ENHANCED NEURAL ARCHITECTURE SEARCH USING SUPER LEARNER AND ENSEMBLE APPROACHES

Seamus Lankford
Adapt Centre, Dublin City University, Dublin, Ireland.
seamus.lankford@adaptcentre.ie

Diarmuid Grimes
Munster Technological University, Cork, Ireland.
diarmuid.grimes@cit.ie

ABSTRACT

Neural networks, and in particular Convolutional Neural Networks (CNNs), are often optimized using default parameters. Neural Architecture Search (NAS) enables multiple architectures to be evaluated prior to selection of the optimal architecture. A system integrating open-source tools for Neural Architecture Search (OpenNAS) of image classification problems has been developed and made available to the open-source community. OpenNAS takes any dataset of grayscale, or RGB images, and generates the optimal CNN architecture. The training and optimization of neural networks, using super learner and ensemble approaches, is explored in this research. Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and pretrained models serve as base learners for network ensembles. Meta learner algorithms are subsequently applied to these base learners and the ensemble performance on image classification problems is evaluated. Our results show that a stacked generalization ensemble of heterogeneous models is the most effective approach to image classification within OpenNAS.

ACM Reference Format:
Seamus Lankford and Diarmuid Grimes. 2021. ENHANCED NEURAL ARCHITECTURE SEARCH USING SUPER LEARNER AND ENSEMBLE APPROACHES. In *2021 2nd Asia Service Sciences and Software Engineering Conference (ASSE '21), February 24–26, 2021, Macau, Macao.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3456126.3456133

1 INTRODUCTION

Open-source AutoML [1] solutions such as Auto-WEKA [2] and TPOT [3] focus on creating simpler neural architectures whereas more complex CNN networks may be developed through libraries such as AutoKeras [4]. In addition to open-source options, commercial solutions also exist. Many large corporations have developed powerful online platforms to enable the generation of neural architectures automatically. Chief among these solutions is Google's Cloud AutoML and Microsoft Azure's AutoML. However, the alternative of using commercial platforms is expensive leaving users with few practical or viable options.

The development of an open-source NAS tool, OpenNAS [5] integrates multiple open-source NAS approaches. With OpenNAS, CNN architectures for grayscale and RGB image datasets are found through the Swarm Intelligence (SI) heuristics of Particle Swarm optimization (PSO) [6] and Ant Colony optimization (ACO) [7]. Pre-trained models using VGG16, VGG19 [8], ResNet50 [9] and MobileNet [10] architectures were also developed. Finally, models derived using SI and pre-trained approaches, were combined into network ensembles and evaluated.

ASSE '21, February 24–26, 2021, Macau, Macao

ACM ISBN 978-1-4503-8908-2/21/02…$15.00
https://doi.org/10.1145/3456126.3456133

2 BACKGROUND

Initially proposed by LeCun [11], CNNs are feed-forward Deep Neural Networks (DNNs) used for image recognition. In this study, SI and ensemble approaches are used to find better combinations of convolutional, pooling and fully connected layers for CNN architectures.

2.1 Neural Architecture Search

The process of automatically finding and tuning DNNs is referred to as Neural Architecture Search (NAS). Systems implementing NAS typically consist of a search space, a search algorithm and an evaluation strategy. The architectures to be evaluated are set out in the search space, the search algorithm determines how the search space is to be explored and the evaluation strategy determines the best architectures on unseen data. Brute force training and evaluation of all possible model combinations is a crude approach to NAS whereas an improvement is to use SI heuristics. Ensembles, combining multiple models, is an alternative which frequently generates better results.

2.2 Swarm Intelligence

Swarm Intelligence (SI) is an important category of heuristics within the domain of Evolutionary Computing. While many SI algorithms exist, the most prominent are Particle Swarm Optimization (PSO) [12] and Ant Colony Optimization (ACO) [13]. Using a PSO algorithm, an open-source python library for CNN optimization, openCNN, was developed by Fernandes et al [14]. An alternative ACO based approach, known as DeepSwarm, was developed by Byla and Pang [15].

2.3 Ensemble Techniques

Cheng Ju et al [16] explored the available options when designing an ensemble for image classification. A detailed analysis was conducted which encompassed the following ensemble techniques: unweighted average, majority voting, Bayes optimal classifier, stacked generalization and a super learner: a cross-validation based stacking method. In their study, the super learner proved the most accurate across all methods. The super learner approach is an extension of stacking in that it creates an ensemble based on cross-validation. A weighted combination of many candidate learners, developed using different algorithms, are combined to build the super learner [17].

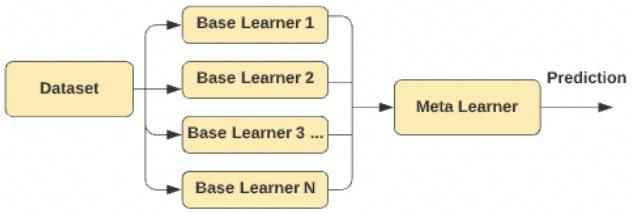

Figure 1: Stacking Approach

3 PROPOSED APPROACH

As part of a neural architecture search, optimization in several areas needs to take place. The number and type of convolutional, pooling and fully connected layers along with their associated parameters must be selected. NAS implementation can be achieved through a variety of approaches including transfer learning using pre-trained networks, network morphism or swarm intelligence. Furthermore, the performance of NAS derived networks can often be enhanced through the use of ensembles.

With this research, ensembles were developed using stacked outputs from base learners. As illustrated in Figure 1, meta learners generated new models by using the stacked ensemble outputs to learn from base learners. Several meta learner algorithms were evaluated. These algorithms include K Nearest Neighbor (KNN), Support Vector Clustering (SVC), Random Forest, Logistic Regression and Multilayer Perceptron (MLP). Combinations of homogeneous or heterogeneous base learners were included in creating the network ensembles. The approach taken in this paper is to focus on stacking ensembles, scikit-learn ensembles and super learner ensembles. With stacking, the accuracy of predictions was improved by combining multiple weaker base learner models. The outputs of N weak learners were combined to form the feature set for a meta learner. Subsequently, the meta learner learns from the prediction outputs of each base learner.

The single level stacking model is further developed with a multi stacked ensemble. With a multi stacked approach, the meta learner is replaced by another set of base learners increasing the model complexity. The super learner approach is an extension of stacking to k-fold cross-validation whereby all models use the same k-fold splits of the data. The meta-model is fit on the out-of-fold predictions from each model. The steps involved in the super learner approach are outlined in Figure 2 from Hubbard's original paper [17].

It can be seen that predictions from the base models, known as candidate learners, serve as the inputs to the meta model which subsequently predicts the target for the training dataset.

3.1 Stacking with neural networks

As outlined in the system design of Figure 4, ensemble outputs are used to create a stacked training dataset for a meta learner. The meta learner is trained by firstly preparing the training dataset and then using the prepared dataset to fit a meta-learner model. In this manner, features of the meta learner dataset are created using predictions from the base learners.

The stacking ensemble approach adopted by OpenNAS enables both heterogeneous and homogeneous ensembles of base learner models to be evaluated. To develop meta learners, the meta-algorithms chosen as the secondary machine learning classifier included Random Forest, Logistic Regression, KNN, MLP and SVC classifiers. With this implementation, there are two principal modes of operation. The first mode involves the creation of base learners. These learners are then used to create ensemble outputs to train meta-learners. The second mode of operation simply loads pre-built base learners to create an ensemble for meta-learners.

3.2 Stacking with Scikit-learn and Super Learners

Using the ML-Ensemble [18] library, a super learner was created using a Random Forest (RF) as the meta-algorithm. Base learners used algorithms from Logistic Regression, SVC, KNN, Bagging, RF and Extra Trees.

Ensemble stacking is achieved using the Stacking Classifier library. Two types of ensembles were implemented: a one-layer stacking ensemble and a multi stacked ensemble consisting of two layers. With the one-layer model, two MLP classifiers (with different learning rates) were used as the base models. These learner outputs feed into a Random Forest which is used as the meta learner.

The multi stacked approach, illustrated in Figure 3, consists of two layers of estimators which are joined using Stacking Classifiers. The first layer consisted of a Random Forest, a KNN and 2 MLP classifiers (with different learning rates). Predictions from layer 1 are passed to a layer consisting of a Decision Tree and a Random Forest Classifier. Layer 2 outputs are then combined with an SVC classifier to make the final prediction.

4 DESIGN

A high-level system architecture overview is presented in Figure 4. Transfer learning, using either feature extraction or fine-tuning of pre-trained networks, is incorporated in the pre-train function. Metaheuristics of particle swarm optimization and ant colony optimization are used to search for the optimal neural architecture as part of the SI design. Particle swarms were created using a psoCNN library [14] and ant colonies were implemented using the DeepSwarm library [15]. Existing AutoML tools, such as AutoKeras [4], are also integrated into the OpenNAS system.

With the ensemble module, there are options to build custom stacked ensembles using either homogeneous or heterogeneous base learners. In addition, there are options to create ensembles using either scikit-learn or a super learner. Base learner outputs are subsequently passed to a suite of meta learner algorithms.

5 EMPIRICAL EVALUATION

5.1 Experimental Setup

Two datasets were chosen for the experimental design, namely CIFAR-10 [19] and Fashion_Mnist [20]. A primary research objective is the development of a neural architecture search tool which chooses the optimal architecture for generic datasets of either grayscale (one channel) or color (three channel) images. The CIFAR-10 dataset meets this requirement in that it is a challenging

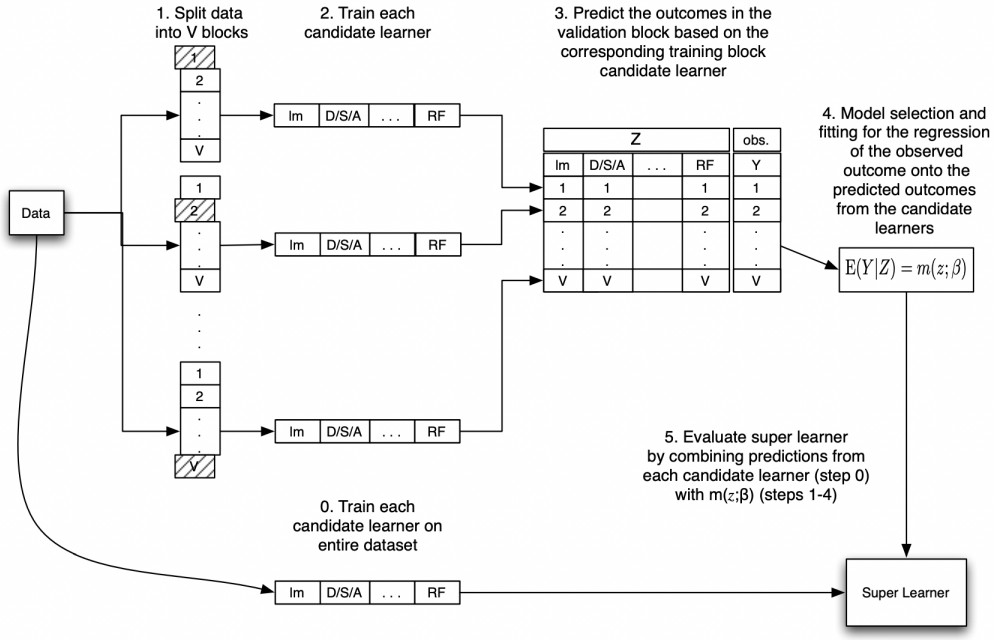

Figure 2: Super Learner Approach [17]

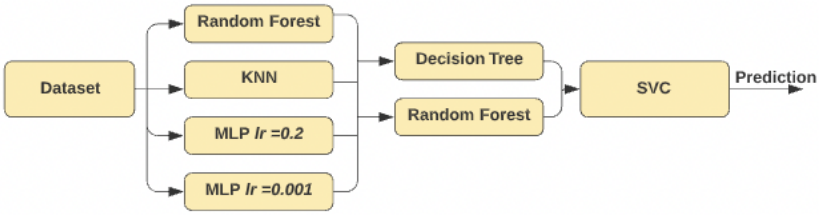

Figure 3: Multi Stacked Approach

dataset of color images. The Fashion_Mnist dataset is also suitable since it is a well-tested and well understood dataset of black and white images.

Models were developed using a lab of machines each of which has an AMD Ryzen 7 2700X processor, 16 GB memory, a 256 SSD and an NVIDIA GeForce GTX 1080 Ti.

For reference, the state of the art (SOA) accuracy achieved on CIFAR-10 is 98.5% whereas with Fashion_Mnist, the SOA accuracy is 94.6% [21].

5.1.1 CIFAR-10. CIFAR-10 is a dataset of 60,000 32x32 color images in 10 classes. There are 6,000 images per class creating a well-balanced dataset. Furthermore, the dataset is divided into five training batches and one test batch, each with 10,000 images. Therefore, there are 50,000 training images and 10,000 test images. The test batch contains exactly 1,000 randomly selected images from each class. Training batches contain the remaining images in random order and contain exactly 5,000 images from each of 10 classes. CIFAR-10 includes the following image categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.

5.1.2 Fashion_Mnist. Fashion_MNIST is a dataset of grayscale images consisting of a training set with 60,000 examples and a test set of 10,000 examples. Each sample is a 28x28 grayscale image, associated with a label from 10 classes. Fashion item images are labelled according to the following classes: T-shirt, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag and Ankle boot.

5.2 Stacking Ensembles

Using CIFAR-10 and Fashion_Mnist datasets, stacking ensembles were evaluated using Random Forest, KNN, MLPC, SVC and Logistic Regression as meta learners. Both homogeneous and heterogeneous stacking ensembles were created.

The ACO ensemble consisted of two models whose architecture was derived from an ACO search whereas the PSO ensemble was composed of two models developed using a PSO heuristic. Likewise, pre-trained homogeneous ensembles consisted of two of the same type of pre-trained network. In addition, the performance of heterogeneous ensembles was also explored. The ensemble, Hetero-4 comprised of four models using two VGG16 and two VGG19 models. The Swarm ensemble was also a four-model ensemble consisting

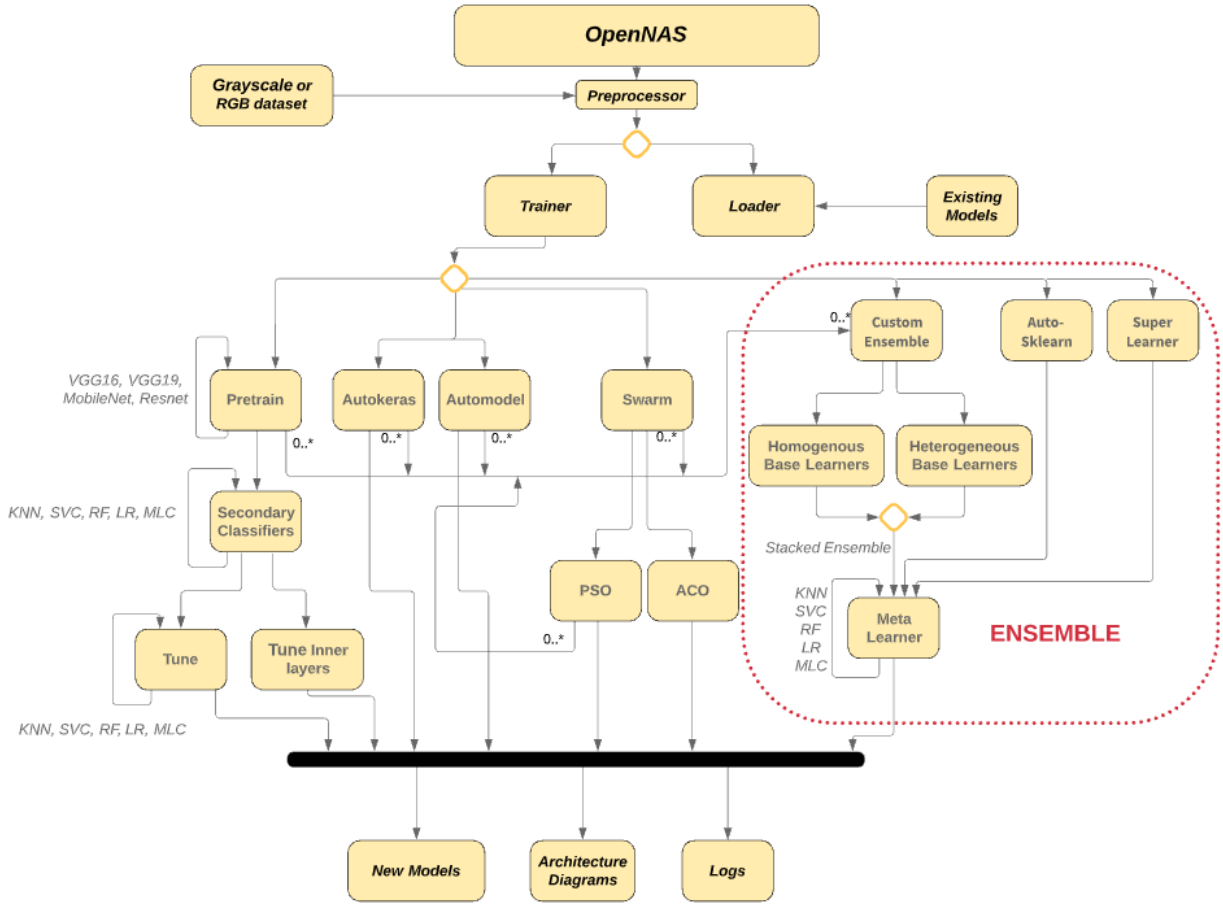

Figure 4: OpenNAS System Design

Table 1: Ensemble Mean Performance of Best Meta Learners on CIFAR-10

	RF	KNN	Best Member	Runtime (s)
Hetero-6	0.931	0.930	0.900	341
Swarm	0.925	0.921	0.900	254
PSO	0.918	0.916	0.900	198
ACO	0.895	0.889	0.848	76
Hetero-4	0.847	0.841	0.755	170
VGG19	0.818	0.822	0.755	86
VGG16	0.817	0.816	0.743	76

of two ACO derived models and two PSO derived models. A total of six models were incorporated into the Hetero-6 ensemble which included two from ACO, two from PSO, one from VGG16 and one from VGG19.

5.2.1 Ensemble Performance on CIFAR-10. The relative performance of all meta learners in classifying CIFAR-10 data is illustrated in Figure 5. Two clear groups emerge. The higher performing group of the RF and KNN classifiers stand out in comparison to the poorer performing group consisting of the MLPC, SVC and Logistic Regression algorithms. In the case of the VGG16 ensemble, there is a difference of 3.7% in accuracy achieved between using a Random Forest and an SVC approach.

The accuracies achieved by higher performing RF and KNN meta learners, across all ensemble types, are summarized in Table 1. Ensembles consisting of weaker members performed worse than ensembles with higher performing members. The Hetero-4 ensemble achieves 84.7% using Random Forest whereas the Swarm ensemble attains an accuracy of 92.5%.

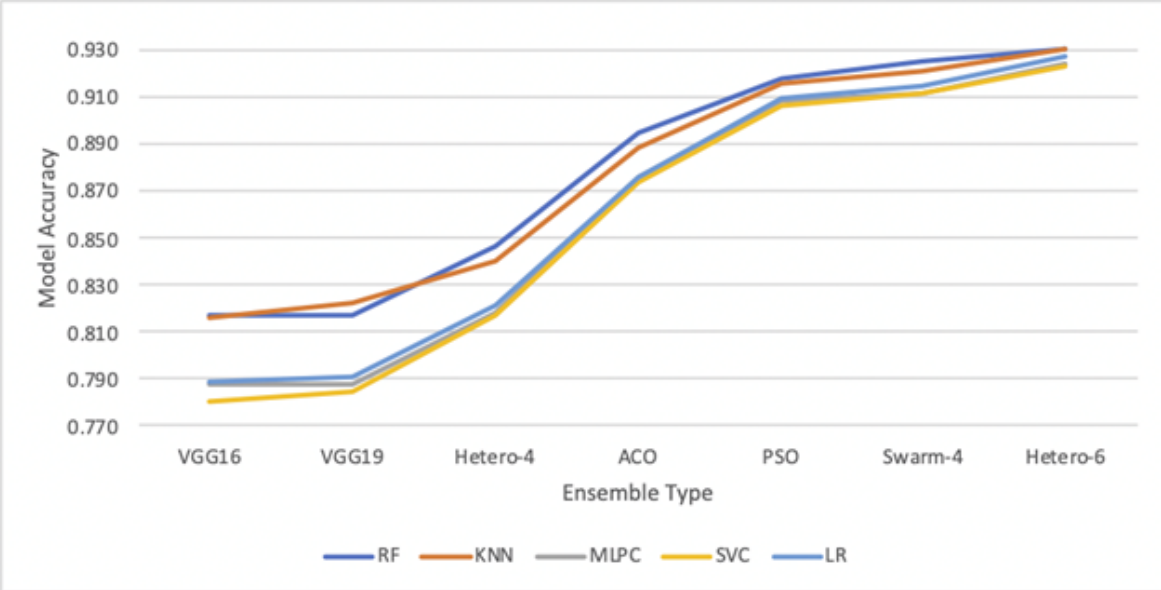

Figure 5: Ensemble Mean Performance of All Meta Learners on CIFAR-10

Table 2: Ensemble Mean Performance of Best Meta Learners on Fashion_Mnist

	RF	KNN	Best Member	Runtime (s)
Hetero-4	0.930	0.924	0.831	156
VGG16	0.922	0.913	0.807	75
VGG19	0.906	0.903	0.831	83
ACO	0.902	0.904	0.867	33
PSO	0.747	0.768	0.686	93

With this study, the impact of the number and diversity of models, on overall ensemble accuracy can be seen. Increasing the number of models within an ensemble often increases ensemble accuracy. The Hetero-6 ensemble performed significantly better (93.1%) compared with its Hetero-4 counterpart (84.7%) using a Random Forest meta learner. Heterogeneous ensembles, containing diverse models, were seen to offer better performance compared to their homogeneous counterparts.

5.2.2 Ensemble Performance on Fashion_Mnist. The relative performance of meta learners in classifying Fashion_Mnist data, using different types of ensembles, was investigated. Similar to the CIFAR-10 evaluations, there were two distinct groups of meta learners namely the higher performing set of RF and KNN compared with the weaker performance of MLPC, SVC and LR. The performance of the higher performing RF and KNN meta learners is illustrated in Table 2. For all ensemble types, Random Forest was again the strongest performer of all meta learners. Consistent with a previous observation, it can be seen that the ensemble set with the highest number of members offers the greatest performance. The Hetero-4 ensemble, with 4 members, achieves an accuracy of 93% compared with a lower accuracy of 92.2% on the VGG16 ensemble of two members.

5.3 Scikit-learn Stacking and Super Learner

For the purposes of this study, the effectiveness of scikit-learn in classifying CIFAR-10 and Fashion_Mnist data was evaluated. Scikit-learn stacking was compared with a super learner approach, which is a stacking ensemble variation incorporating cross fold validation.

5.3.1 Scikit-learn Stacking and Super Learners on CIFAR-10. On first inspection of CIFAR-10 classification in Table 3, the accuracy results for both the super learner (49%) and scikit-learn (52%) appear poor. The performance of these approaches is governed by the algorithms chosen for the base learners and meta learners. Several variations of base learner and meta algorithms were tested. Variations included increasing the number, and diversity, of base learners. A multi stacked approach, with 2 layers, was also implemented. The accuracy of all OpenNAS approaches showed little deviation and stayed within a range of 49% to 53%. Experiments conducted, as part of the original Auto-Sklearn paper indicate a baseline accuracy of 51.7% on CIFAR-10 demonstrating a consistency with the results observed as part of this study [22].

5.3.2 Scikit-learn and Super Learners on Fashion_Mnist. Accuracies obtained on Fashion_Mnist, when using either the scikit-learn or the super learner approach, are much improved when compared

Table 3: Scikit-learn Stacking and Super Learner Performance

	CIFAR-10		Fashion_Mnist	
	Acc (Mean)	**Runtime (s)**	**Acc (Mean)**	**Runtime (s)**
Super learner	0.490	5507	0.887	2144
2 layers	0.520	11910	0.877	3366
1 layer	0.524	8852	0.869	2418

with the findings for CIFAR-10. The accuracies achieved, and their associated run times, are illustrated in Table 3. In comparing approaches, the super learner approach offered better performance in both its accuracy (88.7%) and its run time of 2144 seconds. The super learner essentially has a single layer of base models whose predictions create the feature set for a meta learner. Its structure is similar to the one layer scikit-learn ensemble. However, it is not a strict like for like comparison in that the super learner had 5 base models whereas the single layer scikit-learn ensemble had just 2 base models. For Fashion_Mnist, the super learner run time was over 10% faster and achieved nearly 2% improvement in accuracy when compared to single-layer scikit-learn. This finding was reinforced by the 2-layer scikit-learn ensemble which took 50% longer to build but attained a 1% lower accuracy.

6 DISCUSSION

For CIFAR-10, a heterogeneous ensemble of six base models feeding into a Random Forest meta learner is the highest performing with an accuracy of 93.1%. Such an arrangement effectively creates an "ensemble of ensembles", whose accuracy is significantly higher (3.1%) when compared with the best performing models in previous OpenNAS studies [5]. The improvement is even greater (4.41%) when compared with approaches that rely exclusively on SI heuristics [15].

Pre-trained ensembles consisting of either MobileNet or RestNet50 models delivered the poorest performance with CIFAR-10. The other pre-trained ensembles, using VGG architectures, performed very well on the same dataset. However the accuracy of the VGG homogeneous ensembles was still 4% lower than the highest ranking ensemble, Hetero-6.

Many of the characteristics exhibited with CIFAR-10 were also seen in Fashion_Mnist classification. The lowest performing models were again the pre-train set of Resnet50 and MobileNet. VGG16 and VGG19 ensembles performed well on Fashion_Mnist.

The scikit-learn and super learner approaches performed poorly on CIFAR-10. Clearly, they are not suited to the classification of complex triple channel image datasets, which demand a convolutional neural network approach to achieve accuracies greater than 90%. Run times associated with various ensemble types are illustrated in Tables 1 - 3. The difference in run time between stacking ensembles and that of the scikit-learn or super learner approaches is very significant. In fact, with the classification of CIFAR-10, the run time of the slowest stacking ensemble (Hetero-6) is 15 times faster than the quickest of the super learner and scikit-learn approaches.

7 CONCLUSION

With OpenNAS, heterogeneous ensembles achieved the highest accuracy in classifying both CIFAR-10 and Fashion_Mnist data. The accuracy achieved with OpenNAS ensembles is competitive with the current state of the art [21]. Meta learner algorithms have a significant impact in determining stacking ensemble accuracies. The Random Forest classifier was consistently the best meta learner, irrespective of the underlying ensemble.

Super learner and scikit-learn stacking approaches are fundamentally designed for simpler neural networks using classifier algorithms from the scikit-learn suite. However, they have been shown to perform well on CNNs which classify less complex grayscale image datasets such as Fashion_Mnist.

While Keras offers a powerful framework for neural net development, the strengths of the scikit-learn library should not be overlooked. In particular, in the absence of pre-built base learners, it was shown how scikit-learn or a super learner approach can be used to quickly develop high performing ensembles for simpler datasets. However, for color datasets, a heterogeneous stacked ensemble of pre-built SI and pre-trained models, is faster and more accurate than building models from scratch using scikit-learn or a super learner.

ACKNOWLEDGMENTS

This work was supported by the ADAPT Centre, which is funded under the SFI Research Centres Programme (Grant 13/RC/2016) and is co-funded by the European Regional Development Fund.

REFERENCES

[1] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. 2019. Automated machine learning: methods, systems, challenges. Springer Nature.
[2] Lars Kotthoff, Chris Thornton, Holger H Hoos, Frank Hutter, and Kevin Leyton-Brown. 2017. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. The Journal of Machine Learning Research 18, 1 (2017), 826–830.
[3] Randal S Olson and Jason H Moore. 2019. TPOT: A Tree-Based Pipeline Optimization Tool for Automating. Automated Machine Learning: Methods, Systems, Challenges (2019), 151.
[4] Haifeng Jin, Qingquan Song, and Xia Hu. 2019. Auto-keras: An efficient neural architecture search system. In Proceedings of the 25 th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 1946–1956.
[5] Séamus Lankford and Diarmuid Grimes. 2020. "Neural Architecture Search using Particle Swarm Optimization and Ant Colony Optimization", in Proceedings of the 28th AIAI Irish Conference on Artificial Intelligence and Cognitive Science.
[6] Russell Eberhart and James Kennedy. 1995. A new optimizer using particle swarm theory. In MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. IEEE, 39–43.
[7] Marco Dorigo and Luca Maria Gambardella. 1997. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation 1, 1 (1997), 53–66.
[8] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 770–778.
[10] Hong-Yen Chen and Chung-Yen Su. 2018. An enhanced hybrid MobileNet. In 2018 9 th International Conference on Awareness Science and Technology (iCAST). IEEE, 308–312..
[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 11 (1998), 2278–2324.
[12] Beatriz A Garro and Roberto A Vázquez. 2015. Designing artificial neural networks using particle swarm optimization algorithms. Computational intelligence and neuroscience 2015 (2015).
[13] Michalis Mavrovouniotis and Shengxiang Yang. 2015. Training neural networks with ant colony optimization algorithms for pattern classification. Soft Computing 19, 6 (2015), 1511–1522.
[14] Francisco Erivaldo Fernandes Junior and Gary G Yen. 2019. Particle swarm optimization of deep neural networks architectures for image classification. Swarm and Evolutionary Computation 49 (2019), 62–74.
[15] Edvinas Byla and Wei Pang. 2019. Deepswarm: Optimising convolutional neural networks using swarm intelligence. In UK Workshop on Computational Intelligence. Springer, 119–130.
[16] Cheng Ju, Aurélien Bibaut, and Markvan der Laan. 2018. The relative performance of ensemble methods with deep convolutional neural networks for image classification. Journal of Applied Statistics 45, 15 (2018), 2800–2818.
[17] Mark J Van der Laan, Eric C Polley, and Alan E Hubbard. 2007. Super learner. Statistical applications in genetics and molecular biology 6, 1 (2007).
[18] Sebastian Flennerhag. 2017. Mlens Documentation. (2017).
[19] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2014. The cifar-10 dataset. online: http://www.cs.toronto.edu/~kriz/cifar.html (2014)
[20] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv: 1708.07747 (2017).
[21] Xin He, Kaiyong Zhao, and Xiaowen Chu. 2020. AutoML: A Survey of the State-of-the-Art. Knowledge-Based Systems (2020), 106622.
[22] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2019. Auto-sklearn: efficient and robust automated machine learning. In Automated Machine Learning. Springer, Cham, 113–134