Deep Neural Networks for Genomic Prediction Do Not Estimate Marker Effects

Jordan Ubbens^{1,✉}, Isobel Parkin², Christina Eynck², Ian Stavness^{1,3}, and Andrew Sharpe¹

¹Global Institute for Food Security (GIFS), University of Saskatchewan, Saskatoon, SK S7N 0W9, Canada

²Agriculture and Agri-Food Canada, Saskatoon, SK S7N 0X2, Canada

³Department of Computer Science, University of Saskatchewan, Saskatoon, SK S7N 0W9, Canada

Genomic prediction is a promising technology for advancing both plant and animal breeding, with many different prediction models evaluated in the literature. It has been suggested that the ability of powerful nonlinear models such as deep neural networks to capture complex epistatic effects between markers offers advantages for genomic prediction. However, these methods tend not to outperform classical linear methods, leaving it an open question why this capacity to model nonlinear effects does not seem to result in better predictive capability. In this work, we propose the theory that, due to a principle called shortcut learning, deep neural networks tend to base their predictions on overall genetic relatedness, rather than on the effects of particular markers, such as epistatic effects. Using several datasets of crop plants (lentil, wheat, and *Brassica carinata*), we demonstrate the network's indifference to the values of the markers by showing that the same network, provided with only the locations of matches between markers for two individuals, is able to perform prediction to the same level of accuracy.

genomic prediction | genomic selection | machine learning
Correspondence: jordan.ubbens@gifs.ca

Introduction

Genomic Prediction (GP) is the practice of predicting trait values, such as yield, for members of a population using marker data. In the breeding of crop plants, GP has been shown to be an effective way to speed the rate of improvement for agronomically relevant traits (1–3). GP has become increasingly accessible in recent times due to rapid improvements in genotyping technologies using next generation sequencing at a much reduced cost, and has been integrated into many breeding programs around the world (4).

The literature on GP in plants and animals is mature, with linear additive models of genetic effects being the most popular in practice. However, recent years have seen an increasing interest in using machine learning methods, including deep learning, for the GP task. Although it is widely assumed that deep neural networks will learn to model complex nonlinear epistatic effects (5), they rarely perform better than simple linear models such as the *Genomic Best Linear Unbiased Predictor* (GBLUP) (6). This implies that deep neural networks either do not estimate the effects of specific alleles, or do not benefit from doing so.

In this work, we explore the gap between the capacity and the real-world performance of deep learning methods for GP by controlling what information the network is allowed to access. We develop a novel technique for training a deep neural network which performs prediction based on the locations of

Fig. 1. Overview of the proposed method for training a deep neural network without giving the network access to the marker data. Inset, the DeepGS architecture (5) is shown as the feature extractor module, and a two-layer fully-connected network is shown as the estimator module.

matches between markers for pairs of individuals, without giving the network access to the marker data directly. Under the reasonable assumption that a deep neural network estimates marker effects, this training strategy is expected to perform poorly as the network has no information about which alleles are present in any individual. However, through experiments with different plant species and population structures, we show that this prediction strategy, based solely on genetic relatedness, performs just as well as allowing the network to access the values of the markers. The equivalence in predictive performance for deep neural networks with or without access to the marker data implies that these networks tend to rely on estimating overall genetic relatedness, not the effects of specific markers.

Background

Kernel Methods. The term *kernel method* refers to a class of models which rely on a fixed or learnable kernel function K which relates a notion of “similarity” between its two inputs. Kernel machines are a common example of kernel methods in machine learning. In the simplest terms, kernel machines are models in the form

$$y = g \left(\sum_i a_i y_i K(x, x_i) + b \right) \quad (1)$$

where x and y are the input and output of the model respectively, $(x_i, y_i), i \in \{0 \dots N\}$ are the N samples of the training set (the data vectors and the labels), g is an optional transfer function, and $a_i, i \in \{0 \dots N\}$ and b are learnable weight

and bias terms. In this way, the output of the kernel machine for an input is a function of the weighted similarity between it and the training samples, as indicated by the kernel function K . Kernel functions have classically been chosen based on expert knowledge of the data or the problem space, for example, using Gaussian or polynomial kernels.

In GP, the *Genomic Best Linear Unbiased Predictor*, or *GBLUP*, is a common estimator of the form

$$BLUP(\hat{g}) = \left[I + \left(X X^T \right)^{-1} \frac{\sigma_e^2}{\sigma_g^2} \right] y \quad (2)$$

where X is the (mean-centered and normalized) matrix of markers, and σ_e and σ_g are the standard deviation of errors and genetic effects, respectively (7). Here, the covariance matrix $X X^T$ represents a simple linear kernel, capturing linear additive effects among markers. Nonlinear kernels can also be used in the estimation of genetic relatedness. *Reproducing Kernel Hilbert Spaces* (RKHS) regression with a Gaussian kernel has seen some success in the literature, although linear kernels still remain the most popular (4).

Neural Networks and Deep Learning. Nonlinear models such as neural networks have the theoretical capability to capture arbitrarily complex epistatic effects in whole-genome regression. For example, multiple markers may have negligible effects individually, but significant effects when considered in combination. The ability to model these interactions gives nonlinear models a potential representational advantage over linear models for prediction. Several authors have suggested that this advantage makes deep neural networks promising candidates for performing GP (5, 8, 9).

Machine learning methods for GP, including deep learning methods, have been explored in the literature. The results have been mixed, with the performance of deep learning varying between datasets. One of the most commonly used neural network architectures is the Convolutional Neural Network (CNN), which uses layers of filter banks to capture local patterns in the data. Despite their representational power and their success in other subfields of biology, CNNs have been shown to perform similarly to classical linear models in some cases, although somewhat under-performing them in others (6). One extensive survey of GP methods for various traits in six different plant species showed that CNNs performed the worst among all tested models on average (10). Another study in soybean showed similar performance between deep learning and linear models when missing SNPs were imputed (11). Results for blueberry and strawberry again saw no advantage for CNNs over conventional methods for most traits (12).

The similarity in performance between powerful, high-capacity deep learning models and standard linear models suggests that either the genetic architectures of all the examined traits are strongly linear additive in nature, or that deep neural networks are not estimating nonlinear effects between markers despite their ability to do so.

Methods and Materials

Obscuring Marker Data from the Network. Due to the black-box nature of deep neural networks, it is difficult to determine precisely how the network arrives at a prediction (13). Therefore, the most reliable way to determine whether the network benefits from estimating the effects of particular alleles is to completely disallow the network from accessing the values of the markers themselves.

To investigate whether deep neural networks estimate phenotype based on estimating marker effects, we propose a novel training method which obscures the marker data from the network. A schematic of the proposed method is shown in Figure 1. Comparing to the kernel machine formulated in Equation 1, the proposed model can be represented as

$$y = \mathbb{E} [f_{est} (f_{emb} (\phi (x, x_i)), y_i)] \quad (3)$$

where ϕ is the matching filter operation

$$\phi(u, v) = \begin{cases} 1 & \text{if } u_j = v_j, \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

for each marker j , f_{emb} is the *feature extractor module* and f_{est} is the *estimator module*. The matching filter only outputs one when the encodings match exactly, so comparing heterozygous and homozygous individuals will produce a zero. Here, the subscripted (x_i, y_i) is a member of the training population, called the *reference sample*, and (x, y) is the sample x for which the phenotype y is being predicted, called the *query sample*. In a manner similar to kernel machines, the final point estimate produced for a query sample is an aggregate of predictions made over all members of the training population.

During the training phase, the feature extractor module and the estimator module are trained in an end-to-end fashion using random pairs of samples from the training population. The training samples are retained for inference time, during which a forward pass is performed on each query sample using each member of the training population as the reference sample. Because the output is a distribution over predictions, the point estimate is given by the mean of this output distribution. Various other aggregation methods such as ensemble learning and mixture-of-experts models (14) were trialed, with no consistent improvement over using the simple mean. In practical terms, any nonlinear models can be used for the feature extractor module and for the estimator module. Here, we use a simple two-layer feed-forward neural network for the estimator, and we use the DeepGS convolutional neural network architecture from the literature for the feature extractor module (5). DeepGS is a small convolutional architecture, using a single convolutional layer and two fully connected layers. It uses stochastic regularization in the form of Dropout (15). DeepGS was selected as it is the most commonly used example of a deep network built specifically for GP. We use DeepGS here as this allows us to make direct comparisons to the DeepGS architecture used in a whole-genome regression approach, as differences in performance

are therefore not due to differences in architecture or model capacity.

The proposed method is distinct from RKHS methods in the fact that Equation 3 is not a reproducing kernel K (it is not guaranteed that $K: X \times X \rightarrow \mathbb{R}$ is positive semi-definite). It is also distinct from standard whole-genome regression methods, as the network never has access to the marker data. It is only provided with the locations of matches between two otherwise anonymous samples, and the specific alleles of both samples are unknown. This means that the network cannot reduce to existing whole-genome regression methods by regressing on the markers of the query sample while ignoring the reference sample.

Models Evaluated. To test our hypothesis, we compare the proposed method, with DeepGS as the feature extractor, to DeepGS used in the typical fashion, performing whole-genome regression with direct access to the marker data. For completeness, we also compare these networks to two common linear models, GBLUP and LASSO, as well as two other machine learning methods: a fully-connected neural network (FC), and a fully-connected neural network with L_0 regularization (16).

LASSO is a penalized linear model of the form

$$\arg \min_w \left(y - w^T x \right)^2 + \lambda \|w\|_1 \quad (5)$$

where w are the model parameters and λ is a tuneable constant which controls the strength of the regularization term. In practice, the L_1 penalty on the weight vector tends to encourage sparsity in the weights, a desirable quality under the assumption that a low proportion of the total markers have influence on the trait. LASSO and a related model, the Elastic Net, performed the best (and better than the commonly used RR-BLUP) in a survey of penalized linear models for GP (17).

Fully-connected neural networks, also known in some GP literature as Multi-Layer Perceptrons (MLPs), are a common choice for machine learning applications in GP. We use a standard two-layer neural network. Batch normalization is used, as it potentially improves generalization performance and allows training to succeed over a higher range of learning rates.

Motivated by the principle of sparsity, we also investigate a fully-connected neural network equipped with L_0 regularization (16). While L_1 regularization penalizes the absolute magnitude of the weights, L_0 regularization penalizes the proportion of non-zero weights, explicitly encouraging sparsity. Optimizing this regularization term using gradient descent methods is impossible, because the L_0 norm is not differentiable. However, it can be approximated through a *hard concrete distribution* - a continuous relaxation of discrete random variables (18, 19). The result is differentiable feature selection, learned simultaneously with the model (20).

Hyperparameter Selection. In order to fairly represent the performance of each method, care must be taken to tune them

individually for optimum performance. In the machine learning literature, constants used in the model specification or during training, such as the learning rate or the size in units of a hidden layer, are referred to as *hyperparameters*. The process of selecting the values for hyperparameters which are likely to result in the best performance on the test data is known as *hyperparameter tuning*. This is done by holding out a portion of the training data, called the *validation set*. Performance on the validation set can be used as a proxy for performance on the unseen test data, allowing an automated search over values of the hyperparameters in either a random or a structured fashion. In this section we describe how hyperparameters were selected for each of the six methods.

For the LASSO model, the regularization coefficient $\lambda = 0.001$ is used, as this value allows convergence in every dataset. Tuning λ resulted in slightly worse performance overall, and the bounds for tuning must be set carefully or values for λ may be selected which cause non-convergence when training the final model. For the FC and L_0 models, the number of hidden units and the learning rate are determined via hyperparameter search as this consistently improved performance over predetermined constants. We use an exponentially decaying temperature schedule for the L_0 model (20). For the DeepGS network, most hyperparameters are prescribed by the authors of the architecture. We use a learning rate of 1×10^{-5} . Tuning the learning rate does not improve performance, and sometimes results in learning rates which are infeasible for training the final model.

For the proposed model, hyperparameters were predetermined as hyperparameter tuning proved to be both costly and the performance gains inconsistent. We use a value of $d = 1$ for the dimensionality of the embedding emitted by the feature extractor module. Smaller values of d tend to be better for small datasets, and so we find this to be the most general. The number of hidden units in the estimator module is set to 32, as tuning did not change performance significantly.

In addition to hyperparameter tuning, the validation set can also be used for *early stopping*. Some models may see a decrease in error on the validation set until a certain number of training epochs, and increasing error thereafter. Halting training at this inflection point can prevent the model from overfitting the training data. However, in all experiments presented here, we found that it was universally better to use all of the training data to train the model, instead of reserving a portion as a validation set to monitor for early stopping. This is consistent with previous reports that the size of the training population is one of the most important factors in predictive performance (9). For all models, we add the validation set back in to the training data after hyperparameter tuning (when applied) and use the entirety to train the final model. Each model was trained until the training error had ceased to decrease.

Datasets. In order to evaluate the difference between deep neural networks trained with and without access to the raw marker values, we use four datasets of food crops, representing a variety of species and sample sizes. Sources for the data are provided in the final section.

The first two datasets we use for evaluation are published datasets of lentil (*Lens culinaris*) (21). The authors of this study examined GP performance within-population, across-populations, and across-environments using a variety of models. The authors published a lentil diversity panel (LDP) made up of 320 individuals, as well as two biparental populations, termed LR-01 and LR-11. To evaluate the largest and most genetically diverse population possible, we use the combined LR-01 and LR-11 dataset from the paper, constituting a total of 230 individuals. This data is an amalgamation of two different biparental populations, and the authors of the data reported lower predictive performance on average using the two populations combined as opposed to using each individually. This makes an interesting case study, given the unusual population structure in this challenging combined population. The 39k SNPs sampled via exome capture provided by the authors are used.

We also use a population of 2,536 *Brassica carinata* plants. This population is a nested association mapping (NAM) population consisting of recombinant inbred line (RIL) progeny from 50 crosses of different *B. carinata* lines with a common reference line. This dataset includes 17k SNPs.

For the final crop dataset, we use a published dataset of 2,403 landrace accessions of Iranian wheat (*Triticum aestivum*) (22). The individuals were phenotyped for a variety of agronomically relevant traits and the authors of the dataset used these for a GP study with standard models. This is the dataset previously reported in the DeepGS paper, and so we include it in order to be consistent with the analysis from that publication (5). The dataset includes a set of 33k SNPs obtained via genotyping by sequencing (GBS).

Each dataset was split into five folds, and each model was trained five times using each fold as the hold-out set. Hyperparameter tuning was performed within each split when applicable, using one of the four training folds as the validation set.

Results

Prediction results for the lentil diversity panel, lentil biparental population, carinata, and wheat datasets are shown in Figure 2, Figure 3, Figure 4, and Figure 5, respectively.

As is usual in GP, we report Pearson's r between the predicted and actual phenotypes as the measure of predictive performance. A property of r which should be mentioned is the distribution of r values for two independent random variables. For samples drawn from two independent Gaussian distributions, the probability density for r is given by

$$f(x) = \frac{(1-x^2)^{\frac{N-4}{2}}}{B(0.5, \frac{N-2}{2})} \quad (6)$$

where B is the beta function and N is the sample size. Clearly, for small values of N , the expected variance in r values is high. We see this difference in variance between the biparental lentil dataset, which is the smallest in sample size, and the carinata and wheat datasets, which are significantly

larger. In practice, we found r values to be volatile, varying considerably between different random splits of the data, and different random initializations. Even multiple runs with identical random seeds produce slightly different results for the machine learning models based on non-deterministic effects alone. For this reason, we decline to describe a method as superior when the difference in r is small, taking sample size into account. Due to the volatile and opaque nature of r as an estimate of predictive performance (for example, Figure 6), we also provide scatter plots of the results as supplementary figures.

Fig. 6. Example of scatter plots for the lentil diversity panel dataset using GBLUP (left) and FC (right). Although r is similar (0.76 and 0.78), the shape is different.

For all experiments, all six methods seem to provide similar predictive performance, with no method consistently outside the margin of error. The exception is LASSO, for which we observe a small decrease in performance on the LDP and wheat datasets relative to the other methods. LASSO also exhibits more volatile performance than the other methods across the experiments, likely due to the difficulties of training penalized linear models and their dependence on the strength of the regularization term. Among all the methods investigated in our evaluations, GBLUP continues to prove itself as a flexible choice when the underlying genetic structure of the trait is unknown. Although it is only capable of modeling linear additive effects with constant variance, GBLUP showed strong performance across species and traits. Despite this relatively good predictive performance, it remains the simplest method to implement, requires no tuning, and terminates in considerably less time than any of the machine learning methods.

Overall, our results largely mirror the findings elsewhere in the literature – for sufficiently large datasets, most methods offer comparable predictive performance regardless of differences in model capacity or complexity. Most importantly, the proposed method did not perform significantly worse than the standard DeepGS model on any dataset, even though it had no access to the marker data.

Discussion

On Shortcut Learning. It is reasonable to assume that, having the capacity to model complex epistatic effects between markers, deep neural networks would use this capability during training. However, deep neural networks are known to rely on features of the input which happen to correlate with the cost function, but which are not the features the network was intended to attend to. This phenomenon has been termed *shortcut learning*¹ (23). For example, it has been shown that

¹This is distinct from the use of the term *shortcut* to refer to skip connections in some neural network architectures.

Fig. 2. Genomic prediction results for the lentil diversity panel. Additional details are shown in Figure S1.

Fig. 3. Genomic prediction results for lentil biparental populations. Additional details are shown in Figure S2.

Fig. 4. Genomic prediction results for *B. carinata*. Additional details are shown in Figure S3.

Fig. 5. Genomic prediction results for wheat. Additional details are shown in Figure S4.

neural networks attend to color and texture more than the form of objects when performing image classification. Learning these simpler, higher-level features, or shortcuts, provides the network with a path of least resistance during optimization. The network does not need to learn a complicated representation of “cow” if the presence of cows is correlated with green grass in the image – looking for green pixels in the lower half of the image will suffice (23).

The shortcut learning principle may be true of genomic data as well. The network may find it unnecessary to seek out a small number of quantitative trait nucleotides (QTN), which are likely to be associated with a small and noisy gradient signal, when high performing individuals are likely to share large haplotype blocks in common by virtue of being genetically related. From the perspective of shortcut learning, it is not unlikely that the indifference of the network to the markers is due to the fact that it considers high-level genetic relatedness as a viable shortcut for predicting the phenotype.

Limitations. One study reported that for simulated non-additive genetic architectures, neural networks (FC networks and CNNs) performed slightly better than GBLUP for a 100-QTN simulated trait, albeit slightly worse for a 1000-QTN simulated trait (9). This could be taken as evidence to suggest that there may be some nonlinear genetic architectures where deep neural networks are advantaged due to their ability to estimate nonlinear effects. However, it is worth noting that, for practically all real-world datasets (in this study, in (9), as well as elsewhere in the literature (6, 10–12)), this difference between neural networks and linear models is not observed. This discrepancy may be explained by differences in marker effect sizes between real and simulated traits, or how biologically plausible the architecture of the simulated trait was in practice. These issues could have an effect on the behavior of the network in terms of which features are salient for prediction. Finally, these small performance differences could also be accounted for by, as we’ve observed, volatility in Pearson’s r .

Conclusion

Compared to linear models, deep neural networks have the ability to model complex nonlinear effects – however, this capability does not seem to benefit them in GP. In this paper, we show that one can obscure the marker data from the network, blinding it to which alleles are present in each individual – however, doing this does not seem to be at all deleterious to performance. Taken together, these two observations offer evidence that deep neural networks generally do not learn to model epistatic effects at all, but rather depend on estimating overall genetic relatedness. We suggest that that this is consistent with the behavior of neural networks in other domains, and is explained by a concept called shortcut learning. This work reveals a clear direction for future work in deep learning for GP. The development of models which are capable of avoiding the learning of shortcuts in genomic data may unlock some of the potential of these powerful nonlinear models in genomic prediction.

Data Availability

The data used in the wheat and lentil experiments are available from the sources referenced by their authors (21, 22). The carinata data is available on request.

ACKNOWLEDGEMENTS

We thank Kirstin Bett for productive conversations about the lentil data. We also thank Mitchell Feldmann for his comments on the manuscript.

Bibliography

1. ME Goddard and BJ Hayes. Genomic selection. *J. Anim. Breed. Genet.*, 124(6):323–330, 2007.
2. Michael E Goddard, Naomi R Wray, Klara Verbyla, Peter M Visscher, et al. Estimating effects and making predictions from genome-wide marker data. *Stat. Sci.*, 24(4):517–529, 2009.
3. José Crossa, Gustavo de los Campos, Paulino Pérez, Daniel Gianola, Juan Burgueño, José Luis Araus, Dan Makumbi, Ravi P Singh, Susanne Dreisigacker, Jianbing Yan, et al. Prediction of genetic values of quantitative traits in plant breeding using pedigree and molecular markers. *Genetics*, 186(2):713–724, 2010.
4. José Crossa, Paulino Pérez-Rodríguez, Jaime Cuevas, Osval Montesinos-López, Diego Jarquín, Gustavo de los Campos, Juan Burgueño, Juan M. González-Camacho, Sergio Pérez-Elizalde, Yoseph Beyene, Susanne Dreisigacker, Ravi Singh, Xuecai Zhang, Manje Gowda, Manish Roorkiwal, Jessica Rutkoski, and Rajeev K. Varshney. Genomic Selection

- in Plant Breeding: Methods, Models, and Perspectives. *Trends in Plant Science*, 22(11): 961–975, 2017. ISSN 13601385. doi: 10.1016/j.tplants.2017.08.011.
5. Wenlong Ma, Zhixu Qiu, Jie Song, Jiajia Li, Qian Cheng, Jingjing Zhai, and Chuang Ma. A deep convolutional neural network approach for predicting phenotypes from genotypes. *Planta*, 248(5):1307–1318, 2018. ISSN 14322048. doi: 10.1007/s00425-018-2976-9.
 6. Osval Antonio Montesinos-López, Abelardo Montesinos-López, Paulino Pérez-Rodríguez, José Alberto Barrón-López, Johannes WR Martini, Silvia Berenice Fajardo-Flores, Laura S Gaytan-Lugo, Pedro C Santana-Mancilla, and José Crossa. A review of deep learning applications for genomic selection. *BMC genomics*, 22(1):1–23, 2021.
 7. Gota Morota, Daniel Gianola, Eduardo Manfredi, and Paola Sebastiani. Kernel-based whole-genome prediction of complex traits: a review. *Frontiers in genetics*, 5:363, 2014. doi: 10.3389/fgene.2014.00363.
 8. Guillaume P. Ramstein, Sarah E. Jensen, and Edward S. Buckler. Breaking the curse of dimensionality to identify causal variants in Breeding 4. *Theoretical and Applied Genetics*, 132(3):559–567, 2019. ISSN 00405752. doi: 10.1007/s00122-018-3267-3.
 9. Rostam Abdollahi-Arpanahi, Daniel Gianola, and Francisco Peñagaricano. Deep learning versus parametric and ensemble methods for genomic prediction of complex phenotypes. *Genetics Selection Evolution*, 52(1):1–15, 2020. ISSN 12979686. doi: 10.1186/s12711-020-00531-z.
 10. Christina B. Azodi, Emily Bolger, Andrew McCarren, Mark Roantree, Gustavo de los Campos, and Shin Han Shiu. Benchmarking parametric and machine learning models for genomic prediction of complex traits. *G3: Genes, Genomes, Genetics*, 9(11):3691–3702, 2019. ISSN 21601836. doi: 10.1534/g3.119.400498.
 11. Yang Liu, Duolin Wang, Fei He, Juexin Wang, Tupti Joshi, and Dong Xu. Phenotype Prediction and Genome-Wide Association Study Using Deep Convolutional Neural Network of Soybean. *Frontiers in Genetics*, 10(November):1–10, 2019. ISSN 16648021. doi: 10.3389/fgene.2019.01091.
 12. Laura M. Zingaretti, Salvador Alejandro Gezan, Luis Felipe V. Ferrão, Luis F. Osorio, Amparo Monfort, Patricio R. Muñoz, Vance M. Whitaker, and Miguel Pérez-Enciso. Exploring Deep Learning for Complex Trait Genomic Prediction in Polyploid Outcrossing Species. *Frontiers in Plant Science*, 11(February):1–14, 2020. ISSN 1664462X. doi: 10.3389/fpls.2020.00025.
 13. Bowen Wang, Serge Sverdllov, and Elizabeth Thompson. Efficient estimation of realized kinship from SNP genotypes. *Genetics*, 2020. ISSN 0016-6731. doi: 10.1534/genetics.116.197004.
 14. R A Jacobs, M I Jordan, S E Nowlan, and G E Hinton. Adaptive mixture of experts, 1991.
 15. Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958, 2014. ISSN 15337928. doi: 10.1214/12-AOS1000.
 16. Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through L0 regularization. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, pages 1–13, 2018.
 17. Joseph O Ogutu, Torben Schulz-Streeck, and Hans-Peter Piepho. Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions. *BMC Proceedings*, 6(Suppl 2):S10, 2012. ISSN 1753-6561. doi: 10.1186/1753-6561-6-S2-S10.
 18. Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–13, 2017.
 19. Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–20, 2019.
 20. Abubakar Abid, Muhammed Fatih Balin, and James Zou. Concrete autoencoders: Differentiable feature selection and reconstruction. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:694–711, 2019.
 21. Teketel A. Haile, Taryn Heidecker, Derek Wright, Sandesh Neupane, Larissa Ramsay, Albert Vandenberg, and Kirstin E. Bett. Genomic selection for lentil breeding: Empirical evidence. *Plant Genome*, 13(1):1–15, 2020. ISSN 19403372. doi: 10.1002/tpg2.20002.
 22. José Crossa, Diego Jarquín, Jorge Franco, Paulino Pérez-Rodríguez, Juan Burgueño, Carolina Saint-Pierre, Prashant Vikram, Carolina Sansaloni, Cesar Petrolí, Deniz Akdemir, et al. Genomic prediction of gene bank wheat landraces. *G3: Genes, Genomes, Genetics*, 6(7): 1819–1834, 2016.
 23. Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Supplementary Figures

Fig. S1. Detailed results for the first three traits in the lentil diversity panel dataset. The line $y = x$ is shown in black.

Fig. S2. Detailed results for the first three traits in the lentil biparental dataset. The line $y = x$ is shown in black.

Fig. S3. Detailed results for the first three traits in the carinata dataset. The line $y = x$ is shown in black.

Fig. S4. Detailed results for the first three traits in the wheat dataset. The line $y = x$ is shown in black.