Neural Architecture Search For Skin Cancer Detection

Nikhil Kandukuri (✉ nikhilkand@gmail.com)

Birla Institute of Technology and Science - Hyderabad Campus

Sangeetha Sakthivel

University of California, San Diego

Pengtao Xie

University of California, San Diego

Research Article

Keywords: NAS, learning, performance, LPT

Posted Date: October 11th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-953342/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Neural Architecture Search for Skin Cancer Detection

Nikhil Kandukuri(Corresponding Author)¹, Sangeetha Sakthivel², and Pengtao Xie³

¹Birla Institute of Technology and Science - Hyderabad Campus, India,nikhilkand@gmail.com

²University of California, San Diego, United States,svsakthivel@ucsd.edu

³University of California, San Diego, United States,pengtaoxie2008@gmail.com

ABSTRACT

Neural Architectural Search (NAS) is a novel method capable of achieving state-of-the-art performance with limited computational resources and time. These coupled factors have resulted in its increasing popularity in many domains. NAS helps to discover an effective architecture for a given task. In parallel, learning through tests, a technique used in human learning aims at improving learning results: a chain of new assessments are conducted with increasing difficulty; the learner uses them to discover susceptible points, and those susceptible points are further addressed to pass the evaluation effectively. When applied in the case of learning in machines, this technique enhances their learning ability and is called Learning by passing tests(LPT). We propose to use the LPT technique in combination with NAS, particularly for Differentiable Architecture Search(DARTS), Progressive Differentiable Architecture Search(PDARTS) and, Partially Connected Differentiable Architecture Search(PCDARTS) to solve the medical challenge of Skin Cancer Classification. A bilevel optimization algorithm is formulated using LPT and is applied on the HAM10000 dataset and the Kaggle Skin Cancer: Malignant Vs. Benign dataset. Our LPT algorithm coupled with NAS can attain better performance than the traditional NAS methods and different state-of-the-art models for the given classification task.

Introduction

The advent of Neural Architecture Search (NAS) concept has recently garnered much attention from diverse industries. The central ideology of NAS is to build a space of network architectures, develop an efficient algorithm to explore the space, and discover the optimal structure for a given problem statement. This optimal structure is achieved via an automated procedure in place of a manual procedure, which saves time and human effort. [1, 2]. Numerous NAS strategies were formulated and studied over the last few years, out of which we will use three for our study, i.e, DARTS, PDARTS and, PCDARTS. DARTS was mainly introduced to tackle the issue of scalability in NAS, i.e., the evaluation of a large number of architectures. DARTS reduces the search space to be continuous which can be optimized using gradient descent. In DARTS, an optimal cell is repeated many times with an increased depth in the evaluation scenario. PDARTS (Progressive DARTS) reduces the gap between the depth of the training and evaluation scenarios by gradually increasing the depth of the searched architecture. PCDARTS(Partially Connected DARTS) minimizes the memory usage of DARTS by randomly selecting a subset of channels thereby reducing redundancy in the operation selection block. These methods of NAS result in efficiently selecting a good architecture for training our model. We attempt to include the concept of LPT (Learning by passing tests) to improve the overall performance of the state-of-the-art DARTS and its variants. In human learning, a common strategy adopted to improve tangible results is to learn by passing tests. A tester creates more and more difficult tests so that the learner can learn the task more efficiently. The learner, on the other hand, learns to perform the tasks more efficaciously and pass the test created by the tester. This procedure is along the lines of a bilevel technique. Studies have been done recently³, where this approach is applied in the learning of machines, particularly in the task of Neural Architecture Search. It uses two models, a learner that learns to carry out the architecture search and a tester that aims at learning to test the architecture searched more strictly. Our approach aims at proving an improvement over the original NAS methods. We present the use of this technique to solve a significant challenge, i.e., the challenge of skin cancer detection. Skin Cancer is one of the most common types of cancer prevalent in the world today. In spite of being an uncommon type of skin cancer, melanoma is responsible for 75 percent of skin cancer deaths. Melanoma, the most serious type of skin cancer, is diagnosed in over 96,400 people per year, with around 7,200 succumbing to it. The detection of skin cancer at an early stage can make treatment effective and positively impact millions of people. The diagnosis of a malignant mole is made by visual examination of the suspicious skin area. The difficulty of identifying skin moles has been a long-standing one. Automated classification can enable fast and timely diagnosis. The main aim of this study is to diagnose the early stages of skin cancer using the method of Neural Architecture Search, improved by the technique of LPT. Our method is compared with the performance of state-of-the-art baseline models and shows significant improvement.

Related works

Neural Architectural Search

Neural Architecture Search aims to search for an optimal architecture of a neural network to get the best predictive performance. According to the search space, search technique, and performance estimation strategy used, NAS methods can be classified as:⁴

- The type(s) of ANN that can be constructed and optimized are defined by the search space.
- The approach utilized to explore the search space is defined by the search strategy.
- The performance estimation approach assesses a potential ANN's performance based on its design.

Based on search space, the architectures that can be represented and searched by the algorithm may vary from fully connected feed-forward networks to convolution networks. The search space is an important criterion to be considered, as a small search space will result in poor performance, while a large search space might take a lot of time and computation. The search space can be distinguished into two types: a network-based search space or a cell-based search space^{5,6}. The network-based technique discovers the entire architecture while the cell-based technique finds optimal cells and stacks them. The criteria based on evaluation strategy rank the architectures to find the optimal solution. The simplest way is to train each network until convergence and validation accuracy is measured. Many techniques such as weight prediction, weight reuse, and training from scratch are used for this phase. Some evaluations take time, and many approaches to speed up training were also proposed, like using the limited dataset, limited training time, or smaller-sized images in the dataset. The three broad implementations based on search strategy in NAS are reinforcement-based learning, evolutionary-based and differentiable approaches. In reinforcement learning⁷⁻⁹, learning occurs iteratively to generate new architectures by maximizing the accuracy over the validation set. In evolutionary learning^{10,11} the architectures are considered as individuals in a population. Individuals garnering a high validation accuracy have the ability to generate offspring, which are used as substitutes for the individuals garnering a low validation accuracy. However, the above two methods are computationally demanding. For example, the RL-based approach⁵, and evolutionary method¹¹ each requires over 2000 GPU days. Differentiable designs have demonstrated good results while lowering the search time to a few GPU days. They make use of the network pruning strategy. Along with an over-parametrized network, the weights between nodes are learned using gradient descent. Weights having values close to zero are pruned later on. There are constant developments to improve the performance of NAS. LPT framework does not interfere with the functioning of the NAS approaches and thus can be applied to all the NAS methods, which will be discussed in the next section.

Melanoma Classification

Melanoma skin cancer classification has been a topic of interest in the medical industry. Conventional methods require trained dermatologists to view and identify cancerous moles based on symmetry, irregularities, patterns, and diameter. Such conventional strategies require a lot of time and effort from trained dermatologists. Moreover, the distinction between malignant and benign moles is ambiguous and may give different results when examined by experts. The challenge arises from high intra-class variance and high inter-class similarity. The benign and malignant moles look very similar, with the slight difference being that the malignant moles have inconsistent color and asymmetric borders. Over time, a plethora of machine learning approaches have been applied for the diagnosis of malignant moles. Some of them include¹² a k-nearest neighbors based approach,¹³ a support vector machine-based approach and¹⁴ a random forest-based approach. However, these methods require feature extraction to be carried out under a dermatologist's expertise. These features are further not invariant with respect to the difference in input images. The upcoming deep learning technology solves these challenges faced in earlier works^{15,16}. The availability of high-power GPUs and publicly available large datasets further helped in the success of these deep learning models. Some works using¹⁷ CNN, ¹⁸ Deep CNN and ¹⁹ GANs gave promising results. However, finding the right architecture for these remains a manual effort. We aim at solving this using our LPT-based NAS method.

Methods

Differentiable Architecture Search(DARTS)²⁰

Implementing the correct state-of-the-art neural network architectures can be achieved after a considerable effort of human experts in the domain. Recent developments have focused on developing an automated algorithmic approach to replace the manual design process of the training architecture. The automated process of selecting an architectural design has achieved very competitive quantitative results in tasks like image classification.

Current top architecture search algorithms having high-performance benchmarks are computationally expensive and require close to 2000 GPU days to carry out Reinforcement Learning (RL). In DARTS(Differentiable Architecture Search), a new method is developed for efficient architecture search. The main difference lies in that instead of searching over a discrete set of candidate architectures; we allow the search space to be continuous. By adopting this approach, we can optimize the

architecture with respect to its validation set performance by gradient descent. By incorporating gradient-based optimization in comparison to inefficient black-box search, DARTS can achieve high-performance benchmarks using significantly less computational resources.

Progressive Differentiable Architecture Search(PDARTS)²¹

Handcrafted neural networks have traditionally carried out many perceptual tasks by making use of deep learning. The recent emergence of NAS(Neural Architectural Search) has allowed a paradigm shift from manual to automated model design and selection. Early works in NAS were more geared towards adopting optimal configurations in terms of layer type, filter size and number, activation function to construct the network taking inspiration from handcrafted networks like ResNet and DenseNet. The main catch in these methods was that they used Reinforcement Learning and Exploratory Analysis, which led to the computational requirement being unfeasible. Thousands of GPU hours were the staple in such approaches. DARTS was a breakthrough approach that allowed us to do away with such large computational overheads. However, the manner in which DARTS functioned was that it aimed at replicating operations in the search stage that worked well in shallow architectures and applied them for evaluation. The problem that arises is that the evaluation might be better suited by operations that function well in deep architectures. This is known as a depth gap and can hinder performance benchmarks in tasks like visual recognition. A novel approach is taken in Progressive DARTS (PDARTS) to counter this problem and bridge the depth gap. In PDARTS, the search process is divided into multiple stages, and the network depth is progressively increasing at each stage. To reduce the problem of large computational overhead associated with the process, PDARTS makes use of search space regularization, which incorporates operation level dropout to reduce the dominance of skip-connect during training and control the appearance of skip-connect during evaluation.

Partially Connected Differentiable Architecture Search(PCDARTS)²²

Differentiable architecture search (DARTS) gave a template for devising a fast solution that could find effective network architectures; however, it still faced the problem of computational overheads in jointly training a super-network and searching for an optimal architecture. Partially Connected DARTS (PCDARTS) aims at countering these issues by sampling a small part of the super-network to reduce the repetition and redundancy in carrying out explorations in the network space. This method, therefore, performs the neural architecture search in a more efficient manner without hindering any performance benchmark. To expand further, the algorithm carries out an operation search in a subset of channels while bypassing the held-out part in a shortcut. An issue that sometimes results from such a methodology is that there might be an inconsistency in the selection of edges of the super-net primarily caused by sampling different channels. A counter to this is using edge normalization, which augments a new set of edge-level parameters to reduce uncertainty in search. An important outcome of adopting PCDARTS is that training can now take place with a larger batch size allowing for more GPU efficiency leading to faster speeds and higher training stability.

Learning By Passing Tests

³ Studies show that human learning is constantly enhanced when paired with evaluation in the form of tests. The more appropriate the tests are to the content that is being delivered, there is a direct proportion in terms of effective learning. Similar to the underlying principle in human learning, the LPT model applies a similar approach to machine learning. There are two models, the teacher model, and the learner model, which mirror the processes occurring in human learning. The teacher (tester model) creates tests, and the level of difficulty is gradually improved before being fed into the learner model. The learner model has the task of continuously improving its learning prowess to deliver better results on increasingly improving tests in terms of difficulty. The teacher creates test T by selecting samples from a collection of examples called the "test bank." The learner applies its model M to predict T with an accuracy of R. From the tester's perspective; a high R means the test was easy. Hence the tester tries to improvise itself to obtain a small R. From the learner's perspective, R indicates how well it performs, and it focuses on increasing R. This is a typical example of adversarial learning. Figure 1 provides a pictorial representation of the interaction between the two models.

In LPT, a multi-level optimization framework based on one-step gradient descent is adopted where the tester learns to select hard validation examples that cause the learner to make large prediction errors, and the learner adapts its model to counter these prediction errors. The learner's and tester's learning is divided into three levels. The learner learns its network weights W in the first stage by minimizing the training loss $L(\alpha, W, D_{train})$ on the training data D_{train} . $W^*(\alpha)$ is the optimal weights learned at stage one. In the next stage of the algorithm, the tester is able to model its data encoder E and target-task executor Q by minimizing the training loss $L(E, Q, D_{train}) + \alpha (E, Q, \alpha(C, Q, D_b))$. The first part of the loss is defined on the training set. The second part of the loss is defined on the test set created by the test creator. $E^*(C)$ and $Q^*(C)$ are the optimally trained E and Q at this stage. In the third stage, the learner updates its model architecture by minimizing the predictive loss $L(\alpha, W^*(\alpha), \sigma C, E^*(C), D_b)$ on the test created by the tester; the tester updates its test creator by maximizing $L(\alpha, W^*(\alpha), \sigma C, E^*(C), D_b)$ and minimizing the loss on the validation set $L(E^*(C), Q^*(C), D_{val})$.

Figure 1. Learner and Tester Interaction

Optimization and Approximate Architecture Gradient

The task now is to learn the architecture weights, encoder weights, creator weights, and model weights. The three stages are mutually dependent: The objective function in the third stage is defined using the $W^*(\alpha)$ learned in stage one and $E^*(C)$ and $Q^*(C)$ learned in stage two; the updated C and α in the final stage, in turn, change the objective functions in the first two stages, which propagate and result in the values $W^*(\alpha)$, $E^*(C)$, and $Q^*(C)$ being updated. The optimization problem is formulated by nesting three layers of optimization. The inner two layers of optimization, which directly correlate with the first two learning stages, are levied on the outer optimization problem's constraints. The outer optimization problem's objective function relates to the third learning stage. It is further summarized in simple steps in the algorithm given below Figure 2.

$$\begin{aligned}
 & \max_C \min_{\alpha} \frac{1}{|\sigma(C, E^*(C), Db)|} L(\alpha, W^*(\alpha), \sigma(C, E^*(C), Db)) - \beta(E^*(C), Q^*(C), Dval)) [Stage III] \\
 & s.t. E^*(C), Q^*(C) = \min_{E, Q} L(E, Q, Dtrain) + \delta L(E, Q, \sigma(C, E, Db)) [Stage II] \\
 & W^*(\alpha) = \min_W L(\alpha, W, Dtrain) [Stage I]
 \end{aligned}$$

Figure 2. Three Stages of Optimization

Algorithm 1: Optimization algorithm for learning by passing tests

while not converges **do**

1. The learner's architecture is updated by gradient descent;
2. The tester's test creator ascends the gradient computed in the previous step;
3. The tester's data encoder and target-task are updated;
4. The learner's network weights are updated;

end

Deriving discrete cell, assembling and training searched architecture

The weights get updated during the training of the above algorithm. Upon completion of the training, the discrete architecture can be obtained by retrieving the most likely operations in the cells. This retrieval involves retaining the top k strongest operations. The discrete cell obtained is encoded in the form of a genotype. The genotypes discovered are used to assemble the final architecture. The final architecture is assembled by stacking the cells to achieve the desired depth. The depth is increased in creating the final architecture compared to the architecture obtained during the search to reduce memory consumption. The final architecture obtained at the end of this process is used to train the skin cancer dataset and HAM10000 dataset.

Experiments

In this section, we evaluate the performance of the DARTS, PDARTS, PCDARTS ,and their LPT versions on a skin cancer dataset sourced from ISIC and made publicly available and the HAM10000 dataset.

HAM10000 Dataset²³

Despite being full of promise, earlier studies in skin cancer classification were always limited to small dermatoscopy datasets. As an alternative to solve this problem; the HAM10000 dataset was produced in 2018 with a significant resource of dermatoscopy images. The dataset is a collection of 10,015 skin lesion images of 7 different subclasses including (1) Vascular Skin Lesion(VASC), (2) Actinic Keratosis(AKIEC), (3) Melanoma(MEL), (4) Benign Keratosis(BKL), (5) Basal Cell Carcinoma(BCC), (6) Dermatofibroma(DF), (7) Melanocytic nevi(NV). The images were aggregated and published by the Department of Dermatology at the Medical University of Vienna, Austria, and Skin Cancer practice of Cliff Rosendahl Queensland, Australia, spanning over two decades. The images labels were confirmed either by histopathology, reflectance confocal microscopy, follow-up, or by expert consensus. The images present in the dataset have a 600 x 450-pixel resolution. As a whole, the dataset consists of 10,015 skin lesion images; however, a close study of the metadata reveals that there are only 7,470 distinct images. The remaining images comprise duplicates of different magnifications and different angles, as shown in Figure 3. Another parameter to consider is that the dataset is highly imbalanced, Figure 4 provides a visual representation of the disparity. To put the preceding information into context, the smallest class DF contains 115 images while the largest class NV has 6,705 images.

Figure 3. Duplication with multiple angles

Figure 4. Imbalance in raw data

Skin Cancer Dataset: Malignant vs Benign

The dataset has been compiled by International Skin Imaging Collaboration and made publicly available. It comprises high-resolution dermoscopic photographs obtained from individuals of all ages and genders at clinics around Europe, Australia, and the United States. The images are annotated by highly skilled experts into benign and malignant moles. The dataset consists of two directories, one directory containing all the training data and the other directory containing the test data.

The training data consists of 2637 images, out of which 1440 are malignant, and 1197 images are benign. The dataset has only two classes, and they are quite balanced. The test directory contains a total of 660 images, 360 malignant and 300 benign. We split the original training data into a training set and a validation set before running the algorithm to find the optimal genotype/architecture. The training set is used to train the learner and tester during architecture search. The validation set is

considered as the test bank and also the tester's validation data. The training and validation data are taken as input to train an extensive network composed of multiple searched cell copies for architecture evaluation.

Figure 5 further shows a sample of the benign and malign images that are used in the dataset. It also shows some of the features on which manual classification can be done but is inefficient.

Figure 5. Samples of the two classes in the dataset

Pre-Processing

For pre-processing the training images, a patch of 32×32 pixels is cropped at random. The next operation in the chain is random horizontal flipping, which is applied to the 32×32 patches from the previous step. Additional regularization in the form of Cutout, which is a hyper-parameter, is also used to augment the data. Random horizontal and vertical flips are applied to the dataset to allow for better classification. Along similar lines, a section of images is passed through a random rotation of 30 degrees. These two operations allow the model to grasp minute changes in data and prevent misclassification. Finally, each channel's images are mean-centered and normalized (not per pixel). A similar procedure was followed for both datasets.

Experimental Settings

We tested the framework on the two datasets to the following NAS methods 1) DARTS; 2) PDARTS; 3) PCDARTS. The search space in these methods is identical. The operations included in the candidate set are zero, identity, separable convolutions with kernel size 3×3 and 5×5 , dilated separable convolutions with kernel size 3×3 and 5×5 , max pooling with kernel size 3×3 , and average pooling with kernel size 3×3 , identity. In LPT, each cell consists of 7 nodes, and the cells are further stacked up to form the learner's network. We used ResNet-18 as the tester's data encoder. A stack of 8 cells is used as the network during architectural searches, with the starting channel number set to 16. In most experiments, β and δ are set to 1 except for P-DARTS and PC-DARTS. For PDARTS, β , δ are set to 0.5. For PC-DARTS, we use $\beta = 3$, $\delta = 1$.

With a batch size of 64, the search runs for 50 epochs. The approach used in DARTS, PDARTS, and PCDARTS is followed to set the hyperparameters for the weights and architecture of the learner network. SGD optimizer with a weight decay of $3e-4$ and momentum of 0.9 is used to optimize the tester's target-task executor and data encoder. Learning rate is set to 0.0006 with a cosine decay scheduler. The Adam optimizer generates the test creator, with a learning rate of $3e-4$ and a weight decay of $1e-3$. The learner's network is formed by stacking 20 copies of the searched cell during the architectural search assessment, with the initial channel number set to 48. The training of the network is carried out for 150 epochs and a batch size of 96 for both datasets. The experiments were performed on a single 1080 TI. The diagrams below (Figures 6,7,8) display the various genotypes generated for the different configurations we used to perform classification tasks on our datasets. The model is optimized using SGD with a momentum of 0.9 and weight decay of $3e-4$. Evaluation accuracies are calculated by taking the

mean test accuracy after a stipulated number of epochs, in our case, 150 epochs. These values have been reported and presented in our results section.

Figure 6. DARTS-LPT Architecture/Genotype

Figure 7. PDARTS-LPT Architecture/Genotype

Figure 8. PCDARTS-LPT Architecture/Genotype

Baselines

The Neural Architecture Search Models that are present publicly are DARTS, PDARTS, PCDARTS. We use these configurations as baselines and observe for an accuracy increase when integrating this NAS model and LPT for the skin cancer datasets.

Following this train of thought, we first ran our NAS classification tasks using the publicly available code for DARTS, PDARTS, and PCDARTS on the two datasets. We then applied the LPT versions of each configuration on the datasets to observe and record any improvements. In order to gain an insight into how our LPT algorithm fares against the reputed classification models, we have presented the results in the tables below. Tables 1 and 2 show how NAS coupled LPT performs in comparison with other state-of-the-art classification models and traditional NAS models. The number of trainable parameters has also been reported.

Model	Evaluation Accuracy	Model Parameters
VGG-16	72.76%	138M
Xception	73.44%	23M
Inception v3	75.37%	24M
Inception ResNet-v2	78.19%	56M
DenseNet-121	73.69%	7.2M
DenseNet-169	78.49%	12.8M
DARTS	80.33%	1.92M
DARTS-LPT	84.66%	1.96M
PDARTS	81.88%	1.96M
PDARTS-LPT	83.42%	1.96M
PCDARTS	81.40%	0.7M
PCDARTS-LPT	84.06%	0.7M

Table 1. Model Accuracies for HAM10000 Dataset

Model	Evaluation Accuracy	Model Parameters
VGGNet	81.2%	138M
GoogLeNet	81.7%	7M
ResNet50	83.0%	25.6M
ResNet101	83.4%	44.6M
DenseNet100	84.2%	7.19M
DARTS	85.90%	1.9M
DARTS-LPT	87.30%	1.9M
PDARTS	86.27%	1.2M
PDARTS-LPT	88.23%	1.2M
PCDARTS	85.93%	0.7M
PCDARTS-LPT	88.09%	0.7M

Table 2. Model Accuracies for Skin Cancer: Malignant vs. Benign Dataset

Quantitative Results

Observing the tabulated accuracy values for the different Neural Architecture Searches, we obtain accuracy values of greater than 80% for all the NAS configurations coupled with LPT for both datasets. Table 3 shows the evaluation accuracies and standard deviations of the different algorithms on the Skin Cancer: Malignant Vs. Benign dataset and Table 4 holds the results for the HAM10000 dataset. From the tables, an analysis can be drawn. Firstly, the usage of LPT on different NAS approaches for the skin cancer: malignant vs. benign dataset has led to a decrease in classification errors. Specifically, the error has reduced from 14.07% to 11.91% for PCDARTS, 13.73% to 11.77% for PDARTS and, 14.1% to 12.7% for DARTS. A similar trend can also be observed in the case of HAM10000 data. The misclassification rates have dropped from 18.6% to 15.94% for PCDARTS; 18.12% to 16.58% for PDARTS, and 19.67% to 15.34% for DARTS. Reduced classification error shows the increase in efficiency of the models. A closer inspection of the low values of standard deviations demonstrates that our algorithm is consistent over many runs and can be easily replicated for any subsequent research.

On applying Neural Architecture Search coupled with Learning by Passing Tests(LPT), we observe a spike in the accuracy values for each configuration. A maximum increase was observed for the DARTS-LPT configuration in the case of the HAM10000 dataset, with the accuracy increasing by four percent. The PCDARTS-LPT and PDARTS-LPT configurations also obtained a good increase in accuracy from increasing close to three percent in both cases. Analyzing the results obtained for the

Skin Cancer: Malignant vs. Benign dataset, all three configurations have shown improvement of close to two percent when coupled with LPT. These results obtained further demonstrate the potential of the LPT approach for the given challenge.

Skin Cancer: Malignant Vs Benign Dataset

Algorithm	PCDARTS	PCDARTS-LPT	PDARTS	PDARTS-LPT	DARTS	DARTS-LPT
Mean Evaluation Accuracy	85.93%	88.09%	86.27%	88.23%	85.90%	87.30%
Standard Deviation	0.22%	0.31%	1.13%	0.22%	0.44%	0.31%

Table 3. Evaluation Accuracies

HAM 10000 Dataset

Algorithm	PCDARTS	PCDARTS-LPT	PDARTS	PDARTS-LPT	DARTS	DARTS-LPT
Mean Evaluation Accuracy	81.40%	84.06%	81.88%	83.42%	80.33%	84.66%
Standard Deviation	0.89%	0.54%	0.80%	0.55%	0.27%	0.51%

Table 4. Evaluation Accuracies

Qualitative Results

An explanation for the increased accuracies is that assigning greater weights to specific examples in the training set, it allows the model to overcome the difficulties faced by it. Using specific training examples and testing against validation set better equips the model to the nuances in the data and allows for better classification. The rationale behind Learning by Passing Tests is to devise tests with the right mix of difficulty and meaningfulness to further boost the model and achieve accuracy higher than the benchmarks set by current neural architecture search models present. The three configurations of neural architecture approach the problem statement using varied techniques, resulting in a difference in computational time and accuracy between the various configurations. Considering the case of both datasets, the jump in accuracy by using LPT is significant for all the NAS algorithms. Such a performance displays the universality of the method and its ability to be incorporated into diverse setups. The main takeaway is that there is always a significant improvement with the incorporation of LPT into the current state-of-the-art neural architecture searches.

Discussion

Further analysis of the results and the underlying methods which drive our LPT-based classification tasks allows us to infer the following information. In our methodology, the learner is continuously tasked with improving the architecture by clearing tests devised by the tester model, where the tests are continually increasing in terms of difficulty.

LPT creates a new test on the fly, taking into account how the learner performs in the previous round of results. Based on these previous observations, the tester selects a subset of complex examples from the test bank to evaluate the learner. The new test presents a more significant challenge for the learner to overcome and, in turn, motivates the learner to improve its architecture to tackle the new challenge.

However, when you take the cases of the baseline models, a single validation set is used to test the learner. This leaves room for the learner to achieve a performance that is higher than expected. It happens by simply learning the easy examples well and, in turn, performing very well on the majority of uncomplicated examples and not considering the minority of difficult examples. Thus, the learner does not have the capability to deal with challenging cases when unseen data is presented to it.

Another fact to consider while considering baseline models is that the traditional models have a large number of trainable parameters. This number goes as high as 138M in the case of VGG models. Even in the newer ResNet and DenseNet models, the models have the number of trainable parameters over 20M. Comparatively as a whole, Neural Architectural Search models have considerably fewer parameters to train. Despite the difference, they are able to achieve accuracies on par or even higher than many of the robust traditional models used for image classification.

To further this claim, LPT-based NAS configurations can achieve an additional spike in accuracies, simultaneously not increasing the search times even on limited GPU resources; a single GPU in many runs. This performance shows the effectiveness of this methodology and its impact in the medical industry, where a vast amount of image analysis occurs every day.

Ablation Study

In order to prove the capacity of an individual module in Learning by Passing Tests(LPT) and how effective it can function in the absence of another module, a study of an ablation setting has been carried out. In this ablation configuration, the tester module creates tests only considering the parameter of maximizing the level of difficulty without accounting for their meaningfulness. In correspondence to the logical understanding presented, the module pertinent to where the tester learns to perform target-task by leveraging the created tests is removed. For each example d , the tester learns a selection scalar $s(d) \in [0, 1]$ from the test bank without going through a data encoder or test creator. The algorithmic setup for the ablation setting is presented in Figure 9.

$$\max_s \min_{\alpha} \frac{1}{\sum_{d \in D_b} s(d)} \sum_{d \in D_b} s(d) l(\alpha, W * (\alpha), d)$$

$$W * (\alpha) = \min_W L(\alpha, W, D_{train})$$

Figure 9. where $S = s(d) | d \in D_b$. In this study, β and δ are both set to 1.

The results of the ablation experiments are presented in Tables 5 and 6. “Difficult only” denotes that the tester creates tests solely by maximizing their level of difficulty without considering their meaningfulness, i.e., the tester does not use the tests for learning to perform the target task. “Difficult + meaningful” denotes the full LPT framework where the tester creates tests by maximizing both difficulty and meaningfulness.

Configuration	Classification error
Difficult Only(DARTS)	19.40%
Difficult and Meaningful(DARTS)	15.34%
Difficult Only(PDARTS)	19.74%
Difficult and Meaningful(PDARTS)	16.58%
Difficult Only(PCDARTS)	16.77%
Difficult and Meaningful(PCDARTS)	15.94%

Table 5. HAM10000 Dataset

Configuration	Classification error
Difficult Only(DARTS)	14.7%
Difficult and Meaningful(DARTS)	12.7%
Difficult Only(PDARTS)	14.38%
Difficult and Meaningful(PDARTS)	11.77%
Difficult Only(PCDARTS)	14.18%
Difficult and Meaningful(PCDARTS)	11.91%

Table 6. SkinCancer Dataset

Thus on observing the trends for the two datasets tested by us, difficult and meaningful tests which constitute the backbone of the LPT algorithm helps reduce the classification error further.

Conclusion

In this research article, we evaluated the performance of Neural Architecture Search on the classification of Skin Cancer datasets. The main advantage of using NAS coupled with LPT is that the model is not predefined. There exists a search space where the algorithm aims to find the best model for the given task at hand. We observe that by incorporating LPT into our NAS model, we observe a spike in accuracy values compared to the standalone NAS algorithm. For each of the architectures DARTS, PDARTS, and PCDARTS, there was an improvement in the accuracy values when Learning by Passing Tests was incorporated into the code. In an industry such as the medical industry, accuracy is vital for the correct identification and swift diagnosis of a complication. Our model is able to achieve a greater degree of accuracy, which bodes well with what the industry

is aiming for and lays the foundation for subsequent research. Thus, this is an up-and-coming technique that can be used in the classification of skin cancer moles.

References

1. Zhang, M. *et al.* One-shot neural architecture search: maximising diversity to overcome catastrophic forgetting. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE* <https://ieeexplore.ieee.org/document/9247292> (2020).
2. Cheng, X. *et al.* Hierarchical neural architecture search for deep stereo matching. *NeurIPS Proceedings* <https://proceedings.neurips.cc/paper/2020/file/fc146be0b230d7e0a92e66a6114b840d-Paper.pdf> (2020).
3. Du, X., Zhang, H. & Xie, P. Learning by passing tests, with application to neural architecture search. *arXiv* <https://arxiv.org/abs/2011.15102> (2019).
4. Elsken, T., Metzen, J. & Hutter, F. Neural architecture search: A survey. *figshare* <https://arxiv.org/pdf/1808.05377.pdf> (2019).
5. Zhong, B., Vasudevan, V., Shlens, J. & Quoc, V. Learning transferable architectures for scalable image recognition. *Conference on Computer Vision and Pattern Recognition*, <https://arxiv.org/abs/1707.07012> (2018).
6. Zhong, Z., Yan, J., Wu, W., Shao, J. & Liu, C. Practical block-wise neural network architecture generation. *Conference on Computer Vision and Pattern Recognition* <https://arxiv.org/abs/1708.05552> (2018a).
7. Baker, B., Gupta, O., Naik, N. & Rasker, R. Designing neural network architectures using reinforcement learning. *International Conference on Learning Representations* <https://arxiv.org/abs/1611.02167> (2017).
8. H., C., T., C., W., Z., Y., Y. & J., W. Efficient architecture search by network transformation. *Thirty-Second AAAI Conference on Artificial Intelligence* <https://arxiv.org/abs/1707.04873> (2018).
9. Zhong, B. & Le, Q. Neural architecture search with reinforcement learning. *International Conference on Learning Representations* <https://arxiv.org/pdf/1611.01578.pdf> (2017).
10. Real, E., Aggarwal, A., Huang, Y. & Le, Q. Large-scale evolution of image classifiers. *Accepted for publication at ICML 2017* <https://arxiv.org/abs/1703.01041> (2017).
11. Real, E., Aggarwal, A., Huang, Y. & Le, Q. Regularized evolution for image classifier architecture search. *Accepted for publication at AAAI 2019* <https://arxiv.org/abs/1802.01548> (2018).
12. Linsangan, N. & Adtoon, J. *ICBRA '18*: <https://dl.acm.org/doi/10.1145/3309129.3309141> (2015).
13. Gautam, D. & Ahmad, M. Melanoma detection and classification using svm based decision support system. *Annual IEEE India Conference (INDICON)* <https://ieeexplore.ieee.org/document/7443447> (2015).
14. Nandhini, S., Abdul, M., Kumar, S. & Afridi, A. Skin cancer classification using random forest. *International Journal of Management and Humanities* <https://www.ijmh.org/wp-content/uploads/papers/v4i3/C0434114319.pdf> (2019).
15. Alec, X. & Wang, F. Towards interpretable skin lesion classification with deep learning models. *Annual Symposium proceeding AMIA, 2019* <https://pubmed.ncbi.nlm.nih.gov/32308922/> (2019).
16. Jing, W., Wen, Y., Hu, W., Wen, Y. & Liu, X. Darts: Differentiable architecture search. *Skin Lesion Classification Using Densely Connected Convolutional Networks with Attention Residual Learning, 2020* <https://www.mdpi.com/1424-8220/20/24/7080> (2020).
17. Joseph, T. *et al.* Skin cancer classification using convolutional neural networks: Systematic review. *JMR Publications* <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6231861/> (2018).
18. Nasiri, S., Helsper, J., Jung, M. & Fathi, M. Depict melanoma deep-class: a deep convolutional neural networks approach to classify skin lesion images. *6th International Work-Conference on Bioinformatics and Biomedical Engineering* <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-3351-y> (2020).
19. Adhikari, A. Skin cancer detection using generative adversarial network and an ensemble of deep convolutional neural networks. https://etd.ohiolink.edu/apexprod/rws_etd/send_file/send?accession=toledo1574383625473665&disposition=inline (2019).
20. Liu, H., Simonyan, K. & Yang, Y. Darts: Differentiable architecture search. *International Conference on Learning Representations, 2019* <https://arxiv.org/pdf/1806.09055.pdf> (2019).
21. Chen, X., Xie, L., Wu, J. & Tian, Q. Progressive darts: Bridging the optimization gap for nas in the wild. *International Journal of Computer Vision* <https://arxiv.org/pdf/1912.10952.pdf> (2019).

22. Xue, Y. *et al.* Pc-darts: Partial channel connections for memory-efficient architecture search. *International Conference on Learning Representations, 2019* <https://arxiv.org/pdf/1907.05737.pdf> (2020).
23. Tschandl, P. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions, DOI: [10.7910/DVN/DBW86T](https://doi.org/10.7910/DVN/DBW86T) (2018).

Author Contributions

N.K. and S.S. worked equally in performing the various experiments and presenting the results in the form of a research article. P.X. overviewed the manuscript and provided critical analysis and inputs.

Competing Interests

The authors declare no competing interests.