REGULARIZATION COCKTAILS

Anonymous authors
Paper under double-blind review

ABSTRACT

The regularization of prediction models is arguably the most crucial ingredient that allows Machine Learning solutions to generalize well on unseen data. Several types of regularization are popular in the Deep Learning community (e.g., weight decay, drop-out, early stopping, etc.), but so far these are selected on an adhoc basis, and there is no systematic study as to how different regularizers should be combined into the best "cocktail". In this paper, we fill this gap, by considering cocktails of 13 different regularization methods and framing the question of how to best combine them as a standard hyperparameter optimization problem. We perform a large-scale empirical study on 40 datasets, concluding that, firstly, regularization cocktails substantially outperform individual regularization methods, even if the hyperparameters of the latter are carefully tuned; secondly, the optimal regularization cocktail depends on the dataset; and thirdly, regularization cocktails yield a higher gain on small datasets.

1 INTRODUCTION

In most supervised learning application domains, the available data for training predictive models is both limited and noisy with respect to the target variable. Therefore, it is paramount to regularize machine learning models for generalizing the predictive performance on future unseen data. The concept of regularization is well-studied and constitutes one of the pillar components of machine learning. Throughout this work we use the term "regularization" for all methods that explicitly or implicitly take measures that reduce the overfitting phenomenon; we categorize these non-exhaustively into weight decay, data augmentation, model averaging, structure and linearization, and implicit regularization families (detailed in Section 2). In this paper, we propose a new principled strategy that highlights the need for automatically learning the optimal combination of regularizers, denoted as *regularization cocktails*, via a hyperparameter optimization procedure.

Truth be told, combining regularization methods is of course far from being a novel practice per se. As a matter of fact, most modern deep learning models use combinations of a number of regularizers. For instance, EfficientNet (Tan & Q.Le, 2019) mixes components of structural regularization and linearization via ResNet-style skip connections (He et al., 2016), learning rate scheduling, Drop-Out ensembling (Srivastava et al., 2014) and AutoAugment data augmentation (Cubuk et al., 2019). However, even though each of those regularizers is motivated in isolation, the reasoning behind a specific combination of regularizers is largely based on accuracy-driven manual trial-and-error iterations, mostly on image classification benchmarks like CIFAR (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009).

Unfortunately, the manual search for combinations of regularizers is sub-optimal, unsustainable, and in essence consists of an example of manual hyperparameter tuning, which in turn is easily outperformed by automated algorithms (Snoek et al., 2012; Thornton et al., 2013; Feurer et al., 2015; Olson & Moore, 2016; Jin et al., 2019; Erickson et al., 2020; Zimmer et al., 2020a). Following the spirit of AutoML (Hutter et al., 2018), we therefore propose a strategy for learning the optimal dataset-specific regularization cocktail by means of a modern hyperparameter optimization method.

To the best of our knowledge, there exists no study providing empirical evidence that a mixture of numerous regularizers outperforms individual regularizers; this paper fills this gap. More precisely, the research hypothesis of this paper is that a properly mixed regularization cocktail outperforms every individual regularizer in it, in terms of accuracy under the same run-time budget, and that the best cocktail to use depends on the dataset. To validate this hypothesis, we executed a large-

scale experimental study employing 40 diverse tabular datasets and 13 prominent regularizers with a thorough hyperparameter tuning for all regularizers. Thereby, we make the followings contributions:

1. We demonstrate the empirical accuracy gains of regularization cocktails in a systematic manner via a large-scale experimental study;
2. We challenge the status-quo practices of designing universal dataset-agnostic regularizers, by showing that an optimal regularization cocktail is highly dataset-dependent;
3. We demonstrate that regularization cocktails achieve a higher gain on smaller datasets;
4. As an overarching contribution, this paper provides previously-lacking in-depth empirical evidence to better understand the importance of combining different mechanisms for regularization, one of the most fundamental concepts in machine learning.

2 Related Work

Weight decay: The classical approaches of regularization focused on minimizing the norms of the parameter values, concretely either the L1 (Tibshirani, 1996), the L2 (Tikhonov, 1943), or a combination of L1 and L2 known as the Elastic Net (Zou & Hastie, 2005). A recent work fixes the malpractice of adding the decay penalty term before momentum-based adaptive learning rate steps (e.g., in common implementations of Adam (Kingma & Ba, 2015)), by decoupling the regularization from the loss and applying it after the learning rate computation (Loshchilov & Hutter, 2019).

Data Augmentation: A different treatment of the overfitting phenomenon relies on enriching the training dataset via instance augmentation. The literature on data augmentation is vast, especially for image data, ranging from basic image manipulations (e.g., geometric transformations, or mixing images) up to parametric augmentation strategies such as adversarial and controller-based methods (Shorten & Khoshgoftaar, 2019). For example, Cut-Out (Devries & Taylor, 2017) proposes to mask a subset of input features (e.g., pixel patches for images) for ensuring that the predictions remain invariant to distortions in the input space. Along similar lines, Mix-Up (Zhang et al., 2018) generates new instances as a linear span of pairs of training examples, while Cut-Mix (Yun et al., 2019) suggests super-positions of instance pairs with mutually-exclusive pixel masks. A recent technique, called Aug-Mix (Hendrycks et al., 2020), generates instances by sampling chains of augmentation operations. On the other hand, the direction of reinforcement learning (RL) for augmentation policies was elaborated by Auto-Augment (Cubuk et al., 2019), followed by a technique that speeds up the training of the RL policy (S.Lim et al., 2019). Last but not least, adversarial attack strategies (e.g., FGSM (Goodfellow et al., 2015)) generate synthetic examples with minimal perturbations, which are employed in training robust models (Madry et al., 2018).

Model Averaging: Ensembled machine learning models have been shown to reduce variance and act as regularizers (Polikar, 2012). A popular ensemble neural network with shared weights among its base models is Drop-Out (Srivastava et al., 2014), which was extended to a variational version with a Gaussian posterior of the model parameters (Kingma et al., 2015). A follow-up work known as Mix-Out (Lee et al., 2020) extends Drop-Out by statistically fusing the parameters of two base models. Furthermore, ensembles can be created using models from the local optima discovered along a single convergence procedure (Huang et al., 2016).

Structural and Linearization: One strategy of regularizing deep learning models is to discover dedicated neural *structures* that generalize on particular tasks, such as image classification or Natural Language Processing (NLP). In that context, ResNet adds skip connections across layers (He et al., 2016), while the Inception model computes latent representations by aggregating diverse convolutional filter sizes (Szegedy et al., 2017). The attention mechanism gave rise to the popular Transformer architecture in the realm of NLP (Vaswani et al., 2017). Recently, EfficientNet is an architecture that easily scales deep convolutional neural networks by controlling only a few hyperparameters (Tan & Q.Le, 2019).

Besides the aforementioned manually-designed architectures, the stream of Neural Architecture Search (Elsken et al., 2019) focuses on exploring neural connectivity graphs for finding the optimal architectures via reinforcement learning (Zoph & Le, 2017), black-box search (Real et al., 2019) or differentiable solvers (Liu et al., 2019). A recent trend adds a dosage of *linearization* to deep models, where skip connections transfer embeddings from previous less non-linear layers (He et al.,

2016; Huang et al., 2017). Along similar lines, the Shake-Shake regularization deploys skip connections in parallel convolutional blocks and aggregates the parallel representations through affine combinations (Gastaldi, 2017), while Shake-Drop extends this mechanism to a larger number of CNN architectures (Yamada et al., 2018).

Implicit: The last family of regularizers broadly encapsulates methods which do not directly propose novel regularization techniques, but have an *implicit* regularization effect as a virtue of their 'modus operandi' (Arora et al., 2019). For instance, Batch Normalization improves generalization by reducing the internal covariate shifts (Ioffe & Szegedy, 2015), while early stopping of the optimization procedure also yields a similar generalization effect (Yao et al., 2007). On the other hand, stabilizing the convergence of the training routine is another implicit regularization, for instance by introducing learning rate scheduling schemes (Loshchilov & Hutter, 2017). The recent strategy of stochastic weight averaging relies on averaging parameter values from the local optima encountered along the sequence of optimization steps (Izmailov et al., 2018), while another approach conducts updates in the direction of few 'lookahead' steps (Zhang et al., 2019).

Positioning in the realm of AutoML: In contrast to the prior literature, we do not propose a new individual regularization method, but empirically identify the superiority of learning regularization cocktails among a set of existing regularizers from the aforementioned categories. We train dataset-specific cocktails as a hyperparameter optimization (HPO) task (Feurer & Hutter, 2019). In that regard, our work is positioned in the realm of AutoML, and is a special case of a combined algorithm selection and hyperparameter optimization (Thornton et al., 2013). We learn the regularization cocktails and optimize the joint hyperparameter configuration space by means of BOHB (Falkner et al., 2018b), which is a variation of Hyperband (Li et al., 2017) with model-based surrogates and is one of the current state of the art approaches in efficient HPO.

3 Mixing the Regularization Cocktail

3.1 The Recipe as Problem Statement

A training set is composed of features $X^{(\text{Train})}$ and targets $y^{(\text{Train})}$, while the test dataset is denoted by $X^{(\text{Test})}, y^{(\text{Test})}$. A parameterized function approximates the targets as $\hat{y} = f(X; \theta)$ whereas the parameters θ are trained to minimize a differentiable loss function $\mathcal{L}$ as:

$$\theta^* \in \arg\min_{\theta} \; \mathcal{L}\left(y^{(\text{Train})}, f\left(X^{(\text{Train})}; \theta\right)\right). \tag{1}$$

To generalize into minimizing $\mathcal{L}\left(y^{(\text{Test})}, f(X^{(\text{Test})}; \theta\right)$, the parameters of f are controlled with a regularization technique Ω that avoids overfitting to the peculiarities of the training data. With a slight abuse of notation we denote $f\left(X;\, \Omega\left(\theta; \lambda\right)\right)$ to be the predictions of the model f whose parameters θ are optimized under the regime of the regularization method $\Omega(\cdot; \lambda)$, where $\lambda \in \Lambda$ represent the hyperparameters of Ω. The training data is further divided into two subsets as training and validation splits,[1] the later denoted by $X^{(\text{Val})}, y^{(\text{Val})}$, such that λ can be tuned on the validation loss via the following hyperparameter optimization objective:

$$\lambda^* \in \arg\min_{\lambda \in \Lambda} \; \mathcal{L}\left(y^{(\text{Val})}, f\left(X^{(\text{Val})}; \Omega\left(\theta^*; \lambda\right)\right)\right), \tag{2}$$

$$\text{s.t.} \quad \theta^* \in \arg\min_{\theta} \; \mathcal{L}\left(y^{(\text{Train})}, f(X^{(\text{Train})}; \Omega\left(\theta; \lambda\right)\right). \tag{3}$$

While the search for optimal hyperparameters λ is an active field of research in the realm of AutoML (Hutter et al., 2018), still the choice of the regularizer Ω mostly remains an ad-hoc practice, where practitioners select few combinations among popular regularizers (Dropout, L2, Batch Normalization, etc.). In contrast to prior studies, we hypothesize that the optimal regularizer is a cocktail mixture of a large set of regularization methods, all being simultaneously applied with different strengths (i.e., dataset-specific hyperparameters). Given a set of K regularizers $\left\{\left(\Omega^{(k)}\left(\cdot; \lambda^{(k)}\right)\right\}_{k=1}^{K} := \left\{\Omega^{(1)}\left(\cdot; \lambda^{(1)}\right), \ldots, \Omega^{(K)}\left(\cdot; \lambda^{(K)}\right)\right\}$, each with its own hyperparameters

[1]For simplicity, we only discuss hold-out validation scheme here, but in principle any other validation scheme, such as cross validation and bootstrap sampling, would be possible.

$\lambda^{(k)} \in \Lambda^{(k)}, \forall k \in \{1, \ldots, K\}$, the problem of finding the optimal cocktail of regularizers is:

$$\lambda^* \in \underset{\lambda^{(1)} \in \Lambda^{(k)}, \ldots, \lambda^{(K)} \in \Lambda^{(k)}}{\arg\min} \mathcal{L}\left(y^{(\text{Val})}, f\left(X^{(\text{Val})}; \left\{\Omega^{(k)}\left(\theta^*, \lambda^{(k)}\right)\right\}_{k=1}^{K}\right)\right) \tag{4}$$

$$\text{s.t.} \quad \theta^* \in \underset{\theta}{\arg\min} \mathcal{L}\left(y^{(\text{Train})}, f\left(X^{(\text{Train})}; \left\{\Omega^{(k)}\left(\theta, \lambda^{(k)}\right)\right\}_{k=1}^{K}\right)\right) \tag{5}$$

The intuitive interpretation of Equations 4-5 is searching for the optimal hyperparameters λ (i.e., strengths) of the cocktail's regularizers using the validation set (Equation 4), given that the optimal prediction model parameters θ are trained under the regime of all the regularizers being applied jointly (Equation 5). We stress that the hyperparameters $\lambda^{(k)}$ include a conditional hyperparameter controlling whether the k-th regularizer is applied at all, or skipped. Therefore, the best cocktail might consist of combinations of a subset of regularizers.

3.2 Regularization Ingredients and the Search Space

To build the regularization cocktails we combine the 13 methods shown in Table 1, which are selected among the categories of regularizers covered in Section 2, each having its own hyperparameter search space. We set the other hyperparameters regarding the architecture and the optimizer as detailed in Table 2 in Appendix A.

In the defined search space some of the combinations are not technically feasible, therefore, we introduce the following constraints to the proposed search space: (i) We restrict stochastic weight averaging and snapshot ensembles from being active concurrently since they use snapshots of the optimization process in different, incompatible, ways. (ii) Shake-shake and Shake-Drop are not simultaneously active since the latter builds on the former. (iii) Only one data augmentation technique out of Mix-Up, Cut-Mix, Cut-Out, and FGSM adversarial learning can be active at once due to a technical limitation of the base library (Zimmer et al., 2020a).

As an optimizer, we decided to use BOHB (Falkner et al., 2018a) since it achieves a strong anytime performance by combining Hyperband (Li et al., 2017) and Bayesian Optimization (Shahriari et al., 2016), and still has the convergence guarantees of Hyperband. Furthermore, BOHB can deal with the categorical hyperparameters for enabling or disabling regularization techniques and the corresponding conditional structures.

4 Experimental Protocol

4.1 Experimental Setup

We use a collection of tabular datasets (listed in Table 3 in Appendix C) from the recent open-source OpenML AutoML Benchmark (Gijsbers et al., 2019), as well as popular online repositories, such as UCI (Asuncion & Newman, 2007) and Kaggle. Our benchmark of datasets includes tabular datasets from different domains, containing between 452 and 416188 instances, and between 4 and 2001 features, varying in terms of the number of numerical and categorical features. The datasets are retrieved from OpenML (Vanschoren et al., 2014) and split as 60% training, 20% validation and 20% testing sets. All data is standardized to have 0 mean and unit variance. For datasets with missing values, the median value on the known values is used as an imputation strategy.

We ran all experiments on a CPU cluster, each node of which contains two Intel Xeon E5-2630v4 at 2.2GHz with 20 CPU cores and a total memory of 128GB. We chose the PyTorch library (Paszke et al., 2019) as the main deep learning framework for our work and we extended the AutoDL-framework Auto-Pytorch (Mendoza et al., 2018; Zimmer et al., 2020a) with our implementations for the regularizers, as shown in Table 1.

To optimally utilize resources, we ran BOHB with 10 workers in parallel, where each worker had access to 2 CPU cores and 12GB of memory, executing one configuration at a time. In view of limited computational resources and taking into account the dimensions D of the considered configuration spaces, we ran BOHB for at most 4 days, or at most $40 \times D$ hyperparameter configurations, whichever came first. During the training phase, each configuration was run for 105 epochs. For the sake of studying the effect on more datasets, we only evaluated a single train-val-test split.

Regularizer	Hyperparameter	Type	Range	Conditionality
BN	BN-active	Boolean	{True, False}	–
WD	WD-active	Boolean	{True, False}	–
	Decay factor	Continuous	$\{10^{-5}, \ldots, 0.1\}$	WD-active
DO	DO-active	Boolean	{True, False}	–
	Dropout shape	Nominal	{funnel, long funnel, diamond, hexagon, brick, triangle, stairs}	DO-active
	Drop rate	Continuous	$\{0.0, \ldots, 0.8\}$	DO-active
SC	SC-active	Boolean	{True, False}	–
	MB choice	Nominal	{SS, SD, Standard}	SC-active
SS	SS-active	Boolean	{True, False}	SC-active $\wedge$ MB choice = SS
SD	SD-active	Boolean	{True, False}	SC-active $\wedge$ MB choice = SD
	Max. probability	Continuous	$\{0.0, 1.0\}$	SD-active
LA	LA-active	Boolean	{True, False}	–
	Step size	Continuous	$\{0.5, \ldots, 0.8\}$	LA-active
	Num. steps	Integer	$\{1, \ldots, 5\}$	LA-active
SWA	SWA-active	Boolean	{True, False}	¬SE-Active
SE	SE-active	Boolean	{True, False}	¬SWA-Active
	Num. models	Constant	$\{3\}$	SE-active
–	Augment	Nominal	{MU, CM, CO, AT, None}	–
MU	MU-active	Boolean	{True, False}	Augment = MU
	Mix. magnitude	Continuous	$\{0.0, \ldots, 1.0\}$	MU-active
CM	CM-active	Boolean	{True, False}	Augment = CM
	Probability	Continuous	$\{0.0, \ldots, 1.0\}$	CM-active
CO	CO-active	Boolean	{True, False}	Augment = CO
	Probability	Continuous	$\{0.0, \ldots, 1.0\}$	CO-active
	Patch ratio	Continuous	$\{0.0, \ldots, 1.0\}$	CO-active
AT	AT-active	Boolean	{True, False}	Augment = AT

Table 1: The configuration space for the regularization cocktail regarding the **explicit regularization hyperparameters** of the methods and the conditional constraints enabling or disabling them. (BN: Batch Normalization, WD: Weight Decay, DO: Drop-Out, SC: Skip Connection, MB: Multi-branch choice, SS: Shake-Shake, SD: Shake-Drop, LA: Lookahead Optimizer, SWA: Stochastic Weight Averaging, SE: Snapshot Ensembles, MU: Mix-Up, CM: Cut-Mix, CO: Cut-Out, and AT: FGSM Adversarial Learning)

4.2 Fixed Architecture and Optimization Hyperparameters

In order to focus exclusively on investigating the effect of individual regularization methods, we fix the hyperparameters that are related to the model architecture and general training procedure in the search-space, as specified in Table 2 of Appendix A. These hyperparameter values are tuned for maximizing the performance of an unregularized neural network on our dataset collection (see Table 3 in Appendix C). Moreover, we use a 9-layer feed-forward neural network with 512 units for each layer, a choice based on a previous related work (Orhan & Pitkow, 2017). We emphasize that the network has a sufficiently large capacity, to ensure that the effect of regularization methods would be noticeable.

Moreover, we set a low learning rate of 10^{-3} after performing a grid search for finding the value performing best across all datasets. We use the AdamW implementation (Loshchilov & Hutter, 2019), which implements decoupled weight decay, and cosine annealing with restarts (Loshchilov & Hutter, 2019) as a learning rate scheduler. Using a learning rate scheduler with restarts helps in our case because we keep a fixed initial learning rate. For the restarts, we use an initial budget of 15 epochs, with a budget multiplier 3, following published practices (Zimmer et al., 2020b). Addition-

ally, since our benchmark includes imbalanced datasets, we use a weighted version of categorical cross-entropy and balanced accuracy (Brodersen et al., 2010) as the evaluation metric.

4.3 Hypotheses and Associated Experiments

Hypothesis 1: The regularization cocktails achieve better generalization performance compared to the individual regularization methods over all datasets.

Experiment 1: We regularize the plain neural network (Section 4.2) with each method from Table 1, one at a time. For every regularizer, we tune its hyperparameters on each dataset, then finally measure the regularized network's performance on the test set after training the best hyper-parameter configuration on the joint train and validation set. For each dataset, we compare the results against the results of a cocktail optimized the same way as each of the individual ingredient regularizers.

Hypothesis 2: The optimal regularization cocktails are dataset-dependent.

Experiment 2: We study the best found regularization cocktails of every dataset and frequencies of the regularizers that were chosen to be activated by BOHB, to demonstrate that no combination of regularizers is frequent. Furthermore, we regularize the plain network with the most frequent regularizers and compare it against our proposed method of Section 3.

Hypothesis 3: The regularization cocktails achieve a higher gain on smaller datasets.

Experiment 3: We scale-down the 3 largest datasets in our collection, sampling random subsets of the ratios $(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32})$ of the original dataset size. We compare the gain of a cocktail vs. an unregularized neural network across different data sizes.

5 Experimental Results

Regularization cocktail performance (Experiment 1): Figure 1 shows that the cocktail outperforms each individual regularizer. We plotted the rank distribution of the different regularization experiments over our collection of datasets. For each method, this distribution is computed from the test set performance on the collection of 40 datasets, with the full per-dataset numerical results found in Table 4 at Appendix C.

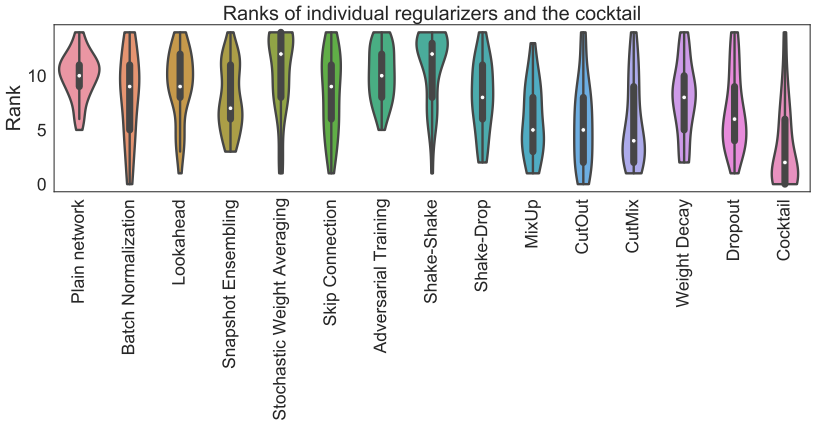

Figure 1: **Rank distribution (lower is better) for the individual regularization methods and the regularization cocktail.** The rank distribution is calculated on the test set over all datasets.

Furthermore, Figure 2 shows the critical difference diagram of the ranks, where we demonstrate that our regularization cocktail has a statistically significantly margin over the individual regularization techniques. The critical difference diagram is generated by performing a post-hoc analysis based on the Wilcoxon-Holm method (Wilcoxon, 1992; Holm, 1979). Both figures confirm our hypothesis that well-tuned regularization cocktails substantially outperform well-tuned individual regularization techniques across a diverse suite of tabular datasets.

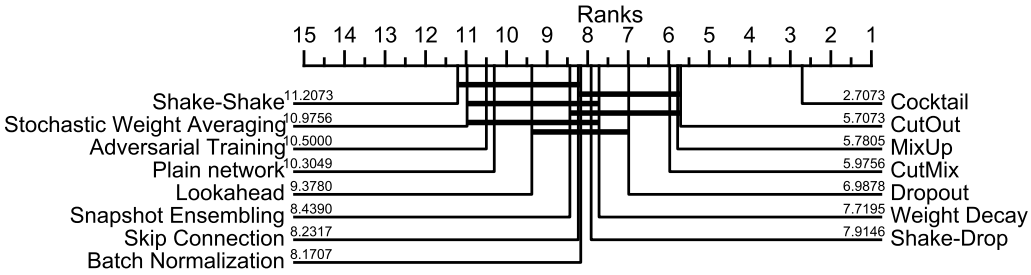

Figure 2: **Critical difference diagram** generated with the Wilcoxon-Holm post-hoc analysis on 40 datasets. The diagram shows the ranks and the statistical significance of the results for every individual regularization technique and the regularization cocktails

.

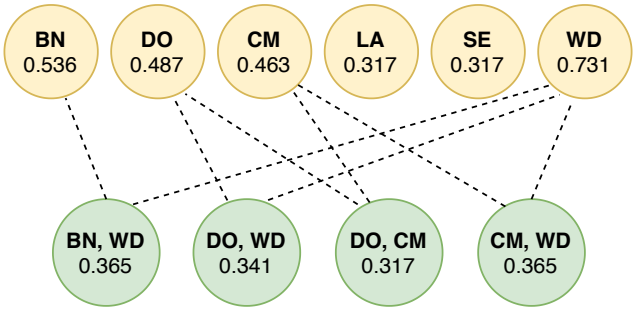

Figure 3: All the frequent cocktail ingredients with occurrence ratios above 0.3, indicating ingredients present in at least 30% of the datasets.

Dataset-dependent optimal cocktails (Experiment 2): Figure 3 shows that the optimal regularization cocktail depends on the dataset at hand, since no combination of regularizers was active on the majority of the datasets. In detail, the figure shows all the frequent regularizers and combinations of regularizers that occur on a minimal threshold of at least 30% of the datasets, based on how often they were part of the per-dataset cocktails. The most frequent combinations of pairs of regularizers are selected on at most 36.5% of the datasets, which highlights the fact that regularization cocktails are dataset-specific and there is no frequent universal combination. For a more in depth summary on the frequencies that correspond to the regularization methods, we refer to Figure 3 in Appendix B.

To further validate this point, we conducted another experiment by creating a cocktail of the most frequent regularizers (Batch Normalization, Dropout and Weight Decay) and by tuning their combined hyper-parameters specifically for each dataset. The combination of these frequent regularizers still underperforms our large cocktail as indicated in Figure 4, which shows again that optimal cocktails are dataset-dependent. Additionally, a Wilcoxon signed rank test with $\alpha = 0.05$ between both sets of results confirmed that the results are statistically significant (p-value 0.00009).

Gain of the cocktail on smaller datasets (Experiment 3): In Figure 5, we present the results of the best-found regularization cocktails on differently-sized subsets of the largest datasets (details in Section 4.3). It can be clearly seen that the gain of the regularization cocktail is higher as the subset size decreases. This is an important empirical finding that might indicate that strong regularization (by means of optimized regularization cocktails) can help deep learning achieve much better performance for smaller datasets than generally perceived by the deep learning community.

6 Conclusions and Future Work

Even though combining regularizers is a relatively frequent practice by researchers, to date there exists no prior work that systematically studies the effect of **optimally** combining regularization

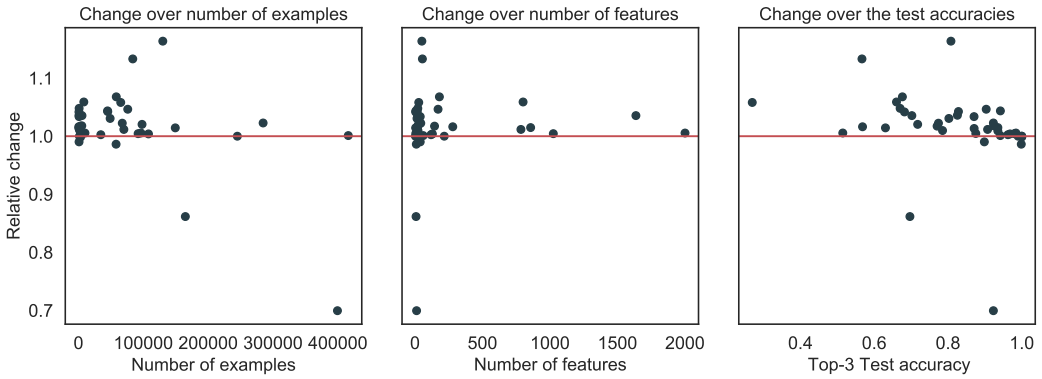

Figure 4: **Gain of the full search space regularization cocktail** A summary of the gain that the full search space regularization cocktail achieves compared to the constrained version with only the top 3 methods. **From the left:** The gain on the test set over the number of examples, the gain over the number of features, and the gain over the test accuracy of the constrained version.

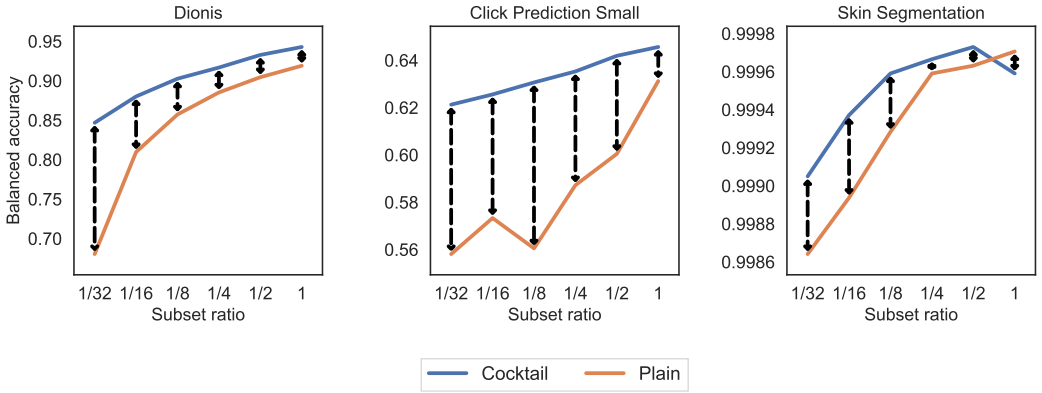

Figure 5: **Gain of the regularization cocktails**. The gain of the best found regularization cocktails over the plain network is shown with the interrupted arrow lines.

methods. This paper presented the first step in empirically studying regularization cocktails, by posing the problem as a standard hyper-parameter optimization challenge. We conducted a large-scale experiment involving 13 regularization methods and 40 datasets, with a thorough hyperparameter optimization procedure for each technique. The findings of this study can be summarized as **three simple take-home messages** for practitioners:

1. Instead of applying a single regularization technique, we recommend to exploit the complementary effects of regularization cocktails.
2. In particular for small datasets, we recommend to leverage the benefits of regularization cocktails.
3. To obtain a well-performing, dataset-specific regularization cocktail, using state-of-the-art hyperparameter optimization techniques is recommended.

As a future work, we would like to combine regularization cocktails for neural networks with automated data preprocessing pipelines and architecture search, in order to make deep learning achieve state-of-the-art performance on small tabular data.

REFERENCES

S. Arora, N. Cohen, W. Hu, and Y. Luo. Implicit regularization in deep matrix factorization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 7413–7424. Curran Associates, Inc., 2019.

A. Asuncion and D. Newman. Uci machine learning repository, 2007.

K. Henning Brodersen, C., and E. Klaas Stephan J. M Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pp. 3121–3124. IEEE, 2010.

E. Cubuk, B. Zoph, D. Mane, and V.Vasudevanand Q. Le. Autoaugment: Learning augmentation strategies from data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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

T. Devries and G. Taylor. Improved regularization of convolutional neural networks with cutout. *ArXiv*, abs/1708.04552, 2017.

T. Elsken, J. H. Metzen, and F. Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola. Autogluon-tabular: Robust and accurate automl for structured data. *CoRR*, abs/2003.06505, 2020.

S. Falkner, A.Klein, and F. Hutter. BOHB: Robust and efficient hyperparameter optimization at scalae. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, pp. 1436–1445, July 2018a.

S. Falkner, A. Klein, and F. Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In J. Dy and A. Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1437–1446, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018b. PMLR.

M. Feurer and F. Hutter. Hyperparameter optimization. In F. Hutter, L. Kotthoff, and J. Vanschoren (eds.), *Automated Machine Learning - Methods, Systems, Challenges*, The Springer Series on Challenges in Machine Learning, pp. 3–33. Springer, 2019. doi: 10.1007/978-3-030-05318-5_1.

M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pp. 2755–2763, Cambridge, MA, USA, 2015. MIT Press.

X. Gastaldi. Shake-shake regularization of 3-branch residual networks. In *5th International Conference on Learning Representations, ICLR*. OpenReview.net, 2017.

P. Gijsbers, E. LeDell, S. Poirier, J. Thomas, B. Bischl, and J. Vanschoren. An open source automl benchmark. *arXiv preprint arXiv:1907.00909 [cs.LG]*, 2019. URL `https://arxiv.org/abs/1907.00909`. Accepted at AutoML Workshop at ICML 2019.

I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. 2015. URL `http://arxiv.org/abs/1412.6572`.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

D. Hendrycks, N. Mu, E. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1gmrxHFvB`.

S. Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

G. Huang, Y. Li, G. Pleiss, Z. Liu, J. Hopcroft, and K. Weinberger. Snapshot Ensembles: Train 1, Get M for Free. *International Conference on Learning Representations*, November 2016.

G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

F. Hutter, L. Kotthoff, and J. Vanschoren (eds.). *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2018. In press, available at http://automl.org/book.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach and D. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, Lille, France, 07–09 Jul 2015. PMLR.

P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. Wilson. Averaging weights leads to wider optima and better generalization, 2018. cite arxiv:1803.05407Comment: Appears at the Conference on Uncertainty in Artificial Intelligence (UAI), 2018.

H. Jin, Q. Song, and X. Hu. Auto-keras: An efficient neural architecture search system. In A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis (eds.), *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD*, pp. 1946–1956. ACM, 2019. doi: 10.1145/3292500.3330648.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

D. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pp. 2575–2583, Cambridge, MA, USA, 2015. MIT Press.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

C. Lee, K. Cho, and W. Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HkgaETNtDB`.

L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.*, 18(1):6765–6816, January 2017. ISSN 1532-4435.

H. Liu, S. Karen, and Y. Yiming. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=S1eYHoC5FX`.

I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR) 2017 Conference Track*, April 2017.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rJzIBfZAb`.

H. Mendoza, A. Klein, M. Feurer, J. Tobias Springenberg, M. Urban, M. Burkart, M. Dippel, M. Lindauer, and F. Hutter. Towards automatically-tuned deep neural networks. In F. Hutter, L. Kotthoff, and J. Vanschoren (eds.), *AutoML: Methods, Sytems, Challenges*, chapter 7, pp. 141–156. Springer, December 2018. To appear.

R. Olson and J. Moore. TPOT: A tree-based pipeline optimization tool for automating machine learning. In F. Hutter, L. Kotthoff, and J. Vanschoren (eds.), *Proceedings of the 2016 Workshop on Automatic Machine Learning, AutoML 2016, co-located with 33rd International Conference on Machine Learning (ICML 2016)*, volume 64 of *JMLR Workshop and Conference Proceedings*, pp. 66–74. JMLR.org, 2016.

A. Emin Orhan and X. Pitkow. Skip connections eliminate singularities. *arXiv preprint arXiv:1701.09175*, 2017.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pp. 8026–8037, 2019.

R. Polikar. Ensemble Learning. In C. Zhang and Y. Ma (eds.), *Ensemble Machine Learning: Methods and Applications*, pp. 1–34. Springer US, Boston, MA, 2012. ISBN 978-1-4419-9326-7. doi: 10.1007/978-1-4419-9326-7_1.

E. Real, A. Aggarwal, Y. Huang, and Q. Le. Regularized evolution for image classifier architecture search. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 4780–4789. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33014780. URL https://doi.org/10.1609/aaai.v33i01.33014780.

B. Shahriari, K. Swersky, Z. Wang, R. Adams, and N. de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

C. Shorten and T. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:1–48, 2019.

S.Lim, I. Kim, T. Kim, C. Kim, and S. Kim. Fast autoaugment. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alche-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 6665–6675. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8892-fast-autoaugment.pdf.

J. Snoek, H. Larochelle, and R. Adams. Practical bayesian optimization of machine learning algorithms. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger (eds.), *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings*, pp. 2960–2968, 2012.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014. ISSN 1532-4435.

C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

M. Tan and Q.Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri and R. Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/tan19a.html.

C. Thornton, F. Hutter, and H. Hoos K. Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 847–855, 2013.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

A. Tikhonov. On the stability of inverse problems. In *Doklady Akademii Nauk SSSR*, 1943.

J. Vanschoren, J. Van Rijn, B. Bischl, and L. Torgo. Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

F. Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pp. 196–202. Springer, 1992.

Y. Yamada, M. Iwamura, and K. Kise. Shakedrop regularization, 2018. URL https://openreview.net/forum?id=S1NHaMW0b.

Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, August 2007. ISSN 1432-0940. doi: 10.1007/s00365-006-0663-2. URL https://doi.org/10.1007/s00365-006-0663-2.

S. Yun, D. Han, S. Joon, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, 2019. to appear.

H. Zhang, M. Cisse, Y. Dauphin, and D. Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

M. Zhang, J. Lucas, J. Ba, and G. Hinton. Lookahead optimizer: k steps forward, 1 step back. In H. Wallach, H. Larochelle, A. Beygelzimer, F. Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 9593–9604, 2019.

L. Zimmer, M. Lindauer, and F. Hutter. Auto-pytorch tabular: Multi-fidelity metalearning for efficient and robust autodl. *CoRR*, abs/2006.13799, 2020a.

L. Zimmer, M. Lindauer, and F. Hutter. Auto-pytorch tabular: Multi-fidelity metalearning for efficient and robust autodl. *arXiv preprint arXiv:2006.13799*, 2020b.

B. Zoph and Q. Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR Conference Track Proceedings*. OpenReview.net, 2017.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.