A Review on Neural Architecture Search built upon Convolution Neural Networks for Medical Image Classification

Tomas Slaven
slvtom001@myuct.ac.za
University of Cape Town
South Africa

ABSTRACT

Automatic Machine Learning (AutoML) presents an approach to automating the creation of Machine Learning (ML) models through various interconnected stages - leading to the rapid growth of a hot AutoML sub-topic known as Neural Architecture Search (NAS). NAS aims to create a maximal neural network architecture through automated optimization. NAS details a tool to improve current Convolutional Neural Networks (CNNs) to make them more applicable to real-world situations as in medical image classification, where they remain relatively incompetent. In utilizing NAS for medical image classification, we are looking to find a solution to overcome the difficulties of sparse data for training as witnessed in medical data sets. This review covers the above and presents NAT-M4 and DeepMAD as the promising but untested solutions to the outlined classification problem. Furthermore, we showcase areas requiring further research and innovation.

1 INTRODUCTION

Neural networks, a machine learning process, are based on the structure and function of the human brain. Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs) constitute the current state-of-the-art (SOTA) architectures employed in image classification contexts as a result of their ability to automatically learn a set of features from a labelled data set [33]. However, this ability is restricted to distinct problem sets and does not translate effectively to medical data sets and imaging. The primary reason for this is large labelled data sets, on a scale of millions of images, that have enabled machine learning to make breakthroughs in computer vision by studying this abundance of data. Unlike medical data sets with sizes ranging from 267 to 65 000 subjects [27]. This stark difference in learning environments emphasises the need for a novel approach to medical imaging.

Incorporating additional subjects into a medical data set is not always feasible due to the expensive and time-consuming constraints of curating accurately labelled task-specific data [2]. A potential solution is Data Augmentation (DA) - a strategy for increasing our data set by artificially creating new label-conforming data from our existing data [32]. Further research into DA highlights that DA is not a be-all and end-all solution to the shortcomings of medical image classification. While initial results and experiments indicate that DA yields only approximately a 2.5% increase in image classification task performance on small data sets [26], this does not render DA useless. DA increases task performance and is still an evolving field of research, suggesting that there are other - potentially more beneficial - improvements/mechanics for current image classification models.

The foundation for developing effective image classification models stems from choosing the correct model in the first place. CNNs have been the predominant model in computer vision for decades now [10]. However, the recent emergence of ViTs has put this free-standing position to the test. ViTs introduce a modern learning scheme that surpasses CNNs due to a self-attention mechanism and a cohort of other dedicated components [28][22]. Issues with ViTs when compared to CNNs do include: the need for quadratic computational complexity according to token size [16]; ViTs typically generalize worse than CNNs when trained on small data sets [12]; and ViTs have not yet been well optimized and integrated into hardware platforms in industry [22]. For the goal of medical image classification, CNNs then appear to be our model of choice.

We now embark on the journey of improving current CNN models. The nature of deep learning (multiple layers in a Neural Network) has opened the doors to a vast search space when designing a CNN and promoted the development of better-performing models over time. The vastness of this search space indicates an almost infinite scope for improvement, with the main restriction being the actual design process of a CNN. CNN design is historically a manual process, requiring significant expertise and trial-and-error optimization by experts. However, this can be a time-consuming process and limits our ability to explore the vast design space of CNNs. Given the potential for almost infinite improvement, the question arises: How can we design better CNNs while avoiding the heartaches of the manual design process?

Research into Neuroevolution has led to a potential solution in the form of evolutionary algorithms. This approach encompasses using automated methods to generate architectures for a given task. The topic first gained traction when traditional methods in neuro-evolving topologies outperformed, at the time, some of the best-fixed topology models on challenging benchmarks [24]. Presenting not only a theoretical solution to automated CNN design but a practical example to illustrate its real-world potential. The rise in modern computing power has furthered neuroevolution algorithms as a competitive alternative since they can automatically scale to match this increase in computing resources and thus impact across the entire scope of neural network optimization problems [23].

Conference'17, July 2017, Washington, DC, USA
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

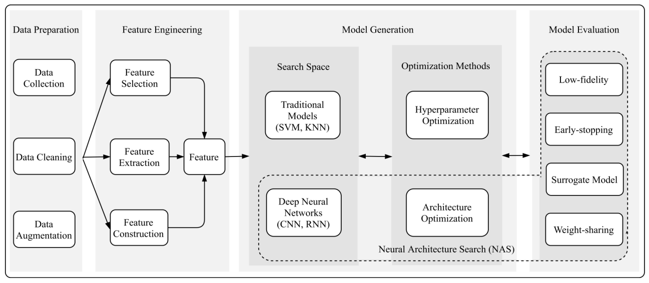

Figure 1: An overview of the AutoML pipeline [6]

Automatic Machine Learning is the latest development from these evolutionary programming revelations that is taking the entire machine learning production line one step further. AutoML has various definitions but boils down to automating the construction of an ML pipeline per a defined computing budget [6]. Complete AutoML systems, such as Cloud AutoML by Google, form easy-to-use solutions to allow people with little ML knowledge to create complex, task-specific models. Analyzing the AutoML process gives an insightful view of the various applications of automation to ML model development. The significant ingredients to the model generation of this AutoML pipeline are Neural Architecture Search (NAS) and Hyper-parameter Optimization (HO). For the sake of our objective, it makes sense to focus on NAS as this defines the optimization of the network's architectural design compared to HO which assumes a given architecture and focuses on tuning hyper-parameters such as learning rate, activation function, and batch size. To create an optimal model, you need an optimal architecture before then optimizing the specified architecture.

This literature review will critique papers on AutoML and NAS while trying to find evolutionary methods to automate the architecture design of CNNs for image classification on small data sets. Specifically, top-performing NAS methods will be discussed to evaluate their strengths, weaknesses and potential for architecture optimization in medical image classification.

2 AUTOML

This section provides a detailed insight into the AutoML pipeline as observed in Figure 1. The AutoML pipeline is a collection of interconnected pieces that go through the entire process of automating the preparation, design, and construction of an ML model.

2.1 Data Preparation

The entry to the ML pipeline, as shown, is Data Preparation. Data Preparation is essential for training in ML, and more so is the data quality. A common concept in computer science is Garbage-In-Garbage-Out (GIGO) which describes how the output quality is dependent on the input quality. Data Preparation aims to overcome the challenge of acquiring good quality data sets through data collection, cleaning and augmentation.

2.1.1 Data Collection.

Data Collection can navigate through two avenues: Data Searching and Data Synthesis.

Data Searching is relevant when extracting Web Data and combines three steps. First, filter results to match keywords by using methods to filter unrelated data and then re-rank the relevant results according to the necessary keywords. Second, ensure the data is correctly labelled using semi-supervised learning self-labelling. Third, ensure the distribution of the data matches the target data set [6].

Data Synthesis differs from Data Searching since it uses a data simulator to generate data. Providing use for particular tasks where data is needed to mimic environments such as those used to test robots and machinery [29].

2.1.2 Data Cleaning.

It is usual for collected data to incorporate unwanted noise. Data Cleaning removes this noise when it becomes a hindrance to model learning. Data cleaning is a complex task requiring domain-specific knowledge. Thankfully the age of automated methods connects us with tools to perform these operations for us. There are many methods for cleaning data from a fixed data set, but these do not accommodate continual learning. Mahdavi et al. [14] details a cleaning workflow orchestrater that learns from past cleaning tasks and presents new cleaning workflows for new data sets. This is beneficial as discerning the data-quality problems when encountering unknown data sets is almost always non-trivial.

2.1.3 Data Augmentation.

Introduction has already provided a description of Data Augmentation (DA). Yet, we must still note its standing in the automated pipeline of AutoML as part of the Data Collection process due to its ability to create label-preserving data from existing data. The DA processes implemented in AutoML solely differ with respect to the type of data collected: images, text and audio. These will be augmented through different techniques as they exhibit unique features [32]. New research in DA has promoted the use of GANs. Generative Adversarial Networks (GANs) are utilized for producing synthetic training data to aid in machine learning tasks. This practice is highly advantageous in cases where the data sets are imbalanced and when working with sensitive information, where it is preferable to minimize the use of original data sets, as is the case with medical data [25]. The particular application of DA through GANs is thus beneficial for this paper's objective as it concerns the augmentation of medical data sets.

2.2 Feature Engineering

Feature Engineering represents the next stage of the pipeline. Features constitute the inherent data properties that ML models seek to interpret and learn. The value features pose to ML is then the upper limit of the capabilities of our learning models. Feature Engineering, therefore, aims to maximize feature extraction from input data via feature selection, feature extraction and feature construction.

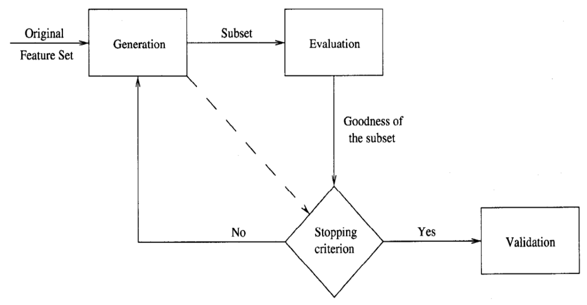

Figure 2: Iterative process of feature selection [3]

2.2.1 Feature Selection.
Feature Selection creates a subset of useful features by removing unnecessary features to reduce overfitting and better model performance. As a result, feature selection strives for distinct features that correspond to object values. The feature selection process is iterative and selects features based on a particular search strategy before evaluating them. Search strategies include Complete Search, Heuristic Search, and Random Search [3]. Once the feature subset has been identified and validated, it is ready for feature construction.

2.2.2 Feature Construction.
Feature Construction, as the name suggests, creates new features from the validated feature subset identified in Feature Selection. This enhances the distinguishability of the feature subset to encourage better generalization of our models. There are numerous techniques that are applicable to this process, depending on the type of features. These esoteric techniques are only advantageous to experienced industry members who often fail to produce optimal feature construction due to the magnitude of possibilities. Automated feature construction methods have since been proposed to aid in this situation and have achieved comparable, if not better, results than industry leaders [20].

2.2.3 Feature Extraction.
Feature extraction is a dimension-reduction process to improve overall efficiency and resource usage, categorising it as pre-processing. Mapping functions execute the extraction process by modifying the existing feature subset according to specific metrics. Traditional approaches include principle component analysis, independent component analysis, non-linear dimensionality reduction, and linear discriminant analysis. Autoencoder algorithms are the modern feature extraction algorithms since they can be automated and stacked in multiple layers to achieve SOTA performance. Autoencoder algorithms are also applicable to various contexts such as supervised learning, unsupervised learning, and now AutoML [19].

2.3 Model Generation

The model generation process of AutoML splits into: *search space* and *optimization methods*. The search space indicates the design principles behind the models, namely traditional models and DNNs, while the optimization methods indicate the optimizations applicable to our models. This displays the precise definition of Architecture Optimization (AO) as it relates to a specific model identified and solidifies our understanding of NAS as it relates to creating a maximal learning model through exploring model search space, applying AO, and evaluating the models to identify the best one.

2.3.1 Search Space.
The search space serves as the blueprint for which AO methods can tinker with. Careful consideration is employed for search space design to ensure it covers a broad range of model architectures to realise the full potential of AO methods. A cohort of search spaces exists, but the commonly used ones in NAS include [6]: Entire-structured Search Space, Cell-based Search Space, Hierarchical Search Space, and Morphism Search Space. The designs of these search spaces are highly detailed, so we have chosen not to explore all of them. Instead, we will be discussing the search spaces of the well-performing NAS models when we talk through NAS in more depth.

2.3.2 Architecture Optimization.
AO aims to discover novel architectures automatically and does so through a variety of methods. The commonly used Architecture methods include Evolutionary Algorithms (EA), Reinforcement Learning, Gradient Descent, Surrogate Model-Based Optimization, and Hybrid Optimization Methods.

EA is an iterative approach where a group of architectures is selected from a previous generation.In each new generation, the most promising architectures undergo variation through cross-over and mutation to produce offspring [13]. This approach mimics the theory behind biological evolution through methods of mutation and cross-over being operations used to replicate the biological understandings of mutation and cross-over. The disadvantages of EA are that they are computationally expensive and the evolutionary operations are performed randomly [18].

Reinforcement learning (RL) is a machine learning training method that revolves around rewarding the system for desired behaviours and punishing the system for inaccuracies. For AO, reinforcement learning maximizes the expected accuracy for a model through this reward and punishment mechanism to great success, as seen with Recurrent Neural Networks [34]. A block in utilizing RL for AO is that it can incur significant time and computational resources, making it infeasible. Other adaptions of the RL version of AO have been proposed, such as ENAS, which offers much faster training times [17]. However, these increased training times are still somewhat longer than other SOTA AO methods since the RL controller may take several actions to obtain a reward.

Gradient Descent differs from the above AO methods in that it does not sample architectures from a discrete space, but explores architectures over a continuous and differentiable search space. This allows gradient descent to cut search times, but comes at an optimization cost since it operates with high-dimensional architecture and weight parameters that are optimized individually. There have been proposed solutions such as optimizing weight and architecture together, however, these models commonly over-fit [5]. Mixed-level optimizations were introduced to overcome this shortfall and have since seen considerable success [5]. However, a disadvantage of gradient descent is that in order to improve search efficiency they relax the categorical candidate operations on continuous variables [6].

The Surrogate model-based optimization method describes searching for a global minimum by constructing a surrogate model of the objective function by continually tracking the results of past evaluations. The surrogate model is then utilized to estimate the optimal architecture. Adaptations to the surrogate models, such as EPNAS, have taken advantage of using neural networks as the surrogate model to improve optimization performance.

Hybrid Optimization is the optimization technique used to combine the other approaches defined above to combine the best of their advantages and also try to offset their disadvantages.

2.3.3 *Hyperparameter Optimization.*

Most NAS methods work with a fixed set of hyperparameters when searching through candidate architectures. After identifying a candidate architecture, it should then be refined through hyperparameter optimization. The commonplace HO methods include Grid Search, Random search and Bayesian Optimization.

Grid Search employs a simple divide and search mechanism where the search space is split into intervals and after total interval evaluation, the best performing point is selected. It is a simple approach that allows parallel exploitation to speed it up. The overhead of this parallelization is manageable for small hyperparameter spaces but quickly becomes inefficient when hyperparameter spaces grow large as such in many deep learning architectures. Adaptions to Grid Search, such as coarse-to-fine Grid Search, help lessen the computational burden of this method [7]. However, it remains more inefficient when compared to other algorithms, such as Random Search.

Random Search differs from Grid Search by choosing the best point among a set of randomly drawn points in the search space. Random search has shown to outperform Grid Search in efficiency and practicality [1], however, due to the nature of selecting random points we are never given the assurance of Random Search producing the optimal value. This introduces a trade-off between search time and performance when comparing Grid Search to Random Search.

Bayesian Optimization involves two key attributes. Firstly, a probabilistic surrogate model similar to the type used in Surrogate Model-based Optimization is used. Drawing from the Bayesian approach to statistics, this model implements a prior distribution of our ideas towards the unknown objective function and an observation model detailing the data production process. Secondly is the integration of a loss function. The loss function is then minimized through an iterative process on the surrogate model to discover an optimal value [21]. This traditional form of Bayesian Optimization has been around for a long time and has had much research dedicated towards improving it. The current SOTA procedure is FABOLAS which reports between 10-100 times performance increases on other SOTA Bayesian Optimization algorithms [9].

2.4 Model Evaluation

Model Evaluation is self-explanatory as it aims to evaluate the performance of generated models. Early approaches in model evaluation consisted of training a neural network until convergence before examining its performance. This straightforward approach effectively evaluates models but has quickly become outdated due to the complexities of modern networks and the computational time required to perform these evaluations. Several algorithms have been proposed to assist in this task, such as Low Fidelity, Weight Sharing, and Early Stopping.

2.4.1 *Low Fidelity.*

Basic intuition lends us to understand that increases in model complexity and data-set size contribute to increased evaluation time. Low fidelity aims to overcome this by adopting approaches that reduce the training data-set and model size during evaluation. While these approaches would appear ineffective in model evaluation since a proportion of the data set or the model size is excluded, they have achieved success. FABOLAS [9], as previously mentioned in Bayesian Optimization, trains on a subset of the data set and achieves SOTA results. Other approaches that reduce model size during training have also found models that work well with much-reduced training times [35]. This outlines that an extended training period does not drastically impact model performance and can often be unnecessary.

2.4.2 *Weight Sharing.*

Straight-forward model evaluations forget about past networks upon completion. Weight sharing proposes to retrieve value from these past evaluations, such as in Network Morphism-based algorithms that can extract weights from previous models [30]. Other variations of weight sharing accomplish this task differently, such as ENAS, by propagating parameters among child nodes [5].

2.4.3 *Early Stopping.*

Early stopping, as commonly known in ML, is a method to reduce overfitting. This idea has transferred to model evaluations by speeding up the process through quitting model evaluations estimated to provide inadequate results on the validation set. A successful novel approach breaks the training data into training-sets and small validation sets. These provide an ongoing prediction of validation performance [15] and can notify the evaluation process when an evaluation requires termination.

3 NEURAL ARCHITECTURE SEARCH

Neural Architecture Search, as mentioned throughout this paper, details an approach designed to tackle the increasing difficulty of architecture engineering. The components of NAS are, as seen in Figure 1: Deep Neural Network Search, Architecture Optimization, and Model Evaluation. These components are designed to automate architecture design and have already beaten manually-designed architectures [4].

As mentioned in Introduction, NAS presents a possible solution to improve medical image classification where medical data-set size is the primary concern in this context. We look towards NAS models to improve medical image classification, but this requires a benchmark that mimics this environment. Medical data sets range from 267 to 65 000 subjects [27], which is very small in image classification terms. Finding a popular benchmark with data sets in the ballpark of 20 000 then becomes almost non-existent. CIFAR-10 presents the best benchmark available, boasting 60 000 images and ten classification groups; falling within the range of current medical data sets and a popular enough benchmark for comparing model performances. Promising NAS models that host the best results on CIFAR-10 include NAT-M4, NAT-M3, Shapley-NAS, and β-darts.

Another intriguing NAS model, DeepMAD, hosts the highest performance on the ImageNet benchmark, and while ImageNet contains over 14 million images, DeepMAD is novel and new. Published in March 2023, DeepMAD, is yet to be properly administered to other benchmarks. For the purpose of this review, we will be looking at DeepMAD, due to its novelty and potential, NAT-M4, and Shapley-NAS. β-darts does not perform as well as the other models, while NAT-M3 is simply a different version of NAT-M4. As such, we will not include them in this review.

3.1 DeepMAD

Mathematical Architecture Design for Deep CNNs (DeepMAD) describes a procedure for designing high-performance convolutional neural network models in a systematic manner. DeepMAD is based on recent advancements in deep learning theories and utilizes a constrained Mathematical Programming (MP) problem to optimize the architecture of CNN models. This approach yields optimized structural parameters, including network width and depth [22].

The MP problem in DeepMAD has a low dimension of a few dozen, which means it is easily solvable on CPUs without the need for customized MP solvers. Therefore, the need for a GPU or creating a deep model in memory is voided. This makes DeepMAD extremely fast, even on servers with limited memory. Once the MP problem is solved, the optimized CNN architecture can be derived from the solution [22].

As mentioned in Introduction, Vision Transformers (ViTs) are the de facto modern image classification technique. ViTs achieve strong results due to a superior learning paradigm to CNNs, however, they come with significant training overheads [8]. DeepMAD showcases that even though CNNs have already been extensively explored, their full potential is yet to be realised. Resulting from DeepMAD achieving performances on par with, if not better than, SOTA ViTs. The authors of DeepMAD demonstrate the effectiveness of DeepMAD by optimizing architectures that utilize conventional convolutional layers as their foundation. DeepMAD produces results on par with or superior to ViT models of the same size and FLOPs. DeepMAD achieves an 82.8% top-1 accuracy on ImageNet-1k with 4.5G FLOPs and 29M parameters, which is better than ConvNeXt-Tiny and Swin-Tiny models of equivalent sizes. Moreover, DeepMAD achieves a top-1 accuracy of 77.7%, outperforming the original ResNet-18 by 8.9% and narrowly surpassing the performance of ResNet-50. All while maintaining an equivalent CNN size to that of its competitors [22].

Although extremely promising, DeepMAD certainly displays some weaknesses from its initial research. DeepMAD involves tuning several hyper-parameters, namely $\{\alpha_i\}$ and $\{\beta,\rho\}$, which will require further optimization to obtain a maximal model. In addition, DeepMAD's current stage focuses on CNN layers, although more advanced options are available, such as transformers. This suggests that it may be possible to extend DeepMAD to other learning paradigms in future research [22].

Regarding DeepMAD's potential towards medical image classification, it has shown to achieve accuracies of around 80% when trained on the CIFAR-100 data set [22]. These accuracies were calculated as part of an experiment to display the importance of the hyperparameter ρ and not primarily executed to excel on the CIFAR-100 benchmark. This still provides evidence of strong model performance on a data set similar in size to medical data sets and, in fact, even more complicated than the CIFAR-10 data set we discussed earlier. This makes a case for DeepMAD being suitable for medical image classification.

3.2 NAT-M4

The basis of NAT-M4 is Neural Architecture Transfer (NAT). NAT automates the discovery of optimal architectures and their associated weights for image classification tasks. NAT uses multiple subnets, creating a supernet, that train simultaneously via weight sharing. This is accomplished through an alternating procedure between supernet adaptation and evolutionary search. This discovers custom neural networks optimized for conflicting objectives. This method is much more efficient than running Neural Architecture Search (NAS) from scratch for each new task, as NAT is able to obtain multiple custom neural networks that cover the entire range of objectives. The resulting supernet then spans the trade-off front of the objectives and can be used for all future deployment-specific NAS without any additional training [13].

The biggest concern facing NAT is the initial prior-search training of the supernet since it takes around 50 days. Thankfully, this is a one-time computational burden as once the supernet is trained it can deploy within various contexts without requiring much training time [13]. The pre-trained network accomplishes this by providing a good starting point for the development of new networks - reducing the number of training iterations required.

This does lead to another weakness of NAT. If the data set used to train the supernet is significantly different from the new task, then the transferred architecture may not be useful. It may even hurt the performance of the new model.

While these are concerns for utilizing NAT, Transfer learning has been shown to improve the performance of new tasks by leveraging the pre-trained model's knowledge, which may have been learned from a greater and more diverse data set [13] and has shown to achieve SOTA performance in image classification against baselines of benchmarks such as CIFAR-10, CIFAR-100 and ImageNet. This leverage of previous model knowledge is evident in NAT's ability to yield the greatest benefits for small, detailed data sets [13]. This is particularly applicable for the case of medical image classification since these are the types of data sets encountered in this field.

3.3 Shapley-NAS

Shapley-NAS introduces a novel approach for neural architecture search (NAS). It is built upon the findings of DARTS and aims to overcome the issues associated with DARTS [11]. Shapley-NAS evaluates the contribution of individual components within a supernet to the validation accuracy of a neural network by utilizing the Shapley value. Unlike previous methods that consider only the magnitude of architecture parameters, Shapley-NAS also takes into account their practical influences on task performance. The Shapley value is an effective tool for handling complex relationships between individual elements by quantifying the average marginal contribution of all possible combinations of operations. Monte Carlo sampling is used to approximate the Shapley value efficiently and employs a momentum update mechanism to reduce fluctuations caused by the sampling process. Shapley-NAS outperforms previous methods, such as DARTS, by showing a higher correlation with task performance. The proposed method achieves superior results on different data sets and search spaces, including a 2.43% error rate on CIFAR-10 and a top-1 accuracy of 23.9% on ImageNet under the mobile setting [31].

A weakness of Shapley-NAS is that the computation of the Shapley value is typically costly, especially in common search spaces where it requires a multitude of evaluations. This can impose significant search costs that limit the practicality of task-specific deployment. As a solution, Monte-Carlo sampling is employed as a method to estimate the Shapley Value for the assessment of the contribution of operations during architecture search. This method provides an unbiased estimate of the Shapley value. Although Monte-Carlo sampling is computationally efficient, it is perhaps not as precise as the exact computation achieved through evaluating all possible subsets for the optimal Shapley Value [31].

Shapley-NAS excels on CIFAR-10, ImageNet, and NAS-Bench201 benchmarks, indicating its effectiveness in identifying optimal architectures [31]. It has also shown particular success in a mobile setting, highlighting its success as a lightweight mechanism for NAS. Based upon its performance on CIFAR-10 and its ability to assume a lightweight form, it displays great potential for developing medical image classification models.

4 DISCUSSION

This literature review has indicated that although there has been limited success with Machine Learning (ML) methods as pathology classifiers for medical studies, this limitation is most likely not a result of ML incapabilities. Rather, it indicates that we have not witnessed the full potential of ML methods. We have observed how traditional ML methods have required a manual design process preventing us from exploiting the entire search space of ML. Empirical results from AutoML models provide an optimistic outlook in the form of automating the design process of ML models. However, AutoML is extremely complex containing multiple stages that each require optimization. As a result, AutoML cannot be executed through a collection of intuitive processes as these would require extraordinary computing budgets. Recent research has advanced steps in the AutoML pipeline to provide more efficient methods and increase the practicality of AutoML, however, due to the nature of Big Data and the number of procedures involved in creating an AutoML pipeline, this remains a computational burden only feasible to world-class institutions such as Google. We have seen that if we seek to dedicate computational resources to a single section of AutoML to improve current ML models in pathology classification, then it should be Neural Architecture Search (NAS).

NAS suffers the same weakness as AutoML - it can be computationally expensive. However, due to NAS being a sub-topic of AutoML, it has shown to be an affordable expense. The three most prominent components for evaluating NAS methods for medical image classification, as discussed in this review, have indicated to be computational complexity, training times, and ability to perform with sparse data.

NAT-M4 is a SOTA NAS method that utilizes a transfer learning paradigm. This allows NAT-M4 to reduce training times, although this only occurs once the initial supernet has been trained which requires a lengthy training time of around 50 days [31]. This remains affordable as it is a once-off cost that can be used to train models for different sub-domains. This increases the efficiency and variety of NAT-M4 solutions. An Achilles heel of NAT-M4 remains to be that it is required for the initial supernet to train on a data set that is similar to the data set used for the task to which the model is applied. This is easily avoided by ensuring the data sets are similar through exploratory data analysis. These weaknesses are not detrimental since approaches have been proposed to mitigate their impact. NAT-M4 has proven to be highly effective for medical image classification, especially for sparse data sets, due to extensive testing and demonstrated impact.

Shapley-NAS presented a different approach to NAS. Instead of incorporating transfer learning as NAT-M4 does, Shapley-NAS uses a variant of differentiable architecture search. There are no particular detrimental weaknesses of Shapley-NAS, and it achieves SOTA performance. However, compared to DeepMAD and NAT-M4 (excluding initial supernet training time) it requires more computational resources. It also does not reach the same performance heights as DeepMAD and NAT-M4 on benchmark data sets. This is potentially a result of the need to utilize Monte Carlo simulations to generate the Shapley value as reported. Shapley-NAS, therefore, does not produce a picture-perfect method for NAS, but rather a well-performing and reliable method that has performed on medical-sized data sets such as CIFAR-10.

DeepMAD presented an extremely novel method for NAS, whereas the other two methods reviewed made advances in commonplace optimization techniques. DeepMAD's novel approach allows it to be by far the most computationally efficient method while still attaining the top spot on the ImageNet benchmark. The MP problem optimization allows for a much more intuitive interpretation of the method and its relevant optimizations to network architecture - compared to the other methods reviewed. A limitation of DeepMAD is the need for further optimization of its parameters. While this does provide concern for the computation required, it has been observed that due to the limited number of parameters and their ranges that this does not render DeepMAD inapplicable. Unlike NAT-M4 and Shapley-NAS, DeepMAD is yet to be extensively applied to sparse data. This does not mean that DeepMAD will not produce an ineffective model for sparse data. As discussed, initial experiments on parameter choice with CIFAR-100 have shown SOTA baseline results. So while its full impact on sparse data is unknown, it is likely to be competitive. DeepMAD is furthermore yet to be integrated with other paradigms, such as transformers, thus highlighting its promising albeit unknown potential.

The three methods reviewed represent a mixture of abilities while utilizing convolutional neural networks. DeepMAD remains the most efficient method, but its effectiveness on sparse data is not yet proven. Shapley-NAS represents the least efficient but remains in the same ballpark allowing it to be competitive. NAT-M4 defines the best-performing method across various benchmarks, while not exhibiting a particular fault - besides its additional reliance on the quality of the data set provided and its initial training time. Due to the mentioned methods' capabilities, these methods outlay a very promising ability to produce accurate models for medical image classification.

What requires improvement for evaluating methods of medical image classification is the development of a robust medical image benchmark for these methods. Currently, several benchmarks are available, but they do not exhibit the required characteristics of a strong benchmark such as CIFAR-10. Furthermore, while many novel NAS methods have been proposed, there has yet to be significant work done in creating hybrid NAS methods. This field would seek to incorporate the advantages of certain approaches and simultaneously aim to mitigate their weaknesses. It would also draw more attention towards improving existing methods rather than attempting to devise novel approaches.

5 CONCLUSIONS

Although AutoML aims to produce a complete solution to automating ML, it is currently infeasible due to computational demands. A sub-topic, NAS, presents a feasible option to improve the performance of image classification models through automation. These modern approaches have shown to increase overall classification accuracy and efficiency when compared to the traditional manual design process - emphasising ML automation as the way forward.

NAS methods have yet to be specialized for image classification on small data sets. Much of the research has been focused on large data sets and creating mobile-setting methods that are computationally efficient. SOTA NAS methods provide much promise to small medical data sets, however, this is yet to be empirically tested on a large scale. The best NAS method for medical data sets appears to be NAT-M4. It has proven benefits for small data sets compared to other NAS methods mentioned - it is reliable and produces SOTA results. DeepMAD remains a dark horse in NAS for medical data sets and more experimentation is needed to evaluate its ability.

Modern NAS methods for image classification are more focused towards large data sets. Medical image classification proposes a niche that requires precise focus and dedicated research that is currently not happening on the same scale as with large data sets. A promising outlook is the increase in recent medical imaging publications that seem to point towards a change in this mindset. A widely used benchmark for medical image classification that captures the inherent challenges of medical data sets would greatly benefit this kind of research.

In final, although we are yet to experience ML methods that can evaluate medical images to the same degree as computer vision methods in autonomous vehicles, the research field is far from stagnant and proposes many possible solutions.

REFERENCES

[1] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of machine learning research* 13, 2 (2012).

[2] Phillip Chlap, Hang Min, Nym Vandenberg, Jason Dowling, Lois Holloway, and Annette Haworth. 2021. A review of medical image data augmentation techniques for deep learning applications. *Journal of Medical Imaging and Radiation Oncology* 65, 5 (2021), 545–563.

[3] Manoranjan Dash and Huan Liu. 1997. Feature selection for classification. *Intelligent data analysis* 1, 1-4 (1997), 131–156.

[4] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *The Journal of Machine Learning Research* 20, 1 (2019), 1997–2017.

[5] Chaoyang He, Haishan Ye, Li Shen, and Tong Zhang. 2020. Milenas: Efficient neural architecture search via mixed-level reformulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11993–12002.

[6] Xin He, Kaiyong Zhao, and Xiaowen Chu. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems* 212 (2021), 106622.

[7] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. 2003. A practical guide to support vector classification.

[8] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2022. Transformers in vision: A survey. *ACM computing surveys (CSUR)* 54, 10s (2022), 1–41.

[9] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2017. Fast bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial intelligence and statistics*. PMLR, 528–536.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.

[11] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055* (2018).

[12] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11976–11986.

[13] Zhichao Lu, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. 2021. Neural architecture transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 9 (2021), 2971–2989.

[14] Mohammad Mahdavi, Felix Neutatz, Larysa Visengeriyeva, and Ziawasch Abedjan. 2019. Towards automated data cleaning workflows. *Machine Learning* 15 (2019), 16.

[15] Maren Mahsereci, Lukas Balles, Christoph Lassner, and Philipp Hennig. 2017. Early stopping without a validation set. *arXiv preprint arXiv:1703.09580* (2017).

[16] Sachin Mehta and Mohammad Rastegari. 2021. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178* (2021).

[17] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. 2018. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*. PMLR, 4095–4104.

[18] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. 2017. Large-scale evolution of image classifiers. In *International Conference on Machine Learning*. PMLR, 2902–2911.

[19] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. 2011. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on international conference on machine learning*. 833–840.

[20] Dan Roth and Kevin Small. 2009. Interactive feature space construction using semantic information. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*. 66–74.

[21] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* 104, 1 (2015), 148–175.

[22] Xuan Shen, Yaohua Wang, Ming Lin, Yilun Huang, Hao Tang, Xiuyu Sun, and Yanzhi Wang. 2023. DeepMAD: Mathematical Architecture Design for Deep Convolutional Neural Network. *arXiv preprint arXiv:2303.02165* (2023).

[23] Kenneth O Stanley, Jeff Clune, Joel Lehman, and Risto Miikkulainen. 2019. Designing neural networks through neuroevolution. *Nature Machine Intelligence* 1, 1 (2019), 24–35.

[24] Kenneth O Stanley and Risto Miikkulainen. 2002. Evolving neural networks through augmenting topologies. *Evolutionary computation* 10, 2 (2002), 99–127.

[25] Fabio Henrique Kiyoiti dos Santos Tanaka and Claus Aranha. 2019. Data augmentation using GANs. *arXiv preprint arXiv:1904.09135* (2019).

[26] Luke Taylor and Geoff Nitschke. 2018. Improving deep learning with generic data augmentation. In *2018 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 1542–1547.

[27] Gaël Varoquaux and Veronika Cheplygina. 2022. Machine learning for medical imaging: methodological failures and recommendations for the future. *NPJ digital medicine* 5, 1 (2022), 48.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[29] Qiang Wang, Shizhen Zheng, Qingsong Yan, Fei Deng, Kaiyong Zhao, and Xiaowen Chu. 2019. Irs: A large synthetic indoor robotics stereo dataset for disparity and surface normal estimation. *arXiv preprint arXiv:1912.09678* 6 (2019).

[30] Tao Wei, Changhu Wang, Yong Rui, and Chang Wen Chen. 2016. Network morphism. In *International conference on machine learning*. PMLR, 564–572.

[31] Han Xiao, Ziwei Wang, Zheng Zhu, Jie Zhou, and Jiwen Lu. 2022. Shapley-NAS: Discovering Operation Contribution for Neural Architecture Search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11892–11901.

[32] Larry Yaeger, Richard Lyon, and Brandyn Webb. 1996. Effective training of a neural network character classifier for word recognition. *Advances in neural information processing systems* 9 (1996).

[33] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. 2018. Convolutional neural networks: an overview and application in radiology. *Insights into imaging* 9 (2018), 611–629.

[34] Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).

[35] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. 2018. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 8697–8710.