FINDING LOST DG: EXPLAINING DOMAIN GENERALIZATION VIA MODEL COMPLEXITY

Anonymous authors
Paper under double-blind review

ABSTRACT

The domain generalization (DG) problem setting challenges a model trained on multiple known data distributions to generalise well on unseen data distributions. Due to its practical importance, a large number of methods have been proposed to address this challenge. However most of this work is empirical, as the DG problem is hard to model formally; and recent evaluations have cast doubt on existing methods' practical efficacy – in particular compared to a well chosen empirical risk minimisation baseline. We present a novel learning-theoretic generalisation bound for DG that bounds novel domain generalisation performance in terms of the model's Rademacher complexity. Based on this, we conjecture that the causal factor behind existing methods' efficacy or lack thereof is a variant of the standard empirical risk-predictor complexity trade-off, and demonstrate that their performance variability can be explained in these terms. Algorithmically, this analysis suggests that domain generalisation should be achieved by simply performing regularised ERM with a leave-one-domain-out cross-validation objective. Empirical results on the DomainBed benchmark corroborate this.

1 INTRODUCTION

Machine learning systems have shown exceptional performance on numerous tasks in computer vision and beyond. However performance drops rapidly when the standard assumption of i.i.d. training and testing data is violated. This domain-shift phenomenon occurs widely in many applications of machine learning (Csurka, 2017; Zhou et al., 2021; Koh et al., 2021), and often leads to disappointing results in practical machine learning deployments, since data 'in the wild' is almost inevitably different from reference training sets.

Given the practical significance of this issue, a large number of methods have been proposed that aim to improve models' robustness to deployment under a different distribution than used for training (Zhou et al., 2021), a problem setting known as domain generalisation (DG). These methods span diverse approaches such as specialized neural architectures, data augmentation strategies, and regularizers. Nevertheless, the DG problem setting is difficult to model formally for principled derivation and theoretical analysis of algorithms, the target domain of interest is unobservable during training, and can't be directly approximated by the training domains due to domain-shift. Therefore the majority of these existing approaches are based on poorly understood empirical heuristics.

To make matters worse, a recent study by Gulrajani & Lopez-Paz (2021) assessed the state of DG research with a carefully conducted comparative evaluation of algorithms on a large benchmark suite under a common platform. They found that published methods were not as effective as claimed, and in particular reported that 'no existing method reliably beats a well tuned empirical risk minimization (ERM) baseline'. We argue that this negative result highlights the need for better theory in this area in order to understand why existing algorithms have such erratic performance, and to guide the development of principled algorithms that are more effective and reliable.

To this end, our first contribution is to present an intuitive learning-theoretic bound for DG performance. Intuitively, while the held-out domain of interest is indeed unobservable during training, we can bound its performance using learning theoretic tools similar to the standard ones used to bound the performance on (unobserved) testing data given (observed) training data. In particular we show that the performance on a held out target domain is bounded by the performance on known source

domains, plus two additional model complexity terms, that describe how much a model can possibly have overfitted to the training domains. This theoretical contribution leads to several insights.

Firstly, our theory suggests that DG performance is governed by a trade-off between empirical risk and model complexity that is analogous to the corresponding and widely understood trade-off that explains generalisation in supervised learning as an overfitting-underfitting trade-off (Geman et al., 1992). Based on this, we hypothesise that the causal factor behind the performance variability of existing DG methods is actually simply implicit or explicit regularization. That is, the plethora of different strategies available (Zhou et al., 2021) – from data-augmentation to specialized optimizers – actually affect DG performance by explicitly or implicitly choosing different fit-complexity trade-offs. We corroborate this hypothesis by evaluating a number of models in the DomainBed suite in terms of complexity, and showing that their apparently erratic performance in Gulrajani & Lopez-Paz (2021) is actually consistent with an explanation in terms of implied complexity.

Practically, our analysis suggests that the model selection strategy (Hastie et al., 2009) is a factor in DG performance that is at least as important as the actual mechanism of model complexity control (i.e., Tuning of regularisation strength vs specific parametric design of regularizer). In particular, regularisation should be stronger when optimizing for future DG performance than when optimizing for performance on seen domains. Unfortunately, model complexity is hard to carefully control in deep learning due to the large number of relevant factors (architecture, regularisers, implicit regularization from optimizer, etc). Gulrajani & Lopez-Paz (2021) attempted to address this by hyper-parameter search in the DomainBed benchmark, but are hampered by the computational infeasibility of accurate hyper-parameter search. In this paper, we use linear models and off-the-shelf self-supervised features to demonstrate much more clearly how cross-domain performance depends on complexity. Specifically, our theoretical and empirical results show that, contrary to the conclusion of Gulrajani & Lopez-Paz (2021), simple domain-wise cross-validation is a better objective to drive DG model selection.

In summary, based on our new generalisation bound, and associated empirical analysis, our take-home messages are: (i) Model fit vs complexity trade-off is a key determinant of DG performance, that explains existing DG algorithm performance variability. (ii) The complexity control strategy used to determine bias-variance trade-off is crucial in practice, with peak DG performance achieved when optimizing model complexity based on domain-wise validation. (iii) Regularisation required for optimal DG is greater than for conventional optimization for within-domain performance.

2 Bounding Risk for Domain Generalization

Supervised learning is concerned with learning a mapping from some input space $\mathcal{X}$, to a label space $\mathcal{Y}$, given data drawn from a distribution on $\mathcal{X} \times \mathcal{Y}$. One aims to find a model, $f^* \in \mathcal{F}$, that minimises the expected loss (also called risk) on unseen data,

$$f^* = \arg\min_{f \in \mathcal{F}} R_p(f), \qquad R_p(f) = \mathbb{E}_{(\vec{x},y)\sim p}[\mathcal{L}(f(\vec{x}), y)], \tag{1}$$

where p is the data distribution and $\mathcal{L}(\cdot,\cdot)$ is the loss function. In practice, we only have access to a finite sample of data, $S = \{(\vec{x}_i, y_i)\}_{i=1}^m$, from this distribution, so must minimise an empirical risk estimate,

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \hat{R}_p(f), \qquad \hat{R}_p(f) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(f(\vec{x}_i), y_i). \tag{2}$$

where m is the number of training examples. One of the central focuses of statistical learning theory is to bound the difference between these two types of risk. For example, in the standard single domain setting this can be done via

$$R_p(f) \leq \hat{R}_p(f) + 2\mathcal{R}_m(\mathcal{F}) + \mathcal{O}\left(\sqrt{\frac{\ln(1/\delta)}{m}}\right), \tag{3}$$

which holds with probability at least $1-\delta$, and $\mathcal{R}(\mathcal{F})$ is known as the empirical Rademacher complexity of the hypothesis class, $\mathcal{F}$. This complexity term is defined as

$$\mathcal{R}_m(\mathcal{F}) = \mathbb{E}_{\vec{\sigma}}\left[\sup_{f \in \mathcal{F}} \frac{1}{m}\sum_{i=1}^{m} \sigma_i \mathcal{L}(f(\vec{x}_i), y_i)\right]. \tag{4}$$

For a hypothesis class consisting of norm-constrained linear classifiers, one can achieve the following upper bound on the empirical Rademacher complexity (Shalev-Shwartz & Ben-David, 2014),

$$\mathcal{R}(\mathcal{F}) \leq \frac{XB}{\sqrt{m}}, \qquad \mathcal{F} = \{\vec{x} \mapsto \vec{x} \cdot \vec{w} : \|\vec{w}\|_2 \leq B\},$$

where we assume that each input, $\vec{x}$, has a Euclidean norm of at most X. There are a variety of ways to define hypothesis classes for neural networks, but most recent approaches take the view of fixing a particular architecture and specifying constraints on the norms of weights or distances they can move from their initialisations (Bartlett et al., 2017; Neyshabur et al., 2019; Gouk et al., 2021).

Domain generalisation. While supervised learning assumes all data come from the same distribution, the DG problem setting assumes the existence of an *environment* $\mathcal{E}$, of distributions p. The conceptually simplest—and often implicitly assumed—goal of DG methods is to minimise the expected risk across different distributions that could be sampled from the environment,

$$R^{\mathcal{E}}(f) = \mathbb{E}_{p \sim \mathcal{E}}[R_p(f)]. \tag{5}$$

As with the single domain learning problem, we only have access to an empirical estimate of the risk,

$$\hat{R}^{\mathcal{E}}(f) = \frac{1}{n} \sum_{j=1}^{n} \hat{R}_{p_j}(f), \tag{6}$$

where we assume for ease of exposition that all n domains have the same number of examples.

2.1 Bounding the Generalisation Gap

We next bound the generalisation gap between the observed empirical risk, $\hat{R}^{\mathcal{E}}$, on the source domains and the expected risk, $R^{\mathcal{E}}$, on unseen domains that holds uniformly for all hypotheses in $\mathcal{F}$.

Theorem 1. *For a 1-Lipschitz $\mathcal{L}(\cdot, \cdot)$ loss taking values in $[0, 1]$, with confidence at least $1 - 2\delta$ for all $f \in \mathcal{F}$ we have that*

$$R^{\mathcal{E}}(f) \leq \hat{R}^{\mathcal{E}}(f) + 2\mathcal{R}_{mn}(\mathcal{F}) + 2\mathcal{R}_n(\mathcal{F}) + \mathcal{O}\left(\sqrt{\frac{ln(1/\delta)}{mn}}\right) + \mathcal{O}\left(\sqrt{\frac{ln(1/\delta)}{n}}\right),$$

where n is the number of training domains and m is the number of training examples in each training domain.

Proof. Standard tools from learning theory (Bartlett & Mendelson, 2002) tell us that with probability at least $1 - \delta$,

$$\frac{1}{n} \sum_{j=1}^{n} R_{p_j}(f) \leq \hat{R}^{\mathcal{E}}(f) + 2\mathcal{R}_{mn}(\mathcal{F}) + \mathcal{O}\left(\sqrt{\frac{\ln(1/\delta)}{mn}}\right). \tag{7}$$

This is possible because Rademacher complexity does not require all instances to be identically distributed—just independent. However it does require that the i-th instance of the ghost sample is drawn from the same distribution as the i-th distribution of the "true" sample. Nevertheless, Equation 7 is only useful if the same domains seen during training are also seen at test time. Thus, we must provide a (high confidence) upper bound on

$$R^{\mathcal{E}}(f) - \frac{1}{n} \sum_{j=1}^{n} R_{p_j}(f). \tag{8}$$

We can use the same idea as Bartlett & Mendelson (2002) to show how Rademacher complexity controls generalisation to novel domains, rather than novel instances within the same domain,

$$\mathbb{E}_{p_{1:n}\sim\mathcal{E}}\left[\sup_{f\in\mathcal{F}}\left(\mathbb{E}_{q\sim\mathcal{E}}[R_q(f)]-\frac{1}{n}\sum_{j=1}^{n}R_{p_j}(f)\right)\right] \tag{9}$$

$$=\mathbb{E}_{p_{1:n}\sim\mathcal{E}}\left[\sup_{f\in\mathcal{F}}\mathbb{E}_{q_{1:n}\sim\mathcal{E}}\left[\frac{1}{n}\sum_{j=1}^{n}\Big(R_{q_j}(f)-R_{p_j}(f)\Big)\right]\right] \tag{10}$$

$$\leq\mathbb{E}_{p_{1:n},q_{1:n}\sim\mathcal{E}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{j=1}^{n}\Big(R_{q_j}(f)-R_{p_j}(f)\Big)\right] \tag{11}$$

$$=\mathbb{E}_{p_{1:n},q_{1:n}\sim\mathcal{E}}\mathbb{E}_{\vec{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{j=1}^{n}\sigma_j\Big(R_{q_j}(f)-R_{p_j}(f)\Big)\right] \tag{12}$$

$$\leq 2\mathbb{E}_{p_{1:n}\sim\mathcal{E}}\mathbb{E}_{\vec{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{j=1}^{n}\sigma_j R_{p_j}(f)\right] \tag{13}$$

$$=2\mathbb{E}_{p_{1:n}\sim\mathcal{E}}\mathbb{E}_{\vec{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{j=1}^{n}\sigma_j\mathbb{E}_{(\vec{x},y)\sim p_j}[\mathcal{L}(f(\vec{x}),y)]\right] \tag{14}$$

$$\leq 2\mathbb{E}_{p_{1:n}\sim\mathcal{E}}\mathbb{E}_{(\vec{x}_j,y_j)\sim p_j}\mathbb{E}_{\vec{\sigma}}\left[\sup_{h\in\mathcal{F}}\frac{1}{n}\sum_{j=1}^{n}\sigma_j\mathcal{L}(f(\vec{x}_j),y_j)\right] \tag{15}$$

$$=2\mathcal{R}_n(\mathcal{F}) \tag{16}$$

where the first inequality comes from moving the supremum inside the expectation, $\vec{\sigma}$ is a vector of Rademacher random variables, the second inequality is due to the subadditivity of suprema, and the final inequality comes from moving the expectation outside of the supremum. Using a standard application (and inversion) of McDiarmid's inequality, we can transform this bound on the *expected* generalisation gap to be a high confidence bound on a particular realisation of the generalisation gap. □

Discussion Theorem 1 tells us that expected risk on unseen domains is bounded by the empirical risk (training loss) on seen domains, plus Rademacher complexity terms and sampling error terms that decay with the number of domains and training instances per domain. As with typical single-domain bounds, the latter sampling error terms are not under control of the model designer. The former Rademacher terms describe the complexity of the chosen model class and govern how much it could possibly overfit to the seen domains while minimising empirical risk. In the case of linear models, these terms depend on weight norms, while in the case of deep models they further depend on properties of the chosen network architecture. Mirroring conventional generalisation in supervised learning, a very simple model may minimise the Rademacher terms, $\mathcal{R}$, while producing high empirical risk, $\hat{R}^{\mathcal{E}}$, and vice-versa. Thus good generalisation critically depends on a carefully chosen empirical risk vs model complexity trade-off. The difference between Theorem 1 and single domain bounds (e.g., Equation 3) is the additional dependence on the number of domains n and additional Rademacher complexity term. This implies that when the goal is to generalise to new domains, the risk of overfitting is higher. Therefore we expect a lower complexity model to me optimal for held out domain performance compared to seen domain performance in standard supervised learning.

2.2 Bounding the Excess Risk of ERM

As a second theoretical contribution, we next bound the excess expected risk between the ERM solution $\hat{f}$ and the *best possible model* f^* within the function class $\mathcal{F}$. Note that bounding excess risk, as opposed to the generalisation gap, overcomes some of the known issues with theoretically analysing the generalisation properties of deep networks (Belkin et al., 2018).

Theorem 2. *With probability as least $1-2\delta$, the excess risk of the empirical risk minimiser in $\mathcal{F}$ is bounded as*

$$R^{\mathcal{E}}(\hat{f}) - R^{\mathcal{E}}(f^*) \leq 2\mathcal{R}_{mn}(\mathcal{F}) + 2\mathcal{R}_n(\mathcal{F}) + \mathcal{O}\left(\sqrt{\frac{ln(1/\delta)}{mn}}\right) + \mathcal{O}\left(\sqrt{\frac{ln(1/\delta)}{n}}\right),$$

The proof uses the same technique as the usual proof for excess risk (see, e.g., (Pontil & Maurer, 2013)), but we include it in Appendix A for completeness.

Discussion Theorem 2 tells us that the gap between ERM and the *best possible* predictor in the function class depends on the same complexity terms observed in Theorem 1. In particular, for any typical hypothesis class, ERM converges to the optimal classifier at a rate of $\mathcal{O}(1/\sqrt{mn}) + \mathcal{O}(1/\sqrt{n})$. To justify itself theoretically, any future DG method that claims to be better than ERM should either: (1) demonstrate a faster convergence rate than this—at least by an improved constant factor; or (2) formally show that the chosen hypothesis class is composed of models that can extrapolate to new domains without additional data.

3 EXPERIMENTS

Based on our previous theoretical analysis, we conduct experiments on DomainBed (Gulrajani & Lopez-Paz, 2021). In particular we aim to address the following questions: (1) Theorem 1 shows that novel domain generalisation performance is governed by empirical-risk complexity trade-off. *Can we directly demonstrate the dependence of generalisation on complexity by controlling for model complexity?* (2) *Can the erratic performance of state of the art methods previously observed on DomainBed be explained in terms of model complexity?* (3) Given that model complexity is a crucial determinant of generalisation, *what is the best objective for tuning regularisation strength?*

3.1 DOMAIN GENERALISATION PERFORMANCE DEPENDS ON MODEL COMPLEXITY

To directly investigate the impact of model complexity on domain generalisation performance, we first work with Linear SVM applied to pre-computed deep features[1]. In this function class model complexity can be directly controlled by a scalar hyper-parameter, the objective is convex so confounding factors in deep network training (optimizer choice, early stopping, etc) disappear, and training is fast enough that we can densely and exhaustively evaluate a wide range of complexities.

Setup We use DINO (Caron et al., 2021) pre-trained models, such as DINO-ViTB8 and DINO-ViTS8 to extract features, and then train LinearSVC classifiers. We experiment on six different DG benchmarks, including RotatedMNIST (Ghifary et al., 2015), PACS (Li et al., 2017), VLCS (Fang et al., 2013), OfficeHome (Venkateswara et al., 2017), SVIRO (Dias Da Cruz et al., 2021) and TerraIncognit (Beery et al., 2018). Different trials have different numbers of training instances in practice, so we used a linear LinearSVC objective of the form $\frac{C}{n}\sum_i^n \mathcal{L}(f_w(x_i), y_i) + \|w\|^2$, with parameter C controlling the complexity of model w through loss-regularization trade-off. We searched a wide range of $\log_2 C$ in $\{-10, \ldots, 10\}$. We then conduct two experiments, holding training and test split size constant across both: (i) Training on the train splits of all domains, and testing on the test splits of all domains (i.e., standard supervised learning). (ii) Training on the train splits of three domains, and testing on the test splits of a held out domain (i.e., DG performance).

Results The results in Fig 1 average over 5 random seeds for dataset splitting, and all choices of hold-out target domain. From these we can see that: (i) all experiments exhibit the classic trade-off between fitting the data well and constraining hypothesis class complexity appropriately. We observe underfitting for high regularization (small C), and overfitting at low regularization (large C). (ii) Comparing the within- and across-domain evaluation: across-domain leads to lower performance—which is to be expected, due to the distribution shift. A more noteworthy observation is that, the optimal regularization for novel-domain performance is stronger than for seen-domain performance (red vertical lines left of blue). This corroborates our theoretical result that the ideal model complexity is lower for DG than for conventional supervised learning. Note that these results are using

[1]Note that when a *fixed* feature extractor, or any other preprocessing is used, it does not impact the model complexity or associated generalisation bound.

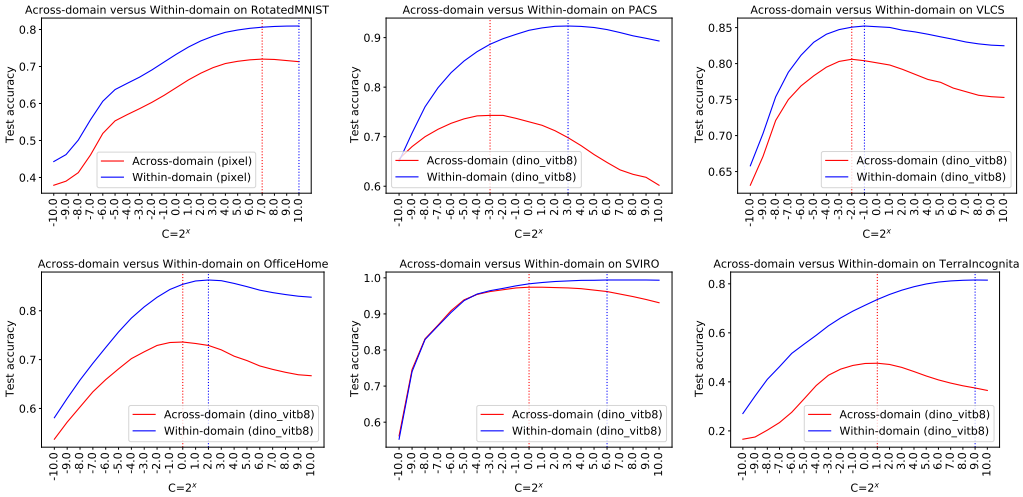

Figure 1: Linear SVC performance on DomainBed benchmark datasets is governed by model complexity parameter C. Optimal tuning for performance on novel target domains (DG condition, red) always requires stronger regularization (lower C) than for performance on seen domains (blue).

an exhaustive oracle to explore mode complexity, and do not constitute a complete algorithm, which would need to chose a specific complexity. We discuss an algorithm in Sec 3.3.

3.2 Explaining DomainBed DG performance in terms of complexity

Our experiment in Section 3.1 showed that linear models exhibit a very clean trade-off between complexity and generalization. However there are many subtle factors that influence effective model complexity in neural networks besides architecture: learning rate, learning rate schedule, weight decay, early stopping, etc; and the repeatably of learning is hampered by non-convexity, stochastic initialization, and stochastic mini-batches, etc. The previous evaluation study of Gulrajani & Lopez-Paz (2021) attempted some degree of control over these factors for state of the art neural DG methods by applying a common hyperparameter search strategy that aimed to find the best tuning (cf: Fig 1) for each competitor. They nevertheless found existing methods to perform unreliably. Our conjecture is that the number of seed samples (3) and hyper-parameter search trials (20) used was far too few to reliably tune the deep networks evaluated, and therefore the tuning of each model was inaccurate and performance estimates unreliable. Since dense and accurate hyper-parameter tuning for neural networks is more challenging than our linear models in Section 3.1, we take a different approach. Rather than trying to control complexity directly, we take trained DomainBed methods and *measure their complexity retrospectively*.

Setup To ensure that complexity can be measured accurately, we work with 2-layer MLPs. Specifically, we take fixed ImageNet pre-trained ResNet-18 features, and feed them to MLPs, which are then trained using the DomainBed framework. 2-layer MLPs are sufficient to instantiate many of the state of the art DG algorithms in DomainBed. We train and compare ERM, CORAL (Sun & Saenko, 2016), IRM (Arjovsky et al., 2019), Mixup (Wang et al., 2020), MMD (Li et al., 2018), RSC (Huang et al., 2020), SD (Pezeshki et al., 2020) and VRex (Krueger et al., 2020). We also report the results of an ERM model, checkpointed at a range of training iterations.

Measuring Model Complexity Because we have limited our attention to 2-layer MLPs, we can take advantage of a model capacity metric that is specialised for this class of models. We retrospectively determine the complexity of a trained network using the measure proposed by Neyshabur et al. (2019), who also use this measure to bound the Rademacher complexity of a 2-layer MLP hypothesis classes. More concretely, the expression used for computing complexity is

$$\mathcal{C}(f_{U,V}) = \|V\|_F(\|U - U^0\|_F + \|U^0\|_2), \tag{17}$$

where U is the weight matrix of the first layer, U^0 is its random initialisation, and V is the weight matrix of the second layer. We use $\|\cdot\|_F$ to denote Frobenius norm, and $\|\cdot\|_2$ to indicate the spectral

norm. Note that for simplicity we have omitted constant factors that depend only on the architecture and problem setting, and not the learned weights, as we use the same architecture for all methods we investigate.

Results The results in Figure 2 summarize the trade-off between measured model complexity (x-axis), and held-out domain test accuracy (y-axis), averaged over all choices of held out domain for each dataset (See Supplementary for full breakdown across all held-out domains). The top plot compares several of the published neural DG methods implemented in DomainBed. The main message of the top plot is that results are consistent with the hypothesis that the resulting complexity of the models trained by various DG methods is a key causal factor in domain generalisation accuracy (compare Fig. 2 with the clean result for linear where models we are able to intervene and control complexity directly in Fig. 1)—thus explaining the previously erratic behaviour observed in domain-bed.

To provide a different view of the same issue, the bottom plot reports a vanilla untuned ERM model check-pointed every 300 training iterations between (up to a total of 15,000) but expressed in the *same* complexity units as above. Because neural models can gain complexity with iterations (see, e.g., Prechelt (1998); Hardt et al. (2016)) we also see typical overfitting-underfitting trade-off curves. This shows that, as expected, proper choice of early stopping criterion is important. It also shows that over a similar dynamic range of complexity (0.5-3), ERM (below) and alternative models (above) span a similar dynamic range of accuracy (e.g., 6% for PACS). This suggests that complexity, as controlled by whatever mechanism, is a key determinant of DG performance.

How Do DG Models Control Complexity? By introducing different modifications to standard ERM, DG models explicitly or implicitly modify the bias and variance of the function class to be learned. Gulrajani & Lopez-Paz (2021) highlight neural models as being dependent on learning rate, batch size, dropout rate, and weight decay; with other factors being choice of optimizer and weight initializer, etc. For example, higher dropout rate and weight decay tend to reduce complexity, while some other factors influence complexity in a less transparent way. RSC introduce more sophisticated dropout-like mechanisms, which would be expected to reduce complexity. Meanwhile alignment-based methods like CORAL and MMD effectively add auxiliary losses, which will implicitly affect complexity, but are hard to explicitly link to it. Consistency based methods like IRM and VRex penalize loss variance, which also tends to reduce the generalisation gap in the single task case (Maurer & Pontil, 2009). The specific setting of all the corresponding hyper-parameters (e.g., regularizer strength, dropout rates) influence final model complexity, which we argue is the key determinant of performance.

3.3 Solving domain-bed with Linear models

Having demonstrated control of generalisation with linear model regularization (Sec. 3.1), and explained DomainBed performance post-hoc via measuring complexity of neural models (Sec. 3.2), we next evaluate algorithms for practical complexity tuning. From Fig. 1 we saw that within-domain and cross-domain evaluation have different optimal tuning. If we consider automating the search for a good regularization parameter, these two final evaluation criteria correspond respectively to hyper-parameter selection based on a validation set from the seen source domains, vs based on a validation set drawn from a held out source domain. The latter validation criterion corresponds to a unbiased estimator of expected cross-domain generalisation performance $R^{\mathcal{E}}(f)$, which our theorem bounds.

Setup We performed DG evaluation on DomainBed using ERM with linear models using the same DINO features as Section 3.1. For each held out domain, we performed either instance-wise or domain-wise cross-validation for hyperparameter tuning.

Results The results in Table 1 report the average accuracy across held out domains, and the average selected $\log C$ regularization strength (See Supplementary for the full breakdown of results for each held-out domain). We can see that the domain-wise cross-validation objective leads to similar or better accuracy, and similar or smaller C value selection (i.e., stronger regularization). This outcome is opposite to the observation made by Gulrajani & Lopez-Paz (2021). Given the theoretical support for our domain-wise objective, and our clear empirical outcome when freed of confounding factors such as stochasticity in neural network training, we consider our result to be decisive and attribute the different observation Gulrajani & Lopez-Paz (2021) to the inaccuracy and stochasticity of hyperparameter tuning in neural networks.

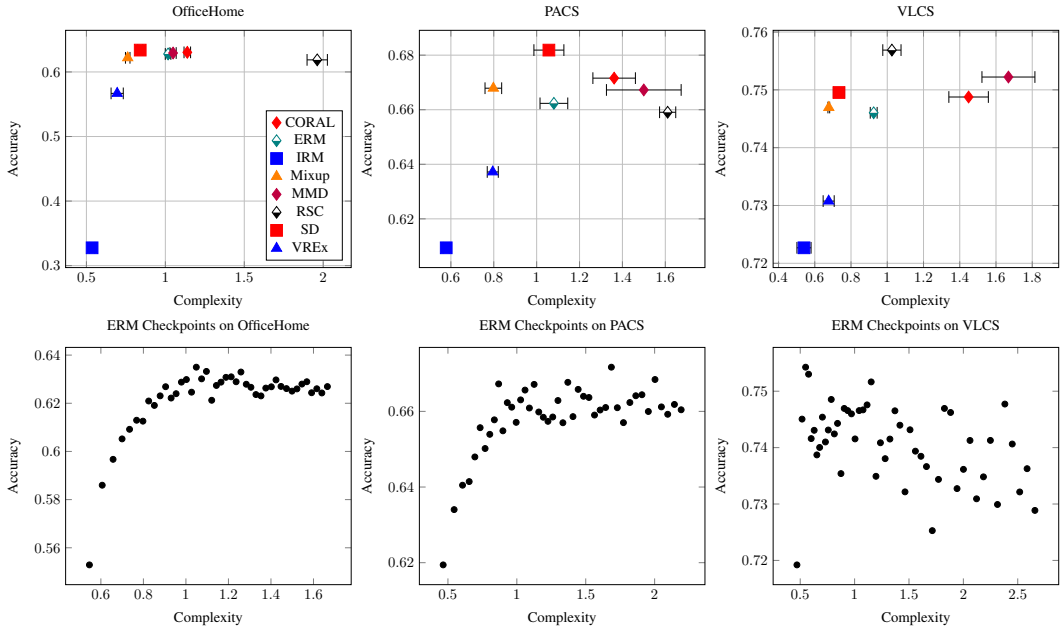

Figure 2: Performance of neural networks as a function of their measured model complexity after training using DomainBed (DB). The overall results are consistent with a standard bias-variance trade-off (cf. Fig 1 for linear models): performance depends on how well each model was tuned by DB's hyper-parameter search procedure. Top: Leave-one-domain-out cross-validation performance of various neural DG algorithms evaluated by DomainBed. Horizontal error bars correspond to the standard deviation of the model complexity measured for each cross-validation iteration. Bottom: Performance of ERM model checkpointed at different training iterations.

		Domain Wise		Instance Wise	
		Acc	$\log C$	Acc	$\log C$
Raw pixel	RotatedMNIST	71.6	4.97 ($\pm$0.52)	71.2	6.82 ($\pm$0.28)
DINO VIT-B8	PACS	74.3	-1.73 ($\pm$0.40)	70.6	1.56 ($\pm$0.66)
	VLCS	80.4	-1.04 ($\pm$0.69)	80.5	-0.87 ($\pm$0.35)
	OfficeHome	73.6	-0.17 ($\pm$0.35)	73.1	1.04 ($\pm$0.40)
	SVIRO	97.2	0.21 ($\pm$0.33)	95.3	4.85 ($\pm$0.46)
	TerraIncognita	45.4	-0.17 ($\pm$0.87)	38.8	5.37 ($\pm$0.35)
DINO VIT-S8	PACS	73.0	-1.91 ($\pm$1.43)	71.6	1.04 ($\pm$0.89)
	VLCS	80.4	-1.91 ($\pm$0.66)	80.6	-1.21 ($\pm$0.35)
	OfficeHome	72.2	-0.69 ($\pm$0.57)	72.3	0.35 ($\pm$0.40)
	SVIRO	94.5	-1.18 ($\pm$0.47)	91.3	5.68 ($\pm$0.97)
	TerraIncognita	37.1	0.00 ($\pm$1.50)	33.1	4.85 ($\pm$0.57)

Table 1: Comparing model selection criteria using LinearSVC. Accuracy and selected SVM regularization parameter 'C'. Domain-wise validation outperforms instance-wise and selects stronger regularization (lower C).

4 Related Work

Theoretical Analysis of the DG Setting and Algorithms The DG problem setting was first analysed in Blanchard et al. (2011). Since then there have been some attempts to analyse DG algorithms from a generalisation bound perspective (Muandet et al., 2013; Blanchard et al., 2021; Rosenfeld et al., 2021; Arjovsky et al., 2019; Hu et al., 2020). However these studies either have theoretical results restricted to shallow models – in contrast to our Rademacher complexity approach that can apply to neural networks (Section 3.2); or rely on an informal link between domain *adaptation*

objectives (where target domains are observable for alignment to source domains) and domain generalisation (where target domains are not observable and thus cannot correctly be used in a learning objective).

Empirical Analysis The main existing empirical analysis on DG is Gulrajani & Lopez-Paz (2021), who compared several state of the art DG methods under a common evaluation and hyper-parameter tuning protocol called DomainBed. They ultimately defend Empirical Risk Minimization (ERM) over more sophisticated alternatives on the grounds that no competitor consistently beats it across the benchmark suite. We also broadly defend ERM, and build on the same benchmark, but differently we provide a much deeper analysis into when and why ERM works. More specifically: (i) We provide a new theoretical analysis of ERM's generalisation quality unlike the prior purely empirical evaluation, (ii) We re-use the DomainBed benchmark to directly corroborate this theory under controlled conditions using linear models where model complexity can be tractably and accurately tuned. (iii) We use our complexity-based analysis to explain the previously erratic results of prior DomainBed competitors in terms of model complexity. (iv) We identify, and empirically validate, the preferred model selection criterion for DG, a point which was inconclusive in Gulrajani & Lopez-Paz (2021).

4.1 Discussion

At the start of this section we set out to address three questions, which can be answered as: (1) The dependence of cross-domain generalisation on complexity can be directly and precisely when using linear models. (2) The erratic performance of state of the art methods in DomainBed can be largely explained in terms of implied model complexity. (3) If the goal is to optimize for domain generalisation, then then domain-wise validation is preferred to instance-wise validation as a model-selection objective.

Our analysis suggests that several existing methods that tried to tune learnable DG hyper-parameters by performance on a held-out domain (Balaji et al., 2018; Li et al., 2019) were broadly on the right track, and concurs with Gulrajani & Lopez-Paz (2021) that those methods with underspecified hyper-parameter and model selection procedures are unhelpful. However, given that most neural methods have many more complexity hyperparameters than the single hyperparameter that we were able to carefully control for linear models, obtaining accurate tuning and reliable performance evaluation is likely to be a challenge. Gradient-based hyper-parameter estimation methods, as initially attempted in Balaji et al. (2018); Li et al. (2019), together with efficient methods for long-inner loop hyper-gradient calculation (Lorraine et al., 2020), may benefit the former problem by making search in larger numbers of hyperparameters more feasible. Alternatively, using general purpose pre-trained features (Caron et al., 2021) as we did here, and focusing on learning shallow models that can be accurately tuned for DG may be another promising avenue in practice. Although achieving state of the art performance is not our focus, we note that our results in Table 1 are quite competitive with end-to-end trained state of the art (Gulrajani & Lopez-Paz, 2021), despite using fixed features and shallow models.

5 Conclusion

In this paper we explained the performance of domain generalisation methods in terms of model complexity (bias-variance trade-off) from both theoretical and empirical perspectives. Both perspectives show that complexity impacts cross-domain generalisation in a way that closely mirrors the bias-variance trade-off in conventional supervised learning – but where stronger regularisation is required if optimising for cross-domain generalisation than if optimising for conventional within-domain generalisation. We clarified the preferred model selection criterion in each case. This analysis demystifies the problem being posed by the DG problem setting, why existing algorithms succeed or fail to work, and sets the bar for future theoretical theoretical studies to surpass in terms of convergence rates, as well as for future empirical studies to surpass in terms of strong baselines.

References

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv*, 2019.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR*, 2002.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *NeurIPS*, 2017.

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, 2018.

Mikhail Belkin, Daniel J Hsu, and Partha Mitra. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. *NeurIPS*, 2018.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *NIPS*, 2011.

Gilles Blanchard, Aniket Anand Deshmukh, Ürün Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *JMLR*, 2021.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

Gabriela Csurka. *Domain Adaptation in Computer Vision Applications*. Springer, 2017.

Steve Dias Da Cruz, Bertram Taetz, Thomas Stifter, and Didier Stricker. Illumination normalization by partially impossible encoder-decoder cost function. In *WACV*, 2021.

Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *CVPR*, 2013.

Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 1992.

Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015.

Henry Gouk, Timothy Hospedales, et al. Distance-based regularisation of deep networks for fine-tuning. In *ICLR*, 2021.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *ICLR*, 2016.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

Shoubo Hu, Kun Zhang, Zhitang Chen, and Laiwan Chan. Domain generalization via multidomain discriminant analysis. In *UAI*, 2020.

Zeyi Huang, Haohan Wang, and Eric P Xing. Self-challenging improves cross-domain generalization. In *ECCV*, 2020.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts. In *ICML*, 2021.

David Krueger, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Rémi Le Priol, and Aaron C. Courville. Out-of-distribution generalization via risk extrapolation (rex). *CoRR*, abs/2003.00688, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018.

Yiying Li, Yongxin Yang, Wei Zhou, and Timothy M. Hospedales. Feature-critic networks for heterogeneous domain generalization. In *ICML*, 2019.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. In *COLT*, 2009.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. In *ICLR*, 2019.

Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*, 2020.

Massimiliano Pontil and Andreas Maurer. Excess risk bounds for multitask learning with trace norm regularization. In *COLT*, 2013.

Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pp. 55–69. Springer, 1998.

Elan Rosenfeld, Pradeep Kumar Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. In *ICLR*, 2021.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, 2016.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017.

Yufei Wang, Haoliang Li, and Alex C Kot. Heterogeneous domain generalization via domain mixup. In *ICASSP*, 2020.

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey, 2021.

A PROOF OF THEOREM 2

Proof. We begin with the expected excess risk, which can be bounded from above by

$$\mathbb{E}_S[R^{\mathcal{E}}(\hat{f}) - R^{\mathcal{E}}(f^*)] = \mathbb{E}_S[R^{\mathcal{E}}(\hat{f}) - \hat{R}^{\mathcal{E}}(\hat{f})] + \mathbb{E}_S[\hat{R}^{\mathcal{E}}(\hat{f}) - \hat{R}^{\mathcal{E}}(f^*)] + \mathbb{E}_S[\hat{R}^{\mathcal{E}}(f^*) - R^{\mathcal{E}}(f^*)] \tag{18}$$

$$\leq \mathbb{E}_S[R^{\mathcal{E}}(\hat{f}) - \hat{R}^{\mathcal{E}}(\hat{f})] + \mathbb{E}_S[\hat{R}^{\mathcal{E}}(f^*) - R^{\mathcal{E}}(f^*)] \tag{19}$$

$$= \mathbb{E}_S[R^{\mathcal{E}}(\hat{f}) - \hat{R}^{\mathcal{E}}(\hat{f})], \tag{20}$$

where the inequality arises because, by definition, the *empirical* risk of $\hat{f}$ must be less than or equal to the empirical risk of the optimal model. The second inequality comes from the fact that f^* is determined independently of the particular training set that we sample, so $\mathbb{E}_S[\hat{R}^{\mathcal{E}}(f^*)] = R^{\mathcal{E}}(f^*)$. Applying McDiarmid's inequality completes the proof. □

B DETAILED RESULTS

Complexity Measurement for DomainBed Competitors In Figure 2 we illustrated the performance of DomainBed competitors in terms of their resulting model complexity by averaging over held-out domains for each dataset. In Figure 3, we present detailed results broken down by each held out dataset.

Detailed results on each DG benchmark In Table 1 we reported results of our linear model competitors on DomainBed summarized over choice of held out domain. In Tables 2-7, we report the detailed results of accuracy and choice of C paramater for each held out domain.

Cross Validation		0	15	30	45	60	75	Ave.
Raw pixel								
Domain Wise	Acc	0.577	0.785	0.799	0.783	0.769	0.586	0.716 (0.096)
	C	256.0	128.0	128.0	128.0	64.0	256.0	
Instance Wise	Acc	0.582	0.774	0.788	0.783	0.765	0.579	0.712 (0.093)
	C	1024.0	512.0	1024.0	1024.0	1024.0	1024.0	

Table 2: Accuracy and selected 'C' on Rotated-MNIST, using LinearSVC.

Cross Validation		A	C	P	S	Ave.
DINO ViT-B8						
Domain Wise	Acc	0.877	0.676	0.971	0.446	0.743 (0.201)
	C	0.25	0.125	0.25	0.125	
Instance Wise	Acc	0.866	0.620	0.891	0.445	0.706 (0.184)
	C	4.0	8.0	8.0	2.0	
DINO ViTS8						
Domain Wise	Acc	0.864	0.666	0.948	0.441	0.730 (0.196)
	C	0.5	0.0625	0.5	0.03125	
Instance Wise	Acc	0.867	0.676	0.858	0.464	0.716 (0.164)
	C	2.0	4.0	8.0	1.0	

Table 3: Accuracy and selected 'C' on PACS, using LinearSVC.

Cross Validation		C	L	S	V	Ave.
DINO ViT-B8						
Domain Wise	Acc	0.972	0.648	0.787	0.811	0.804 (0.115)
	C	0.5	0.5	0.125	0.5	
Instance Wise	Acc	0.977	0.648	0.785	0.811	0.805 (0.117)
	C	0.25	0.5	0.5	0.5	
DINO ViTS8						
Domain Wise	Acc	0.959	0.648	0.785	0.823	0.804 (0.111)
	C	0.25	0.125	0.0625	0.25	
Instance Wise	Acc	0.959	0.651	0.789	0.823	0.806 (0.109)
	C	0.25	0.5	0.25	0.25	

Table 4: Accuracy and selected 'C' on VLCS, using LinearSVC.

Cross Validation		A	C	P	R	Ave.
DINO ViT-B8						
Domain Wise	Acc	0.748	0.512	0.832	0.850	0.736 (0.135)
	C	1.0	1.0	1.0	0.5	
Instance Wise	Acc	0.730	0.514	0.833	0.846	0.731 (0.133)
	C	4.0	2.0	2.0	4.0	
DINO ViTS8						
Domain Wise	Acc	0.740	0.496	0.820	0.834	0.722 (0.136)
	C	1.0	0.25	0.5	0.5	
Instance Wise	Acc	0.726	0.508	0.824	0.833	0.723 (0.131)
	C	2.0	1.0	1.0	2.0	

Table 5: Accuracy and selected 'C' on OfficeHome, using LinearSVC.

Cross Validation		aclass	escape	hilux	i3	lexus	tesla	tiguan	tucson	x5	zoe	Ave.
DINO ViT-B8												
Domain Wise	Acc	0.979	0.980	0.967	0.997	0.959	0.981	0.942	0.986	0.975	0.954	0.972 (0.016)
	C	1.0	1.0	1.0	1.0	2.0	1.0	2.0	1.0	1.0	2.0	
Instance Wise	Acc	0.922	0.991	0.910	0.998	0.902	0.960	0.937	0.983	0.976	0.955	0.953 (0.032)
	C	128.0	64.0	128.0	64.0	256.0	128.0	256.0	128.0	128.0	128.0	
DINO ViTS8												
Domain Wise	Acc	0.961	0.959	0.959	0.971	0.958	0.964	0.933	0.979	0.957	0.814	0.945 (0.045)
	C	0.125	0.5	0.25	0.5	0.5	0.5	0.25	0.25	0.25	0.25	
Instance Wise	Acc	0.914	0.969	0.952	0.996	0.966	0.838	0.843	0.989	0.810	0.851	0.913 (0.067)
	C	1024.0	128.0	512.0	256.0	1024.0	64.0	512.0	128.0	512.0	128.0	

Table 6: Accuracy and selected 'C' on SVIRO, using LinearSVC.

Cross Validation		L100	L38	L43	L46	Ave.
DINO ViT-B8						
Domain Wise	Acc	0.444	0.457	0.516	0.399	0.454 (0.041)
	C	0.25	1.0	1.0	2.0	
Instance Wise	Acc	0.472	0.391	0.366	0.324	0.388 (0.054)
	C	256.0	128.0	256.0	256.0	
DINO ViTS8						
Domain Wise	Acc	0.375	0.231	0.480	0.398	0.371 (0.090)
	C	0.125	1.0	2.0	4.0	
Instance Wise	Acc	0.407	0.157	0.438	0.322	0.331 (0.109)
	C	256.0	64.0	128.0	128.0	

Table 7: Accuracy and selected 'C' on TerraIncognita, using LinearSVC.

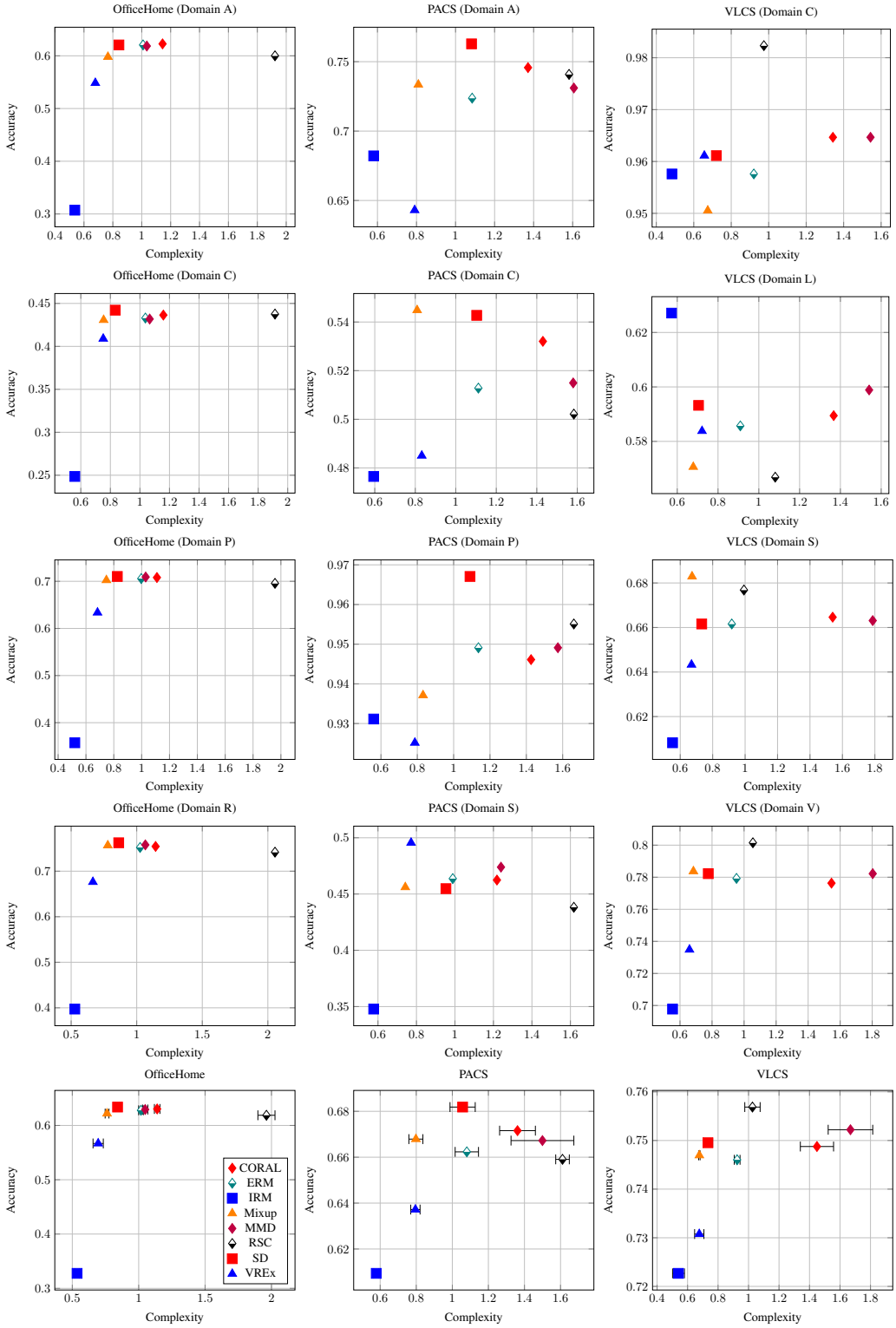

Figure 3: Performance of various neural network DG algorithms as a function of their measured model complexity after training using DomainBed. Breakdown by held-out dataset. The top four rows correspond to individual domains, the bottom row is the leave-one-domain-out cross-validation estimate of the DG performance, and horizontal error bars correspond to the standard deviation of the model complexities measured in each iteration of cross-validation.