaces outperforms the complete DARTS by approximately 1.72%. The accuracy rates of the subspace models shows a small difference between them.

TABLE II
EXPERIMENTAL RESULTS ON CIFAR-10. ACCURACY, NUMBER OF PARAMETERS AND GPU COST.

Architecture	ACC	Params (M)	GPU (Hours)
DARTS	90.36%	1.93	6
DARTS space A	90.58%	1.93	6
DARTS space B	89.80%	0.92	4
DARTS space C	89.56%	1.14	4
DARTS ensemble	**92.08%**	**3.99**	**14***

* This time considers we run each search space sequentially.

It is worth mentioning that we executed the DARTS ensemble sequentially (one at a time in the GPU). However, CoNAS is embarrassingly parallel. If we execute the base models simultaneously, we will obtain a runtime similar to the maximum runtime for a single model. In this case, approximately six hours.

Figure 6 presents the confusion matrix obtained by our method and DARTS using the complete search space.

The ensemble proves to be more efficient than DARTS with the complete search space. The results show a substantial increase in the number of correctly classified examples in almost all classes of CIFAR-10. The only exception is the class "cat", where CoNAS mislabeled two examples more than DARTS.

Figures 7, 8, and 9 illustrate the differences obtained in the final architectures caused by the segmentation of the search spaces, showing that it is possible to obtain diversity of representations by segmenting the DARTS search space.

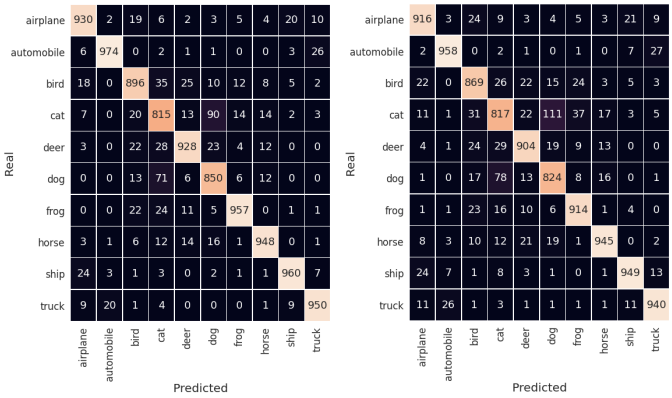


Fig. 6. Confusion matrix obtained by CoNAS (left) and DARTS (right) on CIFAR-10.

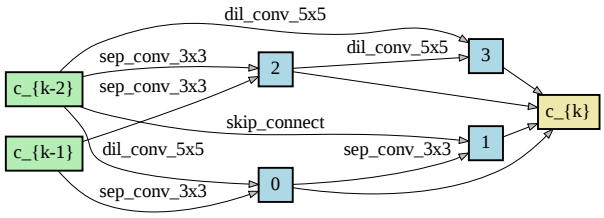


Fig. 7. Snapshots of the most likely normal conv from search space 1.

The Imagenette dataset [41] consists of a subset of 10 "easy" classes from the Imagenet dataset [42]. The goal behind assembling a small version of the Imagenet dataset was mainly because running new experiments using Imagenet takes a long time.

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

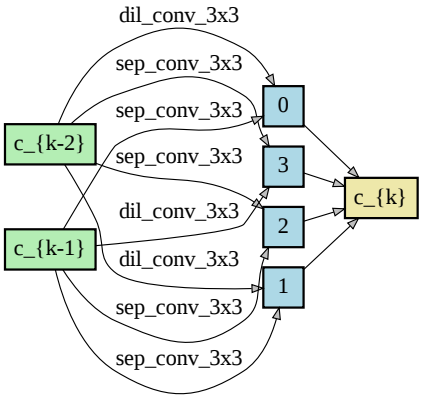


Fig. 8. Snapshots of the most likely normal convfrom search space 2.

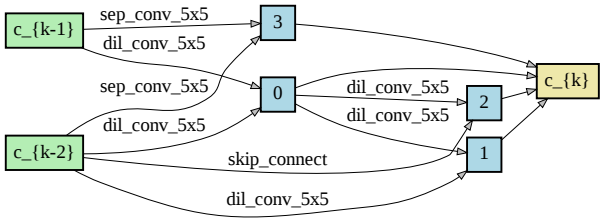


Fig. 9. Snapshots of the most likely normal conv from search space 3.

We experimented with the Imagenette training with transfer learning from the networks trained in CIFAR-10. Since Imagenette does not have a test set, the results shown in Table III refer to the validation set. Overall, the results show that CoNAS presents an increase of about 0.76% in accuracy when compared to the regular DARTS.

TABLE III
EXPERIMENTAL RESULTS ON IMAGENETTE. ACCURACY, NUMBER OF PARAMETERS AND GPU COST.

Architecture	ACC	Params (M)	GPU (Hours)
DARTS	88.05%	4.71	2
DARTS space A	87.23%	4.83	2
DARTS space B	85.63%	5.47	2
DARTS space C	86.67%	4.65	2
DARTS ensemble	**88.81%**	**14.95**	**6***

* This time considers we run each search space sequentially.

Figure 10 illustrates the results of the evaluations in the training stage. The results show the possibility of using early stopping to avoid a possible overfit and reduce the time to evaluate the architecture.

VII. CONCLUSION

We presented CoNAS, a committee of NAS-based models, to classify images. To the best of our knowledge, this is the first proposal of an ensemble algorithm based on automated neural architecture search to explore different search spaces to provide diversity to the induced models.

CoNAS resulted in superior results in the experimented datasets over sub-spaces of DARTS and full DARTS. These results may indicate that investigation on ensemble-based network architecture is a promising path for NAS researchers' future efforts.

As future work, we intend to extend CoNAS to a broader set of data, including different modalities, such as text and time series. Moreover, we plan to evaluate how the limitation of different functions impacts creating more (or less) diverse models, improving the committee's effectiveness. Finally, to generalize our findings and allow CoNAS to scale to more complex domains, we plan to investigate the effects of randomly defining the functions of each sub-spaces and allowing the user to define the number of sub-spaces CoNAS will use.

ACKNOWLEDGMENT

We would like to thank B2W Digital - Innovation Lab - and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - for the financial support.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.
[2] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automatic machine learning: methods, systems, challenges*. Springer, 2019.
[3] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annual review of biomedical engineering*, vol. 19, pp. 221–248, 2017.
[4] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.
[5] *A unified architecture for natural language processing: Deep neural networks with multitask learning*. ACM, 2008.
[6] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Information Fusion*, vol. 42, pp. 146–157, 2018.
[7] *Practical block-wise neural network architecture generation*, 2018.
[8] Z. Zhong, Z. Yang, B. Deng, J. Yan, W. Wu, J. Shao, and C.-L. Liu, "Blockqnn: Efficient block-wise neural network architecture generation," *arXiv preprint arXiv:1808.05584*, 2018.
[9] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," *arXiv preprint arXiv:1802.01548*, 2018.
[10] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," 2019.
[11] T. G. Dietterich, "Machine-Learning Research," *AI Magazine*, vol. 18, no. 4, p. 97, 1997.
[12] S. Tao, "Deep neural network ensembles," *CoRR*, vol. abs/1904.05488, 2019.
[13] M. Ponti, "Combining classifiers: from the creation of ensembles to the decision fusion," in *2011 24th SIBGRAPI Conference on Graphics, Patterns, and Images Tutorials*. IEEE, 2011, pp. 1–10.
[14] C. Liu, L. Chen, F. Schroff, H. Adam, W. Hua, A. L. Yuille, and L. Fei-Fei, "Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation," *CoRR*, vol. abs/1901.02985, 2019.
[15] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," *CoRR*, vol. abs/1707.07012, 2018.
[16] *Searching for efficient multi-scale architectures for dense image prediction*, 2018.
[17] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *CoRR*, vol. abs/1611.01578, 2016.
[18] *Xception: Deep Learning with Depthwise Separable Convolutions*, July 2017.

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.

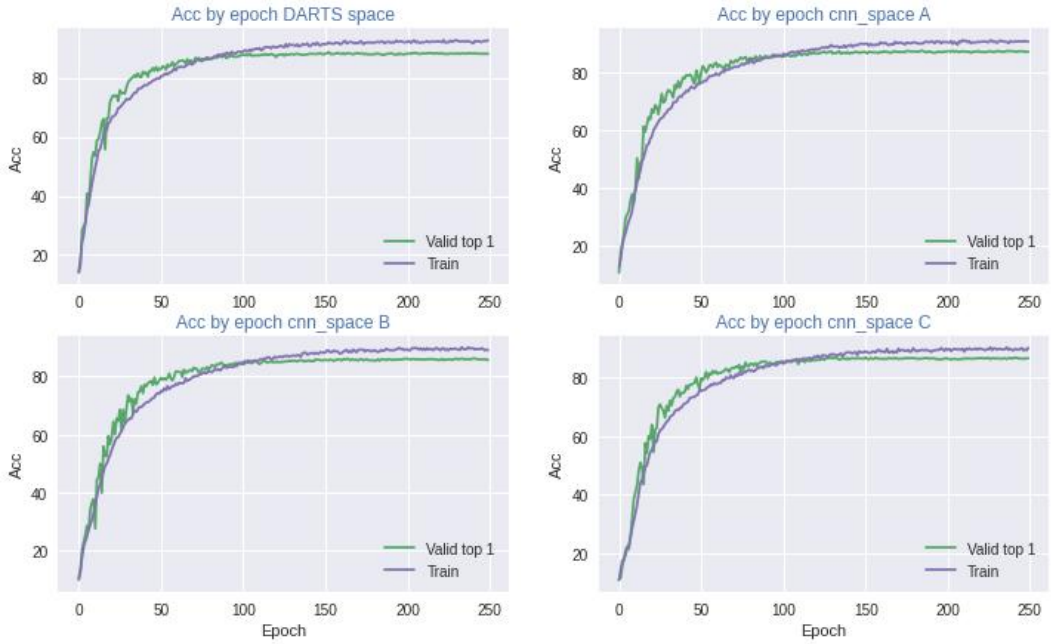


Fig. 10. Results of the validation and training set in the training of neural networks in Imagenette.

[19] B. Baker, O. Gupta, N. Naik, and R. Raskar, "Designing neural network architectures using reinforcement learning," *CoRR*, vol. abs/1611.02167, 2017.

[20] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Towards Automatically-Tuned Neural Networks*, ser. Proceedings of Machine Learning Research, vol. 64. New York, New York, USA: PMLR, 24 Jun 2016.

[21] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," *arXiv preprint arXiv:1806.10282*, 2018.

[22] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Aging evolution for image classifier architecture search," in *AAAI 2019*, 2019.

[23] A. Klein, S. Falkner, S. Bartels, P. Hennig, and F. Hutter, "Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets," in *International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. Fort Lauderdale, FL, USA: PMLR, 20–22 Apr 2017, pp. 528–536.

[24] X. Dong and Y. Yang, "Searching for A robust neural architecture in four GPU hours," *CoRR*, vol. abs/1910.04465, 2019.

[25] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[26] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *International Conference on International Conference on Machine Learning - Volume 28*. JMLR.org, 2013, pp. I–115–I–123.

[27] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, Jan 2016.

[28] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *International Conference on Learning and Intelligent Optimization*. Springer, 2011, pp. 507–523.

[29] G. F. Miller, P. M. Todd, and S. U. Hegde, "Designing neural networks using genetic algorithms," in *International Conference on Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1989, pp. 379–384.

[30] *Large-scale Evolution of Image Classifiers*, ser. ICML'17. JMLR.org, 2017.

[31] H. Liu, K. Simonyan, O. Vinyals, C. Fernando, and K. Kavukcuoglu, "Hierarchical representations for efficient architecture search," *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, pp. 1–13, 2018.

[32] H. Liang, S. Zhang, J. Sun, X. He, W. Huang, K. Zhuang, and Z. Li, "Darts+: Improved differentiable architecture search with early stopping," *arXiv preprint arXiv:1909.06035*, 2019.

[33] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[34] J. Kittler, "Combining classifiers: A theoretical framework," *Pattern Analysis and Applications*, vol. 1, pp. 18–27, 2005.

[35] H. Krishnakumar and D. S. Williamson, "A comparison of boosted deep neural networks for voice activity detection," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2019, pp. 1–5.

[36] H. Nagahamulla, U. Ratnayake, and A. Ratnaweera, "Optimizing member selection for neural network ensembles using genetic algorithms," in *2016 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*, 2016, pp. 1–5.

[37] M. Marino, K. Virupakshappa, and E. Oruklu, "A stacked ensemble neural network classifier for ultrasonic non-destructive evaluation applications," in *2020 IEEE International Ultrasonics Symposium (IUS)*, 2020, pp. 1–4.

[38] Y. Abouelnaga, O. S. Ali, H. Rady, and M. Moustafa, "Cifar-10: Knn-based ensemble of classifiers," in *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2016, pp. 1192–1195.

[39] J. Yang, X. Zeng, S. Zhong, and S. Wu, "Effective neural network ensemble approach for improving generalization performance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 6, pp. 878–887, 2013.

[40] N. Rijal, R. T. Gutta, T. Cao, J. Lin, Q. Bo, and J. Zhang, "Ensemble of deep neural networks for estimating particulate matter from images," in *2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC)*, 2018, pp. 733–738.

[41] J. Howard, "imagenette," 2019. [Online]. Available: https://github.com/fastai/imagenette/

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

Authorized licensed use limited to: UNIVERSIDADE DE SAO PAULO. Downloaded on January 26,2024 at 20:16:31 UTC from IEEE Xplore. Restrictions apply.