Aquaculture Reports 39 (2024) 102468

Contents lists available at ScienceDirect

Aquaculture Reports

journal homepage: www.elsevier.com/locate/aqrep

Breeding evaluations in aquaculture using neural networks

Christos Palaiokostas

Department of Animal Biosciences, Swedish University of Agricultural Sciences, Uppsala, Sweden

ARTICLE INFO

Keywords:
Neural networks
Machine learning
Aquaculture breeding

ABSTRACT

Deep learning comes with a portfolio of highly flexible models, known as neural networks (NNs), capable of solving various problems and setting new high standards for prediction accuracy. Nevertheless, whether NNs could be of value in aquaculture selective breeding settings is unclear as the whole topic is underexplored. Furthermore, fine-tuning a plethora of hyperparameters before fitting a neural network is a daunting task. Using simulated and a publicly available dataset on genetic resistance in carp against koi herpes virus (KHV), various neural network architectures were benchmarked against commonly used animal breeding models. More specifically, the simulated datasets comprised 36000 phenotyped animals genotyped for 54000 single nucleotide polymorphisms (SNPs). In contrast, the carp dataset included 1255 carp juveniles with survival recordings for KHV that were genotyped for 15615 SNPs. The assessed NN architectures included multilayer perceptrons (MLPs), convolution neural networks (CNNs) and local convolution neural networks (LCNNs). In addition, the effect of various hyperparameters of neural networks, such as the number of hidden layers, neurons per layer, activation function, learning rate, batch size, and regularisation techniques like dropout, were examined. In the simulated datasets, fully connected models with 5 hidden layers and 100 neurons per layer performed slightly better (1 – 4 %) than ridge-regression best linear unbiased prediction (rrBLUP), while the CNNs gave the lowest prediction accuracies (~ 14 % lower than MLPs) and the ones from LCNN in between the above (~ 8 lower than MLPs). Nevertheless, the estimated breeding values from NNs appeared more biased than rrBLUP (mean regression slope of 1.2 for the NN with the highest prediction accuracy vs 1.08). A reverse picture was observed in the case of the carp dataset, where following the application of receiver operating characteristic (ROC) curves, the animal breeding models outperformed neural networks by more than 2 % (based on the area under the curve index). In this case the LCNN had the highest area under the curve index from all NNs. Overall, NNs could be valuable tools in aquaculture breeding programs, though large training datasets of tens of thousands or more of phenotyped and genotyped animals seem to be required.

Introduction

The central aim of a breeding program is to determine which animals have the highest potential to be used as breeders. At its essence, selective breeding attempts to predict the productivity of a future generation by deciding on its genetic makeup through appropriate mating crosses. This decision is based on available information from current and past generations. Understandably, the above task is accompanied by uncertainty, constituting prediction accuracy a key success-determining factor. Therefore, fine-tuned statistical models capable of producing accurate predictions are necessary.

The best linear unbiased prediction (BLUP) methodology (Henderson, 1975) and its modern extensions, which use genomic information (Meuwissen et al., 2001; Legarra et al., 2014; Misztal et al., 2020), have allowed for substantial improvements in practically all food production systems (Hickey et al., 2017). In aquaculture, compelling evidence from early on pointed towards a considerable increase in productivity through selective breeding (Gjedrem and Rye, 2018), which recent studies have emphatically verified (Kause et al., 2022; Vandeputte et al., 2022; Faggion et al., 2023).

As selective breeding results in cumulative and, most often, permanent improvements, even a small increase in prediction accuracy can result in substantial economic benefits for the industry in the long term. Despite the success of the original BLUP methodology, notable efforts have been made over the years to enrich the statistical machinery responsible for ranking the breeding candidates (de los Campos et al., 2013; Lourenco et al., 2020) with models that could further boost prediction accuracy (GBLUP family, Bayesian-based models).

Machine learning constitutes a diverse family of models where accurate predictions are often the ultimate goal. Providing solutions in a

E-mail address: christos.palaiokostas@slu.se.

https://doi.org/10.1016/j.aqrep.2024.102468
Received 1 June 2024; Received in revised form 16 October 2024; Accepted 29 October 2024
Available online 15 November 2024

diverse range of tasks (Wilmott, 2019), including applications in aquaculture (Karras et al., 2023), machine learning models have attracted attention regarding their potential in breeding evaluations. Compared to other models, where assumptions and domain expertise are hard coded, machine learning models are designed to detect patterns in data automatically (Eraslan et al., 2019). Even though previous studies in aquaculture species suggested that, in some cases, more accurate predictions could be obtained compared to ones derived from the golden standard breeding models, this trend is certainly not universal (Bargelloni et al., 2021; Palaiokostas, 2021; Wang et al., 2022; Song et al., 2023b).

Deep learning (DL) is a subfield of machine learning associated recently with fascinating advancements in artificial intelligence. A key characteristic of deep learning is that it allows learning to progress through successive layers of increasingly meaningful representations (Chollet, 2021). The machinery responsible for predictions in DL is known as a neural network. Following a simplistic mental model of how a neural network functions, one can imagine information funnelled from the input layer (equivalent term for input data) through successive layers (known as hidden layers) to finally emerge in a distilled format suitable to address the problem in question.

The building block of each layer is the neuron, which, in the case of the input layer, represents an explanatory variable known in machine learning terminology as a feature. At its core, a neuron performs an affine transformation to its input, which is subsequently passed to an activation function. More specifically, the affine transformation involves the dot product between a feature vector and a matrix of trainable weights, followed by adding an offset term. During training, the matrix of trainable weights is adjusted to minimise a loss function used as a proxy to evaluate the model's predictions. Commonly used loss functions include the mean square error (MSE) for regression and the cross-entropy for classification problems. An essential factor in the above process is the backpropagation algorithm (Rumelhart et al., 1986) that efficiently computes the loss function and its gradient for all trainable parameters (Lindholm et al., 2022).

Overall, the number of hidden layers, the neurons on each layer, and the way those are connected can vary, allowing for practically any model architecture. The above are known as model hyperparameters that users must specify in advance. Neural networks come with many hyperparameters like the learning rate, the choice of an optimiser, the batch size and the activation function. As a detailed explanation of those hyperparameters is beyond the scope of this article, the interested reader is pointed to Geron (2019). In relation to the common observation that neural networks appear to shine in capturing nonlinear associations, (Montesinos-López et al., 2021b) it would worth mentioning that this capacity is due to the usage of activation functions. A wide range of activation functions have been tested over the years, and some of the most popular choices nowadays include the rectified linear unit (ReLU) and its variants, such as the exponential linear unit (ELU) and the scaled ELU (SELU) (Xu et al., 2015; Klambauer et al., 2017).

A typical neural network architecture contains connections between all neurons from two successive layers, characterising a class of models as fully connected ones (Fig. 1). Those models, also known as multilayer perceptrons (MLPs), have been tested recently in plant breeding, where competitive predictions compared to GBLUP-based models were obtained (Montesinos-López et al., 2019b). However, a drawback of MLPs lies in the magnitude of estimable parameters, which can range to several millions, even for models of moderate complexity.

Convolutional neural networks (CNNs) form another popular category of deep learning models (Fig. 2). CNNs have been instrumental in the success of deep learning, allowing remarkable breakthroughs, especially in computer vision (Geron, 2019). Particularly suitable for handling problems where the input data have a grid-like structure, CNNs can also be applied to single or multi-dimensional data. By leveraging sparse interactions and parameter sharing, CNNs tend to have far fewer parameters than MLPs (Lindholm et al., 2022). Finally, an intermediate category of NNs in terms of number of parameters is the local CNNs (LCNNs) where opposed to CNNs the trainable weights of each predefined channel and filter size are not shared.

In general, and regardless of model architecture, neural networks tend to have orders of magnitude more estimable parameters than customed animal breeding models or other machine learning algorithms. As such, the risk of overfitting is substantially higher. Overfitting refers to a generic term in statistics where a model, instead of detecting patterns in the dataset that generalise, also adapts to noise. More specifically, in

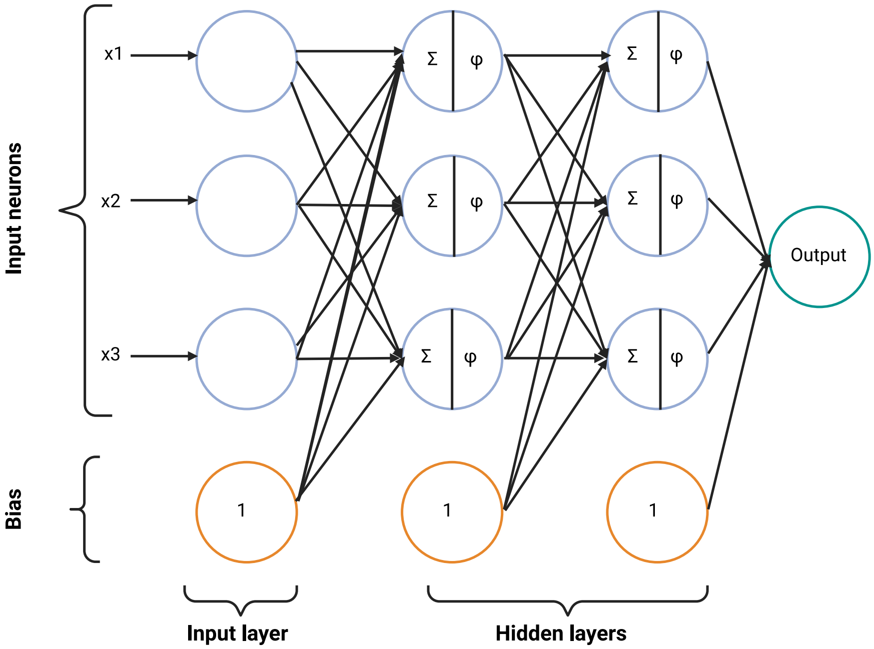

Fig. 1. Architecture of a multilayer perceptron (MLPs). The depicted neural network has three input neurons (x1, x2, x3) and two hidden layers, each containing three neurons. A bias neuron is included in every layer, always outputting 1. **Σ** denotes an affine transformation of the dot product between the feature vector and a matrix of trainable weights, followed by adding an offset term from the layers' bias neuron. **φ** denotes an activation function that allows the model to capture nonlinear associations. The output layer consists of a single neuron referring to a regression problem. Figure was created with BioRender.com.

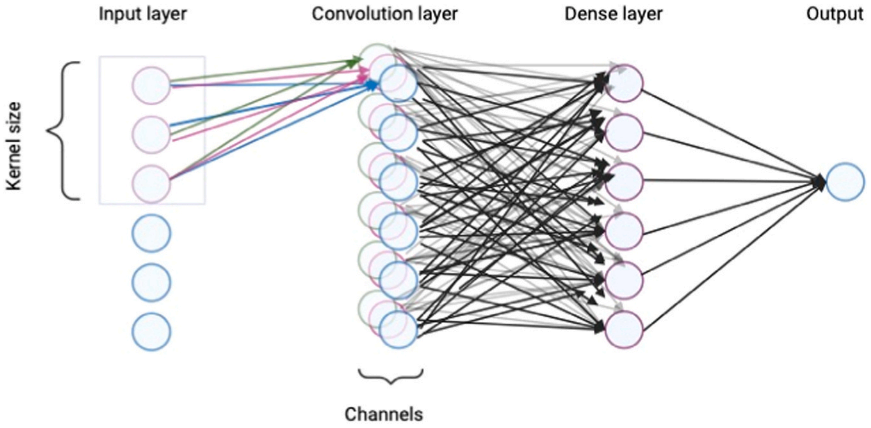

Fig. 2. Architecture of a convolution neural network (CNN). The depicted model uses a filter (or kernel size) of 3. A surrounding rectangle points to the used filter. All the same genotypic sequences share the trainable weights of the used filter regardless of genomic location. The convolutional layer consists of 3 different channels. The first channel is visualised in blue, the second in purple, and the third in green. Each channel has unique trainable weights, all of which simultaneously contribute to the output of each neuron from the dense layer. To avoid cluttering the figure, the bias neurons and their associated links are not depicted. The output layer consists of a single neuron referring to a regression problem. Figure was created with BioRender.com

every dataset, one expects to find both regular and irregular features, with the formers being the target of learning as they tend to generalise and, therefore, contribute towards answering a question of interest (McElreath, 2020). Due to overfitting a model could initially give the impression that highly accurate predictions can be obtained. However, when the model is applied to previously unseen data (which is the reason for using a ML algorithm or more broadly speaking any sort of model) the prediction accuracy tends to be considerably worse than anticipated.

Even though with highly complex datasets such as genomic ones, a certain level of overfitting is practically unavoidable, strategies exist to alleviate it. Those strategies usually come under the umbrella term of regularisation, where model complexity and the magnitude of estimable parameters are penalised. Aside from generic regularisation techniques, ones specific to neural networks exist. In the latter case, some of the most popular ones include batch normalisation, where each input neuron on every hidden layer is zero-centered and normalised (Ioffe and Szegedy, 2015) and dropout, where at every training step, a prespecified and random subset of neurons is ignored (Hinton et al., 2012).

In the current study, MLPs, CNNs and LCNNs were assessed in terms of their efficiency in predicting the most suitable breeding candidates. Both simulated data and a publicly available aquaculture dataset were used. In the case of simulated datasets, the prediction efficiency of the tested neural networks was benchmarked against the corresponding predictions from ridge-regression BLUP (rrBLUP) and random forests (RFs). The former, being mathematically equivalent to GBLUP, has been repeatedly shown to produce robust predictions (Meuwissen et al., 2001; Daetwyler et al., 2013). Equally, RFs are considered one of the most robust ML models being successfully tested in diverse problems, including ones related to genomic prediction (Montesinos López et al., 2022). In addition, the aforementioned neural networks were applied in a publicly available dataset on genetic resistance to koi herpes virus in common carp (Palaiokostas et al., 2019). Finally, the effect of different neural network hyperparameters on prediction efficiency was assessed. Those hyperparameters included the number of hidden layers, neurons per layer, the choice of activation function, optimiser and the application of regularisation with dropout.

Materials and methods

Simulated datasets

Phenotypes and their corresponding genotypes were simulated using the QMSim software (Sargolzaei and Schenkel, 2009). The initial historic population involved 2000 generations, each with a constant size of 10000 animals. An equal sex ratio, random mating and discrete generations were assumed. After that, 12 discrete non-overlapping recent generations were created using a breeding design often encountered in salmonids. In particular, 150 sires were considered to be mated with 300 dams, where a single sire is mated with two dams, and 30 animals from each family were phenotyped for a single trait. For the last four generations, SNP data were created. As such, individuals from generation 9–12 (36000 animals) were genotyped for 54000 SNPs (typical size of a SNP array used in aquaculture breeding) randomly distributed across a genome of 30 chromosomes, each of 100 centimorgans (cM) in length. The heritability of the simulated trait was equal to 0.3, out of which 0.2 was attributed to quantitative trait loci (QTL). Overall, 100 biallelic QTLs randomly located per chromosome were simulated. Those QTLs were sampled from a gamma distribution with a shape of 0.4 Finally, ten replicates of simulated data were created. For each simulated dataset, the last generation was considered the test set, while the previous three generations with genotypic data comprised the training set from which all trainable weights of each model were estimated. The tested models were assessed based on the Pearson correlation coefficient between the predicted values and the test set's true breeding values.

Baseline predictive efficiency using rrBLUP and random forest (RF)

The rrBLUP model was used as a baseline for prediction accuracies. The fitted model had the following form:

$$y_i = \mu + \sum_{k=1}^{p} x_{ik} \quad \beta_\kappa + \varepsilon_i$$

where y_i is the phenotype of the ith individual, μ is the grand mean, x_{ik} is the genotype at the kth marker of the ith individual, with p representing the total number of markers. β_k is the estimated random additive marker effect of the kth marker ~N(0, σ^2_β), and ε_i is the residual error term ~N (0, σ^2_e). σ^2_β was assigned a scaled inverse χ^2 prior distribution with 5 degrees of freedom (df) and a scale value reflecting that 50 % of the observed variance was attributable to the SNP markers. The residual variance (σ^2_e) was assigned a prior following a scaled inverse χ^2 distribution and parameterised as above. The model was implemented using the R/BGLR software (Pérez and de los Campos, 2014).

In the case of the RF, the model was fitted using the Python library scikit-learn v1.2 (Pedregosa et al., 2011). The RF model was considered to have a maximum of 2000 decision tree regressors. Furthermore, when considering for the best split the maximum features hyperparameter was equal to 30. Those hyperparameters were inferred after 3-fold cross-validation on the training set using the *RandomizedSearchCV* function of scikit-learn.

Neural network architectures

The first model architecture that was assessed involved MLPs. As mentioned, in this sort of architecture, all neurons between two successive layers are connected through trainable weights (Fig. 1). In general, the outputs of a fully connected layer are defined as follows:

$$h_{W,b}(\boldsymbol{X}) = \phi(\boldsymbol{XW} + \boldsymbol{b})$$

where **X** denotes the matrix of input features. In the current case, **X** represented the genotypic matrix where the number of rows equals the number of genotyped samples, and the number of columns equals the number of SNPs. Each genotype was initially coded as 0, 1, 2. Before model fitting, the genotypes were normalised using the *StandardScaler* function of scikit-learn. The weight matrix **W** denotes all the connection weights (except those from the bias neuron) and has one row and column per input and output neuron, respectively. The vector **b** denotes all the connection weights between the bias neuron and each other neuron in the same layer. Finally, **ϕ** denotes the activation function. During training, the aim was to minimise a pre-specified loss function. In the case of the simulated datasets, the MSE was the targeted loss function, while in the case of the publicly available carp dataset, the aim was to minimise cross-entropy.

The current study evaluated the effect of various hyperparameters on prediction accuracy. Initially, the effect of the optimisation, the learning rate, and the number of hidden layers were tested. After retaining the values of those hyperparameters, which gave the highest prediction accuracy, models with varying neurons per layer, activation function, and dropout were fitted. More specifically, models with either 1, 10, 100, 300, 500 or 1000 nodes per layer were tested. Furthermore, the effect of using different activation functions was tested, which included the usage of either ReLU, ELU or SELU. In addition, all the above scenarios were also tested using dropout, where a random 20 % of the neurons of each hidden layer were ignored. Overall, by varying the number of neurons per layer, activation function and the inclusion of dropout, 24 different models were fitted (Table 1).

In the case of the CNN architecture, the tested hyperparameters included the number of convolutional layers (CLs), number of channels, kernel size and the inclusion of a max pooling layer with a pool size of 2 (Table 1). CNN architectures with either 1 or 3 CLs were assessed. Furthermore, the number of channels had values of 8, 16 or 32, while in the case of the kernel size, the tested values were 3 and 6. In contrast to the convolutional layers, the max pooling layer by construction does not include any additional trainable parameters. Instead, it condenses information by using only the maximum value of a predefined region, which in this case was equal to 2. Finally, a fully connected layer of 300 neurons was included before the output layer. Similar to the MLPs, 24 different models were fitted.

Table 1
Hyperparameter search among different neural network topologies.

Hyperparameter	Tested values
Multilayer perceptron (MLPs)	
Activation function	ReLU, ELU, SELU
Batch size	32, 64, 128
Dropout	No, Yes
Number of hidden layers	1, 3, 5
Neurons per layer	1, 10, 100, 300, 500, 1000
Optimizer	Adam, Nadam
Learning rate	0.1, 0.01, 0.001
Convolution neural networks (CNNs)	
Convolutional layers	1,3
Channels	8, 16, 32
Kernel size	3, 6
Maxpooling	No, Yes
Local Convolution neural networks (LCNNs)	
Convolutional layers	1,3
Channels	4, 8, 10, 12
Kernel size	8, 10, 12

Finally, a similar network architecture to that previously described for CNN, named local CNN (LCNN), was also assessed. In contrast to CNN, in LCNN, the trainable weights of each predefined channel and filter size are not shared. As such, the number of trainable parameters is larger than CNNs, though not as large as in MLPs. The tested hyperparameters for LCNNs included 1 or 3 CLs, channels of size 4, 8, 10 or 12 and kernel sizes of 8, 10 or 12 (Table 1).

Fitting neural networks

All the above models were fitted using Keras (Chollet and others 2015) with TensorFlow (Abadi et al., 2015) v2.10.1 as the backend. Example code for fitting neural network models can be found at https://github.com/chpalaiokostas/DNN_in_aquaculture_breeding. As is customary in deep learning, model training did not take place by fitting the entire dataset simultaneously but iteratively through pre-specified batch sizes. This process was repeated until the targeted loss function reached a minimum.

Following custom terminology, each iteration involving all available training data is called an epoch. The current study used a batch size of 32 and a maximum number of 100 epochs. Regarding the number of epochs, it is essential to mention that a tradeoff exists between under- and overfitting. If an insufficient number of epochs is used, then the model will not be able to capture efficiently all generalisable features, resulting in poor predictions. On the other hand, as the number of epochs increases, the model will adapt to noise, resulting in overfitting. To avoid the above, the early stopping functionality offered by TensorFlow was used. In this case, as soon as no further progress is observed regarding the reduction of the loss function in the validation set, the trainable weights were reversed back to the ones from the epoch, where the loss function was previously at its minimum. The above is a simple but efficient regularisation technique commonly applied in deep learning (Geron, 2019). Moreover, TensorBoard (Abadi et al., 2015) was used to visualise the learning progress during each epoch. Finally, all computations were performed in the high-performance computing (HPC) cluster Alvis dedicated to artificial intelligence and machine learning research and offered by the national academic infrastructure for supercomputing in Sweden.

Inflation of estimated breeding values

The true breeding values of the test set were regressed on the estimated breeding values of each fitted model to get insights regarding potential bias. The slope regression coefficient indicates inflated values when found to be above one or deflated ones when below one. On the other hand, a value equal to 1 suggests that the results are unbiased.

Carp resistance to koi herpesvirus dataset

A publicly available dataset from Palaiokostas et al. (2019) was utilised to assess the efficiency of a MLP and a CNN model in predicting carp resistance to koi herpesvirus disease (KHVD). In the case of the MLP, the used topology consisted of 3 hidden layers of 300 neurons each, while the CNN had a convolution layer of 32 channels with a filter size of 3 and a fully connected layer of 300 neurons. A binary phenotype represented resistant and non-resistant animals. Predictions were assessed using receiver operating characteristic (ROC) curves, with the tested models ranked based on the area under the curve (AUC) metric. By construction, AUC ranges between zero and one, with the latter representing the perfect classifier. The dataset consisted of 1255 carp juveniles with survival recordings for KHVD that were genotyped for

15615 SNPs using restriction-site associated DNA sequencing (RAD-seq). A five-fold cross-validation was used to rank the fitted models, with the whole procedure repeated five times to account for potential bias in the assignment of animals in training or validation sets. Allocation to training and validation sets was as in Palaiokostas (2021) to allow for direct comparisons.

Results

All fitted NNs converged in less than 100 epochs. The EarlyStopping hyperparameter ensured that models that required more epochs for convergence than others did not overfit the training dataset.

Hyperparameter fine-tuning of MLPs in the simulated datasets

A two-stage exploration of the effect of various hyperparameters on the prediction accuracy of MLPs was performed. This kept the number of screened models to a manageable level. The assessed MLPs contained 1, 3, or 5 hidden layers, while the number of neurons per layer was 1, 10, 100, and 300. The simplest model had 1 hidden layer and 1 neuron, while the most complex had 5 hidden layers with 100 neurons each. Each model was trained on a dataset of 27000 animals from the first three generations with genotypic data and assessed based on the Pearson correlation coefficient between the estimated and true breeding value of each animal from the test dataset (n = 9000). The model with the highest mean prediction accuracy had 5 hidden layers with 100 neurons each (r = 0.79). On the other hand, the model with the lowest prediction accuracy (r = 0.59) was the simplest one with 1 hidden layer and 1 neuron. In general, only slight differences were found regarding the standard deviation of the prediction accuracy of each model amongst the 10 datasets (Fig. 3). Additionally, the aforementioned models were fitted using different learning rate values (0.01 and 0.001), batch sizes (32, 64, 128) and optimisers (Adam or Nadam). As no improvements compared to the default values were found, all presented results were obtained from models with a learning rate of 0.001, a batch size of 32 and Adam as optimiser.

Based on the above, further exploration of the effect of various hyperparameters was attempted by keeping the number of hidden layers fixed at 5 and testing different values of neurons per layer (100, 300, 500 and 1000) and three different activation functions (ReLU, ELU and SELU). Moreover, all the above combinations were tried with and without dropout. Overall, 24 different models were fitted to each of the 10 simulated datasets (Fig. 4).

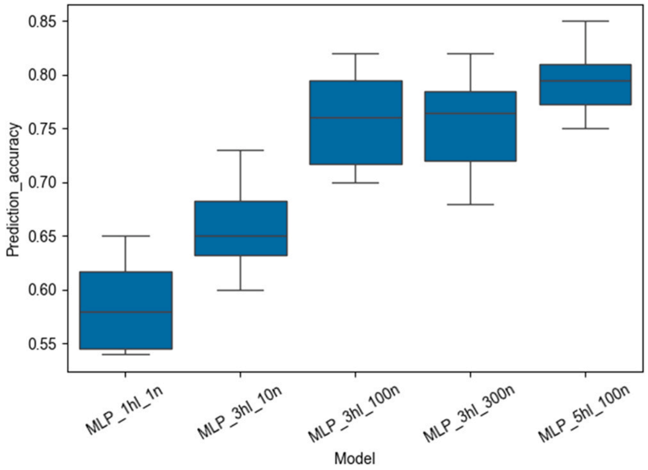

Fig. 3. Prediction accuracies of multilayer perceptrons (MLPs) of 1, 3 and 5 hidden layers (hl) with 1, 10, 100 and 300 neurons (n). The prediction accuracies were based on the Pearson correlation coefficient between the estimated and the true breeding value of each animal. The depicted values were estimated from 10 simulated datasets. In each case, the training and test datasets consisted of 27000 and 9000 animals, respectively.

The model with the highest prediction accuracy (r = 0.81) had 100 neurons on each of the 5 hidden layers with ReLU as activation function and without dropout. In general, for models with the same number of neurons per hidden layer, only slight differences were observed between the 3 activation functions (∼ 1 %), with ReLU consistently having the highest accuracy. The only exception to the above was observed in models containing 100 neurons per layer without dropout.

The use of dropout decreased prediction accuracies by ∼ 3 – 12 % compared to the equivalent models without it. Finally, only slightly higher standard deviations regarding prediction accuracy were observed in the models with 500 or 1000 neurons compared to ones with 100 or 300 (∼ 0.04 vs 0.05 std).

Hyperparameter fine-tuning of CNNs and LCNNs in the simulated datasets

CNNs with different numbers of CLs (1 or 3), kernel size (3 or 6) and number of channels (8, 16, 32) were assessed. At the same time, all the above combinations of models were fitted with and without max pooling (Fig. 5). The above resulted in 24 different CNNs fitted to the same 10 simulated datasets as with MLPs. Compared to MLPs, the obtained prediction accuracies with CNN were lower (r ∼ 0.64 – 0.71). The best-performing CNN had a 14 % lower mean prediction accuracy than the best-performing MLP. At the same time, the accompanying standard deviations in the fitted CNNs were generally slightly higher, ∼ 0.06, than the MLPs. In general, CNNs with 1 CL performed better than the corresponding model with 3 CLs. In addition, CNNs with 8 channels performed better except where max pooling and a filter size of 3 were used.

Similar to CNNs, 24 LCNNs were fitted with varying CLs (1 or 3), kernel size (8,10,12) and number of channels (4,8,10,12). A higher mean accuracy by ∼ 4 % was observed when models with 1 CL were compared to those with 3 CLs, following a similar pattern as previously observed with CNN. The obtained prediction accuracies ranged between 0.69 – 0.74. The model with the higher prediction accuracy had 1 CL, 4 channels and a kernel size of either 12 or 8 (Fig. 6). The accompanying standard deviations were ∼ 0.04, similar to ones from MLPs

Benchmarking the best performing NNs against rrBLUP and RF

Following hyperparameter fine-tuning, the best-performing MLP, LCNN and CNN were benchmarked against rrBLUP and RF. More specifically, a MLP with 5 hidden layers containing 100 neurons per layer, the ReLU activation function and without dropout was used as a representative of the former category. In the case of LCNN its hyperparameters were fixed to 1 convolution layer, a kernel size of 8 and 4 channels and a fully connected layer with 300 neurons before the output layer. For the CNNs, the used topology had 1 convolution and 1 fully connected layer. The former had 8 channels, a filter size of 3 without max pooling, while the latter had 300 neurons similar to the previously described LCNN. The MLP had the highest mean prediction accuracy, while the lowest was recorded in the case of the RF (Fig. 7). More specifically, the MLP outperformed rrBLUP, LCNN, CNN and RF by ∼ 4 %, ∼9 %, 14 % and 31 %, respectively. At the same time, the MLP had the lowest standard deviation amongst the 10 simulated datasets (∼ 0.03 vs 0.03 – 0.06).

Inflation of estimated breeding values

The mean regression slope in most cases was above 1, indicating that the estimated breeding values were deflated. Out of 74 fitted models, the mean regression slope was below one in MLPs when the SELU activation function and dropout were used and in CNNs with 1 CL, 32 channels and no max pooling. However, the corresponding standard deviations were the highest recorded for the latter, with values exceeding 0.2. Overall,

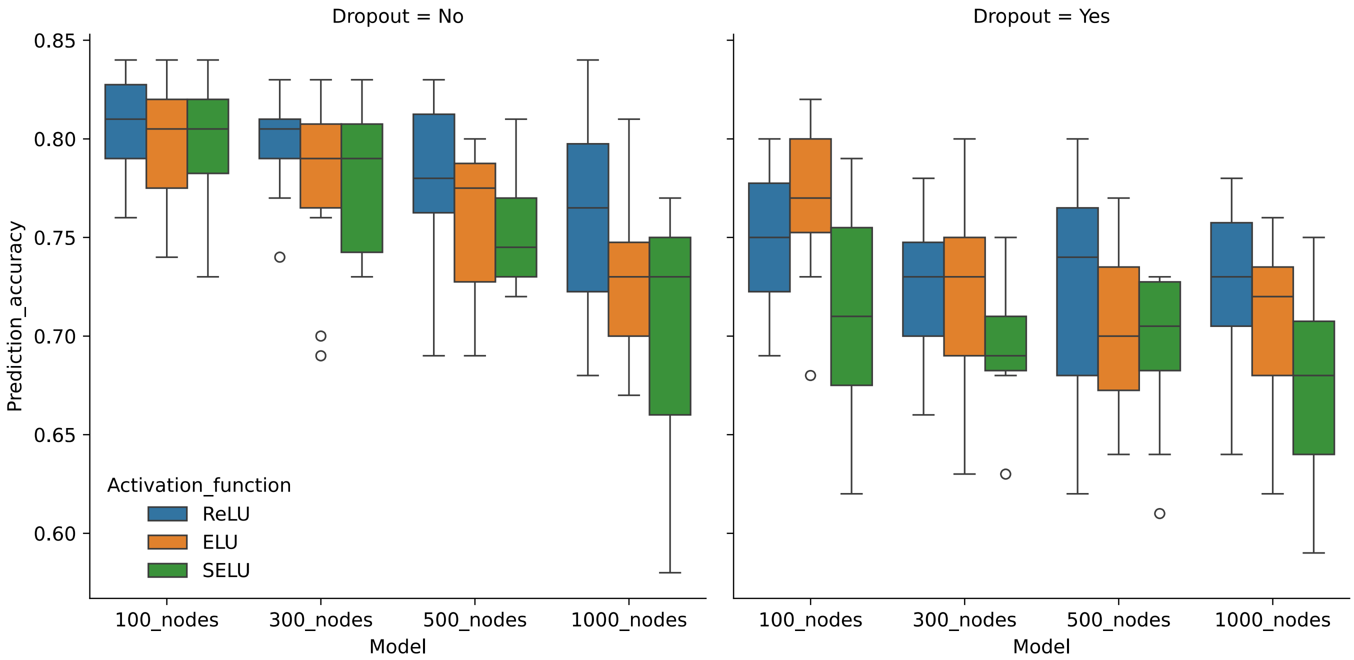

Fig. 4. Prediction accuracies of multilayer perceptrons (MLPs) with 5 hidden layers and 100, 300, 500 and 1000 neurons on each layer. The above models were fitted with the ReLU, ELU or SELU activation functions, each coloured differently. The above combinations were also fitted with and without dropout, as depicted in the two subplots. The prediction accuracies were based on the Pearson correlation coefficient between the estimated and the true breeding value of each animal. The depicted values were estimated from 10 simulated datasets. In each case, the training and test datasets consisted of 27000 and 9000 animals, respectively.

less bias was observed with rrBLUP (mean 1.08; SD 0.04) than with NNs where the mean regression slope ranged between 0.93 – 1.6 with accompanying standard deviations of 0.05 – 0.25. The best performing NN in terms of prediction accuracy (MLP; RELU activation function; 5 hidden layers; 100 neurons on each layer) had a mean regression slope of ~ 1.2 with a SD of 0.06.

Predicting for KHV resistant carp using NNs

Both a MLP, a CNN and a LCNN were benchmarked against GBLUP in terms of predicting KHV resistant carp. As a binary phenotype was used to distinguish between resistant and non-resistant animals, the area under the curve (AUC) score was used to identify the best-performing model with a value of 1, denoting the perfect classifier. The AUC score from the LCNN was the highest from all fitted NNs and was in close proximity to the one from GBLUP (~ 2 % lower). The MLP and the CNN had a lower mean AUC score than GBLUP (Fig. 8). GBLUP outperformed the MLP by ~ 4 % and the CNN by 10 % on average.

Discussion

Genomics is nowadays an indispensable part of several aquaculture breeding programs (Houston et al., 2020). Over the last few years, aiming to unlock the full potential of selective breeding and boost genetic gain, researchers have compared predictions derived from deep learning algorithms to those from animal breeding models (Pérez-Enciso and Zingaretti, 2019; Montesinos-López et al., 2021b). As is usually the case when comparing predictions from different models, it is probably utopic to expect that a single model will be top-ranked across all traits that a modern breeding program targets (Palaiokostas, 2021). Nevertheless, competitive or even higher accuracies have been reported with DL models compared to popular animal breeding models (Waldmann, 2018; Montesinos-López et al., 2019a; Sandhu et al., 2021) in some instances.

Benchmarking the prediction efficiency of neural networks

Overall, the potential of ML models and especially DL ones, to deliver genomic predictions for breeding purposes in aquaculture species is underexplored (Song et al., 2023a). The above is not surprising, considering that the topic is understudied even in more mature industries (plants or terrestrial animals) (Montesinos-López et al., 2021a). The required number of samples for efficient implementation of DL is probably one of the main restrictive factors. More specifically, to efficiently train a model with several hundreds of thousands or even millions of parameters, datasets of corresponding sizes are required. At the same time, DL models are computationally demanding, necessitating access to substantial resources.

In the current study, datasets of moderate size for DL standards were simulated, each consisting of 36000 phenotyped animals with accompanying genotypes from 54000 SNPs. It has to be stressed that in terms of size, the above is probably within the higher upper limit of the datasets that a typical aquaculture breeding company would possess. Across all simulated scenarios, the best-performing neural network performed slightly better in terms of prediction accuracy (by ~ 4 %) than a most commonly BLUP-based animal breeding model (rrBLUP). At the same time, practically all MLPs that were tested had a substantially higher prediction accuracy (~20 % - 40 %) than RF. The above results are in the upper range of what has been previously documented in most studies (Montesinos-López et al., 2021a; Vu et al., 2022). However, increases above 10 % in terms of prediction accuracies using NNs compared to custom breeding models have also been reported (Pook et al., 2020; Nguyen et al., 2022; Luo et al., 2024).

It is likely that since the training dataset of the current study (n = 27000) was considerably higher compared to previous studies (~ n = 400 – 8000), DL models were more adequately trained. In addition, the fact in the simulated datasets, the trait was not fully polygenic, with QTLs accounting for 2/3 of the narrow sense heritability, might explain to a certain degree why rrBLUP did not have the highest prediction accuracy of all tested models. As BLUP models impose a strong prior that heavily shrinks all QTLs towards zero might have restricted the upper level of prediction accuracy in the tested simulation scenarios. On the

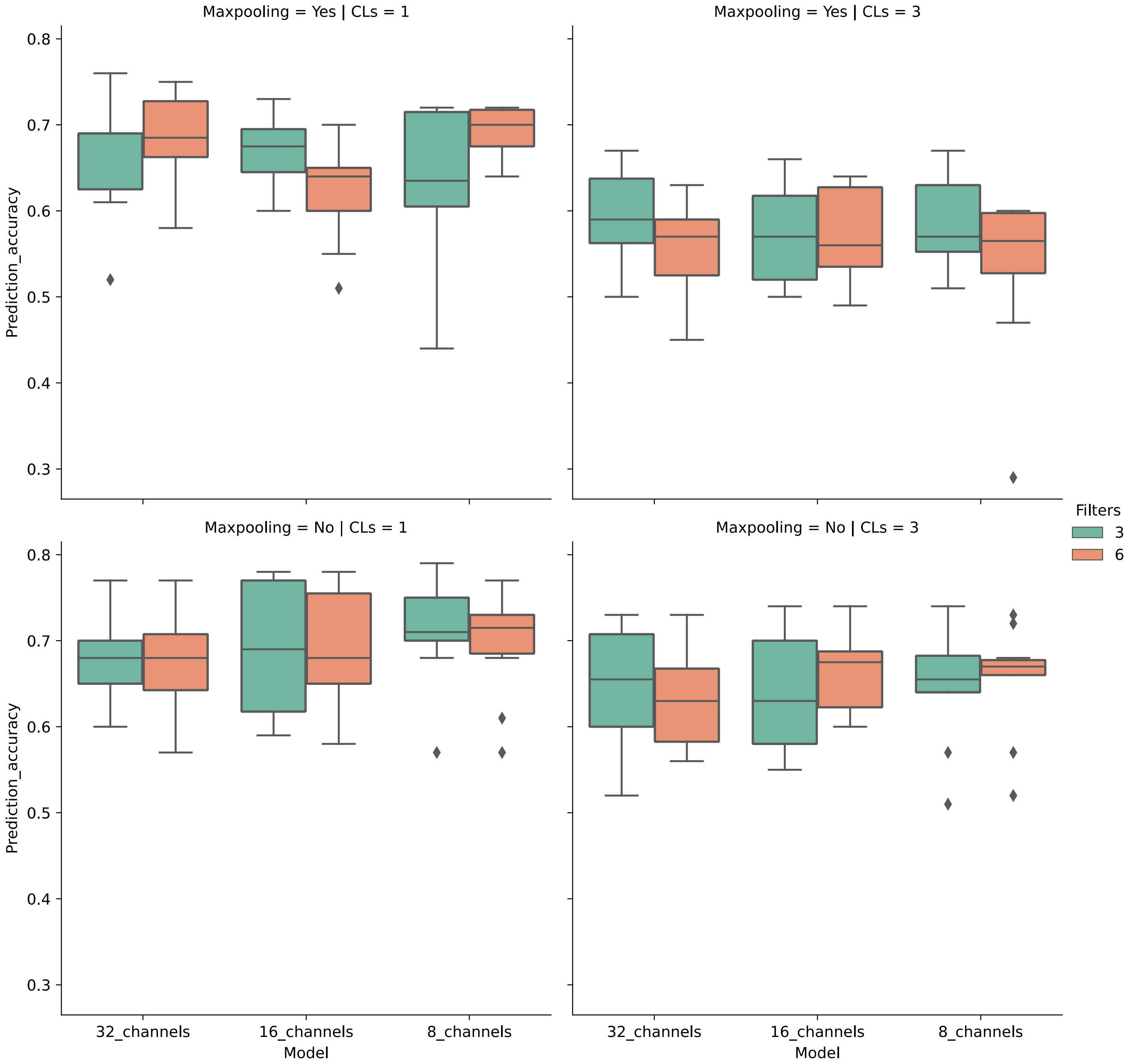

Fig. 5. Prediction accuracies of convolution neural networks (CNNs). CNNs with 1 or 3 convolution layers (CLs), number of channels of 8, 16 or 32 and a kernel size of 3 or 6 were fitted. The above combinations were also fitted with or without max pooling. All the fitted CNNs had 1 dense layer with 300 neurons before the output layer. The prediction accuracies were based on the Pearson correlation coefficient between the estimated and the true breeding value of each animal. The depicted values were estimated from 10 simulated datasets. In each case, the training dataset consisted of 27000 animals and was tested on a dataset of 9000 animals.

other hand, no such prior expectations are imposed on the NNs. Of course, it should be noted that rrBLUP was only slightly worse than the best-performing NNs, which automatically makes identifying the causative reason particularly difficult.

Furthermore, as a word of caution, the above results do not prove that DL models will be more efficient than customed animal breeding models in an actual aquaculture breeding program. As the former are mainly tested in simulated data, while the latter have been battle-tested for several years across a wide range of species and traits, additional studies are needed, preferably using real-life data. Furthermore, it is worth mentioning that the individual estimated breeding values from the NNs appeared to be more biased compared to rrBLUP (∼ 10 % more deflated in the case of the NN with the highest prediction accuracy).

Applying the same DL models in the KHV carp dataset resulted in competitive but slightly worse predictions than previously documented ones from GBLUP (Palaiokostas, 2021). In this case the LCNNs performed better compared to MLPs and CNNs. As the dataset, in this case, consisted of ∼1200 animals only, the highly parameterised neural networks were not the most efficient choice, though nonetheless, competitive predictions were obtained. It should also be pointed out that the structure of this dataset did not allow for an efficient hyperparameter fine-tuning of the NNs in a similar fashion to the simulated datasets. Instead, a cross-validation scheme was used as animals from only a single generation were available. Overall, as the aforementioned dataset represents at least the average size of publicly available data in aquaculture breeding it is fair to infer that more objective evaluations of DL models to the ones of the current study are difficult due to lack of appropriate data.

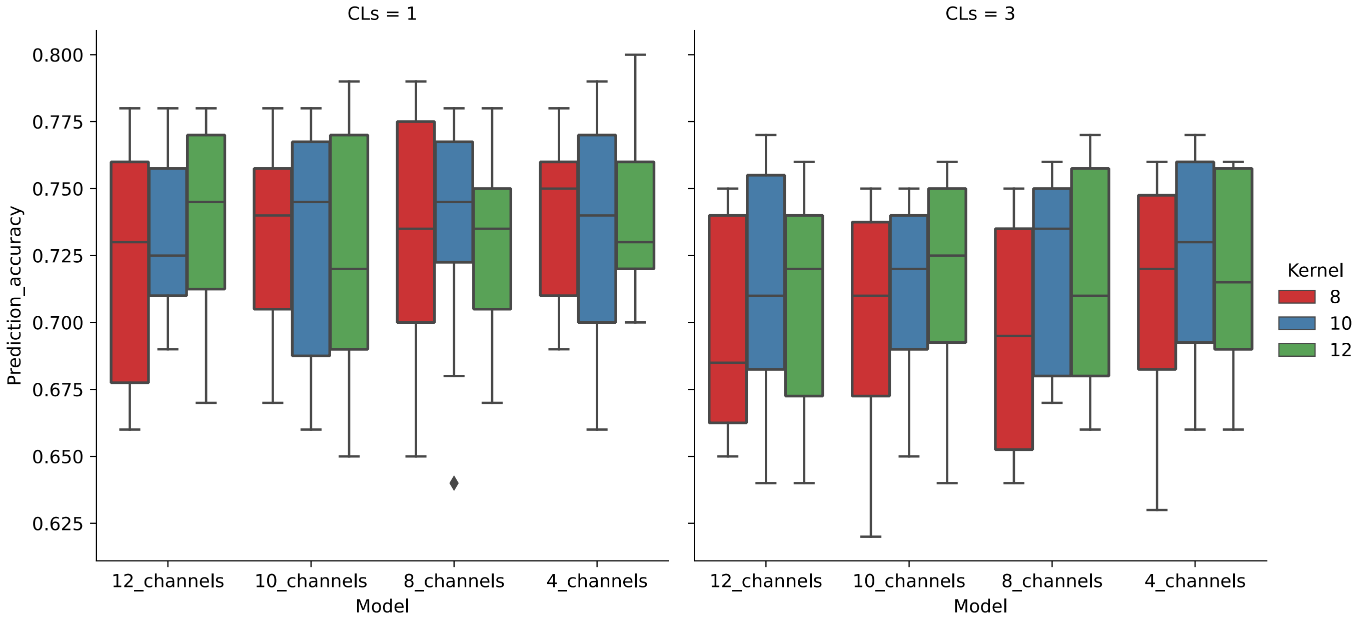

Fig. 6. Prediction accuracies of local convolution neural networks (LCNNs). LCNNs with 1 or 3 convolution layers (CLs), number of channels of 4, 8, 10 or 12 and a kernel size of 8, 10 or 12 were fitted. All the fitted LCNNs had 1 dense layer with 300 neurons before the output layer. The prediction accuracies were based on the Pearson correlation coefficient between the estimated and the true breeding value of each animal. The depicted values were estimated from 10 simulated datasets. In each case, the training dataset consisted of 27000 animals and was tested on a dataset of 9000 animals.

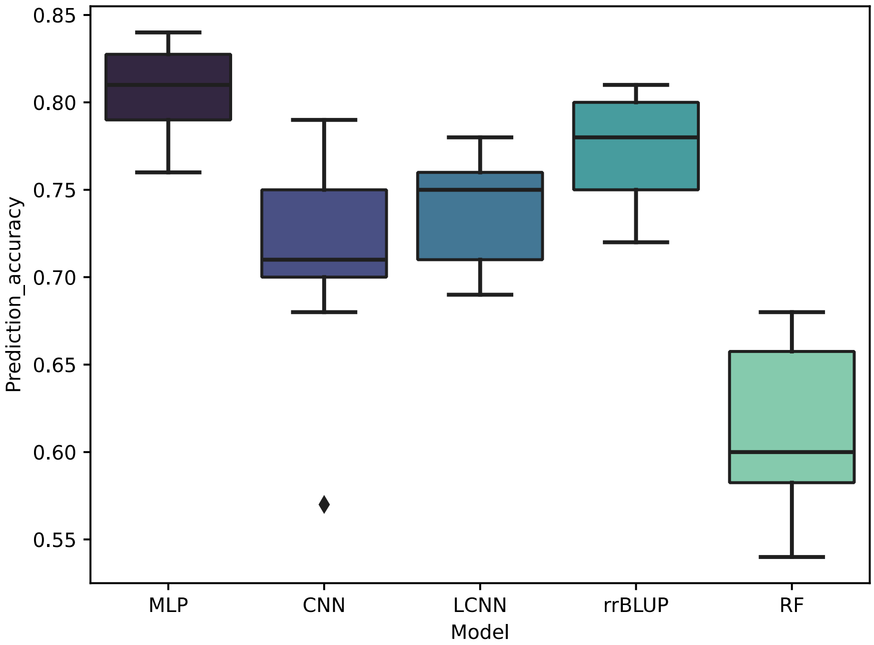

Fig. 7. Prediction accuracies from a multilayer perceptron (MLP), convolution neural network (CNN), local convolution neural network (LCNN), ridge-regression best linear unbiased prediction (rrBLUP) and a random forest (RF) model. The MLP was hyperparametrized with 5 hidden layers, 100 neurons on each layer and with the ReLU activation function. The CNN had 1 convolution and 1 fully connected layer. The former had 8 channels and a kernel size of 3, while the latter had 300 neurons. The LCNN had 1 convolution and 1 fully connected layer. The former had 4 channels and a kernel size of 8, while the latter had 300 neurons The RF model had a maximum of 2000 base estimators, a learning rate 0.1 and a maximum tree depth of 8. The prediction accuracies were based on the Pearson correlation coefficient between the estimated and the true breeding value of each animal. The depicted values were estimated from 10 simulated datasets. In each case, the training and test datasets consisted of 27000 and 9000 animals, respectively.

Neural network architectures

Three DL model architectures were used in this study, including MLPs CNNs and LCNN. The prediction accuracy derived from CNNs was lower (~ 4 – 13 %) from both the LCNN, MLPs and rrBLUP, suggesting that the topology used in this study is most likely not the most suitable for breeding-related predictions. CNNs models were consistently slightly worse in predicting human height than other DL and linear models (Bellot et al., 2018). On the other hand, CNNs models outperformed rrBLUP and other DL models in accurately predicting phenotypes of interest in wheat from corresponding genotypic information (Ma et al., 2018).

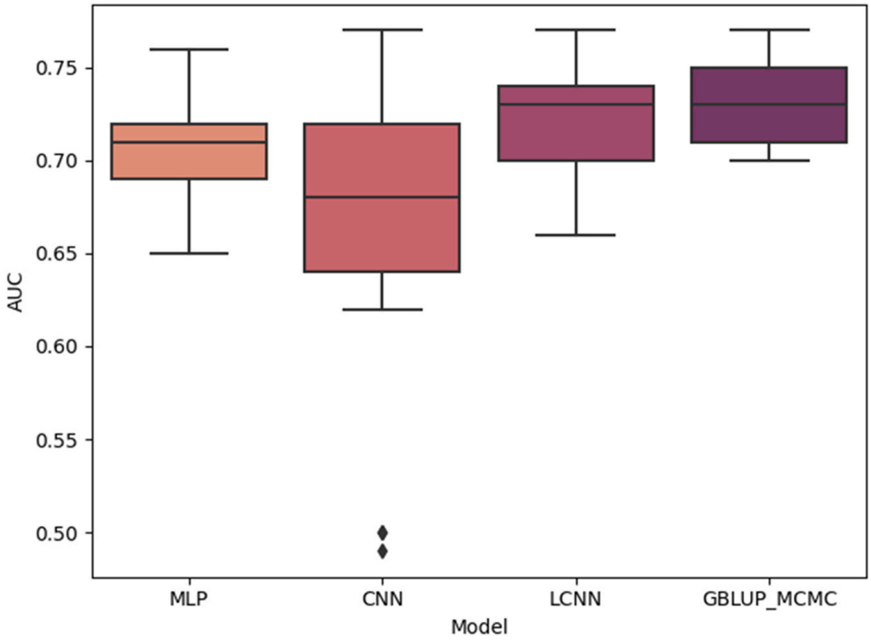

Fig. 8. Prediction of carp resistant to koi herpes virus (KHV). The area under the curve (AUC) scores from a multilayer perceptron (MLP), convolution neural network (CNN), local convolution neural network (LCNN) and a genomic best linear unbiased prediction (GBLUP) model were used as proxies to assess each model. The highest AUC score points to the best-performing classifier. The depicted values were estimated from a 5-fold cross-validation repeated 5 times to reduce potential bias due to the allocation of animals to the training and validation sets.

However, as pointed out by Pook et al. (2020) a convolutional layer in its customed form will attribute the same effect to a particular sequence of SNPs (it's length defined by the pre-selected filter size) regardless of its location in the genome. The above appears to be a restrictive attribute, as ordering SNPs in an array is not associated with their actual effects. As a result, the authors suggested LCNN as a natural extension of CNN where instead of weight sharing a region-specific filter is used (Pook et al., 2020). In the current study LCNN clearly outperformed CNN in practically all cases and, in the case of the KHV dataset performed better than the MLPs. In the study of Pook *et al.* LCNNs had ~10 % or more higher prediction accuracies than other DL models and GBLUP. However, taking into account the inherent flexibility of NNs fine-tuned CNNs could also be value. By applying a sparse CNN whose hyperparameters were fine-tuned through Bayesian optimisation, reductions in terms of prediction error between 3 % and over 25 % in both simulated and real pig datasets were found (Waldmann et al., 2020).

Effect of hyperparameters on the prediction efficiency of neural networks

All things considered, NNs appear to come with a blessing in disguise. On the one hand, the wide variety of available model architectures/topologies, together with a plethora of accompanying hyperparameters, empowers NN with the necessary flexibility to handle efficiently a range of very different problems. At the same time, this exact same attribute constitutes their usage a daunting task as fine-tuning their hyperparameters is a non-trivial task. The current study attempted to understand the effect of various NN hyperparameters on prediction accuracy, focusing on real and simulated data resembling aquaculture breeding scenarios.

Initially, the effect of only a handful of hyperparameters, like the choice of optimiser, learning rate and batch size, were screened as they have been highlighted as some of the most important ones in a wide range of problems (Geron, 2019). Regarding the choice of optimizer, only slight differences were found among members of the adaptive optimisation family, with Adam appearing suitable, as shown previously in different types of problems (Kingma and Ba, 2017). Moreover, learning rate and batch size had a negative effect only when large deviations from the default values were attempted (e.g learning rate 100-fold higher than the default value of 0.001 or a batch size exceeding 128). Otherwise, their effect was negligible. It is important to stress that the above inferences are based on a sparse search of hyperparameter values (e.g., only three different values of learning rate and batch size were attempted). Regardless, the data of this study support that the default values for learning rate and batch size don't seem to require extensive fine-tuning when performing genomic predictions.

Aiming to get further insights regarding the effect of hyperparameters in prediction accuracy, the effect of the number of hidden layers, neurons per layer, choice of activation function and the inclusion of a common regularisation technique known as dropout were further investigated. An architecture consisting of five hidden layers was chosen for the tested MLPs, as it showed a slight but consistent advantage over architectures with three hidden layers for the simulated datasets. Using 100 neurons per layer with the ReLU activation function resulted, on average, in the highest prediction accuracies among the tested models.

Surprisingly, including dropout gave consistently 3 – 12 % lower prediction accuracies. Probably, the fact that a relatively large training dataset was used alleviated the effect of overfitting. In addition, including a simple regularisation technique known as early stopping also contributed to the above. As a generic piece of information, it might be worth mentioning that the batch size (as long as it's lower than the size of the training dataset) also imposes regularisation to the fitted model in a similar format to ensemble ML algorithms.

Non-highlighted advantages of NNs compared to regular breeding models

Aside from focusing solely on comparing prediction accuracies between NNs and other breeding or ML models, there are attributes and tools related to the ecosystem of the former that are worth stressing. Even though NNs are computationally intensive, the fact that their application goes beyond the field of breeding triggered the development of highly efficient software, e.g. TensorFlow. As an example, the analyses of this study were conducted primarily in the Swedish HPC Alvis on a node with four NVIDIA A100 graphical processing units (GPUs). The code did not require any specific changes from the user's side compared to what would be needed if it was run on a laptop, as TensorFlow handled everything behind the scenes. Moreover, the fact that in NNs, training takes place using batches of the available data instead of requiring the entire dataset, as is the case with BLUP- or Bayesian-based breeding models, means that the computational time increases linearly with sample size. In comparison and due to the prohibitive computational costs of applying GBLUP-based models in large datasets, specialised algorithms have been developed where the so-called core animals from the training dataset are identified and used to invert the genomic relationship matrix (Misztal et al., 2014; Pocrnic et al., 2016a, 2016b; Bradford et al., 2017).

An additional and surprisingly non-highlighted aspect of NNs compared to breeding models is the availability of accompanying tools like TensorBoard that greatly facilitate monitoring the training status by providing informative plots. Overall, it is fair to say that the available software packages for DL have prioritised user friendliness and democratised their usage.

Future applications in aquaculture breeding

Even though competitive and, in some instances, higher prediction accuracies compared to rrBLUP were obtained with NNs, it is unlikely for the latter to replace the former, at least in the format used here. A primary disadvantage of NNs compared to BLUP-based evaluations is that the machinery responsible for predictions in the former is rather opaque (Montesinos-López et al., 2021b). Furthermore, the obtained predictions lack an accompanied uncertainty metric like reliabilities in BLUP-based breeding values. A further disadvantage of NNs, at least in the crude form attempted here, are not optimal for providing information regarding the underlying genetic architecture of the trait(s) under study compared to some animal breeding models. However, appropriate adjustments in network architecture and topology could allow the identification of QTLs, at least in some instances (Waldmann et al., 2020; Zhao et al., 2021).

Although the concept of breeding values is centred around additive effects, having reliable machinery for estimating non-additive effects can improve the prediction accuracy of genomic selection (Varona et al., 2018). The inherent attribute of NNs on capturing non-linear associations could open new possibilities, enabling more efficient mating crosses compared to purely additive models. The fact that competitive and even slightly higher prediction accuracies compared to rrBLUP were obtained in this study, even though the underlying genetic architecture of the simulated data was purely additive, suggests that there is potential for further exploring suitable NN architectures and topologies.

Reaping the full potential of NNs in selective breeding could be reached by utilizing information from different sources, e.g., multi-omics data simultaneously. Encouraging results using simulated data were obtained when customed animal breeding models were extended with NNs, allowing multi-omics data to be included in breeding evaluations (Zhao et al., 2022). With sequencing costs continuously dropping, the possibility of generating large datasets that contain information from multi-omics platforms no longer seems like science fiction. Furthermore, using NNs could contribute to recording more inciteful phenotypes (Xue et al., 2023; Føre et al., 2024; Ashraf Rather et al., 2024) that could better reveal the actual genetic potential of the breeding candidates and allow for the most compatible matings.

Conclusions

Neural networks can be a valuable addition to the aquaculturist breeder toolbox. Competitive and, at times, higher prediction accuracies of up to 4 % were obtained using MPLs compared to rrBLUP using simulated datasets. On the other hand, when a publicly available dataset on KHV resistance in carp was used, the tested NNs did not improve previously obtained prediction accuracies. As the size of the training dataset of the above was substantially lower compared to the simulated sets it would be worth investigating in the future the performance of NNs in larger datasets. Finally, the ability of NNs to capture non-linear associations and receive data from different sources (e.g. multi-omics) would be worth exploring.

CRediT authorship contribution statement

Christos Palaiokostas: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of Competing Interest

The author declares no confict of interest

Acknowledgements

The computations and data handling of the study were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Data availability

Data will be made available on request.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z. et al., 2015 TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. arXiv:1603.04467.

Ashraf Rather, M., Ahmad, I., Shah, A., Ahmad Hajam, Y., Amin, A., et al., 2024. Exploring opportunities of Artificial Intelligence in aquaculture to meet increasing food demand. Food Chem. X 22, 101309.

Bargelloni, L., Tassiello, O., Babbucci, M., Ferraresso, S., Franch, R., et al., 2021. Data imputation and machine learning improve association analysis and genomic prediction for resistance to fish photobacteriosis in the gilthead sea bream. Aquac. Rep. 20, 100661.

Bellot, P., de los Campos, G., Pérez-Enciso, M., 2018. Can deep learning improve genomic prediction of complex human traits? Genetics 210, 809–819.

Bradford, H. l, Pocrnić, I., Fragomeni, B. o, Lourenco, D. a l, Misztal, I., 2017. Selection of core animals in the Algorithm for Proven and Young using a simulation model. J. Anim. Breed. Genet. 134, 545–552.

Chollet, F., 2021. Deep Learning with Python. Manning Publications, New York, NY.

Daetwyler, H.D., Calus, M.P.L., Pong-Wong, R., de los Campos, G., Hickey, J.M., 2013. Genomic prediction in animals and plants: simulation of data, validation, reporting, and benchmarking. Genetics 193, 347–365.

Eraslan, G., Avsec, Ž., Gagneur, J., Theis, F.J., 2019. Deep learning: new computational modelling techniques for genomics. Nat. Rev. Genet. 20, 389–403.

Faggion, S., Carnier, P., Franch, R., Babbucci, M., Pascoli, F., et al., 2023. Viral nervous necrosis resistance in gilthead sea bream (*Sparus aurata*) at the larval stage: heritability and accuracy of genomic prediction with different training and testing settings. Genet Sel. Evol. 55, 22.

Føre, M., Alver, M.O., Alfredsen, J.A., Rasheed, A., Hukkelås, T., et al., 2024. Digital Twins in intensive aquaculture — challenges, opportunities and future prospects. Comput. Electron. Agric. 218, 108676.

Geron, A., 2019. Hands-on Machine Learning with Scikit-learn, Keras and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd ed. O'Reilly.

Gjedrem, T., Rye, M., 2018. Selection response in fish and shellfish: a review. Rev. Aquac. 10, 168–179.

Henderson, C.R., 1975. Best linear unbiased estimation and prediction under a selection model. Biometrics 31, 423–447.

Hickey, J.M., Chiurugwi, T., Mackay, I., Powell, W., Hickey, J.M., et al., 2017. Genomic prediction unifies animal and plant breeding programs to form platforms for biological discovery. Nat. Genet. 49, 1297–1303.

Hinton, G.E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov, 2012 Improving neural networks by preventing co-adaptation of feature detectors.

Houston, R.D., Jin, Y.H., Jenkins, T.L., Selly, S.L.C., Martin, S.A.M., et al., 2020. Harnessing genomics to fast-track genetic improvement in aquaculture. Nat. Rev. Genet. 21.

Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv 1502, 03167.

Karras, A., Karras, C., Sioutas, S., Makris, C., Katselis, G., et al., 2023. An integrated GIS-based reinforcement learning approach for efficient prediction of disease transmission in aquaculture. Information 14, 583.

Kause, A., Nousiainen, A., Koskinen, H., 2022. Improvement in feed efficiency and reduction in nutrient loading from rainbow trout farms: the role of selective breeding. J. Anim. Sci. 100, skac214.

Kingma, D.P., Ba, J., 2017. Adam: a method for stochastic optimization. arXiv 1412, 6980.

Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S., 2017. Self-normalizing neural networks. arXiv 1706, 02515.

Legarra, A., Christensen, O.F., Aguilar, I., Misztal, I., 2014. Single Step, a general approach for genomic selection. Livest. Sci. 166, 54–65.

Lindholm, A., Wahlström, N., Lindsten, F., Schön, T., 2022. Machine learning: a first course for engineers and scientists. Cambridge University Press.

de los Campos, G., Hickey, J.M., Pong-Wong, R., Daetwyler, H.D., Calus, M.P.L., 2013. Whole-genome regression and prediction methods applied to plant and animal breeding. Genetics 193, 327–345.

Lourenco, D., Legarra, A., Tsuruta, S., Masuda, Y., Aguilar, I., et al., 2020. Single-Step Genom. Eval. Theory Pract.: Using SNP Chips Seq. Data BLUPF90. Genes 11, 790.

Luo, Z., Yu, Y., Bao, Z., Li, F., 2024. Evaluation of machine learning method in genomic selection for growth traits of Pacific white shrimp. Aquaculture 581, 740376.

Ma, W., Qiu, Z., Song, J., Li, J., Cheng, Q., et al., 2018. A deep convolutional neural network approach for predicting phenotypes from genotypes. Planta 248, 1307–1318.

McElreath, R., 2020. Statistical Rethinking: A Bayesian Course with Examples in R and STAN. Chapman and Hall/CRC, New York.

Meuwissen, T.H.E., Hayes, B.J., Goddard, M.E., 2001. Prediction of total genetic value using genome-wide dense marker maps. Genetics 157, 1819–1829.

Misztal, I., Legarra, A., Aguilar, I., 2014. Using recursion to compute the inverse of the genomic relationship matrix. J. Dairy Sci. 97, 3943–3952.

Misztal, I., Lourenco, D., Legarra, A., 2020. Current status of genomic evaluation. J. Anim. Sci. 98, 1–14.

Montesinos López, O.A., Montesinos López, A., Crossa, J., 2022. Random Forest for Genomic Prediction. In: Montesinos López, O.A., Montesinos López, A., Crossa., J. (Eds.), in Multivariate Statistical Machine Learning Methods for Genomic Prediction. Springer International Publishing, Cham, pp. 633–681.

Montesinos-López, O.A., Martín-Vallejo, J., Crossa, J., Gianola, D., Hernández-Suárez, C. M., et al., 2019a. A benchmarking between deep learning, support vector machine and Bayesian threshold best linear unbiased prediction for predicting ordinal traits in plant breeding. G3: Genes Genomes Genet. 9, 601–618.

Montesinos-López, O.A., Martín-Vallejo, J., Crossa, J., Gianola, D., Hernández-Suárez, C. M., et al., 2019b. New deep learning genomic-based prediction model for multiple traits with binary, ordinal, and continuous phenotypes. G3 Genes Genomes Genet. 9, 1545–1556.

Montesinos-López, O.A., Montesinos-López, A., Hernandez-Suarez, C.M., Barrón-López, J.A., Crossa, J., 2021a. Deep-learning power and perspectives for genomic selection. Plant Genome 14, e20122.

Montesinos-López, O.A., Montesinos-López, A., Pérez-Rodríguez, P., Barrón-López, J.A., Martini, J.W.R., et al., 2021b. A review of deep learning applications for genomic selection. BMC Genom. 22, 19.

Nguyen, N.H., Vu, N.T., Patil, S.S., Sandhu, K.S., 2022. Multivariate genomic prediction for commercial traits of economic importance in Banana shrimp *Fenneropenaeus merguiensis*. Aquaculture 555, 738229.

Palaiokostas, C., 2021. Predicting for disease resistance in aquaculture species using machine learning models. Aquac. Rep. 20, 100660.

Palaiokostas, C., Vesely, T., Kocour, M., Prchal, M., Pokorova, D., et al., 2019. Optimizing genomic prediction of host resistance to koi herpesvirus disease in carp. Front. Genet. 10, 543.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., et al., 2011. Scikit-learn: machine learning in python. J. Mach. Learn. Res. 12, 2825–2830.

Pérez, P., de los Campos, G., 2014. Genome-wide regression and prediction with the BGLR statistical package. Genetics 198, 483, 95.

Pérez-Enciso, M., Zingaretti, L.M., 2019. A guide on deep learning for complex trait genomic prediction. Genes 10, 553.

Pocrnic, I., Lourenco, D.A.L., Masuda, Y., Legarra, A., Misztal, I., 2016a. The dimensionality of genomic information and its effect on genomic prediction. Genetics 203, 573–581.

Pocrnic, I., Lourenco, D.A.L., Masuda, Y., Misztal, I., 2016b. Dimensionality of genomic information and performance of the Algorithm for Proven and Young for different livestock species. Genet. Sel. Evol. 48, 82.

Pook, T., Freudenthal, J., Korte, A., Simianer, H., 2020. Using local convolutional neural networks for genomic prediction. Front. Genet. 11, 561497.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536.

Sandhu, K.S., Aoun, M., Morris, C.F., Carter, A.H., 2021. Genomic selection for end-use quality and processing traits in soft white winter wheat breeding program with machine and deep learning models. Biology 10, 689.

Sargolzaei, M., Schenkel, F.S., 2009. QMSim: a large-scale genome simulator for livestock. Bioinformatics 25, 680–681.

Song, H., Dong, T., Yan, X., Wang, W., Tian, Z., et al., 2023a. Genomic selection and its research progress in aquaculture breeding. Rev. Aquac. 15, 274–291.

Song, H., Dong, T., Yan, X., Wang, W., Tian, Z., et al., 2023b. Using Bayesian threshold model and machine learning method to improve the accuracy of genomic prediction for ordered categorical traits in fish. Agric. Commun. 1, 100005.

Vandeputte, M., Corraze, G., Doerflinger, J., Enez, F., Clota, F., et al., 2022. Realised genetic gains on growth, survival, feed conversion ratio and quality traits after ten generations of multi-trait selection in rainbow trout *Oncorhynchus mykiss*, fed a standard diet or a "future" fish-free and soy-free diet. Aquac. Rep. 27, 101363.

Varona, L., Legarra, A., Toro, M.A., Vitezica, Z.G., 2018. Non-additive effects in genomic selection. Front. Genet. 9, 78.

Vu, N.T., Phuc, T.H., Oanh, K.T.P., Sang, N.V., Trang, T.T., et al., 2022. Accuracies of genomic predictions for disease resistance of striped catfish to Edwardsiella ictaluri using artificial intelligence algorithms. G3 Genes Genomes Genet. 12 (1), jkab361, 4.

Waldmann, P., 2018. Approximate Bayesian neural networks in genomic prediction. Genet. Sel. Evol. 50, 1–9.

Waldmann, P., Pfeiffer, C., Mészáros, G., 2020. Sparse convolutional neural networks for genome-wide prediction. Front. Genet. 11, 25.

Wang, K., Yang, B., Li, Q., Liu, S., 2022. Systematic evaluation of genomic prediction algorithms for genomic prediction and breeding of aquatic animals. Genes 13, 2247.

Wilmott, P., 2019. Mach. Learn. Appl. Math. Introd. Panda Ohana 1–219.

Xu, B., N. Wang, T. Chen, and M. Li, 2015 Empirical Evaluation of Rectified Activations in Convolutional Network.

Xue, Y., Bastiaansen, J.W.M., Khan, H.A., Komen, H., 2023. An analytical framework to predict slaughter traits from images in fish. Aquaculture 566, 739175.

Zhao, T., Fernando, R., Cheng, H., 2021. Interpretable artificial neural networks incorporating Bayesian alphabet models for genome-wide prediction and association studies. G3 Genes, Genomes, Genet. 11 (10), jkab228.

Zhao, T., Zeng, J., Cheng, H., 2022. Extend mixed models to multilayer neural networks for genomic prediction including intermediate omics data. Genetics 221 (1), iyac034.