Chapter 4

Prediction and inference on non-linear genetic effects using neural-networks

4.1 Chapter 4 outline

Parallel to my work described in the preceding chapter where I used regression based methods to search for statistical epistasis, I also explored the potential of neural-network (NN) based methods to infer evidence of epistasis indirectly. NNs perform a non-exhaustive random search for interactions between the input features, and their performance is evaluated by predictions in held out data, which is a different standard of evidence than the null hypothesis testing approach that regression based methods rely on.

Section 4.2 reviews relevant previous work, and provides the necessary background on the NN architectures and algorithms that I used in the rest of the chapter. Section 4.3 describes how these methods were applied to synthetic data, where I confirmed via a large-scale simulation study the potential of NNs to be able to infer interactions at a higher accuracy than standard regression based methods. Section 4.4 covers the application of the same NN approaches to the cohorts I prepared and analysed in Chapter 3.

4.2 Neural-networks in genomics

4.2.1 Relevant previous work

The key advantage of NNs is that they can approximate complex non-linear functions and model higher-order interactions between input features, without performing an exhaustive search. For NNs to perform well, there must be a substantial non-linearity in the underlying problem, and the training data needs to be sufficiently large for the NN to learn this non-linearity (Hestness et al., 2017).

As I described earlier in the Introduction (section 1.1.2), non-linearity (epistasis) may occur between individuals in the form of statistical epistasis, or within a genome in the form of functional epistasis. To date in genomics NN have been most successful in inferring functional epistasis. Here, NN classifiers are trained to learn the relationships between labels (such as TF binding) and DNA sequence context. A successful example in this area was the 'DeepSEA' network (Zhou and Troyanskaya, 2015), which was a shallow convolutional neural-network (CNN) classifier that could accurately predict the presence or absence of regulatory elements in a given nucleotide sequence. A similar model was more recently applied to quantify the consequence of non-coding mutations to autism spectrum disorders by Zhou et al. (2019).

Applications of NNs to infer the existence of statistical epistasis have been less convincing. Here, the disease phenotype is modeled directly from genetic differences between individuals; thus, these models are the NN analogues of PRS generation methods. A few recent applications in the field of agricultural science showed early promise. A NN model by Ma et al. (2017) outperformed baseline linear methods in bread wheat yield prediction by up to 65%. Another NN effort found a ~24% improvement over linear methods by deploying a locally connected CNN model to predict plant traits (Pook et al., 2020). However, similar reliable positive results in humans have been lacking. When I started my PhD, most relevant studies relied on small GWAS cohorts and pre 'deep learning' era models (Motsinger-Reif et al., 2008). Parallel to my own work, in the last few years there have been a number of attempts that relied on larger datasets and more modern NN models.

One of the earliest attempts from the more modern efforts was a study by Montañez et al. (2018), which aimed to predict the obesity phenotype using a NN based PRS. They claimed that from a sample size of only ~2000 individuals their NN model was able to predict obesity with a near 100% accuracy, a trait which is not a 100% heritable. Additional issues of this study included a lack of comparison to baseline linear methods, and also that they did not consider haplotype effects. Their choice of 2,465 SNPs was selected based on a simple additive association p-value filtering step with no consideration given to LD. There were another two more recent recent studies that utilised the UKBB cohort, which were also more comparable in scope to my own work.

The study by Bellot et al. (2018) used the smaller interim UKBB release (~150K individuals) on quantitative traits (including height and BMI) to investigate the relative performance of NN and linear methods for building PRS. A notable difference between their approach and mine was their treatment of LD when selecting SNPs to be included in the PRS. Instead of trying to eliminate haplotype effects, which could create non-biologically meaningful statistical epistasis via filtering (Wood et al., 2014), they took the position that incorporating them into the model may improve prediction. As a consequence, their NN models were smaller (maximum 128 neurons), but the number of SNPs considered was larger (~10-50K) than what I had in my analyses. However, their approach of incorporating LD, correlations between SNPs (a linear effect), into prediction was a very different goal than my own objective to find evidence for non-linear genetic effects that contribute to phenotypic variance. They found that their NN models did not outperform the linear baselines; however, even if they would have, their approach could not have been used to find evidence for epistasis, as any potential benefits over the linear baselines could also have been due to haplotype effects. Indeed, if the main benefit of NNs would be to improve PRS by modelling LD, I would argue that it would be more effective to use LDpred instead, as the latter method can accommodate ~1 million markers (Vilhjálmsson et al., 2015), rather than just ~50K, the maximum number that their study considered.

The most recent study that made use of the full UKBB cohort, investigated the potential of NNs to build PRS for blood cell traits (Xu et al., 2020). This study was more similar to my own project with respect to variable selection. However, instead of filtering on a recombination block basis, they used conditional analysis to only keep SNPs that represented unique signal. This filtering process left their study with fewer SNPs (160 - 762) than what I used for my own analyses (450 - 1,732). It is important to note, that their variable selection approach may not have fully controlled for haplotype effects, as they found that when they removed SNPs in their models close together on chromosomes 3, 6 (HLA) or 16, this resulted in a deterioration in prediction accuracy of PRS built by their multivariate linear models (that did not include the interaction terms). Subsequently, when explicitly tested, the interaction terms of these removed variants were found to be significant. Taken together, the behaviour of these SNPs is consistent with haplotype effects that could generate spurious statistical interactions as described by Wood et al. (2014). The conclusion of their study was also similar to that of Bellot et al. (2018), as they found that NNs did not perform better than linear PRS building methods.

A common limitation of all previously described studies that aimed to build PRS with NNs was the attitude they took on using NNs to infer the nature of potential non-linear effects. Specifically, they asserted that a NN could outperform conventional approaches because it would learn non-linear genetic effects. However, just because NNs are capable of learning non-linear functions, there is no guarantee that they will detect non-linear effects if they are too weak, or especially, if there are none. Also, even if NNs did learn non-linear effects, unless addressed on explicit terms, there is no guarantee that they were not due to haplotype effects, or some of the other known artefacts that could generate statistical epistasis (Fish et al., 2016; Wood et al., 2014). Finally, none of the previous studies so far attempted to look for non-linear effects across genomic domains nor have they tried to perform inference using NNs to identify individual interactions.

4.2.2 Opportunities and challenges for NNs in genomics

The reasons why fully-connected NNs (FNNs) and CNNs have faced difficulties in genetic prediction lie in the differences between the domains in which NNs have been applied to successfully, such as images, and genomics. Image labels are predicted from pixel data, where convolutional layers may learn reusable features (I discuss convolutions in more detail in Appendix B in section B). However, convolutions do not fit GWAS data well, and reusable filters are unlikely to be learned from SNPs. The subtle reasons for this have to do with how GWAS SNP data and DNA sequence data (or pixel data) differ. In the domain of sequence classifiers, convolution filters may represent regulatory motifs. However, as SNP data only captures deltas from a reference genome, a series of SNPs do not carry any intrinsic meaning as their sequence context is not preserved. To clarify, a $3x3$ filter in an image classification task may represent an edge. Similarly, for a sequence classifier a motif pattern (example: '*GCA*') may represent a (partial) transcription factor binding site, both of which are reusable features that are likely to carry the same meaning in other areas of the input. In contrast, a pattern of three SNPs (example: '021') with the same values is unlikely to have the same meaning elsewhere. Two SNPs next to each other in the genotype matrix may refer to different types of nucleotides with an arbitrary distance between them. Indeed, both studies (Bellot et al. (2018); Xu et al. (2020)) that empirically evaluated the applicability of convolutions to SNP data found that CNNs perform at a level below FNNs.

An additional important difference between image classification and genetic prediction is that an image's label depends only on the pixels in the image itself; however, complex diseases are never a 100% heritable, and what is not heritable is also not predictable from SNP data. Therefore, the predictive ceiling is correspondingly lower (broad sense heritability).

Finally, images are typically made up of only a few thousand pixels (after downsampling), whereas SNP data ranges from the hundreds of thousands to millions of predictors. In addition, while image data is abundant and cheap, genomic data is still relatively scarce and expensive. These last two factors mean that typical image classification problems have very high samples to predictors ratios (an essentially infinite sample size), whereas in most GWAS there are far more SNPs than individuals. This poses a continued challenge for NNs, as state of the art PRS typically consists of ~500K SNPs (Khera et al., 2018), a dimensionality that would seem difficult for FNNs of the current generation. Table 4.1 summarises the differences between the typical domains where NNs succeed, such as images, and genomics.

	Image classification	**Genomics**
n >> p	YES	NO
prediction ceiling	100%	H^2
noise	NO	1 - H^2
typical predictive accuracy	>95%*	< 10% *
main challenge	problem complexity	low power

Table 4.1 **Summary of the differences between typical image classification and genetic prediction tasks.** n is the number of observations and p is the number of input features. H^2 is broad-sense heritability. *Accuracies taken from He et al. (2016) and Lee et al. (2018) for images and PRS, respectively.

In summary, the challenges that NNs face in genomics originate from the differences in the domains where they traditionally excel, and typical genetic data. However, given their strengths at modeling highly non-linear functions, they also offer potential for revealing the higher-order encoding of genetic information of complex traits. This attribute, taken together with the application of careful QC measures, and the recent availability of large genotyped cohorts such as the UKBB dataset, offer new hope that the challenges that held NN back in human genetics so far can be surmounted.

4.2.3 Neural-network models and data preparation

The NNs used in this chapter were all FNN based models that I described in detail in the Introduction in section 1.7.2. To ensure model stability, and to speed up training, all input data (genotype and gene-level predictors) and phenotypes were standardised to have a zero mean and unit variance based on the training set (LeCun et al., 2012).

4.2.3.1 Choosing the model architecture

A NN model's hyperparameters such as regularization, activation functions, the number, type and size of layers, are collectively known as the architecture of the NN. The ideal architecture is determined by the complexity of the function modeled (the number of input features and the degree of non-linearity between them), and the limits of the computational resources available (the RAM, GPU capabilities and time limits on a computing cluster).

I believe that, if possible, it is better to let the data guide the model hyperparameter selection, rather than to impose my own prior beliefs. Therefore, the only restriction I set on

parameter	**range**	**type**	**description**
first layer size	[100 - 4000]	int	number of neurons in the first layer of the network
epochs	[10 - 100]	int	number of training iterations
#hidden layers	[1 - 20]	int	number of hidden layers between input and output
dropout	[0 - 0.9]	real	percent of neurons to be deactivated at each iteration on all layers
learn rate	[10^{-5} - 0.01]	real	the learning rate at which parameters are changed between epochs
activation layer	SELU / linear	logical	if the non-linear capacity of the network is enabled

Table 4.2 **Summary of the search-space covered by the hyperopt tool.** The SELU activation function is described in the Introduction in section 1.7.4.4.

the NN architecture was that each subsequent hidden layer would be half the size of the one preceding it (a common design pattern employed to reduce the number of hyperparameters of a FNN model). The rest of the hyperparameters were determined by employing a semi-random search performed by the package '*hyperopt*' (Bergstra et al., 2013). I defined a range of possible hyperparameters (Table 4.2), which I based on the limits of the computational resources available, such as the RAM and time limits on the computing cluster. Then, I defined the r^2 between predicted and observed outcomes in the validation set as the criteria for hyperopt to optimise on. Hyperopt was then set to find the best performing hyperparameters in a pre-specified number of trials, which for computational time limit considerations, I set to 50. All models were trained via the ADAM optimizer that I described in the Introduction in section 1.7.4.1 (Kingma and Ba, 2014). Finally, to reduce overfitting, I also employed the early stopping mechanism (Prechelt, 1998) by recording the epoch at which the NN performed at the highest accuracy on the validation set, and after the initial training finished I retrained the model up until this epoch. The model weights for this best performing epoch were saved and reused for the subsequent evaluation on the Test Set. Expression 4.1 summarises the overall NN architecture in a shorthand notation:

$$NN : [In, FC_1, \sigma, DO, FC_2, \sigma, DO, \ldots, FC_k, \sigma, DO, Out], \tag{4.1}$$

where FC, DO and σ denote the k fully-connected layers, the dropout layers and the SELU activation functions, respectively.

4.2.4 NN methods used in this chapter

4.2.4.1 Using NNs to evaluate the evidence for non-linearity

As previously described in section 4.2.1, a common shortcoming of projects similar to mine were assertions about the supposed non-linearity the NN models would have learned from SNP data (Montañez et al., 2018; Xu et al., 2020). However, without explicitly assessing the model for non-linearity such claims lack evidence.

To test if my own NN models have learned any non-linear effects from the data, I evaluated the following two-tier strategy. The previously described hyperopt model selection process had the option to choose between a linear and a non-linear activation function for better prediction performance on the held-out validation set. Additionally, if a non-linear solution was selected, I also evaluated the final model with and without the non-linear function enabled, by turning on and off the activation layers for the final test prediction. To demonstrate why this removes any potential non-linearity from a NN model, consider the original NN equation I presented in the Introduction (1.39):

$$Y = \sigma_k(\ldots\sigma_2(\sigma_1(\mathbf{XW_1})\mathbf{W_2})\ldots W_k). \tag{4.2}$$

where Y, $\mathbf{X}$ and $\mathbf{W}$ denote the phenotype column vector, the SNP input and the learned model weights, respectively. Then, consider the following linear NN which may be derived from the above non-linear NN by switching off all activation functions (σ) as

$$\begin{aligned}
Y &= \cancel{\sigma_k}(\ldots\cancel{\sigma_2}(\cancel{\sigma_1}(\mathbf{XW_1})\mathbf{W_2})\ldots W_k) && (4.3)\\
&= (((\mathbf{XW_1})\mathbf{W_2})\ldots W_k) && (4.4)\\
&= \mathbf{X}(\mathbf{W_1W_2}\ldots W_k) && (4.5)\\
&= \mathbf{X}W_{all}. && (4.6)
\end{aligned}$$

As the two models are identical in every other respect, including the same weights, if (4.6) is at least as accurate as (4.2), then the latter could not have learned interactions between the input features.

After some initial test runs, I found that the performance of the previously described linear NN models was poor due to two different reasons, other than the lack of non-linearity. The weights for the non-linear NN were obtained via training with non-linearity enabled; thus, they may not have been ideal for a model that never had an activation function in the first place. Additionally, my preferred choice for activation (the SELU function), in addition to providing non-linearity, also standardises the output of each layer. This latter property addresses the internal covariate shift problem I described in the Introduction in section 1.7.4.3. Thus, removing the SELU, and not providing a substitute for its normalization capacity may produce sub-optimal linear models. Therefore, to ensure that I fit the best possible linear NN with an architecture closest to the non-linear version, I made the following two changes. I replaced the SELUs with a batch normalization layer (Ioffe and Szegedy, 2015), and I also retrained the linear NNs with early stopping applied to obtain the best possible weights for these models too.

4.2.5 Inference via neural-networks

In the field of the biomedical sciences point estimates are often inadequate, and an explanation as of why a certain prediction was made, or at what confidence level, are considered highly important. Because of the complexity of the models and the non-linear functions they learn, NNs have traditionally been thought of as 'black boxes' that are unable to satisfy this criterion. However, inference for NN based methods is an emerging area of research; thus, this perception has been slowly changing.

4.2.6 Overview of my NN inference strategy

I will provide a detailed background on each component of my approach for inference in the subsequent sections; however, I will first outline my overall strategy, so that each individual component may be understood in the greater scheme of my analysis. Inference begins by training the NN model to the highest possible prediction accuracy after which the NN is then taken forward to the association stage. Interaction association is then performed for each order of interaction (two, three and four), starting from the lowest, and proceeding upward in a three-step process that I will outline next.

The first step in inference is interaction association, where each method attempts to identify combinations of predictors that were most influential for the NN's performance on the original prediction task. This step is performed via either the NID algorithm (section 4.2.6.3) or my own NNPred algorithm (section 4.2.6.5). This is followed by the second step that consist of significance testing, where the previously identified putative interactions are evaluated against the null hypothesis of no association. For the NID approach this is performed via OLS based techniques, and for the NNPred method this is performed via the application of the dropout technique (section 4.2.6.1). Once all candidate interactions have been identified for a given order, their p-values are FDR corrected, and only those below a 0.05 threshold are taken forward. Finally, the last step consists of a common search-space reduction strategy (section 4.2.6.7), that reduces the number of tests the methods need to perform based on the heuristic of only considering those interactions deemed to be possible, given the previously discovered associations.

4.2.6.1 Uncertainty estimation via dropout

In image classification tasks where NNs produce a list of predictions to indicate the probability that a given image belongs to a certain class, one may naively expect a prediction of low confidence to have near equal predictions for all categories. However, this does not precisely quantify the uncertainty of the model. For regression based prediction tasks the problem of lack of uncertainty estimation is even more pronounced. As for regression tasks, the model only produces a real value of a single point estimate, which does not reveal anything about how confident the model was when making that particular prediction.

As I described in the Introduction in section 1.7.4.8, the dropout technique is traditionally applied during training only, and is usually switched off for test predictions. However, it was recently proposed that applying dropout at test-time induces the NN to approximate Bayesian inference in deep Gaussian processes (Gal and Ghahramani, 2016). The intuition behind this is that the application of the binary masks that switch off random subsets of neurons may be considered to create an ensemble of NNs; thus, a prediction with dropout enabled is taken as a single observation from their distribution. The implementation of this technique is very straightforward: during test time, dropout is simply not switched off. Thus, for any given test case, instead of just producing a single prediction, potentially many thousands of predictions may be produced, which are taken as the empirical distribution for that prediction. Initially, it was believed this method would only be feasible for FNNs, and not for CNNs, due to the traditionally poor performance of dropout on convolution layers; however, later it was shown that state of the art performance may be achieved if during testing the mean of this distribution is used as the point estimate, and its standard deviation as the standard error of the estimate (Gal and Ghahramani, 2015).

Deploying dropout for uncertainty estimation may be accomplished at no extra computational cost, as the same test observation may be added repeatedly into a mini batch; thus, all observations from the distribution may be obtained via a single forward propagation pass. Whether using dropout in this manner produces a genuine approximation for Bayesian inference is still subject to debate (Osband, 2016); however, this technique has become a popular method for uncertainty estimation in practice.

4.2.6.2 Estimating the importance of input features

Another important objective of inference is to obtain a list of input features (or their combinations) that were most influential in generating the model prediction. This is a challenge, as NN models fit a non-linear model where combinations between input features are considered randomly in a non-exhaustive search to yield the best overall prediction. Additionally, the learned interactions are not stored in individual neuron weights; instead, they are distributed across the network, where each neuron learns only very small fragments of the overall task. This phenomenon is known as distributed representations (Hinton, 1984).

Approaches that permit one to query a trained NN to produce association-like results are reviewed in the following sections. These may be placed into two broad categories, examining the learned NN weights directly, and inference-via-prediction type approaches (there exist a third approach which is to build Visible NNs (Yu et al., 2018); however, as I have not used this for my own work I will not describe this here).

4.2.6.3 Examining the learned weights of the network directly

These approaches obtain inference by attempting to interpret the weights of a trained NN directly. Most relevant to my work from this class of methods is the NID algorithm by Tsang et al. (2017), which was developed to identify statistical interactions between input features. Briefly, the method proposes to learn interactions from the weighs of the first hidden layer's neurons directly. Their algorithm computes I, the interaction strength of a candidate combination, by

$$I = S_i * min(U). \tag{4.7}$$

Here, the strength of the interaction is estimated by weighting the total output influence of the neuron (S_i) by the top most important inputs to that neuron ($min(U)$). The vector S is defined as

$$S = |W^{out}|^T |\mathbf{W^j}|^T |\mathbf{W^{j-1}}|^T \dots |\mathbf{W^2}|^T. \tag{4.8}$$

Thus, S is a matrix product of all the absolute weights from the output back to the second hidden layer. This produces a vector of length a, where a is the number of neurons in the first hidden layer. Intuitively, this quantifies the total influence of the first hidden layer on the output. So in turn, the element i of the vector S isolates the output influence of neuron i. U is a set $\{1,2,\dots d\}$, defined as the top d largest absolute value elements of the vector $|\mathbf{W^1_i}|$ (the column corresponding to the ith neuron in the first layer's weights), where d is the order of interactions considered. The minimum operation is applied to U, as the total strength of an interaction would be zero, should any individual input feature have zero weight in the first layer.

The algorithm queries each neuron in the first layer, and evidence of association is evaluated based on if a candidate interaction improves the overall model fit on a validation set. A limitation of this method is that the maximum number of interactions it may consider is capped to the number of neurons in the first layer.

4.2.6.4 My NID implementation

The only difference between my implementation and the authors' was that I used regression based models to evaluate the evidence for the strength of association instead of additional NNs. In the original implementation, a NN is fit for each candidate interaction; however, in practice, this would have been infeasible for the scale of my analyses. Also, a NN model of only a few features would not represent a real advantage over a linear model where the interactions are explicitly coded into the design matrix. Thus, my routine for finding a threshold for determining the number of candidate interactions was via forward step-wise regression. I added each candidate interaction, together with their lower-order terms, sequentially into a multiple regression OLS model, and determined the cutoff as the last putative interaction above which the r^2 of the PRS stopped improving on the validation set.

4.2.6.5 Inference-via-prediction

Methods that belong to this class achieve inference via the manipulation of the weights of the model to generate predictions that are informative of feature importance. To explain the rationale behind these techniques, I briefly return to (logistic) regression. There, the slope of the model (β_{OLS}) provides direct interpretability on feature importance. However, because of the non-linearity, the weight matrices of NNs cannot be used directly in the same manner. Instead, as it was shown by Simonyan et al. (2013), the NN analogue of the β_{OLS} is the derivative of the network with respect to the input:

$$\beta_{OLS} \sim \frac{\partial}{\partial \mathbf{W_{in}}}, \tag{4.9}$$

where the above derivative was obtained via the procedure I described in the Introduction by eq 1.41. Intuitively, this quantifies which input predictors would need to be changed the least to affect the final classification the most.

Inference can be obtained on feature importance in two different ways. One approach is called *input-centric* inference, which aims to answer the question: what in a given input contributed most to the final prediction by computing

$$\hat{y}_j = x_j \frac{\partial}{\partial \mathbf{W_{in}}}, \tag{4.10}$$

where the resulting predictions ($\hat{y}_j$), known as heat or saliency maps, are obtained by multiplying a specific input x_j by the network derivative. In the image classification domain, it was found that the clarity of predictions may be improved by adding a small amount of Gaussian noise to x_j, and then taking the average over many predictions to produce the final inference, a technique known as '*SmoothGrad*' (Smilkov et al., 2017).

The other approach is called *network-centric* inference, and this aims to reveal what the NN has learned about a prediction task in a general, input-independent way. This may be implemented by an algorithm known as '*Gradient ascent*', where the derivative of the network is iteratively added to an input over many iterations as

$$x_i = x_{i-1} + \frac{\partial}{\partial \mathbf{W_{in}}_i}. \tag{4.11}$$

In the first iteration, the input (x_0) is initialised with random noise, and the derivative is computed against the desired target class or value for each iteration. One may intuitively understand this process as generating an 'ideal case' for the classifier (this same mechanism was also responsible for the popular imagery behind 'deep dreaming' (Mordvintsev et al., 2015)). To improve on this basic formula, the derivative may be modified by altering the backpropagation algorithm by zeroing out the values of neurons whose derivative was negative. This modification is known as '*Guided Backpropagation*' (Springenberg et al., 2014), and is motivated by the fact that neurons with negative derivatives would only contribute noise to the final classification.

4.2.6.6 My inference-via-prediction implementation

As I have shown in section 4.2.6.2, NN methods that attempt to provide interpretability accomplish it mostly by either dissecting the NN to extract information directly from its weights, such as the NID algorithm, or via the manipulation of the forward/backward propagation process to induce a prediction more informative on the importance of input features, such as the 'guided backpropagation' method.

The problem with the aforementioned approaches is that NNs learn distributed representations, where the learned features are distributed across many neurons (Hinton, 1984). Thus, focusing in on any particular neuron or neurons, hoping that they may reveal information about the reasons behind a prediction, remains challenging. This is especially true for FNNs, where neurons are not restricted to a subset of the input, as in that case all neurons learn from all input features without restriction. There have been attempts to force neurons to learn 'localist' representations by the application of a special type of regularization. Such regularization forces the NN weights to become orthogonal with respect to other neurons, and thereby 'disentangle' representations (Brock et al., 2016; Rodríguez et al., 2016); however, these techniques have not found widespread use yet. The other issue that render these methods unsuitable for my purposes is that they lack the per-predictor precision that one would expect from traditional statistical inference. To clarify, in images these methods produce a heatmap-like inference, which when overlaid on the original input picture, highlight areas that were relevant for the classification. In the image classification domain, this may be sufficient for visually determining what objects in the image were important (Smilkov et al., 2017; Springenberg et al., 2014). However, to obtain precise inference about individual predictors or their combinations, for example SNPs, such an approach would not have been feasible.

I reasoned that, for lower-order interactions, there may be a much simpler and more powerful solution. My rationale for this was the insight that the only location where a NN is forced to produce a human-interpretable result is the output itself. Thus, instead of trying to force the NN models into something that they were not designed for, I opted for obtaining inference-via-prediction by simply observing the phenotype prediction for inputs that only consisted of the candidate interactions. The implementation of my algorithm (*NNPred*) is presented below:

Algorithm 1 NNPred Interaction search algorithm

```
input: trained NN classifier $S_c$
output: list of importance scores $IS$
1: procedure INTERACTION-SEARCH($S_c, c$)
2:     $IS \leftarrow 0$                    ▷ initialise empty array for importance scores
3:     for $i$ in number of $SNP_{set}$ do
4:         $x \leftarrow 0$                 ▷ initialise empty array in the shape of the input data
5:         $x[SNP_{set}[\text{i}]] = 1$     ▷ set each SNP in tested SNP-set to 1
6:         $\hat{y} = S_c(x)$               ▷ forward propagate to obtain phenotype prediction
7:         $IS[i] = \hat{y}$
8:     return $IS$
```

The NNPred algorithm cycles through all possible interactions, and generates a synthetic individual for each, where all input features are zeroed out except the candidate interaction itself. This observation is then forward propagated to produce a phenotype prediction, which is then taken as the evidence for interaction association.

To provide an estimate of uncertainty for each association, I used the dropout method that I described in detail in section 4.2.6.1 (Gal and Ghahramani, 2016). Briefly, this entailed producing a mini-batch number of test predictions with dropout enabled, and taking these as observations from the empirical distribution of the model's prediction. I then used this distribution to obtain p-values for the NN estimate via the usual formulas:

$$\begin{aligned} \beta_{NN} &= \frac{\Sigma \hat{y}}{L}, \\ \sigma_{NN} &= \sqrt{\frac{\Sigma(\hat{y}_l - \beta_{NN})^2}{L}}, \\ t &= \beta_{NN}/\sigma_{NN}, \end{aligned} \tag{4.12}$$

where β_{NN} is the NN estimate of the interaction's effect size, and $\hat{y}$ is an individual prediction out of a total of L predictions in a minibatch. σ_{NN} represents the standard error of the estimate, which is then used to obtain t, the quantile of a normal distribution. Finally, t was used to obtain the appropriate p-value.

The central limit theorem states that the sampling distribution of sample means asymptotically tends to a normal distribution. To ensure that this held true for my datasets, I performed the following test in the simulated experiments (described under 4.3). Using a 1,000 predictions of 1,024 observations each, I performed a Kolmogorov-Smirnov test against normal distributions of the same mean and standard deviation. I obtained a median p-value of 0.588, which confirmed that the assumption of normality was appropriate.

4.2.6.7 Common search space reduction strategy

An exhaustive search for higher-order interactions may quickly become computationally infeasible. Evaluating all possible combinations from even just a 1,000 SNPs up to the fourth-order would require over 40 billion tests. Thus, to reduce the search space, I applied a heuristic I described in the Introduction in section 1.1.7. In brief, this entailed only testing Dth order interactions if all nested $D-1$th order interactions were previously found by the algorithm.

An interaction was deemed to exist based on the following criterion. After a complete search for a given order of interaction, the association test p-values were all Benjamini-Hochberg FDR corrected (the number of tests were always in the thousands; thus, the

FDR correction was appropriate). Candidate interactions were considered valid if their $FDR < 0.05$, a stringent threshold which was motivated by the above described assumption, that an interaction may only exist if all its nested interactions also exist. Also, the FDR correction and filtering step was applied to each order of interaction just once, not repeatedly to all interactions found up until that point.

4.2.6.8 OLS baseline

To serve as a frame of reference for the NN based inference approaches, I also evaluated a standard OLS regression based interaction test. This model was almost identical to the one described in the Introduction (eq 1.7). The only change in this method was that, instead of just testing for second-order interactions, I extended the same model up to the fourth-order by adding the appropriate higher-order terms.

4.3 Simulation experiments on synthetic data

To assess if a NN based strategy was capable of analysing data at the scale of the UKBB, and also to have the potential to identify interactions, I performed a set of simulation experiments. The objective of these tests was to serve as a proof of concept if NNs may be used to infer the existence of statistical epistasis, if it was present.

4.3.1 Genotype dataset

I selected the same FIS genotype panel of 955 SNPs that I used for the two-way interaction tests in Chapter 3. I chose this particular panel (as opposed to the larger height or BMI panels) due to the practical considerations of the immense computational resource requirements needed to perform simulations at the scale of the UKBB. The number of individuals used in these experiments were 137,088, 34,270 and 21,775 for the training, validation and test sets, respectively.

4.3.2 Phenotype simulation details

To obtain conclusions from simulations that may offer insight for my subsequent real data analyses, I aimed to obtain simulated phenotypes that arose from a signal comparable in magnitude to the one observed in the real FIS datasets.

To begin, I had to consider the unique properties of simulations that involve epistasis, as here, there are potentially two distinct parameters that control the way causal SNPs contribute to the phenotype. The first parameter is the causally involved number of SNPs c, which would be the only parameter needed for simulating additive phenotypes. Here however, there may also be a second parameter v, which would control the number of interactions made up from the c SNPs. To determine the causally involved number of SNPs (c) to generate the simulated phenotypes, I evaluated three potential causal fractions of the 955 SNPs: 0.25, 0.5 and 0.95. The upper limit (0.95) for this was motivated by my QC process that involved an $FDR < 0.05$ filter, which implied that ~95% of SNPs were associated with the phenotype. After empirically evaluating three values for v, (c, $c/2$ and $c*2$), I set $v = c$, based on preliminary observations that while v had an impact on the overall accuracy, it did not seem to influence the preference between the linear and non-linear methods. Therefore, to reduce the space for my simulations, I did not pursue experiments that involved alternative choices for v. Thus, the raw genetic values (GV) for individual j were calculated as

$$GV^j = \sum_i^v X_i^j \beta_i \quad ; \quad \beta_i \sim N(0,1), \tag{4.13}$$

where β_i was drawn from a standard normal distribution, and X_i^j denotes the v randomly selected combinations of SNP genotype counts selected to be causal. I simulated four architectures that ranged from the purely additive to second, third and fourth-order interactions. Thus, X_i^j was defined as

$$X_i^j = \prod_d^{D_i} SNP_{dj}, \tag{4.14}$$

where SNP_{dj} is the genotype count for the dth SNP in the ith Dth-order interaction.

Using LDAK (Speed et al., 2012), I estimated the narrow sense SNP heritability of the 955 SNPs to be ~8.1% for the real FIS phenotype. I chose LDAK as opposed to GCTA, as the latter was incapable of working with a kinship matrix of 137,088 individuals due to RAM limitations.

To simulate phenotypes with an additive genetic architecture with a pre-defined h^2, the final phenotype (y_{sim}) is determined to be the sum of the genetic (g) and noise (e) components as

$$y_{sim} = g + e, \tag{4.15}$$

where both g and e are scaled in proportion to the desired h^2. The noise component is drawn from a standard normal distribution, with zero mean and a variance of $1 - h^2$

$$e \sim N(0, \sqrt{1-h^2}). \tag{4.16}$$

Thus, the noise contributes all the remaining variance not due to h^2. The scaled genetic value (g) is in turn defined as

$$g = GV * s, \tag{4.17}$$

where s is a scaling factor given by

$$s = \sqrt{\frac{h^2}{var(GV)}}, \tag{4.18}$$

where $var(GV)$ denotes the sample variance of the genetic values. The above would generate a simulated phenotype, arising from additive genetic effects, with a pre-specified level of h^2 (Speed et al., 2012). However, generating such an additive phenotype was not my objective; instead, I was interested in if the observed additive genetic architecture may have been generated by latent non-linear effects. Thus, the above formula would not have been expected to create a phenotype with a given narrow sense heritability, if in reality the phenotype arose from non-linear effects. To test this, I generated a phenotype as above with the desired 8.1% h^2 for all four interaction levels, and estimated the h^2 once again with LDAK. I found that their actual estimated h^2 was 0.077, 0.05, 0.03 and 0.017 for additive, second, third and fourth-order interactions, respectively. These values indicated that the apparent additive signal was rapidly diminishing at higher-orders interactions. Therefore, I decided to modify the original formula.

I reasoned that the scaling factor (s) needed to be adjusted to be proportionate to the apparent additive effect of SNPs. I attempted to adjust it by fitting a multiple linear regression model, and regressing the individual genetic values (GV) on the genotype matrix as

$$GV = \mathbf{X}'\beta' + \varepsilon, \tag{4.19}$$

where $\mathbf{X}'$, β' and ε denote the genotype matrix of the individual SNPs involved in interactions, the interaction coefficients and a random noise term, respectively. I reasoned that the fitted values from this model ($\widehat{GV}$), would represent the genetic values due to the apparent main effects of the SNPs involved in interactions. Therefore, I proceeded to alter eq 4.18 by using this new $\widehat{GV}$ to produce an adjusted scaling factor by

$$s' = \sqrt{\frac{h^2}{var(\widehat{GV})}}. \tag{4.20}$$

From this point onward, with the exception of using this new scaling factor s', the rest of the simulation steps remained identical to those previously described.

% of sample size	additive	2nd order	3rd order	4th order
10%	0.43	0.42	0.50	0.54
25%	0.27	0.27	0.31	0.55
50%	0.25	0.29	0.41	0.71
75%	0.34	0.25	0.44	0.82
100%	0.20	0.25	0.57	0.92

Table 4.3 **Fractions of experiments where a non-linear solution was found by the NN out 100 simulations with a causal fraction of 0.25 of SNPs involved in statistical epistasis.** The values in the 'additive' column represent experiments where the ground truth genetic architecture was purely additive.

To evaluate the impact of my altered process, I re-estimated the h^2 for the simulated phenotypes generated from the new formula. I found that their estimated h^2 changed to 0.077, 0.074, 0.067 and 0.058 for additive, second, third and fourth-order interactions, respectively. These were still less than the target h^2 of 0.081; however, they were closer than the ones produced by the original naive simulation formula. I note that for the fourth-order scenario the gap between the target and realised h^2 was still ~30%, which I expect would result in a corresponding decrease in power to detect interactions for that scenario. Using this protocol, I simulated a 100 phenotypes for each order of interaction, and for each of the three causal fraction of SNPs. Finally, to evaluate the effect of the sample size on accuracy, I also down sampled the full cohort to 10%, 25%, 50% and 75% of the total available. In total, this produced 6,000 simulated experiments and 300,000 NN models (as each model required 50 hyperopt trials).

4.3.3 Prediction results

The fraction of experiments where a non-linear solution was identified by the model selection procedure are summarised in Tables 4.3 and 4.4 for the causal fractions 0.25 and 0.95, respectively. The relationship between the degree of non-linearity and sample size to the final prediction accuracy of the linear and non-linear NN models are shown in Figs 4.1 and 4.2 for the causal fractions 0.25 and 0.95, respectively (the same information for the causal fraction of 0.5 can be found under A.1 in Appendix A).

It is also important to note that the non-linear results represent the possibility of non-linearity, rather than that a non-linear solution was actually identified in each instance (see Figs 4.1 and 4.2). Thus, it is a more conservative estimate of the benefit of enabling non-linearity, as it counts the experiments into the non-linear result where the best performing model was still in fact linear.

% of sample size	**additive**	**2nd order**	**3rd order**	**4th order**
10%	0.47	0.50	0.68	0.63
25%	0.25	0.35	0.34	0.50
50%	0.14	0.29	0.28	0.34
75%	0.21	0.25	0.28	0.35
100%	0.20	0.29	0.30	0.36

Table 4.4 **Fractions of experiments where a non-linear solution was found by the NN out 100 simulations with a causal fraction of 0.95 of SNPs involved in statistical epistasis.** The values in the 'additive' column represent experiments where the ground truth genetic architecture was purely additive.

4.3.4 Inference results

As the difference between the linear and non-linear models was most pronounced in the experiments using a causal fraction of 0.25 at the fourth-order, I chose this series for my inference analyses. Fig 4.3 shows the performance of the three evaluated interaction detection algorithms for the fourth-order interaction series of experiments for all the instances where a result was returned by each method.

I chose ROC curves to visualise my results, which are defined by plotting the true positive rate (TPR) against the false positive rate (FPR). TPR and FPR are in turn defined as

$$\begin{aligned} TPR &= TP/(TP+FN) \\ FPR &= FP/(TN+FP), \end{aligned}$$

where TP and FN denote true positives and false negatives, respectively, and FP and TN denote false positives and true negatives, respectively. I also recognised partial interaction matches by defining both the TP and the putative associations to include, in addition to the fourth-order interactions, also all of their unique nested interactions. To clarify, this meant that each fourth-order interaction carried with it four third-order interactions, and each of those in turn carried three second-order interactions. Finally, I removed any overlapping nested interactions, so as to only keep unique sets of SNPs. To illustrate this with a simple example, consider the following three-way interaction: $(1,2,3)$. To account for partial matches, this interaction would also include the following two-way interactions as valid targets: $(1,2)$, $(2,3)$ and $(1,3)$ (but only if none of these were already part of other interactions).

Due to the heuristic employed to avoid exhaustive searches, I also had to manually correct the TN value by adding to it the number of tests not performed by the method. I obtained this number by subtracting from number of interaction tests possible from 955 SNPs, the number

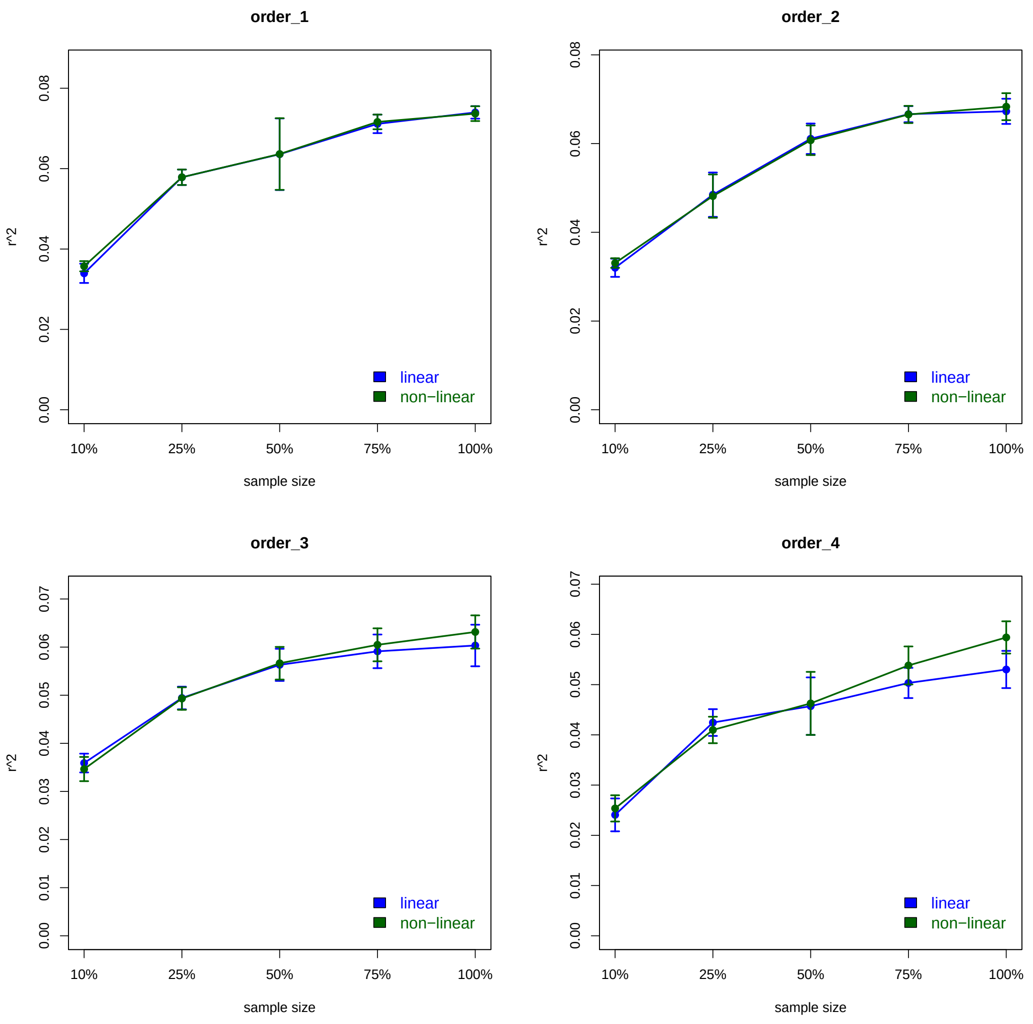


Fig. 4.1 **Neural-network performance on obtaining non-linear solutions under varying conditions for the experiments with a causal fraction of 0.25 of SNPs involved in statistical epistasis.** x-axis represents the % of sample size used and y-axis represents the r^2 of predicted vs observed phenotypes on the test set. Facets display experiments of genetic architectures that involve either additive, second, third and fourth-order interactions.

of tests the methods actually performed. This latter quantity I obtained by enumerating over the total tests performed (including the nested partial matches as I defined above) and adding to it the total number of true interactions.

The results presented in Fig 4.3 evaluate method performance conditioned on the fact that a method actually successfully returned a result. An alternative perspective of the same results is provided by considering all 100 potential experiments regardless if a result was actually obtained, and evaluating each method on that basis. Finally, I also considered evaluating method performance conditioned on the intersection of the experiments where

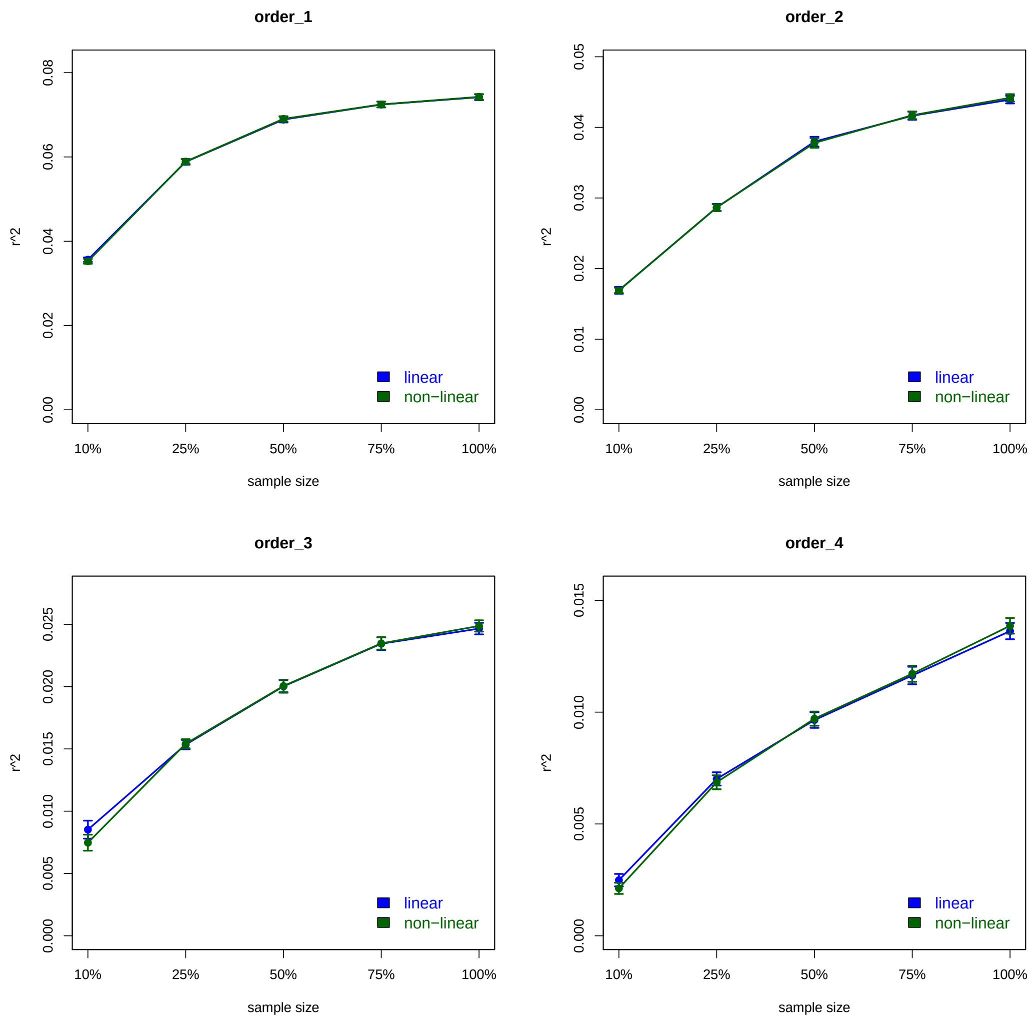
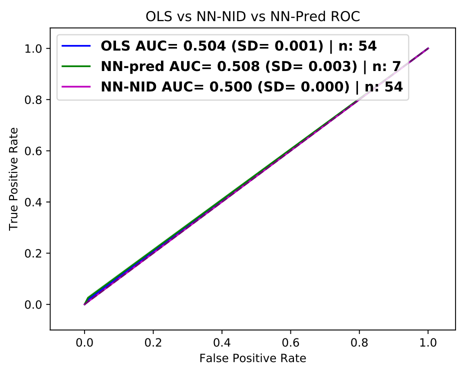


Fig. 4.2 **Neural-network performance on obtaining non-linear solutions under varying conditions for the experiments with a causal fraction of 0.95 of SNPs involved in statistical epistasis.** x-axis represents the % of sample size used and y-axis represents the r^2 of predicted vs observed phenotypes on the test set. Facets display experiments of genetic architectures that involve either additive, second, third and fourth-order interactions.

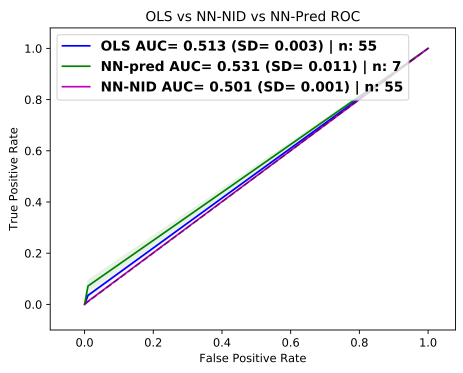
every method returned a result. Table 4.5 summarises method performance from all three perspectives.

4.3.5 Discussion of the simulation experiments

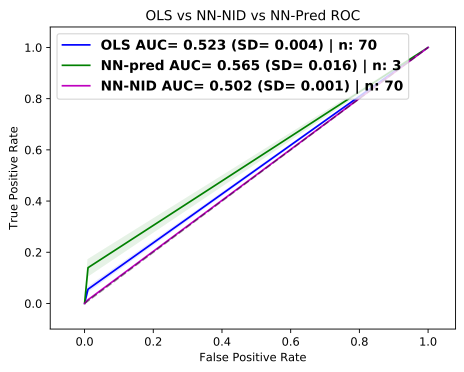
In method development, the goal of simulations is to consider plausible ranges of parameters of data to provide insight under in what scenarios would a novel method offer an advantage over conventional approaches. The level of insight is in turn proportionate to the degree



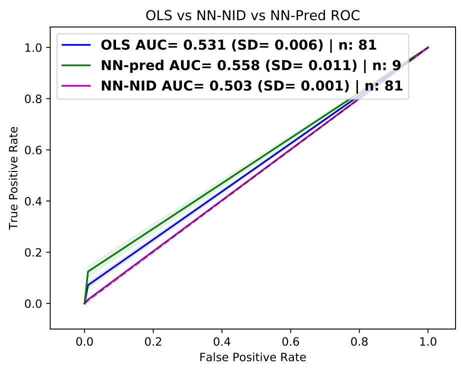
(a) 10% sample size



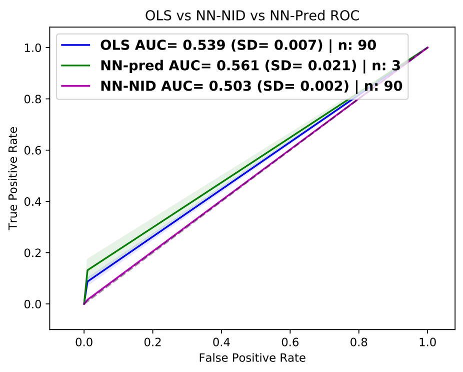
(b) 25% sample size



(c) 50% sample size



(d) 75% sample size



(e) 100% sample size

Fig. 4.3 **The performance of the three evaluated algorithms for statistical epistasis detection for the fourth-order series of experiments for the five sample sizes evaluated.** The average AUCs for OLS, NID and NNPred are shown blue, purple and green, respectively. *n* is the number of experiments from which the curves were drawn from.

that the simulations can approximate real world processes. My own simulation effort was

	all results		successful results		intersection results	
method	**AUC (SE)**	**n**	**AUC (SE)**	**n**	**AUC (SE)**	**n**
NNPred 10%	0.501 (0.002)	100	**0.508 (0.003)**	7	**0.508 (0.003)**	7
NID 10%	0.500 (0.000)	100	0.500 (0.000)	54	0.501 (0.001)	7
OLS 10%	**0.502 (0.002)**	100	0.504 (0.001)	54	0.505 (0.001)	7
NNPred 25%	0.502 (0.009)	100	**0.531 (0.012)**	7	**0.531 (0.012)**	7
NID 25%	0.501 (0.001)	100	0.501 (0.001)	55	0.501 (0.001)	7
OLS 25%	**0.507 (0.007)**	100	0.513 (0.003)	55	0.513 (0.002)	7
NNPred 50%	0.502 (0.012)	100	**0.565 (0.02)**	3	**0.565 (0.02)**	3
NID 50%	0.501 (0.001)	100	0.502 (0.001)	70	0.502 (0.002)	3
OLS 50%	**0.516 (0.011)**	100	0.523 (0.004)	70	0.529 (0.003)	3
NNPred 75%	0.505 (0.017)	100	**0.558 (0.012)**	9	**0.558 (0.012)**	9
NID 75%	0.502 (0.002)	100	0.503 (0.001)	81	0.503 (0.001)	9
OLS 75%	**0.525 (0.013)**	100	0.531 (0.006)	81	0.528 (0.004)	9
NNPred 100%	0.502 (0.011)	100	**0.561 (0.026)**	3	**0.561 (0.026)**	3
NID 100%	0.503 (0.002)	100	0.503 (0.002)	90	0.505 (0.002)	3
OLS 100%	**0.535 (0.014)**	100	0.539 (0.007)	90	0.532 (0.004)	3

Table 4.5 **Inference results for the simulation experiments for the three methods (NNPred, NID and OLS) at different percentages of the total sample size.** '*AUC*', '*SE*' and '*n*' denote the area under the curve, its standard error and the number of experiments the preceding values were calculated from, respectively. Values under 'all results' represent inference results from all 100 experiments. In case a method did not report a result, its accuracy was substituted by an AUC of 0.5. Values under 'successful results' represent inference results conditioned on individual methods successfully reporting a result. Values under 'intersection results' represent inference results conditioned all three methods reporting a result. Bold text highlights the best method in a given scenario.

therefore limited due to the lack of reliable evidence of what non-linear genetic architectures may comprise of in the real world.

Iterating through all potential factors, such as the number of causal interactions, the proportion of additive to epistatic effects, the ratio between various degrees of non-linear components, including both their number and effect size distribution and their relationship to MAF, would have been intractable. Therefore, my motivation was more modest, I only sought to simulate genetic architectures that covered the extreme scenarios, such as consisting entirely of a given degree of interaction. My aims were limited to illustrate general trends, such as the capacity of NNs to cope with the scale of data, and what general effects do degree of non-linearity and sample size have on expected accuracy at a plausible level of h^2. More specific conclusions, or quantifying relationships between factors (such as the sample size) and outcomes would have been invalid, as all such relationships would have been influenced by the arbitrary decisions that were used to determine the simulation parameters.

4.3.5.1 Prediction performance

I observe the general trend that the greater the degree of non-linearity and the larger the sample size, the more likely that a non-linear solution was preferred by the model selection process. This trend was present for both causal fraction series of experiments (Tables 4.3 and 4.4). Only the lowest sample size experiments were apparently anomalous, where the model selection chose a non-linear solution at ~50% of the time, even with no genuine non-linear signal. I interpret this as an artifact of low power, as the accuracy of such models was so low to begin with that it made no difference whether a linear or a non-linear model was chosen by the hyperopt optimisation process.

With respect to comparing the series across the two causal fractions (0.25 and 0.95), I observe the following. The lower the number of SNPs involved in the interactions, the better the non-linear models performed in comparison to the linear models, an observation which was true both in terms of model selection, as well as for prediction accuracy (Figs 4.1 and 4.2). This impression was also supported by the 0.5 causal fraction series, which exhibited intermediary results between the two extremes (Figs A.1 and A.1 in Appendix 1). These results indicate that, given a fixed level of h^2, the relationship between the ability to detect non-linearity and the causal fraction is inversely related. My interpretation of this is that a smaller causal fraction requires a larger effect size per interaction to achieve the same level of signal. This observation is also consistent with theoretical results that suggest that the number of loci involved in interactions and the epistatic variance they may each explain are inversely related (Mäki-Tanila and Hill, 2014).

For the series of experiments with a causal fraction of 0.95 of SNPs, I note that despite the positive association between genuine non-linear effects and the choice of a non-linear model, even at the largest sample size and highest degree of non-linearity, a non-linear solution was selected in only 36% of the experiments (Table 4.4). Besides the aforementioned factors, the stochastic nature of the hyperopt search process and the low level of h^2 in the simulated dataset may also have contributed to the lower preference for non-linear models. During my initial exploratory analyses, I simulated a phenotype with a much greater level of h^2 of 0.5, and in that case, the same experiments yielded a non-linear solution over 95% of the time. With respect to the prediction results (Fig 4.2), I find that the linear versus non-linear models were almost identical in performance across all sample sizes and degrees of non-linearity. I performed paired t-tests, and I found that the linear and non-linear means were not significantly different. Even with a 100% of the sample size at the highest degree on non-linearity, the gain of the non-linear NN over the linear version was not significant (paired t-test p-value = 0.361). In contrast, for the series of experiments with a causal fraction of 0.25 of SNPs (Table 4.3), I observe a greater preference for non-linearity. Here, the corresponding metric is 92% for the highest degree of non-linearity and the largest sample size, which indicates a stronger detectable effect at the stage of model selection. Additionally, I found that in the same experiment (Fig 4.1 lower right), the prediction accuracy of the non-linear model was also significantly higher than the linear version (paired t-test p-value of 0.020).

In summary, my prediction results suggest that, given the range of parameters evaluated in my simulations, NN based models are capable of inferring the presence of statistical epistasis in GWAS SNP data at the scale of UKBB.

4.3.5.2 Inference performance

From the perspective where the methods were evaluated based on the maximum number of successful results for each approach (Fig 4.3 and 'successful results' column in Table 4.5), I note that the ROC curves of all methods have a peculiar shape, there is a short curved rise near (0,0), after which there is a long straight line to (1,1). The former represents the tests performed, and the latter represents the tests not performed due to the search-space reducing heuristic I described in section 4.2.6.7. This shape provides support to the advantages of employing the aforementioned heuristic, as the long straight line suggests that my methods were much more likely to carry out tests for true positive interactions than for true negatives ones.

I observe a monotonous trend that indicates that all methods performed better with each increase in sample size that increased from 0.500 (NID at 10%) to 0.561 (NNPred at 100%). My own NNPred method exhibited the highest AUC at all % of sample sizes that ranged from 0.508 (at 10%) to 0.561 (at 100%).

Relative to the other methods, NNPred had a much lower number of scenarios where it could identify interactions. The method reported only three results for sample size scenarios 50% and 100%, and at most nine for the 75% sample size run. Due to the low number of observations for NNPred at the 50% and 100% sample size scenarios, I only performed significance tests to compare NNPred against NID and the OLS baseline in the remaining scenarios. The advantage of NNPred was significant in all of these tests, except at the comparison against OLS at the 50% sample size scenario (p-value=0.065). At the 75% sample size scenario, where NNPred had the most observations (nine), I found it to be significantly better than both NID and OLS with p-values of $7.025 * 10^{-7}$ and $1.288 * 10^{-4}$, respectively. NNPred consistently outperformed all other methods in all scenarios; however, it had the disadvantage of being able to identify a solution only at a fraction of the time. Out of the 100 replicates, it only found interactions between three to nine times, whereas the other methods obtained results at a far higher rate, ranging between 54 to 90 times.

The OLS baseline method's performance was intermediate between NNPred and NID. OLS found solutions at the same rate as NID, and it significantly outperformed the latter at every level with p-values that decreased from $1.338 * 10^{-26}$ to $5.577 * 10^{-69}$ for the 10% and 100% sample size scenarios, respectively. I consider OLS to represent a good balance between accuracy and reliability, as although it was not as accurate as NNPred, it proved to be more robust, as it found interactions in the majority of all experiments.

The NID algorithm performed the worst out of the three evaluated methods. NID's AUCs stayed near the chance level of 0.5, only increasing slightly from ~0.500 to 0.503 at the at 10% and 100% sample size scenarios, respectively. Although these AUCs were all very low, I found that they were still all significantly different from the no skill baseline of 0.5 (the largest p-value was $4.373 * 10^{-7}$ at the 10% sample size scenario). One potential reason for NID's low performance may have been that its algorithm assumes that the strength of interactions would be well captured by neurons in the first hidden layer, rather than being more evenly dispersed across the network. However, NNs are known to learn via distributed representations (Hinton, 1984), which implies that the learned features would be distributed across the deeper layers of the network, and therefore less detectable at the hidden first layer.

Considering the alternative perspectives on method performance (Table 4.5) I make the following observations. If I evaluate method performance on all 100 experiments ('all results' column), which may be interpreted as evaluating how well a method does on a random dataset, then the OLS method emerges as the clear winner in all scenarios. NNPred's advantage over the other two methods disappears due to the low number of experiments where it returned a result. NID remained the least performant method in all but the 100% sample size scenario, where it slightly outperformed NNPred (AUCs of 0.502 vs 0.503).

The high performance but an overall low number of results returned by NNPred poses an important question about this method. That is, if it outperformed the other methods only because it returned a result in the subset of experiments where the other methods have done similarly well. This question is answered by the column 'intersection results' in Table 4.5, where results are conditioned on the intersection of experiments where all methods obtained a result. That is, this is a like-for-like comparison, where OLS and NID are evaluated on the same subset of experiments where NNPred obtained a result. I observe that the performance of the other two methods (NID and OLS) did not improve, which suggests that these experiments were a random subset, rather than the 'easy' cases where all methods would have done well. However, this leads to a further question, which is why NNPred returned a result successfully in these instances but not in the other experiments.

To investigate why NNPred has failed to return any results for such a large number of experiments, I decided to examine if there was a difference in hyperparameter selection between NNs that succeeded or failed to return inference results for NNPred. I found that out of all the hyperparameters (listed in Table 4.2) only the learn rate, dropout and the number of layers were significantly different between failed and successful experiments. NNs that successfully returned an inference result for NNPred had a higher dropout (0.432 vs 0.646, p-value=$8.784 * 10^-6$), a lower average number of layers (3.849 vs 1.862, p-value=$1.951 * 10^-10$) and a higher learn rate (0.004 vs 0.006, p-value=0.002). The finding of a positive association between dropout and inference performance makes intuitive sense, as switching on and off a larger fraction of the network via dropout may provide a better estimate of uncertainty in predictions. The finding of a positive relationship between inference performance and a higher learn rate or a shallower NN architecture are more difficult to interpret directly. These parameters may have influenced inference performance via a combination with other factors, such as the genetic architecture in a given simulation. An auxiliary explanation for this phenomenon may be found in the reported shortcomings of dropout based uncertainty estimation for NNs, where some researchers argued that this approach does not always provide a genuine approximation of a Gaussian process (Osband, 2016). The overall conclusion I draw from this investigation is that optimising NN architectures for performance on prediction may not always yield architectures that are compatible with inference tasks.

4.4 Neural-network tests on real data

As I was interested in whether NN based approaches may offer a viable alternative to the standard methods I evaluated in Chapter 3 to infer non-linear effects, I assessed their utility on the same datasets. I applied the NN models described in section 4.2 onto the previously prepared cohorts that comprised of the four UKBB traits and the two IBD sub-phenotypes.

4.4.1 Data preparation and model selection

I used the same set of predictors and subsets of individuals that I finished with in Chapter 3. For the SNP datasets I had 955, 1,732, 1,671 and 450 SNPs for FIS, height, BMI and asthma, respectively. For the protein score datasets I used 99, 781, 317 and 38 protein scores for FIS, height, BMI and asthma, respectively. For the asthma phenotype, I also used the three TWAS tissue gene expression datasets of 264, 218 and 253 gene-scores for monocytes, neutrophils and T-cells, respectively. Finally, I also applied the NN models onto the concatenated cross-domain datasets that integrated SNPs, protein scores and TWAS scores (for asthma), which comprised of 966, 1,805, 1,695 and 579 predictors for FIS, height, BMI and asthma, respectively.

I brought the IBD datasets in line with the UKBB datasets by processing them through the same filtering steps. I only kept the LD-clumped top FDR < 0.05 corrected SNPs from additive associations, and I also filtered out all variants that were within the same recombination block. This process left 308 and 285 SNPs for CD and UC, respectively.

After determining the model architecture via hyperopt on the first bootstrap sample, I trained NN models for all 20 bootstrap samples using the same hyperparameters with early stopping enabled. The number of epochs used for this was $+20$, relative to the ideal number of epochs identified in the first sample. I added this redundancy to allow for slight variations in the ideal number of epochs between bootstrap samples, which was expected, given that each sample is a different observation from the same distribution.

4.4.2 Prediction results on real data

A non-linear solution was preferred for only the following experiments (given in the format of phenotype/domain): BMI/SNP, asthma/SNP, asthma/neutrophils and asthma/cross-domain, together with the two IBD sub-phenotypes/SNP. The results from these are presented in Fig 4.4. Among these, only the asthma cross-domain test was significant with a paired t-test p-value of $1.366 * 10^{-9}$.

4.4.3 Inference results on the asthma cross-domain data

I performed NN based interaction association tests in all 20 bootstrap samples for the asthma cross-domain data. Among the three evaluated methods, only the NID and OLS methods reported putative interactions. To assess why NNPred did not return any inference results for this cohort, I examined the hyperparameters of the model used for this analysis. I found that this model's hyperparameters were closer to those models that did not return an inference result in the simulations (detailed in section 4.3.5.2) than to those which did, with dropout of 0.517, six hidden layers and a learn rate of 0.005.

On average, NID and OLS found 706 and 3,357 associations across all bootstrap samples, respectively, and 1,100 and 749 of associations were present in at least half of the bootstrap samples for NID and OLS, respectively. I reasoned that as the different bootstrap samples are the resampling of the same single original pool of observations, the strongest associations should manifest through all of them. Thus, to reduce the potential for false positives generated by the stochastic nature of bootstrap sampling, I intersected the 20 association results, and only considered candidate interactions further that manifested across all 20 samples. This left no results for OLS and six candidate interactions for NID.

Genuine associations are expected to show consistent signal across different sources of evidence; therefore, I run the following diagnostic tests to assess the credibility of these six pairs of interactions. I retrieved the association p-values for the six pairs of two-way interaction tests from my earlier interaction tests in Chapter 3. Then, I performed new regression based two-way interaction tests (described in Chapter 3 by eq 3.3) in an attempt to replicate the six pairs of associations in the Test Set. Finally, I also performed cases only tests to obtain an additional source of support. This test (described in the Introduction in section 1.1.7) evaluates the hypothesis that cases that carry the interacting alleles at both loci should be over-represented relative to those that only carry a single copy (Vittinghoff and Bauer, 2006). However, unlike standard SNP based tests where the relationship between allele counts may be evaluated in a contingency table, I was also dealing with continuous values for the gene-based predictors. Evaluating correlation between the continuous predictor and the allele counts within cases may be considered an analogous test; therefore, I performed correlation tests for those pairs. Table 4.6 presents the results from these diagnostics, which includes the NID importance scores, p-values from the original two-way tests, p-values from the interaction tests from the Test Set, and finally, the p-values from the cases only tests.

All standard statistical tests unanimously indicated that none of the six pairs were genuine interactions. The p-values for the interaction terms from both the Training Set and the Test Set, together with the correlation test were all non-significant. Given their non-linearity, NNs are better at capturing structure in the data than linear models, which I thought may explain this apparent discrepancy between their results and of those reported by standard methods. Therefore, I decided to examine the raw data more closely to inspect it for factors that may indicate anomalies, such as outliers or structure.

Table 4.7 summarises the diagnostic statistics that I obtained for these predictors, which included their genomic location and MAF (where applicable). Next, to examine the data more closely, I plotted the genotype counts for cases and controls separately for the three pairs involving only SNPs. Finally, to obtain an analogue of the same information that involved continuous predictors, I created a scatter plot for the two gene-scores (ENSG00000238818 vs SPARC) and box-plots the SNP/gene-score pairs (rs16858573 vs GLMN and rs2970932 vs ENSG00000232528). Visual examination of the data (Fig 4.5 and Table 4.8), did not reveal any anomalies, such as outliers or groupings, that may have explained the discrepancy

predictors	$\mathbf{NN_{IS}}$	$\mathbf{p_{train}}$	$\mathbf{p_{test}}$	$\mathbf{p_{testCorr}^{cases}}$
ENSG00000238818, SPARC	0.169	0.792	0.261	0.096
GLMN, rs16858573	0.167	0.412	0.580	0.770
rs903361, rs2112535	0.172	0.164	0.572	0.636
rs2970932, ENSG00000232528	0.185	0.856	0.823	0.828
rs3813308, rs2830962	0.177	0.385	0.315	0.184
rs2492419, rs2832662	0.173	0.392	0.345	0.435

Table 4.6 **Comparison between the significance metrics of the NN and standard statistical methods for the variants identified as potentially interacting.** $\mathbf{NN_{IS}}$ is the importance score produced by the NID algorithm (arbitrary scale). $\mathbf{p_{train}}$ is the raw interaction p-value from Chapter 3 that considered all predictors which survived the filtering process in the Training Set. $\mathbf{p_{test}}$ is the raw interaction p-value for the same pairs in the Test set. $\mathbf{p_{testCorr}^{cases}}$ is the p-value of the correlation between the predictors in the cases only test in the Test Set.

between the NN and the standard methods. All plots appeared to support the standard formal test results that indicated no difference between cases and controls for the pairs investigated.

4.4.4 Summary and limitations

The results from the IBD datasets resemble the results from the small sample size simulation experiments (Fig 4.4). That is, a non-linear solution may have been preferred in situations where the NN had a very low power. A non-linear solution may have been chosen due to chance, or potentially due to the overfitting on the validation set with the help of non-linearity, which then ultimately fell short on the predictions for the Test set. Indeed, I found this to be the case for both CD and UC, as the non-linear models were significantly worse than their linear versions, indicated by their t-test p-values of $1.071 * 10^{-12}$ and $2.314 * 10^{-6}$ for CD and UC, respectively. This finding underlines why NNs require large sample sizes, and that any positive results that originate from smaller cohorts have to be treated with caution.

In the UKBB, only the asthma cross-domain experiment showed a genuine non-linear effect (paired t-test p-value $1.366 * 10^{-9}$). One possible explanation for this may have been that interactions exist across SNP and gene-level predictors. To see if I could localise this non-linear advantage, I performed NN based association tests which identified six putative pairs that were particularly relevant for this improved non-linear prediction. All of the gene-level variants originated from the TWAS panels, two from monocytes (ENSG00000232528, SPARC) and two from neutrophils (FOXK1 and ENSG00000238818). Among the genes, SPARC was the only one with an established link to asthma (Wong and Sukkar, 2017).

predictor	chr	bp	$\mathrm{MAF}_{\mathrm{case}}$	$\mathrm{MAF}_{\mathrm{control}}$
ENSG00000238818.1	1	15237862	-	-
SPARC	5	150043046	-	-
GLMN	1	91712200	-	-
rs16858573	2	143875725	0.119 (0.003)	0.125 ($9*10^{-4}$)
rs903361	1	203091274	0.327 (0.007)	0.338 (0.001)
rs2112535	5	176531075	0.252 (0.003)	0.245 (0.001)
rs2970932	2	162858200	0.415 (0.004)	0.423 (0.001)
ENSG00000232528.3	20	748610	-	-
rs3813308	5	118690781	0.444 (0.004)	0.433 (0.001)
rs2830962	21	28844948	0.172 (0.003)	0.166 (0.001)
rs2492419	6	83407023	0.42 (0.004)	0.428 (0.001)
rs2832662	21	31596876	0.012 ($8*10^{-4}$)	0.01 ($3*10^{-4}$)

Table 4.7 **Diagnostic statistics for each variant potentially involved in interactions.** The values in parentheses in the 'MAF' columns are the standard error of the mean.

Among the SNPs, three have been previously associated with asthma or respiratory diseases in the GWAS catalogue (rs16858573, rs903361 and rs3813308).

I attempted to find support for the putative interactions for the asthma phenotype by comparing these results to those obtained for the same pairs in Chapter 3. I also performed additional tests on the Test Set to obtain an orthogonal source of evidence. None of these putative associations found any support from the standard statistical methods. All standard statistical tests yielded results consistent with the null hypothesis.

There are several additional reservations that further reduce the credibility of these alleged interactions. The asthma phenotype was the one UKBB trait where I have not used a different chip for the Test Set due to its prior links with asthma, and had to evaluate it on an independent subset of the main UK Biobank Axiom™ chip which may have given rise to a non-linear batch effect. Additionally, given that asthma is a binary phenotype, putative interactions are susceptible to the thresholding artefact I described in the Introduction in section 1.1.6.3. Finally, the NID search method is an experimental algorithm with no proven empirical track record. From my own experience in the simulation experiments it actually proved to be consistently the worst among all evaluated methods (section 4.3.5). Thus, NID may have made false positive associations due to low power, and the algorithm's unrealistic assumptions that interactions would be well captured by the first hidden layer's weights, rather than be more widely distributed across the network (Hinton, 1984).

In conclusion, I deemed that there is insufficient evidence to consider any of these six interactions to be real. Given how easy it is to fit a convincing biological narrative onto false cases

	rs2112535		
rs903361	0	1	2
0	0.256	0.171	0.028
1	0.242	0.165	0.028
2	0.060	0.041	0.008

controls

	rs2112535		
rs903361	0	1	2
0	0.250	0.162	0.026
1	0.255	0.165	0.027
2	0.065	0.043	0.007

cases

	rs2830962		
rs3813308	0	1	2
0	0.256	0.171	0.028
1	0.242	0.165	0.028
2	0.060	0.041	0.008

controls

	rs2830962		
rs3813308	0	1	2
0	0.250	0.162	0.026
1	0.255	0.165	0.027
2	0.065	0.043	0.007

cases

	rs2832662		
rs2492419	0	1	2
0	0.256	0.171	0.028
1	0.242	0.165	0.028
2	0.060	0.041	0.008

controls

	rs2832662		
rs2492419	0	1	2
0	0.250	0.162	0.026
1	0.255	0.165	0.027
2	0.065	0.043	0.007

Table 4.8 **Training Set genotype fraction tables for the asthma phenotype for cases and controls for the three putative two-way interactions that involved SNP pairs.**

positive associations (Biedrzycki et al., 2019), I refrain from further speculation about how these interactions could have contributed to the pathogenesis of asthma.

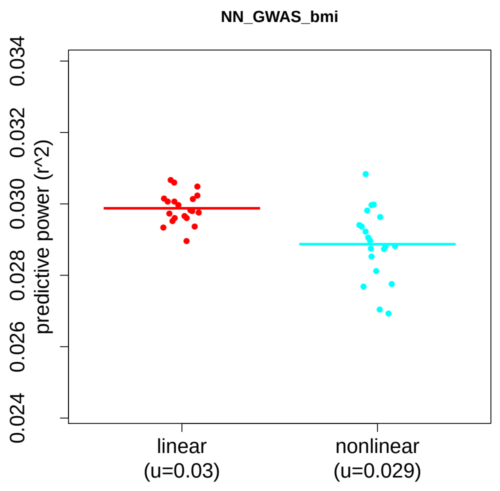
4.4.5 The outlook of NNs for building PRS

My project of using NNs to build more accurate PRS offered numerous improvements over the comparable efforts on humans by Bellot et al. (2018) and Xu et al. (2020). My contribution included a more rigorous treatment to eliminate haplotype effects, an explicit test if NNs learned genuine non-linear genetic effects, the novelty of considering interactions across domains, and finally, the application of NN based inference methods to identify individual interactions. However, despite these advances, I found that my results are consistent with the aforementioned studies, both of which found no consistent or substantial contributions from non-linear effects to phenotypic variance. Our results are in stark contrast with agricultural applications, where similar efforts have been much more successful (Ma et al., 2017; Pook et al., 2020). I speculate that this may be due to the fact that the plant and animal subjects of those experiments were not natural populations, but rather products of recent, artificial breeding programmes. Such recent crosses between distantly related populations (or inbred lines) may 'convert' functional epistasis into statistical epistasis by the creation of novel heterozygotes at loci previously only involved in functional epistasis (Mackay, 2014).

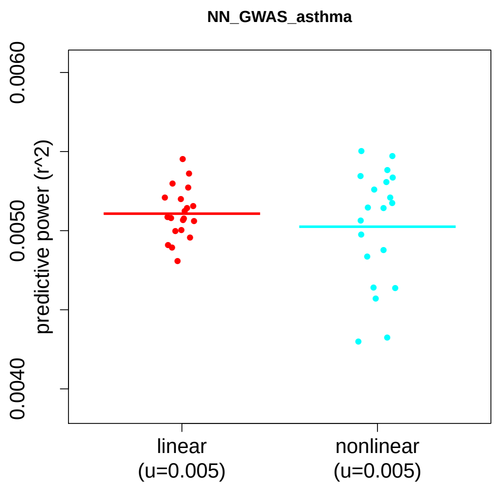
FNNs face numerous additional practical limitations due to the fact that they require genotype level data to build PRS. My simulations suggested that the sample size required for NNs to reliably obtain a non-linear solution is likely to be far beyond non-biobank-scale cohorts. Although biobanks are increasing in size and popularity (GEL, 2020; The All of Us Research Program Investigators, 2019), many traits still rely on the pooling of smaller cohorts in meta-analyses. This presents an additional problem for NNs, as building NN derived PRS directly from SNP data requires the integration of cohorts on the genotype level. This is incompatible with the normal practice of meta-analyses, where QC and the association step are performed within each cohort, and results are only integrated via summary statistics. The only way NNs could work with such data would be if QC would be performed on the basis of the 'lowest common denominator' (removing all variants that failed in any of the datasets), and the cohort batch effects regressed out from the phenotypes. This would result in a loss of many variants, and potentially in the imperfect controlling of batch effects. Finally, even if such technical integration was feasible, genotype level access may not be granted on the basis of legal or privacy considerations.

There are also computational challenges that would have to be addressed before NNs could become a viable option for phenotype prediction. Even if statistical interactions did contribute to phenotypic variance substantially, most of the phenotypic variance would still be due to additive effects. Most state-of-the-art PRS consists of ~500K or more SNPs (Khera et al., 2018), a dimensionality that would seem insurmountable for fully connected NNs trained on GPUs and sample sizes of the current generation.

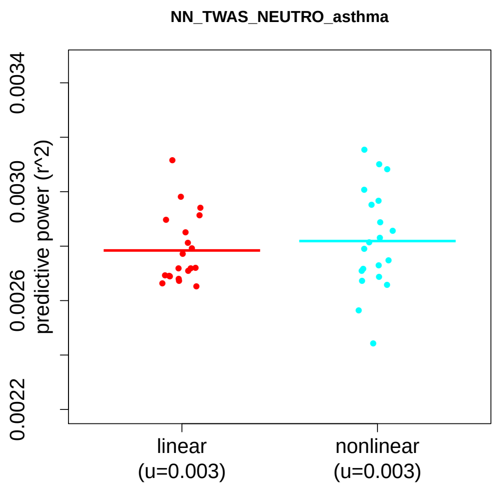
In the future, if single cohort biobanks on the scale of the millions become available with perfect coverage to eliminate haplotype effects, together with GPUs powerful enough to fit the number of SNPs comparable to that of the state-of-the-art additive PRS, experiments similar to mine may be repeated with the hope of a more positive verdict. For the time being however, I see limited real-world applications for NNs to build PRS directly from GWAS SNP data.



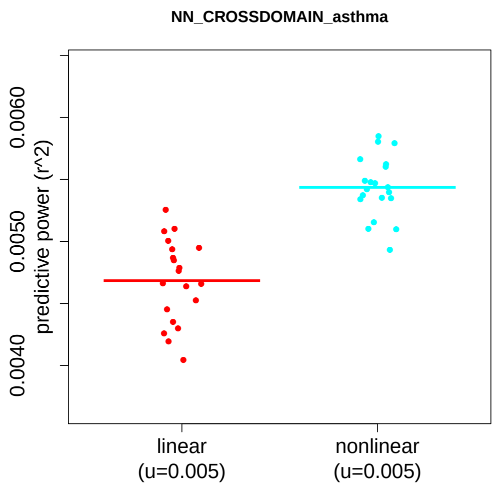
(a) BMI - SNP



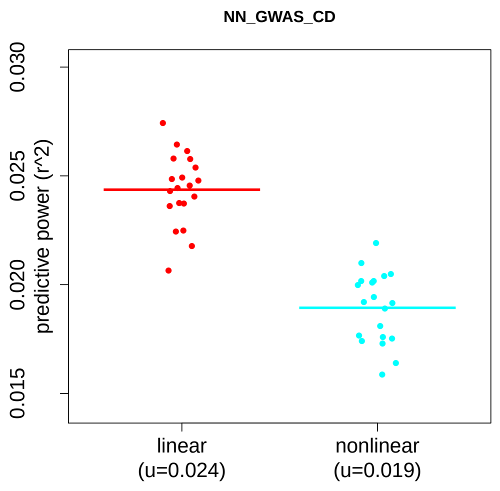
(b) Asthma - SNP



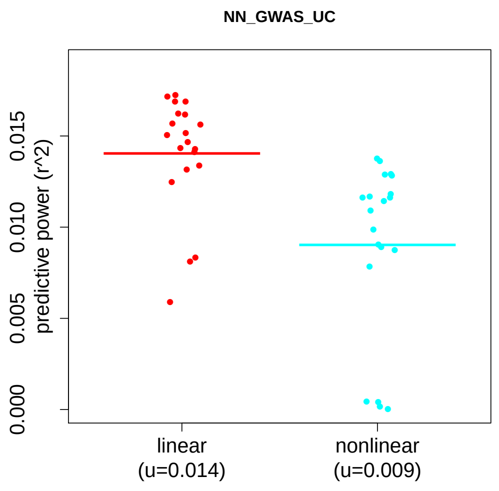
(c) Asthma - TWAS neutrophils



(d) Asthma - cross-domain

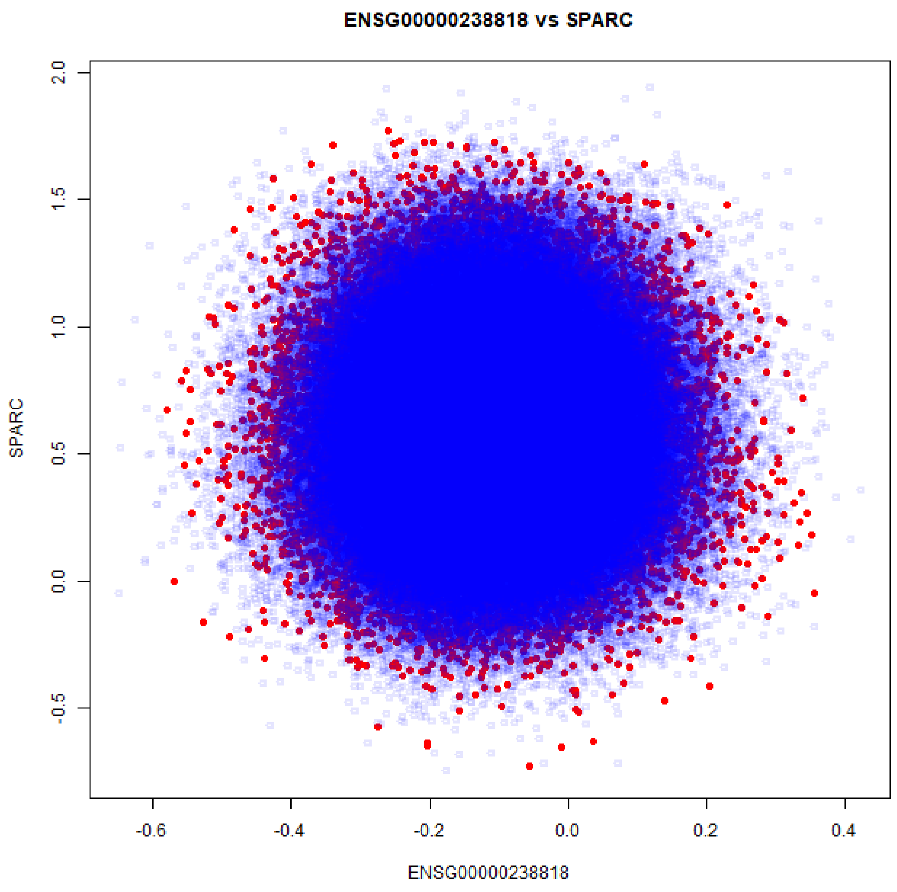


(e) CD - SNP

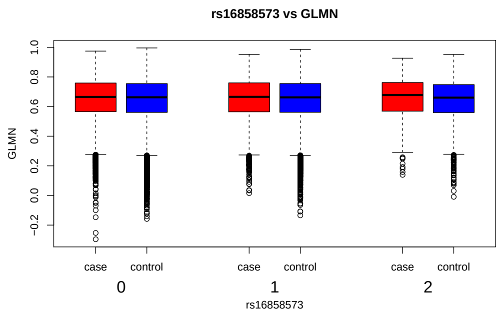


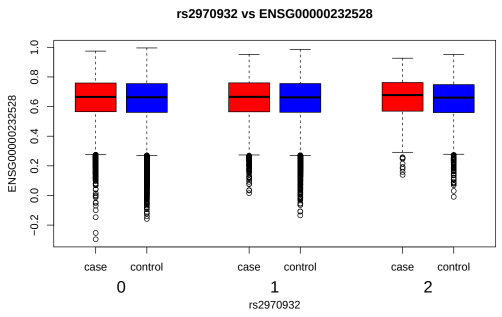
(f) UC - SNP

Fig. 4.4 **NN performance in the six experiments where a non-linear solution was preferred.** Results given in the format of *phenotype - domain*. y-axis represents r^2 of predicted vs observed phenotypes on the Test Set. For CD and UC the Test Sets were GWAS1 and GWAS2, respectively.



(a) ENSG00000238818 vs SPARC





(b) rs16858573 vs GLMN (c) rs2970932 vs ENSG00000232528

Fig. 4.5 **Diagnostic plots for the asthma cross-domain experiments for putative interaction pairs involving gene-level predictors.** Red and blue represent cases and controls, respectively.