Hyperparameter Importance for Image Classification by Residual Neural Networks

Abhinav Sharma[1(✉)], Jan N. van Rijn[1,2(✉)], Frank Hutter[3], and Andreas Müller[1]

[1] Columbia University, New York City, USA
{as5414,acm2248}@columbia.edu
[2] Leiden University, Leiden, The Netherlands
j.n.van.rijn@liacs.leidenuniv.nl
[3] Albert-Ludwigs-Universität Freiburg, Freiburg, Germany
fh@cs.uni-freiburg.de

Abstract. Residual neural networks (ResNets) are among the state-of-the-art for image classification tasks. With the advent of automated machine learning (AutoML), automated hyperparameter optimization methods are by now routinely used for tuning various network types. However, in the thriving field of deep neural networks, this progress is not yet matched by equal progress on rigorous techniques that yield information beyond performance-optimizing hyperparameter settings. In this work, we aim to answer the following question: Given a residual neural network architecture, what are generally (across datasets) its most important hyperparameters? In order to answer this question, we assembled a benchmark suite containing 10 image classification datasets. For each of these datasets, we analyze which of the hyperparameters were most influential using the functional ANOVA framework. This experiment both confirmed expected patterns, and revealed new insights. With these experimental results, we aim to form a more rigorous basis for experimentation that leads to better insight towards what hyperparameters are important to make neural networks perform well.

Keywords: Hyperparameter importance · Residual neural networks

1 Introduction

Residual neural networks [10] are among the state-of-the-art for image classification tasks. Given sufficient data and proper hyperparameter settings, residual neural networks can achieve remarkable results, but their performance (and that of other neural networks) highly depends on their hyperparameter settings. As a consequence, there has been a lot of recent work and progress on hyperparameter optimization, with methods including Bayesian optimization [1,29], meta-learning [4] and bandit-based methods [18]; see [8] for a review.

Despite impressive results both on common benchmarks and various application domains, the experiments in many academic machine learning papers are

P. Kralj Novak et al. (Eds.): DS 2019, LNAI 11828, pp. 112–126, 2019.
https://doi.org/10.1007/978-3-030-33778-0_10

designed to answer *which* particular method works better, typically by introducing a new algorithm and demonstrating success over a limited set of baselines or benchmarks [31]. In a recent paper, Sculley et al. (2018) identify this as a problem: 'Empirical studies have become challenges to be won, rather than a process for developing insight and understanding' [27]. Additionally, many advances in deep learning have been evaluated on a small number of datasets. It has long been recognized that small-scale studies can create a false sense of progress [9]. Recht et al. (2018) speculate that by overly using the same test set, reported results tend to overfit and demonstrate that performance results of many introduced models does not generalize to other (newly assembled) test sets [24].

In this work, we aim to provide a more rigorous approach to the following question: Given a residual neural network architecture, what are generally (across datasets) its most important hyperparameters? In order to answer this question, we assembled an image classification benchmark suite consisting of 10 popular datasets from the literature. On each of these datasets we obtained performance results with varying hyperparameter settings. Although the aim of this paper is not to improve predictive performance, we compare the results with state-of-the-art results reported by other researchers, to ensure that the results are credible and applicable. We see this as a first step towards creating more rigorous insights about the conditions under which residual neural networks perform well and which hyperparameters influence this.

Our contributions are the following: (i) We assembled a benchmark suite of 10 well-known image classification datasets, allowing researchers to draw conclusions across datasets. We made all code, data and results publicly available;[1] (ii) we apply functional ANOVA [30] on performance results of residual neural networks, to identify the importance of the various hyperparameters to predictive accuracy; and (iii) we verified expected behaviour regarding hyperparameter interactions, and gained new insights regarding typical marginal curves and hyperparameter interactions. Most notable is the observation that for the concerning datasets the marginals of important hyperparameters exhibit very similar landscapes. Overall, this work is the first to provide large-scale quantitative evidence for which hyperparameters of residual neural networks are important, providing a better scientific basis for the field than previous knowledge based on small-scale studies and intuition.

2 Related Work

In this section we review related work on residual neural networks, hyperparameter importance and landscape analysis.

Residual Neural Networks. Deep residual neural networks were introduced in [10] and have set the benchmark for image recognition tasks in recent years. They provide good predictive accuracy while maintaining an affordable model size. Their defining characteristic is the use of *residual learning*, in which deeper layers of the network are linked to shallower layers directly using 'shortcut connections' skipping several layers in between. These shortcut connections perform

[1] https://www.github.com/janvanrijn/openml-pimp

an identity mapping which ensures the convergence of the deep network is at least as good as its shallower counterpart and hence limit divergence during training. In this way, the residual learning framework eases the training of networks that are substantially deeper. Furthermore, empirical evidence suggests that these residual neural networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset the residual nets were evaluated with a depth of up to 152 layers – 8 times deeper than VGG nets [28] but still having lower complexity. Furthermore, residual learning can be used on networks of varying depth to fit the task at hand. Smaller residual neural networks, like 'ResNet18' (as the name suggests, consisting of 18 layers), provide great performance while being very efficient in terms of size and speed [10].

Hyperparameter Importance. When using a new algorithm on a given dataset, it is typically a priori unknown which hyperparameters should be tuned, what are the good ranges for these hyperparameters to sample from, and which values in these ranges are most likely to yield high performance. Various techniques exist that allow for the assessment of hyperparameter importance. These techniques generally consider either local importance (dependent on a specific setting for other hyperparameters) or global importance (independent of specific hyperparameter settings).

Forward selection [12] is based on the assumption that important attributes in a dataset have high impact on the performance of classifiers trained on it. It trains a model which predicts the performance of a configuration based on a subset of hyperparameters. This set is initialized empty and greedily filled with the next most important hyperparameter. *Ablation analysis* [2] requires a default setting and an optimized setting and calculates a so-called ablation trace, which embodies how much the hyperparameters contributed towards the difference in performance between the two settings. *Local Parameter Importance* [3] studies the performance changes of a configuration along each parameter using an *empirical performance model* (sometimes also called a 'surrogate' model). *Functional ANOVA* [30] is a global hyperparameter importance framework that can detect the importance of both individual hyperparameters and interaction effects between arbitrary subsets of hyperparameters. It is the key technique upon which this research is built.

Functional ANOVA depends on the concept of the *marginal* of a hyperparameter, i.e., how a given value for a hyperparameter performs, averaged over all possible combinations of the other hyperparameters' values. While there are an exponential number of combinations, the authors of [13] showed how this can be calculated efficiently using tree-based surrogate models.

All the aforementioned techniques are post-hoc techniques, i.e., when confronted with a new dataset, these do not reveal what hyperparameters are important prior to experimenting on that particular dataset. Contrary, various researchers argued that it is more useful to generalize the notion of hyperparameter importance across datasets [22,25,26]. In particular, it has been shown how to apply functional ANOVA across datasets for a given algorithm [25,26]. These works build upon the assumption that if this hyperparameter importance quantification method is applied on a large enough set of datasets, we can draw

conclusions regarding which hyperparameters are generally important. However, neither of these studies applied this methodology to convolutional neural networks. To the best of our knowledge, this is the first work that addresses hyperparameter importance for residual neural networks.

Landscape Analysis. The interaction between configurations and the respective results can be seen as an high-dimensional landscape, which in turn can be analyzed for mathematical properties [23]. Although this particular study is executed on 'satisfiability', 'mixed integer programming' and 'traveling salesman problems' benchmarks, it shows evidence that configuration landscapes are often uni-model and even convex.

3 Background and Methods

We follow the notation that was introduced in [13]. We assume that a given residual neural network model has n hyperparameters with domains $\Theta_1, \ldots, \Theta_n$ and *configuration space* $\boldsymbol{\Theta} = \Theta_1 \times \ldots \times \Theta_n$. Let $N = \{1, \ldots, n\}$ be the set of all hyperparameters of the classifier. An instantiation (or configuration) of the classifier is a vector $\boldsymbol{\theta} = \langle \theta_1, \ldots, \theta_n \rangle$ with $\theta_i \in \Theta_i$. A partial instantiation is a vector $\boldsymbol{\theta}_U = \langle \theta_i, \ldots, \theta_j \rangle$ with a subset $U \subseteq N$ of the hyperparameters fixed, and the values for other hyperparameters unspecified. The *marginal* $\hat{a}^p_U(\boldsymbol{\theta}_U)$ is defined as the average performance on measure p of all complete instantiations $\boldsymbol{\theta}$ that agree with $\boldsymbol{\theta}_U$ in the instantiations of hyperparameters U. The variance of $\hat{a}^p_U(\boldsymbol{\theta}_U)$ is denoted as $\mathbb{V}^p_U$. Intuitively, if the marginal $\hat{a}^p_U(\boldsymbol{\theta}_U)$ has a high variance, this means that hyperparameter was of high importance to performance measure p, and vice versa. For a more complete description, the reader is referred to [13].

In this research, we address the following problem. Given (i) a residual neural network architecture with configuration space $\boldsymbol{\Theta}$, (ii) a set of datasets $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(M)}$, with M being the number of datasets, and (iii) for each of the datasets, a set of empirical performance measurements $\langle \boldsymbol{\theta}_i, Y_i \rangle_{i=1}^K$ for different hyperparameter settings $\boldsymbol{\theta}_i \in \boldsymbol{\Theta}$, where Y_i is a tuple of all relevant performance measures (in this case, predictive accuracy), we aim to determine which hyperparameters affect the algorithm's empirical performance most, and which values are likely to yield good performance.

For a given dataset, we use the performance data $\langle \boldsymbol{\theta}_i, Y_i \rangle_{i=1}^K$ collected on this dataset to fit an internal tree-based surrogate model, in this case, a random forest with 16 trees. Functional ANOVA then uses this surrogate model to calculate the variance contribution $\mathbb{V}^p_j / \mathbb{V}^p$ of every hyperparameter $j \in N$, with high values indicating high importance. We then study the distribution of these variance contributions across datasets to obtain empirical data regarding which hyperparameters tend to be most important.

It is possible that a hyperparameter is responsible for a high variance on many datasets, but that its best value is the same across all of them. We note that functional ANOVA will flag such hyperparameters as important, although it could be argued that they have appropriate defaults and do not need to be

tuned [22,26]. For example, it is reasonable to expect that for any type of neural network the marginal of the number of epochs has a high variance, where obviously better performances are achieved for higher values (at the cost of additional run-time). For this reason, it is always important to consider the individual marginals, as well as the generalizations across datasets.

4 Experimental Setup

Section 4.1 describes the training procedure of the residual neural network, Sect. 4.2 the configuration space from which we sampled the various configurations, and Sect. 4.3 the datasets that we included in this study.

4.1 Models

In this work, we focus on the fixed architecture of 'ResNet18'. This model gives good predictive accuracy for datasets while being small in size which allows for faster training [10]. As the datasets in this research all contain images, with relatively similar dimensions, we could use the same architecture for all of them (see Sect. 4.3). The optimizer is fixed to Stochastic Gradient Descent (SGD), parameterized by momentum and weight decay. The training starts with an initial learning rate. Thereafter an adaptive learning rate scheduler is used which decays the learning rate by a factor (hyperparameter: learning rate decay) when the test accuracy plateaus for a given number (hyperparameter: patience) of epochs. The details of the hyperparameter space are described in Sect. 4.2.

We record the time taken (in seconds) and the accuracy on the test set after every epoch. The goal is not to identify an optimum parameter setting, as using a test-set simply computing the maximum would result in overly optimistic evaluation. If one would be interested in using the results for hyperparameter optimization a proper nested cross-validation procedure should be applied [5]. We performed all runs on single NVIDIA P100 GPU.

4.2 Configuration Space

We selected twelve hyperparameters. This selection was made based on visual inspection of the modules in the 'Torch' package, as well as personal experience. Even though it feels natural to fix the values for some hyperparameters to seemingly good values, in this work we aim to verify the applicability of these values.

Our hyperparameter space contains six hyperparameters for the SGD optimizer (number of epochs, initial learning rate, learning rate decay, momentum, batch size, and whether to shuffle the data), two early stopping hyperparameters (tolerance and patience), and four regularization hyperparameters (weight decay, and data augmentation by resize crop, horizontal flips, and vertical flips). Note that since we keep a fixed network structure, we do not modify any architectural hyperparameters (this would be very interesting, but is out of scope for the current study).

Table 1. Overview of the hyperparameters used in this research.

Hyperparameter	Range	Description
Batch size	$2^{\{3,4,5,6,7,8,9\}}$	Number of samples in one batch of gradient descent used during training
Epochs	[1–200]	The number of times each training observation is passed to the network
Horizontal flip	Boolean	Whether to apply data augmentation by flipping the image horizontally
Vertical flip	Boolean	Whether to apply data augmentation by flipping the image vertically
Learning rate	$[10^{-6}$–1] (log)	The learning rate with which the network starts training
Learning rate decay	[2–1000] (log)	Factor to reduce the learning rate with, if no improvement is obtained after several epochs
Momentum	[0–1]	Value of momentum multiplier used during gradient descent
Patience	[2–200]	Number of epochs without improvements that are being tolerated before learning rate is reduced
Shuffle	Boolean	Whether to shuffle the train set before an epoch
Resize crop	Boolean	Whether to apply data augmentation by resizing and then cropping the image
Tolerance	$[10^{-5}$–$10^{-2}]$ (log)	Tolerance for early stopping criterion
Weight decay	$[10^{-6}$–$10^{-2}]$ (log)	L2 loss on the weights

We note that some of the hyperparameters we tune are sometimes rather chosen manually on a per-dataset basis based on domain knowledge (e.g., certain data augmentations don't make sense for some types of images, and the batch size is often set to the maximum feasible given the GPU's memory). We still included these in our study to study how large their impact is on performance.

Table 1 lists all the hyperparameters and their maximal ranges we considered. In order to obtain reasonable performance for datasets of different input sizes, we had to use slightly different hyperparameter spaces across datasets; in particular, the datasets with large input size (Fruits 360, Flower, STL-10 and Dog vs. Cat) would have led to memory issues with batch sizes of 256 or 512, and we therefore only considered batch size values of $2^{\{3,4,5,6,7\}}$ in those cases and of $2^{\{5,6,7,8,9\}}$ in all other cases. Also, owing to limited computational resources, some other manual modifications were made to speed up experiments and focus on a region of good hyperparameters: for datasets MNIST, Fashion MNIST and Fruits, the maximum number of epochs was set to 50, for datasets Dog vs. Cat the maximum number of epochs was set to 100, and for datasets Fruits 360, Flower, Dog vs. Cat, MNIST and Fashion MNIST, the maximum learning rate was set to 0.1.

For each dataset, we sampled $K = 200$ configurations uniformly from this configuration space, with the maximum number of epochs. For each run we stored performance results after every epoch, allowing functional ANOVA to model the marginal of this hyperparameter more accurately.

Table 2. Overview of the datasets used in this research.

Name	Description	Dimensions	Class	Train	Test	Ref
MNIST	Handwritten digits	28×28	10	60,000	10,000	[17]
Fashion MNIST	Gray-scale objects	28×28	10	60,000	10,000	[34]
CIFAR-10	Colored objects	$32 \times 32 \times 3$	10	50,000	10,000	[16]
CIFAR-100	Colored objects	$32 \times 32 \times 3$	100	50,000	10,000	[16]
STL-10	Colored objects	$96 \times 96 \times 3$	10	5,000	8,000	[6]
HAM10000	Skin cancer images	$28 \times 28 \times 3$	7	9,013	1,002	[32]
SVHN	House number images	$32 \times 32 \times 3$	10	73,257	26,032	[21]
Flower	Flower images	$96 \times 96 \times 3$	5	3,888	435	[19]
Fruits 360	Fruit images	$96 \times 96 \times 3$	82	41,814	14,041	[20]
Dog vs. Cats	Dog and cat images	$96 \times 96 \times 3$	2	22,500	2,500	[15]

When computing hyperparameter importances across different datasets, the question arises how to treat differing hyperparameter spaces. For the important learning rate hyperparameter we felt it to be important to use identical ranges everywhere and therefore used a reduced range of $[10^{-6}, 0.1]$. However, for the batch size hyperparameter, no single range makes sense for all datasets, and we therefore simply computed hyperparameter importance separately based on the range used for the dataset at hand. Likewise, for the maximum number of epochs, we used 200 throughout; this is justified by the assumption that the internal random forest model would correctly model the plateaued performance.

4.3 Datasets

This section reviews the datasets that were used in this research. We assembled a diverse set of image classification datasets, including often used datasets (e.g., MNIST and CIFAR-100). We excluded the common benchmark ImageNet to keep the computational costs reasonable.

All our datasets, listed in Table 2, are classification tasks (the respective task is briefly described in column 'Description'). For example, in the MNIST dataset the task is to classify hand-written digits, whereas for the CIFAR-10 dataset the task is to identify images into 10 classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship or truck). Column 'Dimensions' represents the size and number of channels of the training images. Black and white or gray-scale datasets have two dimensions (width and height), whereas colored datasets have three dimensions (width, height and number of color channels; in this study the number of color channels is always 3). Column 'Class.' represents the number of classes, column 'Train' the number of train observations and column 'Test' the number of test observations. Finally, column 'Ref.' contains a reference to the publication where the dataset was introduced.

As the dimensions of these datasets are all approximately the same, we could use the same architecture for all of them. There are minor modifications to the

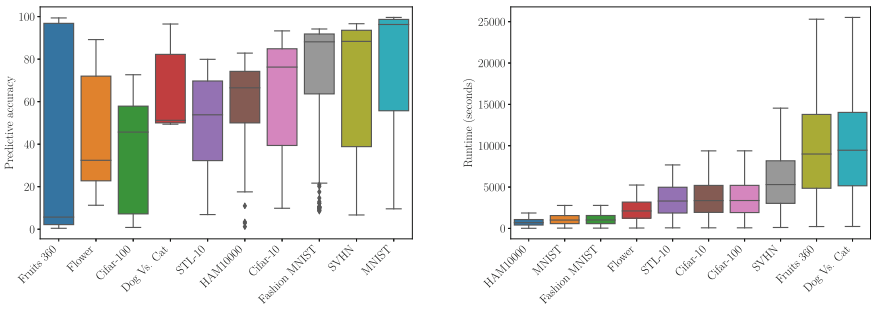


Fig. 1. Performance results of the various configurations per dataset, sorted by median performance. Complementary, Table 3 shows the best obtained result per dataset.

input and output layers due to different input dimensions and output classes of each dataset. For datasets with gray-scale images (i.e., MNIST and Fashion MNIST) the pixel values are duplicated over three dimensions during preprocessing. Whether data augmentation techniques like random crops and random flips were performed is controlled by the respective hyperparameter.

5 Results

In this section we analyze the results of the experiments. In Sect. 5.1 we discuss some basic performance characteristics compared to state-of-the-art algorithms. In Sects. 5.2 and 5.3 we discuss the main contribution of this work, the importance of hyperparameters according to functional ANOVA. Finally, Sect. 5.4 discusses limitations that could inspire future work.

5.1 Performance Results

We explore some basic characteristics about the performance results obtained on the datasets. As mentioned before, obtaining state-of-the-art performance is neither the aim nor the contribution of this paper, but in order for the results to be credible and applicable, it is important to verify that the results are in the same ballpark as good results reported in literature. Figure 1 shows the predictive accuracy (left) and run-time (right) of all hyperparameter configurations $\boldsymbol{\theta}$ grouped per dataset in a box-plot. Both plots include only the measured performance (run-time or accuracy) after the final epoch, and thus not the recorded intermediate results.

We made a best effort to find established state-of-the-art results for existing datasets. Table 3 compares the best results of the conducted experiments with the best found result in literature. We obtained the results of state-of-the-art methods from public sources on the internet[2,3]. For some lesser known datasets,

[2] https://benchmarks.ai/

[3] https://github.com/RedditSota/state-of-the-art-result-for-machine-learning-problems

there was no established state-of-the-art. In these cases, we did not report any state-of-the-art result, as that might be misleading. Column 'ResNet' denotes the best obtained performance (optimistic, as was argued in Sect. 4.1) of the residual neural network through random search. Column 'SOTA' denotes the (also optimistic) performance of the state-of-the-art network.

The comparison between residual neural network results and state-of-the-art results contains various conditions that need to be accounted for. As a consequence, this comparison is somewhat biased. However, it serves the purpose of providing context to the obtained results. In some cases the best results obtained by the trained residual neural networks are close to the best reported results in literature (e.g., for MNIST and Fashion MNIST), while for others the differences are bigger (e.g., CIFAR-100 and STL-10). Overall, the performance results are good enough to expect that some conclusions drawn may carry over to state-of-the-art models.

Table 3. Comparison between the best results obtained by the residual neural networks in this study and state-of-the-art results.

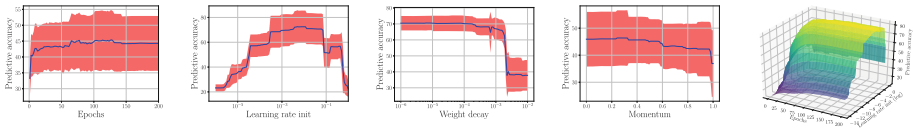
Dataset	ResNet	SOTA	Source
MNIST	99.62	99.79	[33]
Fashion MNIST	94.18	96.35	[35]
CIFAR-10	93.29	99.00	[11]
CIFAR-100	72.66	91.30	[11]
STL-10	79.91	88.80	[14]
HAM10000	82.83	-	
SVHN	96.66	98.98	[7]
Flower	89.20	-	
Fruits 360	99.38	-	
Dog vs. Cats	96.52	-	

5.2 Marginals per Dataset

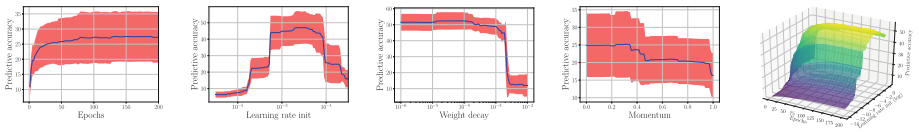
This section details the results of the hyperparameter importance experiment. Figure 2 shows the predictive accuracy marginals of important (combinations) of hyperparameters, per dataset.

For each of the 10 datasets, we plotted the marginal of several important hyperparameters and one pair of hyperparameters. From left to right this image displays the marginals of the 'number of epochs', the 'initial learning rate', 'weight decay', 'momentum' and the combination of 'number of epochs and initial learning rate'. Note that we calculated the marginals for all 12 hyperparameters and all 65 hyperparameter pairs, but only show this subset due to space reasons. The x-axis shows the value of the hyperparameter and the y-axis shows the marginal performance (predictive accuracy). The epochs marginals are most detailed, presumably because of the recorded performance after every epoch.

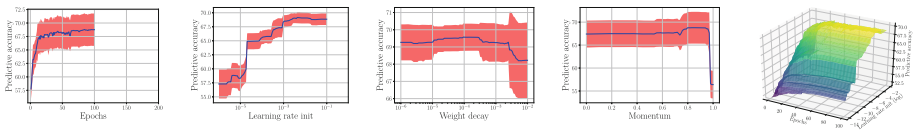
The marginals reveal several patterns about the interaction between hyperparameter values and final performance of the network. Firstly, as the number of epochs increases, so does the performance of the network. Although this is quite obvious, it is important to verify that the proposed methodology can discover known and expected behaviors. Secondly, the marginals for the initial learning rate reveal that there is no perfect default across datasets, but that values between 10^{-3} and 10^{-2} generally perform quite well, whereas setting it much



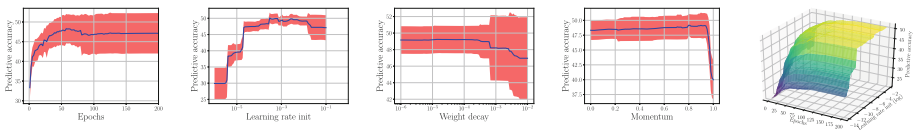
(a) CIFAR-10



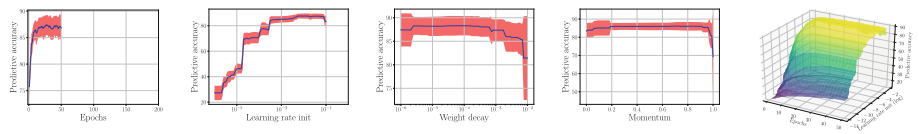
(b) CIFAR-100



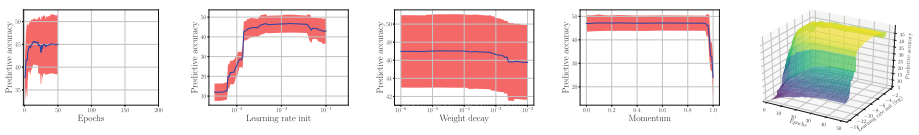
(c) Dog vs. Cat



(d) Flower

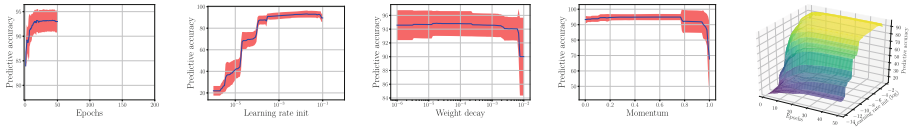


(e) Fashion MNIST

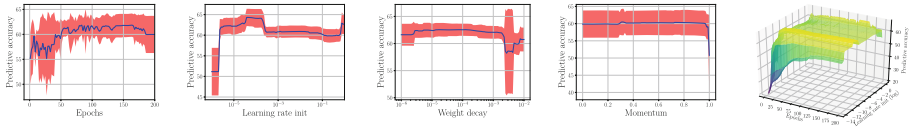


(f) Fruits 360

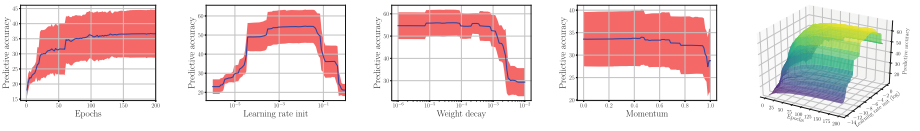
Fig. 2. Marginal plots for predictive accuracy on the test set per dataset. The first four columns show the marginal of a single hyperparameter. The blue line represents the marginal, the red area represents the standard deviation. The fifth column shows the combined marginal of two hyperparameters (note that the ranges per dataset differ). (Color figure online)



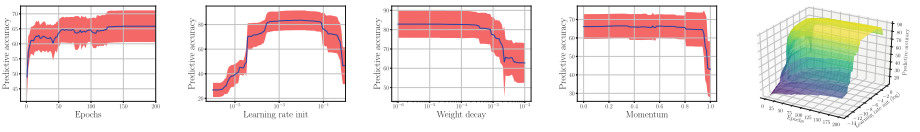
(g) MNIST



(h) HAM10000



(i) STL-10



(j) SVHN

Fig. 2. *(continued)*

lower or higher typically results in suboptimal performance. For weight decay and momentum we see similar trends, however there seems to be a tendency that setting their values too low is less harmful than setting them too high. Thirdly, also the combined marginal of number of epochs and initial learning rate is interesting, as it reveals that there is very little interplay between these two hyperparameters. Interestingly, both hyperparameters are important, but setting one hyperparameter to a specific value does not have a large influence on the optimal value for the other (maximum variance contribution: 0.026 on Cifar-10).

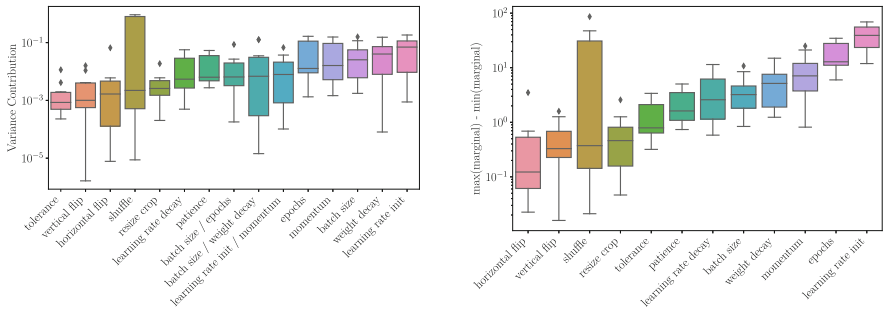
Most interestingly, we observe that for each hyperparameter the marginals follow similar trends across the datasets. Although the marginals exhibit a rough and edgy pattern, based on visual inspection we conclude that after some smoothing the landscapes would be uni-modal and convex. Even though the methodology and application domain are slightly different, these results seem to be in line with earlier reported findings [23].

5.3 Importance Across Datasets

Figure 3(a) shows box-plots for the variance per hyperparameter, presented similarly to [26]. For each partial configuration $\boldsymbol{\theta}_U$ with $U = 1$ and the three most important partial configurations with $U = 2$, we record variance of the marginal $\mathbb{V}_U^p$ per dataset, and present these across datasets in box-plots. We observe various expected results. Hyperparameters related to the optimizer seem generally most important, i.e., 'weight decay', 'momentum', and 'learning rate init'. The data augmentation hyperparameters are among the least important hyperparameters. We note that functional ANOVA is meant as a tool for assessing global hyperparameter importance; data augmentation techniques are generally used to be applied on already good performing models, in order to further improve the performance. As such, the utility of data augmentation techniques might not be detected by functional ANOVA but can most likely be measured with local hyperparameter importance tools, such as Ablation Analysis [2].

Furthermore, we observe that the number of epochs, a hyperparameter which we expected to be important, ranks only 5th when analyzing the marginals. We note that the variance of the marginal (upon which functional ANOVA is built) is highly dependent on the selected ranges. To alleviate this problem, Fig. 3(b) shows the results in an alternative way. For each partial configuration $\boldsymbol{\theta}_U$ with $U = 1$, we record the maximum of marginal (i.e., $max(\hat{a}_U^p(\boldsymbol{\theta}_U))$) and the minimum of the marginal (i.e., $min(\hat{a}_U^p(\boldsymbol{\theta}_U))$) per dataset, and present the difference between these across datasets in box-plots. We observe that this plot confirms the importance of the epochs hyperparameter, making it the second most important hyperparameter after the learning rate. Also note that the other hyperparameters with the highest median variances according to Fig. 3(a) are still ranked as important in Fig. 3(b), albeit in a slightly changed order.

Finally, based on Fig. 3(a) we note that most of the variance can be explained by the effect of single hyperparameters. Apart from the variance contribution of single hyperparameters ($U = 1$) it shows the 3 most important combinations of hyperparameters ($U = 2$). The variance contribution of combined



(a) Functional ANOVA (b) Marginal max-min

Fig. 3. Importance per (combination of) hyperparameters across datasets

hyperparameters seems rather low, as none are ranked highly compared to the variance contribution of single hyperparameters. However, like the data augmentation techniques, we speculate that even though the combined effect is relatively small, it will still be important to consider when optimizing for performance.

5.4 Limitations

Looking at the results in Fig. 3, the following result stands out. The shuffle hyperparameter value has a rather low median but a very high tail. This indicates that for most datasets the marginal is not particularly affected by this hyperparameter, however for some datasets (i.e., Flower, Fruits 360, Dog vs. Cat and HAM10000) it seems extremely important.

Furthermore, Fig. 1 reveals that the median performance is quite low, despite the decent maximal performance. This is confirmed by the marginals in Fig. 2. For example, none of the marginals for CIFAR-10 exceed the 80% accuracy threshold, whereas the best found configuration obtained an accuracy of 93.29% (according to Table 3). This gives rise to the question whether a hyperparameter tool like functional ANOVA can still reveal hyperparameters that are important for fine-tuning models (such as data augmentation), or whether only global trends are detected.

Finally, functional ANOVA highly relies on a proper configuration space. A seemingly important hyperparameter like 'number of epochs' will account for a relative low variance if the range is selected in such a way that it exceeds the values for which the performance reaches the plateau. It is currently an open question how to construct the configuration space to avoid this problem.

6 Conclusions

This work was motivated by the call for more rigor in hyperparameter optimization and neural network research [24,27]. We assembled a benchmark suite with corresponding performance results of residual neural networks, and made it publicly available. Our hyperparameter importance experiment confirmed existing beliefs about which hyperparameters are most influential across datasets, i.e., the initial learning rate and the number of epochs. Other important hyperparameters are the weight decay and momentum. Most of the other hyperparameters did not have a large variance of the marginal, however we note that in many image classification benchmarks the devil is in the detail. In order to go from a reasonable performance to state-of-the-art performance, also hyperparameters with a small effect should be set to adequate values.

We confirmed some well expected patterns, for example the form of the marginals for the number of epochs and the volatility across datasets of the marginals for the initial learning rate.

We acknowledge that this is only a first step towards more rigorous results in neural network research. While this research focused specifically on residual neural networks with a fixed architecture, future work should focus on other network types and also important parameters in architecture search.

References

1. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems, vol. 24, pp. 2546–2554. Curran Associates, Inc. (2011)
2. Biedenkapp, A., Lindauer, M., Eggensperger, K., Fawcett, C., Hoos, H.H., Hutter, F.: Efficient parameter importance analysis via ablation with surrogates. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, pp. 773–779. AAAI Press (2017)
3. Biedenkapp, A., Marben, J., Lindauer, M., Hutter, F.: CAVE: configuration assessment, visualization and evaluation. In: Battiti, R., Brunato, M., Kotsireas, I., Pardalos, P.M. (eds.) LION 12 2018. LNCS, vol. 11353, pp. 115–130. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05348-2_10
4. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: Metalearning. Applications to Data Mining, 1st edn. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-73263-1
5. Cawley, G.C., Talbot, N.L.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**, 2079–2107 (2010)
6. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, vol. 15, pp. 215–223. PMLR (2011)
7. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: AutoAugment: learning augmentation strategies from data. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
8. Feurer, M., Hutter, F.: Hyperparameter optimization. In: Hutter, F., Kotthoff, L., Vanschoren, J. (eds.) Automated Machine Learning: Methods, Systems, Challenges. TSSCML, pp. 3–33. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05318-5_1
9. Hand, D.J.: Classifier technology and the illusion of progress. Stat. Sci. **21**(1), 1–14 (2006)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016)
11. Huang, Y., et al.: GPipe: efficient training of giant neural networks using pipeline parallelism. arXiv preprint arXiv:1811.06965 (2018)
12. Hutter, F., Hoos, H.H., Leyton-Brown, K.: Identifying key algorithm parameters and instance features using forward selection. In: Nicosia, G., Pardalos, P. (eds.) LION 2013. LNCS, vol. 7997, pp. 364–381. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_40
13. Hutter, F., Hoos, H., Leyton-Brown, K.: An efficient approach for assessing hyperparameter importance. In: Proceedings of the 31st International Conference on Machine Learning, vol. 32, pp. 754–762. PMLR (2014)
14. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. arXiv preprint arXiv:1807.06653 (2018)
15. Kaggle: Dogs vs. Cats Redux: Kernels Edition (2016). https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition. Accessed December 2018
16. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, University of Toronto (2009)

17. LeCun, Y.: The MNIST database of handwritten digits (1998). http://yann.lecun.com/exdb/mnist/. Accessed December 2018
18. Li, L., Jamieson, K.G., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: bandit-based configuration evaluation for hyperparameter optimization. In: 5th International Conference on Learning Representations, ICLR 2017. OpenReview.net (2017)
19. Mamaev, A.: Flowers Recognition (version 2). https://www.kaggle.com/alxmamaev/flowers-recognition. Accessed December 2018
20. Mureşan, H., Oltean, M.: Fruit recognition from images using deep learning. Acta Universitatis Sapientiae, Informatica **10**(1), 26–42 (2018)
21. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning (2011)
22. Probst, P., Boulesteix, A.L., Bischl, B.: Tunability: importance of hyperparameters of machine learning algorithms. J. Mach. Learn. Res. **20**(53), 1–32 (2019)
23. Pushak, Y., Hoos, H.: Algorithm configuration landscapes: more benign than expected? In: Auger, A., Fonseca, C.M., Lourenço, N., Machado, P., Paquete, L., Whitley, D. (eds.) PPSN 2018. LNCS, vol. 11102, pp. 271–283. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99259-4_22
24. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do CIFAR-10 Classifiers Generalize to CIFAR-10? arXiv preprint arXiv:1806.00451 (2018)
25. van Rijn, J.N., Hutter, F.: An empirical study of hyperparameter importance across datasets. In: AutoML@ PKDD/ECML, pp. 91–98 (2017)
26. van Rijn, J.N., Hutter, F.: Hyperparameter importance across datasets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2367–2376. ACM (2018)
27. Sculley, D., Snoek, J., Wiltschko, A., Rahimi, A.: Winner's curse? on pace, progress, and empirical rigor. In: Proceedings of ICLR 2018 (2018)
28. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
29. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: Advances in Neural Information Processing Systems, vol. 25, pp. 2951–2959. ACM (2012)
30. Sobol, I.M.: Sensitivity estimates for nonlinear mathematical models. Math. Model. Comput. Exp. **1**(4), 407–414 (1993)
31. Strang, B., Putten, P., Rijn, J.N., Hutter, F.: Don't rule out simple models prematurely: a large scale benchmark comparing linear and non-linear classifiers in OpenML. In: Duivesteijn, W., Siebes, A., Ukkonen, A. (eds.) IDA 2018. LNCS, vol. 11191, pp. 303–315. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01768-2_25
32. Tschandl, P., Rosendahl, C., Kittler, H.: The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci. Data (2018)
33. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., Fergus, R.: Regularization of neural networks using DropConnect. In: International Conference on Machine Learning, pp. 1058–1066 (2013)
34. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
35. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. arXiv preprint arXiv:1708.04896 (2017)