Neural Interaction Detection

Michael Tsang, Dehua Cheng, Yan Liu
Department of Computer Science
University of Southern California
Los Angeles, CA 90089
{tsangm, dehuache, yanliu.cs}@usc.edu

Abstract

We develop a method of detecting statistical interactions in data by interpreting the trained weights of a feedforward multilayer neural network. With sparsity regularization applied to the weights, our method can achieve high interaction detection performance *without* searching an exponential solution space of possible interactions. We obtain our computational savings by first observing that interactions between input features are created by the non-additive effect of nonlinear activation functions, and that interacting paths are encoded in weight matrices. We use these observations to develop a way of identifying both pairwise and higher-order interactions with a simple traversal over the input weight matrix. In experiments on simulated and real-world data, we demonstrate the performance of our method and the importance of discovered interactions.

1 Introduction

Despite their predictive capability, neural networks have traditionally been difficult to interpret, preventing their adoption in many application domains. Healthcare and finance are examples of such domains, where understanding a machine learning model is paramount when using it to make critical decisions (Caruana et al., 2015; Goodman & Flaxman, 2016). This is because models can learn unintended patterns from data, and the risks associated with depending on these models can be consequential for stakeholders (Varshney & Alemzadeh, 2016).

Existing approaches to interpreting feedforward neural networks have focused on explanations of feature importance, for example by computing input gradients (Hechtlinger, 2016; Ross et al., 2017) or by using post-hoc means (Ribeiro et al., 2016). Owing to the importance of interpretation, we add to the existing approaches by introducing a way of finding feature groupings that neural networks model, in this case statistical interactions.

Statistical interactions carry great importance in natural phenomena, where features often have joint effects with other features on predicting an outcome. This is different than correlation because correlations do not involve outcome variables. The discovery of interactions can be very useful for science, where for example, physicists may want to better understand what joint factors provide evidence for new elementary particles. Moreover, interpreting interactions can also be useful for validating machine learning models. For example, doctors may want to know what interactions are accounted for in risk prediction models, to compare against known interactions from scientific literature.

In this work, we developed a simple and efficient algorithm that proposes statistical interactions of variable order in data, by accounting for all weights of a feedforward network that is fully-connected across input features. Our approach is efficient because it avoids searching over an exponential solution space of interaction candidates, which is achieved by making an approximation of hidden unit importance at the first hidden layer via all weights above and doing a 2D traversal of the input

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

weight matrix. We propose our framework, *Neural Interaction Detector* (`NID`), which generates a ranking of interaction candidates solely by interpreting the weights of a feedforward network. Top-K true interactions are then determined by finding a cutoff on the ranking using a special form of generalized additive model, which accounts for interactions of variable order (Wood, 2006; Lou et al., 2013). In experiments on simulated and real-world data, we evaluate the performance of our approach, the results of which show similar interaction detection performance compared to the state-of-the-art while taking orders of magnitude less time.

2 Background and Notations

Interaction Detection Statistical interaction detection has been a well-studied topic in statistics, dating back to the 1920s when two-way ANOVA was first introduced (Fisher, 1925). Since then, two general approaches emerged for conducting interaction detection. One approach has been to conduct individual tests for each combination of features (Lou et al., 2013). The other approach has been to pre-specify all interaction forms of interest, then use lasso to simultaneously select which are important (Tibshirani, 1996; Bien et al., 2013). Our approach to interaction detection is unlike others in that it is both fast and capable of detecting interactions of variable order without limiting their functional forms. The approach is fast because it does not conduct individual tests for each interaction to accomplish higher-order interaction detection. This property has the added benefit of avoiding a high false positive-, or *false discovery rate*, that commonly arises from multiple testing (Benjamini & Hochberg, 1995).

Interpretability Two general approaches to interpreting machine learning are *local* and *global* interpretability. A local interpretation explains how a machine learning model makes predictions over small regions of input data, whereas a global interpretation provides an understanding of how the model behaves over all data (Ribeiro et al., 2016). For feedforward neural networks, there are existing works that address these approaches. For example, the input gradient has been studied as a way of locally explaining predictions at individual data points (Hechtlinger, 2016; Ross et al., 2017), and weight interpretation has been studied for measuring global feature importance (Garson, 1991). Our approach belongs to the global interpretation category, but unlike previous works, this work interprets learned statistical interactions from the weights of a feedforward neural network.

Feedforward Neural Network[1] Consider a feedforward neural network with L hidden layers. Let p_ℓ be the number of hidden units in the ℓ-th layer. We treat the input features as the 0-th layer and $p_0 = p$ is the number of input features. There are L weight matrices $\mathbf{W}^{(\ell)} \in \mathbb{R}^{p_\ell \times p_{\ell-1}}$, $\ell = 1, 2, \ldots, L$, and $L+1$ bias vectors $\mathbf{b}^{(\ell)} \in \mathbb{R}^{p_\ell}$, $\ell = 0, 1, \ldots, L$. Let $\phi(\cdot)$ be the activation function (non-linearity), and let $\mathbf{w}^y \in \mathbb{R}^{p_L}$ and $b^y \in \mathbb{R}$ be the coefficients and bias for the final output. Then, the hidden units $\mathbf{h}^{(\ell)}$ of the neural network and the output y with input $\mathbf{x} \in \mathbb{R}^p$ can be expressed as:

$$\mathbf{h}^{(0)} = \mathbf{x}, \quad y = (\mathbf{w}^y)^\top \mathbf{h}^{(L)} + b^y, \quad \text{and } \mathbf{h}^{(\ell)} = \phi\left(\mathbf{W}^{(\ell)}\mathbf{h}^{(\ell-1)} + \mathbf{b}^{(\ell)}\right), \quad \forall \ell = 1, 2, \ldots, L.$$

Statistical Interaction Let $[p]$ denote the set of integers from 1 to p. An *interaction*, $\mathcal{I}$, is a subset of all input features $[p]$ with $|\mathcal{I}| \geq 2$, and an interaction that is *higher-order* denotes $|\mathcal{I}| \geq 3$. For a vector $\mathbf{w} \in \mathbb{R}^p$ and $\mathcal{I} \subseteq [p]$, let $\mathbf{w}_\mathcal{I} \in \mathbb{R}^{|\mathcal{I}|}$ be the vector restricted to the dimensions specified by $\mathcal{I}$.

Definition 1 (Non-Additive Statistical Interaction (Dodge, 2006; Sorokina et al., 2008))**.** *Consider a function $f(\cdot)$ with input variables $x_i, i \in [p]$, and an interaction $\mathcal{I} \subseteq [p]$. Then $\mathcal{I}$ is a non-additive interaction of function $f(\cdot)$ if and only if there does not exist a set of functions $f_i(\cdot), \forall i \in \mathcal{I}$ where $f_i(\cdot)$ is not a function of x_i, such that*

$$f(\mathbf{x}) = \sum_{i \in \mathcal{I}} f_i\left(\mathbf{x}_{[p]\setminus\{i\}}\right).$$

For example, in $x_1x_2 + \sin(x_2 + x_3 + x_4)$, there is a pairwise interaction $\{1, 2\}$ and a 3-way interaction $\{2, 3, 4\}$. Note that from the definition of statistical interaction, a *d-way interaction can only exist if all its corresponding $(d-1)$-interactions exist* (Sorokina et al., 2008). For example, the interaction $\{1, 2, 3\}$ can only exist if interactions $\{1, 2\}$, $\{1, 3\}$, and $\{2, 3\}$ also exist.

[1] In this paper, we mainly focus on the *multilayer perceptron* architecture with ReLU activation functions, while some of our results can be generalized to a broader class of feedforward neural networks.

Algorithm 1 `NID` Greedy Ranking Algorithm

```
Input: input-to-first hidden layer weights $\mathbf{W}^{(1)}$, aggregated weights $\mathbf{z}^{(1)}$
Output: ranked list of interaction candidates $\{\mathcal{I}_i\}_{i=1}^{m}$
$d \leftarrow$ initialize an empty dictionary mapping interaction candidate to interaction strength
for each row $\mathbf{w}'$ of $\mathbf{W}^{(1)}$ indexed by $r$ do
    for $j = 2$ to $p$ do
        $\mathcal{I} \leftarrow$ sorted indices of top $j$ weights in $\mathbf{w}'$
        $d[\mathcal{I}] \leftarrow d[\mathcal{I}] + z_r^{(1)} \mu\left(|\mathbf{w}'_{\mathcal{I}}|\right)$
$\{\mathcal{I}_i\}_{i=1}^{m} \leftarrow$ interaction candidates in $d$ sorted by their strengths in descending order
```

3 Interaction Detection

Interactions can be detected by first generating an interaction ranking, then finding a cutoff on the ranking to determine top-K interactions. Our approach to interaction ranking is to start with interaction candidates, compute an average of their weights entering common hidden units in the first hidden layer (see common hidden unit proof in Proposition 2), and approximate the influences of these hidden units on the neural networks' final output. Irrespective of interaction candidate, the influences of hidden units can be approximated in the following way via matrix multiplications:

$$\mathbf{z}^{(1)} = |\mathbf{w}^{y}|^{\top} \left|\mathbf{W}^{(L)}\right| \cdot \left|\mathbf{W}^{(L-1)}\right| \cdots \left|\mathbf{W}^{(2)}\right|, \tag{1}$$

where $\mathbf{z}^{(1)} \in \mathbb{R}^{p_1}$ and $z_i^{(1)}$ is the approximated influence of hidden unit i. This approximation satisfies upper bounds on the gradient magnitudes of hidden units (Lemma 3). We can combine this hidden unit influence with a proposed local strength of interaction candidate $\mathcal{I}$ per hidden unit i:

$$\omega_i(\mathcal{I}) = z_i^{(1)} \mu\left(\left|\mathbf{W}_{i,\mathcal{I}}^{(1)}\right|\right), \tag{2}$$

where $\mathbf{W}_{i,\mathcal{I}}^{(1)}$ are the weights associated with $\mathcal{I}$ from the input weight matrix, and $\mu(\cdot)$ is an averaging function that combines said weights into a scalar. Local strengths are to be summed across units.

Architecture We study two architectures: *MLP* and *MLP-M*. *MLP* is a standard multilayer perceptron, and *MLP-M* is an *MLP* with additional univariate networks summed at the output (Figure 1). The univariate networks are intended to discourage the modeling of univariate functions (or main effects) away from the *MLP*, which can create spurious interactions using the main effects. We apply L1 regularization on the *MLP* portions of the architectures to suppress unimportant interacting paths.

Ranking Interactions The key to efficiently detecting interactions of variable order is to determine what interaction candidates to consider first. Thus, we design a greedy algorithm (Algorithm 1) that generates an interaction ranking by only considering, at each hidden unit, the top-ranked interactions of every order, where $2 \leq |\mathcal{I}| \leq p$. Due to this greedy strategy, the search space of interactions is drastically reduced while all interaction orders are still considered. We set the averaging function $\mu(\cdot) = \min(\cdot)$ based on its performance in experimental evaluation (Section 4.1). With this averaging function, the greedy algorithm automatically improves the ranking of higher-order interactions over their redundant subsets (Theorem 4).

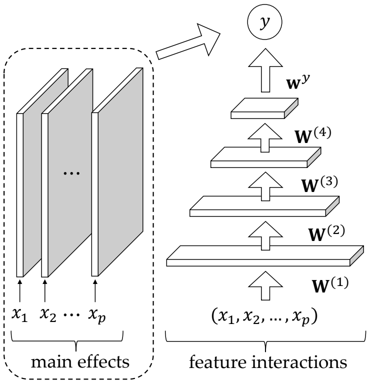


Figure 1: Neural network architecture for interaction detection, with optional univariate networks

Cutoff on Interaction Ranking We obtain a top-K cutoff on the interaction ranking by constructing *MLP-Cutoff*:

$$c_K(\mathbf{x}) = \sum_{i=1}^{p} g_i(x_i) + \sum_{i=1}^{K} g'_i(\mathbf{x}_{\mathcal{I}}),$$

where $g_i(\cdot)$ captures the main effects, $g'_i(\cdot)$ captures the interactions, and both g_i and g'_i are feed-forward networks trained jointly via backpropagation. We gradually add top-ranked interactions to *MLP-Cutoff* until performance on a validation set plateaus. The exact plateau point can be found by early stopping or other heuristic means, and we report $\{\mathcal{I}_i\}_{i=1}^{K}$ as the identified feature interactions.

Table 1: Test suite of data-generating functions

$F_1(\mathbf{x})$	$\pi^{x_1 x_2}\sqrt{2x_3} - \sin^{-1}(x_4) + \log(x_3 + x_5) - \frac{x_9}{x_{10}}\sqrt{\frac{x_7}{x_8}} - x_2 x_7$
$F_2(\mathbf{x})$	$\pi^{x_1 x_2}\sqrt{2\lvert x_3\rvert} - \sin^{-1}(0.5x_4) + \log(\lvert x_3 + x_5\rvert + 1) + \frac{x_9}{1 + \lvert x_{10}\rvert}\sqrt{\frac{x_7}{1 + \lvert x_8\rvert}} - x_2 x_7$
$F_3(\mathbf{x})$	$\exp\lvert x_1 - x_2\rvert + \lvert x_2 x_3\rvert - x_3^{2\lvert x_4\rvert} + \log(x_4^2 + x_5^2 + x_7^2 + x_8^2) + x_9 + \frac{1}{1 + x_{10}^2}$
$F_4(\mathbf{x})$	$\exp\lvert x_1 - x_2\rvert + \lvert x_2 x_3\rvert - x_3^{2\lvert x_4\rvert} + (x_1 x_4)^2 + \log(x_4^2 + x_5^2 + x_7^2 + x_8^2) + x_9 + \frac{1}{1 + x_{10}^2}$
$F_5(\mathbf{x})$	$\frac{1}{1 + x_1^2 + x_2^2 + x_3^2} + \sqrt{\exp(x_4 + x_5)} + \lvert x_6 + x_7\rvert + x_8 x_9 x_{10}$
$F_6(\mathbf{x})$	$\exp\left(\lvert x_1 x_2\rvert + 1\right) - \exp(\lvert x_3 + x_4\rvert + 1) + \cos(x_5 + x_6 - x_8) + \sqrt{x_8^2 + x_9^2 + x_{10}^2}$
$F_7(\mathbf{x})$	$(\arctan(x_1) + \arctan(x_2))^2 + \max(x_3 x_4 + x_6, 0) - \frac{1}{1 + (x_4 x_5 x_6 x_7 x_8)^2} + \left(\frac{\lvert x_7\rvert}{1 + \lvert x_9\rvert}\right)^5 + \sum_{i=1}^{10} x_i$
$F_8(\mathbf{x})$	$x_1 x_2 + 2^{x_3 + x_5 + x_6} + 2^{x_3 + x_4 + x_5 + x_7} + \sin(x_7 \sin(x_8 + x_9)) + \arccos(0.9x_{10})$
$F_9(\mathbf{x})$	$\tanh(x_1 x_2 + x_3 x_4)\sqrt{\lvert x_5\rvert} + \exp(x_5 + x_6) + \log\left((x_6 x_7 x_8)^2 + 1\right) + x_9 x_{10} + \frac{1}{1 + \lvert x_{10}\rvert}$
$F_{10}(\mathbf{x})$	$\sinh\left(x_1 + x_2\right) + \arccos\left(\tanh(x_3 + x_5 + x_7)\right) + \cos(x_4 + x_5) + \sec(x_7 x_9)$

Pairwise Interaction Detection A variant to our interaction ranking algorithm tests for all pairwise interactions. We rank all pairs of features $\{i, j\}$ according to their interaction strengths $\omega(\{i, j\})$ calculated on the first hidden layer, where again the averaging function is $\min(\cdot)$, and $\omega(\{i, j\}) = \sum_{s=1}^{p_1} \omega_s(\{i, j\})$. The higher the rank, the more likely the interaction exists.

4 Experiments

4.1 Experimental Setup

Averaging Function Our proposed NID framework relies on the selection of an averaging function (Equation 2). We experimentally determined the averaging function by comparing representative functions from the generalized mean family (Bullen et al., 1988): maximum, root mean square, arithmetic mean, geometric mean, harmonic mean, and minimum. To make the comparison, we used a test suite of 10 synthetic functions, which consist of a variety of interactions of varying order and overlap, as shown in Table 1. We trained 10 trials of *MLP* and *MLP-M* on each of the synthetic functions, obtained interaction rankings with our proposed greedy ranking algorithm (Algorithm 1), and counted the total number of correct interactions ranked before any false positive. In this evaluation, we ignore predicted interactions that are subsets of true higher-order interactions because the subset interactions are redundant (Section 2). As seen in Figure 2, the number of true top interactions we recover is highest with the averaging function, *minimum*, which we will use in all of our experiments. A simple analytical study on a bivariate hidden unit also suggests that the minimum is closely correlated with interaction strength (Appendix D).

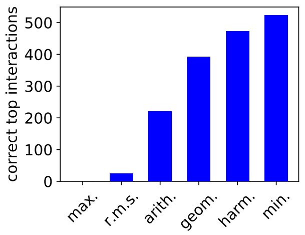


Figure 2: A comparison of averaging functions by the total number of correct interactions ranked before any false positives, evaluated on the test suite (Table 1). x-axis labels are maximum, root mean square, arithmetic mean, geometric mean, harmonic mean, and minimum.

Neural Network Configuration We trained feedforward networks of *MLP* and *MLP-M* architectures to obtain interaction rankings, and we trained *MLP-Cutoff* to find cutoffs on the rankings. In our experiments, all networks that model feature interactions consisted of four hidden layers with first-to-last layer sizes of: 140, 100, 60, and 20 units. In contrast, all individual univariate networks had three hidden layers with sizes of: 10, 10, and 10 units. All networks used ReLU activation and were trained using backpropagation. In the cases of *MLP-M* and *MLP-Cutoff*, summed networks were trained jointly. The objective functions were mean-squared error for regression and cross-entropy for classification tasks. On the synthetic test suite, *MLP* and *MLP-M* were trained with L1 constants in the range of 5e-6 to 5e-4, based on parameter tuning on a validation set. On real-world datasets, L1 was fixed at 5e-5. *MLP-Cutoff* used a fixed L2 constant of 1e-4 in all experiments involving cutoff. Early stopping was used to prevent overfitting.

Table 2: AUC of pairwise interaction strengths proposed by NID and baselines on a test suite of synthetic functions (Table 1). *ANOVA* and *HierLasso* are deterministic.

	ANOVA	HierLasso	AG	NID, *MLP*	NID, *MLP-M*
$F_1(\mathbf{x})$	0.992	1.00	1 ± 0.0	0.970 ± 9.2e−3	0.995 ± 4.4e−3
$F_2(\mathbf{x})$	0.468	0.636	0.88 ± 1.4e−2	0.79 ± 3.1e−2	0.85 ± 3.9e−2
$F_3(\mathbf{x})$	0.657	0.556	1 ± 0.0	0.999 ± 2.0e−3	1 ± 0.0
$F_4(\mathbf{x})$	0.563	0.634	0.999 ± 1.4e−3	0.85 ± 6.7e−2	0.996 ± 4.7e−3
$F_5(\mathbf{x})$	0.544	0.625	0.67 ± 5.7e−2	1 ± 0.0	1 ± 0.0
$F_6(\mathbf{x})$	0.780	0.730	0.64 ± 1.4e−2	0.98 ± 6.7e−2	0.70 ± 4.8e−2
$F_7(\mathbf{x})$	0.726	0.571	0.81 ± 4.9e−2	0.84 ± 1.7e−2	0.82 ± 2.2e−2
$F_8(\mathbf{x})$	0.929	0.958	0.937 ± 1.4e−3	0.989 ± 4.4e−3	0.989 ± 4.5e−3
$F_9(\mathbf{x})$	0.783	0.681	0.808 ± 5.7e−3	0.83 ± 5.3e−2	0.83 ± 3.7e−2
$F_{10}(\mathbf{x})$	0.765	0.583	1 ± 0.0	0.995 ± 9.5e−3	0.99 ± 2.1e−2
average	0.721	0.698	0.87 ± 1.4e−2	**0.92*** ± 2.3e−2	**0.92** ± 1.8e−2

*Note: The high average AUC of NID, *MLP* is heavily influenced by F_6.

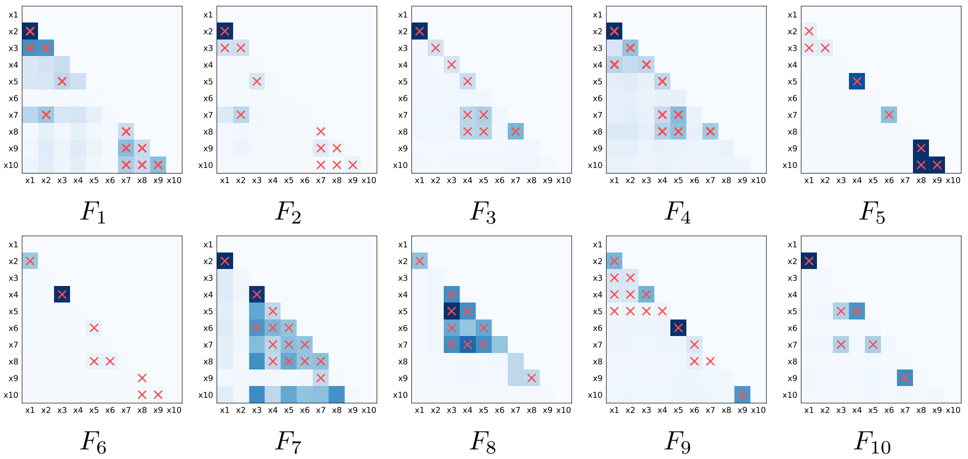
Datasets We study our interaction detection framework on both simulated and real-world experiments. For simulated experiments, we used a test suite of synthetic functions, as shown in Table 1. The test functions were designed to have a mixture of pairwise and higher-order interactions, with varying order, strength, nonlinearity, and overlap. F_1 is a commonly used function in interaction detection literature (Hooker, 2004; Sorokina et al., 2008; Lou et al., 2013). All features were uniformly distributed between -1 and 1 except in F_1, where we used the same variable ranges as reported in literature (Hooker, 2004).

We use four real-world datasets, of which two are regression datasets, and the other two are binary classification datasets. Specifically, the cal housing dataset is a regression dataset with 21k data points for predicting California housing prices (Pace & Barry, 1997). The bike sharing dataset contains 17k data points of weather and seasonal information to predict the hourly count of rental bikes in a bikeshare system (Fanaee-T & Gama, 2014). The higgs boson dataset has 800k data points for classifying whether a particle environment originates from the decay of a Higgs Boson (Adam-Bourdarios et al., 2014). Lastly, the letter recognition dataset contains 20k data points of transformed features for binary classification of letters on a pixel display (Frey & Slate, 1991). For all real-world data, we use random train/valid/test splits of $80/10/10$.

Baselines We compare the performance of NID to that of three baseline interaction detection methods. Two-Way *ANOVA* (Wonnacott & Wonnacott, 1972) utilizes linear models to conduct significance tests on the existence of interaction terms. *Hierarchical lasso* (HierLasso) (Bien et al., 2013) applies lasso feature selection to extract pairwise interactions. *Additive Groves* (*AG*) (Sorokina et al., 2008) is a nonparameteric means of testing for interactions by placing structural constraints on an additive model of regression trees. *AG* is a reference method for interaction detection because it directly detects interactions based on their non-additive definition.

4.2 Pairwise Interaction Detection

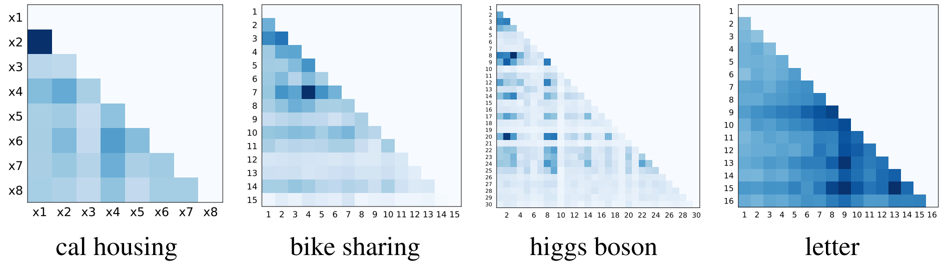
As discussed in Section 3, our framework NID can be used for pairwise interaction detection. To evaluate this approach, we used datasets generated by synthetic functions F_1-F_{10} (Table 1) that contain a mixture of pairwise and higher-order interactions, where in the case of higher-order interactions we tested for their pairwise subsets as in Sorokina et al. (2008); Lou et al. (2013). AUC scores of interaction strength proposed by baseline methods and NID for both *MLP* and *MLP-M* are shown in Table 2. We ran ten trials of *AG* and NID on each dataset and removed two trials with highest and lowest AUC scores. When comparing the AUCs of NID applied to *MLP* and *MLP-M*, we observe that the scores of *MLP-M* tend to be comparable or better, except the AUC for F_6. On one hand, *MLP-M* performed better on F_2 and F_4 because these functions contain main effects that *MLP* would model as spurious interactions with other variables. On the other hand, *MLP-M* performed worse on F_6 because it modeled *spurious main effects* in the $\{8, 9, 10\}$ interaction. Specifically, $\{8, 9, 10\}$ can be approximated as independent parabolas for each variable (shown in Appendix E). In our analyses of NID, we mostly focus on *MLP-M* because handling main effects is widely considered an important problem in interaction detection (Bien et al., 2013; Lim & Hastie, 2015; Kong et al., 2017). Comparing the AUCs of *AG* and NID for *MLP-M*, the scores tend to close, except for F_5, F_6, and F_8, where NID performs significantly better than *AG*. This performance difference may be



F_1 F_2 F_3 F_4 F_5

F_6 F_7 F_8 F_9 F_{10}

Figure 3: Heat maps of pairwise interaction strengths proposed by our NID framework on *MLP-M* for datasets generated by functions F_1-F_{10} (Table 1). Red cross-marks indicate ground truth interactions.



cal housing bike sharing higgs boson letter

Figure 4: Heat maps of pairwise interaction strengths proposed by our NID framework on *MLP-M* for real-world datasets.

due to limitations on the model capacity of *AG*, which is tree-based. In comparison to *ANOVA* and *HierLasso*, NID-*MLP-M* generally performs on par or better. This is expected because *ANOVA* and *HierLasso* are based on quadratic models, which can have difficulty approximating the interaction nonlinearities present in the test suite.

In Figure 3, heat maps of synthetic functions show the relative strengths of all possible pairwise interactions as interpreted from *MLP-M*, and ground truth is indicated by red cross-marks. The interaction strengths shown are normally high at the cross-marks. An exception is F_6, where NID proposes weak or negligible interaction strengths at the cross-marks corresponding to the $\{8, 9, 10\}$ interaction, which is consistent with previous remarks about this interaction. Besides F_6, F_7 also shows erroneous interaction strengths; however, comparative detection performance by the baselines is similarly poor. Interaction strengths are also visualized on real-world datasets via heat maps (Figure 4). For example, in the cal housing dataset, there is a high-strength interaction between x_1 and x_2. These variables mean longitude and latitude respectively, and it is clear to see that the outcome variable, California housing price, should indeed strongly depend on geographical location. We further observe high-strength interactions appearing in the heat maps of the bike sharing, higgs boson dataset, and letter datasets. For example, all feature pairs appear to be interacting in the letter dataset. The binary classification task from the letter dataset is to distinguish letters A-M from N-Z using 16 pixel display features. Since the decision boundary between A-M and N-Z is not obvious, it would make sense that a neural network learns a highly interacting function to make the distinction.

4.3 Higher-Order Interaction Detection

We visualize higher-order interaction detection on synthetic and real-world datasets in Figures 5 and 6 respectively. The plots correspond to the detection process as the ranking cutoff is applied (Section 3). The interaction rankings generated by NID for *MLP-M* are shown on the x-axes, and the blue bars correspond to the validation performance of *MLP-Cutoff* as interactions are added. For example, the plot for cal housing shows that adding the first interaction significantly reduces RMSE. We keep adding interactions into the model until reaching a cutoff point. In our experiments, we use a cutoff

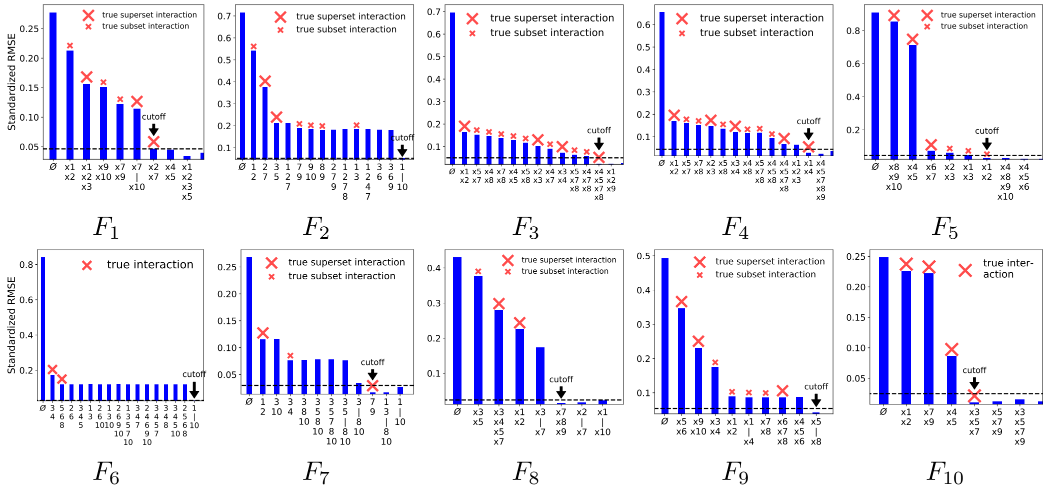


Figure 5: *MLP-Cutoff* error with added top-ranked interactions (along x-axis) of F_1-F_{10} (Table 1), where the interaction rankings were generated by the NID framework applied to *MLP-M*. Red cross-marks indicate ground truth interactions, and Ø denotes *MLP-Cutoff* without any interactions. Subset interactions become redundant when their true superset interactions are found.

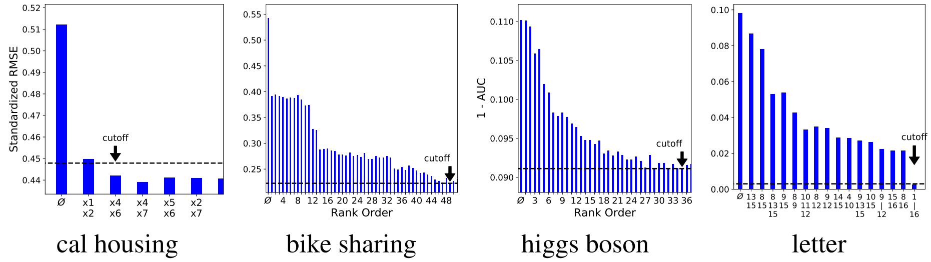


Figure 6: *MLP-Cutoff* error with added top-ranked interactions (along x-axis) of real-world datasets (Table 1), where the interaction rankings were generated by the NID framework on *MLP-M*. Ø denotes *MLP-Cutoff* without any interactions.

heuristic where interactions are no longer added after *MLP-Cutoff*'s validation performance reaches or surpasses *MLP-M*'s validation performance (represented by horizontal dotted lines).

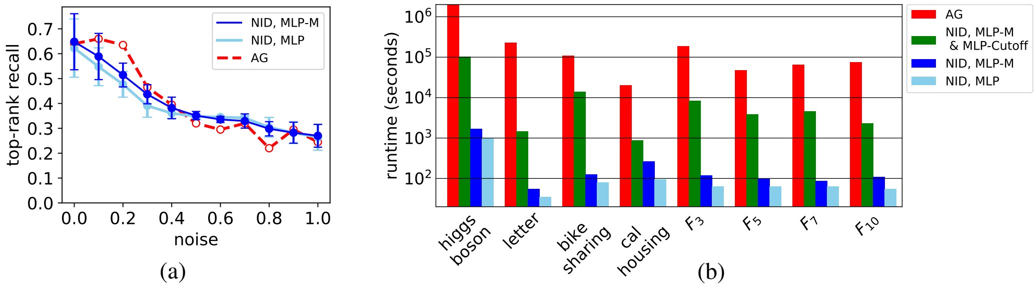
As seen with the red cross-marks, our method finds true interactions in the synthetic data of F_1-F_{10} before the cutoff point. Challenges with detecting interactions are again mainly associated with F_6 and F_7, which have also been difficult for baselines in the pairwise detection setting (Table 2). For the cal housing dataset, we obtain the top interaction $\{1, 2\}$ just like in our pairwise test (Figure 4, cal housing), where now the $\{1, 2\}$ interaction contributes a significant improvement in *MLP-Cutoff* performance. Similarly, from the letter dataset we obtain a 16-way interaction, which is consistent with its highly interacting pairwise heat map (Figure 4, letter). For the bike sharing and higgs boson datasets, we note that even when considering many interactions, *MLP-Cutoff* eventually reaches the cutoff point with a relatively small number of superset interactions. This is because many subset interactions become redundant when their corresponding supersets are found.

In our evaluation of interaction detection on real-world data, we study detected interactions via their predictive performance. By comparing the test performances of *MLP-Cutoff* and *MLP-M* with respect to *MLP-Cutoff* without any interactions (*MLP-Cutoff*^{Ø}), we can measure the relative test performance improvement obtained by including detected interactions. These relative performance improvements are shown in Table 3 for the real-world datasets as well as four selected synthetic datasets, where performance is averaged over ten trials per dataset. The results of this study show that a relatively small number of interactions of variable order are highly predictive of their corresponding datasets, as true interactions should.

We further study higher-order interaction detection of our NID framework by comparing it to *AG* in both interaction ranking quality and runtime. To assess ranking quality, we design a metric,

Table 3: Test performance improvement when adding top-K interactions from *MLP-M* to *MLP-Cutoff* for real-world datasets and select synthetic datasets. Here, the median $\bar{K}$ excludes subset interactions, and $|\bar{\mathcal{I}}|$ denotes average interaction cardinality. RMSE values are standard scaled.

Dataset	p	Relative Performance Improvement	Absolute Performance Improvement	$\bar{K}$	$\lvert\bar{\mathcal{I}}\rvert$
cal housing	8	99% ± 4.0%	0.09 ± 1.3e−2 RMSE	2	2.0
bike sharing	12	98.8% ± 0.89%	0.331 ± 4.6e−3 RMSE	12	4.7
higgs boson	30	98% ± 1.4%	0.0188 ± 5.9e−4 AUC	11	4.0
letter	16	101.1% ± 0.58%	0.103 ± 5.8e−3 AUC	1	16
$F_3(\mathbf{x})$	10	104.1% ± 0.21%	0.672 ± 2.2e−3 RMSE	4	2.5
$F_5(\mathbf{x})$	10	102.0% ± 0.30%	0.875 ± 2.2e−3 RMSE	6	2.2
$F_7(\mathbf{x})$	10	105.2% ± 0.30%	0.2491 ± 6.4e−4 RMSE	3	3.7
$F_{10}(\mathbf{x})$	10	105.5% ± 0.50%	0.234 ± 1.5e−3 RMSE	4	2.3



(a) (b)

Figure 7: Comparisons between *AG* and `NID` in higher-order interaction detection. (a) Comparison of top-ranked recall at different noise levels on the synthetic test suite (Table 1), (b) comparison of runtimes, where `NID` runtime with and without cutoff are both measured. `NID` detects interactions with top-rank recall close to the state-of-the-art *AG* while running orders of magnitude times faster.

top-rank recall, which computes a recall of proposed interaction rankings by only considering those interactions that are correctly ranked before any false positive. The number of top correctly-ranked interactions are then divided by the true number of interactions. Because subset interactions are redundant in the presence of corresponding superset interactions, only such superset interactions can count as true interactions, and our metric ignores any subset interactions in the ranking. We compute the top-rank recall of `NID` on *MLP* and *MLP-M*, the scores of which are averaged across all tests in the test suite of synthetic functions (Table 1) with 10 trials per test function. For each test, we remove two trials with max and min recall. We conduct the same tests using the state-of-the-art interaction detection method *AG*, except with only one trial per test because *AG* is very computationally expensive to run. In Figure 7a, we show top-rank recall of `NID` and *AG* at different Gaussian noise levels, and in Figure 7b, we show runtime comparisons on real-world and synthetic datasets. As shown, `NID` can obtain similar top-rank recall as *AG* while running orders of magnitude times faster.

4.4 Limitations

In higher-order interaction detection, our `NID` framework can have difficulty detecting interactions from functions with interlinked interacting variables. For example, a clique $x_1x_2 + x_1x_3 + x_2x_3$ only contains pairwise interactions. When detecting pairwise interactions (Section 4.2), `NID` often obtains an AUC of 1. However, in higher-order interaction detection, the interlinked pairwise interactions are often confused for single higher-order interactions. This issue could mean that our higher-order interaction detection algorithm fails to separate interlinked pairwise interactions encoded in a neural network, or the network approximates interlinked low-order interactions as higher-order interactions. Another limitation of our framework is that it sometimes detects spurious interactions or misses interactions as a result of correlations between features; however, correlations are known to cause such problems for any interaction detection method (Sorokina et al., 2008; Lou et al., 2013).

5 Conclusion

We presented our `NID` framework, which detects statistical interactions in data without searching an exponential solution space of interaction candidates. The framework detects interactions by interpreting the trained weights of a feedforward neural network.

Acknowledgments

We would like to thank Xinran He, Natali Ruchansky, Meisam Razaviyayn, and reviewers for their helpful feedback and insights. This work is supported by the National Science Foundation under award number IIS-1254206 and USC Annenberg Fellowships for MT and DC.

References

Claire Adam-Bourdarios, Glen Cowan, Cecile Germain, Isabelle Guyon, Balazs Kegl, and David Rousseau. Learning to discover: the higgs boson machine learning challenge. *URL https://higgsml.lal.in2p3.fr/documentation/*, 2014.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the royal statistical society. Series B (Methodological)*, pp. 289–300, 1995.

Jacob Bien, Jonathan Taylor, and Robert Tibshirani. A lasso for hierarchical interactions. *Annals of statistics*, 41(3):1111, 2013.

PS Bullen, DS Mitrinović, and PM Vasić. Means and their inequalities, mathematics and its applications, 1988.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1721–1730. ACM, 2015.

Yadolah Dodge. *The Oxford dictionary of statistical terms*. Oxford University Press on Demand, 2006.

Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2-3):113–127, 2014.

Ronald Aylmer Fisher. *Statistical methods for research workers*. Genesis Publishing Pvt Ltd, 1925.

Peter W Frey and David J Slate. Letter recognition using holland-style adaptive classifiers. *Machine learning*, 6(2):161–182, 1991.

G David Garson. Interpreting neural-network connection weights. *AI Expert*, 6(4):46–51, 1991.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *arXiv preprint arXiv:1606.08813*, 2016.

Yotam Hechtlinger. Interpretation of prediction models using the input gradient. *arXiv preprint arXiv:1611.07634*, 2016.

Giles Hooker. Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 575–580. ACM, 2004.

Yinfei Kong, Daoji Li, Yingying Fan, Jinchi Lv, et al. Interaction pursuit in high-dimensional multi-response regression via distance correlation. *The Annals of Statistics*, 45(2):897–922, 2017.

Michael Lim and Trevor Hastie. Learning interactions via hierarchical group-lasso regularization. *Journal of Computational and Graphical Statistics*, 24(3):627–654, 2015.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 623–631. ACM, 2013.

R Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33 (3):291–297, 1997.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016.

Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 2662–2670, 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Daria Sorokina, Rich Caruana, Mirek Riedewald, and Daniel Fink. Detecting statistical interactions with additive groves of trees. In *Proceedings of the 25th international conference on Machine learning*, pp. 1000–1007. ACM, 2008.

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

Kush R Varshney and Homa Alemzadeh. On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. *arXiv preprint arXiv:1610.01256*, 2016.

Thomas H Wonnacott and Ronald J Wonnacott. *Introductory statistics*, volume 19690. Wiley New York, 1972.

Simon Wood. *Generalized additive models: an introduction with R*. CRC press, 2006.

A Proof and Discussion for Proposition 2

Given a trained feedforward neural network as defined in Section 2, we can construct a directed acyclic graph $G = (V, E)$ based on non-zero weights as follows. We create a vertex for each input features and hidden units in the neural network: $V = \{v_{\ell,i} | \forall i, \ell\}$, where $v_{\ell,i}$ be the vertex corresponding to the i-th hidden unit in the ℓ-th layer. Note that the final output y is not included. We create edges based on the non-zero entries in the weight matrices, i.e., $E = \{(v_{\ell-1,i}, v_{\ell,j}) | \mathbf{W}^{\ell}_{j,i} \neq 0, \forall i, j, \ell\}$. Note that under the graph representation, the value of any hidden unit is a function of parent hidden units. We will also use vertices and hidden units interchangeably.

In feedforward neural networks with nonlinear activation functions, any interacting features must follow strongly weighted connections to a common hidden unit before the final output. That is, in the corresponding directed graph, interacting features will share at least one common descendant. The key observation is that non-overlapping paths in the network are aggregated via weighted summation at the final output without creating any interactions between features. The statement is rigorized in the following proposition with a proof. The reverse of this statement, that a common descendant will create an interaction among input features, holds true in most cases.

Proposition 2 (Interactions at Common Hidden Units)**.** *Consider a feedforward neural network with input feature $x_i, i \in [p]$, where $y = \varphi(x_1, \ldots, x_p)$. For any interaction $\mathcal{I} \subset [p]$ in $\varphi(\cdot)$, there exists a vertex $v_{\mathcal{I}}$ in the associated directed graph such that $\mathcal{I}$ is a subset of the ancestors of $v_{\mathcal{I}}$ at the input layer (i.e., $\ell = 0$).*

Proof. We prove Proposition 2 by contradiction.

Let $\mathcal{I}$ be an interaction where there is no vertex in the associated graph which satisfies the condition. Then, for any vertex $v_{L,i}$ at the L-th layer, the value f_i of the corresponding hidden unit is a function of its ancestors at the input layer $\mathcal{I}_i$ where $\mathcal{I} \not\subset \mathcal{I}_i$.

Next, we group the hidden units at the L-th layer into non-overlapping subsets by the first missing feature with respect to the interaction $\mathcal{I}$. That is, for element i in $\mathcal{I}$, we create a index set $\mathcal{S}_i \in [p_L]$:

$$\mathcal{S}_i = \{j \in [p_L] | i \notin \mathcal{I}_j \text{ and } \forall i' < i, j \notin \mathcal{S}_{i'}\}.$$

Note that the final output of the network is a weighed summation over the hidden units at the L-th layer:

$$\varphi(\mathbf{x}) = b^y + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{S}_i} w^y_j f_j\left(\mathbf{x}_{\mathcal{I}_j}\right),$$

Since $\sum_{j \in \mathcal{S}_i} w^y_j f_j\left(\mathbf{x}_{\mathcal{I}_j}\right)$ is not a function of x_i, we have that $\varphi(\cdot)$ is a function without the interaction $\mathcal{I}$, which contradicts our assumption. □

The reverse of this statement, that a common descendant will create an interaction among input features, holds true in most cases. The existence of counterexamples is manifested when early hidden layers capture an interaction that is negated in later layers. For example, the effects of two interactions may be directly removed in the next layer, as in the case of the following expression: $\max\{w_1x_1 + w_2x_2, 0\} - \max\{-w_1x_1 - w_2x_2, 0\} = w_1x_1 + w_2x_2$. Such an counterexample is legitimate; however, due to random fluctuations, it is highly unlikely in practice that the w_1s and the w_2s from the left hand side are exactly equal.

B Proof for Lemma 3

We show that our definition of hidden unit influence (Equation 1) satisfies upper bounds on the gradient magnitudes of hidden units by proving it computes Lipschitz constants for corresponding units. Gradients have been commonly used as variable importance measures in neural networks, especially input gradients which compute directions normal to decision boundaries (Ross et al., 2017; Goodfellow et al., 2015; Simonyan et al., 2013). Thus, an upper bound on the gradient magnitude approximates how important the variable can be.

Lemma 3 (Neural Network Lipschitz Estimation)**.** *Let the activation function $\phi(\cdot)$ be a 1-Lipschitz function. Then the output y is $z^{(\ell)}_i$-Lipschitz with respect to $h^{(\ell)}_i$.*

Proof. For non-differentiable $\phi(\cdot)$ such as the ReLU function, we can replace it with a series of differentiable 1-Lipschitz functions that converges to $\phi(\cdot)$ in the limit. Therefore, without loss of generality, we assume that $\phi(\cdot)$ is differentiable with $|\partial_x \phi(x)| \leq 1$. We can take the partial derivative of the final output with respect to $h_i^{(\ell)}$, the i-th unit at the ℓ-th hidden layer:

$$\begin{aligned}\frac{\partial y}{\partial h_i^{(\ell)}} =& \sum_{j_{\ell+1},\ldots,j_L} \frac{\partial y}{\partial h_{j_L}^{(L)}} \frac{\partial h_{j_L}^{(L)}}{\partial h_{j_{L-1}}^{(L-1)}} \cdots \frac{\partial h_{j_{\ell+1}}^{(\ell+1)}}{\partial h_i^{(\ell)}} \\ =& \mathbf{w}^{y\top} \mathrm{diag}(\dot{\boldsymbol{\phi}}^{(L)}) \mathbf{W}^{(L)} \cdots \mathrm{diag}(\dot{\boldsymbol{\phi}}^{(\ell+1)}) \mathbf{W}^{(\ell+1)},\end{aligned}$$

where $\dot{\boldsymbol{\phi}}^{(\ell)} \in \mathbb{R}^{p_\ell}$ is a vector that

$$\dot{\phi}_k^{(\ell)} = \partial_x \phi\left(\mathbf{W}_{k,:}^{(\ell)} \mathbf{h}^{(\ell-1)} + b_k^{(\ell)}\right).$$

We can conclude the Lemma by proving the following inequality:

$$\left|\frac{\partial y}{\partial h_i^{(\ell)}}\right| \leq |\mathbf{w}^y|^\top \left|\mathbf{W}^{(L)}\right| \cdots \left|\mathbf{W}_{:,i}^{(\ell+1)}\right| = z_i^{(\ell)}.$$

The left-hand side can be re-written as

$$\sum_{j_{\ell+1},\ldots,j_L} w_{j_L}^y \dot{\phi}_{j_L}^{(L)} W_{j_L,j_{L-1}}^{(L)} \dot{\phi}_{j_{L-1}}^{(L-1)} \cdots \dot{\phi}_{j_{\ell+1}}^{(\ell+1)} W_{j_{\ell+1},i}^{(\ell+1)}.$$

The right-hand side can be re-written as

$$\sum_{j_{\ell+1},\ldots,j_L} |w_{j_L}^y| \left|W_{j_L,j_{L-1}}^{(L)}\right| \cdots \left|W_{j_{\ell+1},i}^{(\ell+1)}\right|.$$

We can conclude by noting that $|\partial_x \phi(x)| \leq 1$. □

C Proof for Theorem 4

In addition to efficiency, a benefit of Algorithm 1 with $\mu(\cdot) = \min(\cdot)$ is that it automatically improves the ranking of a higher-order interaction over its redundant subsets. This allows the higher-order interaction to have a better chance of ranking above any false positives and being captured in the cutoff stage. We justify this improvement by proving Theorem 4 under a mild assumption.

Theorem 4 (Improving the ranking of higher-order interactions)**.** *Let $\mathcal{R}$ be the set of interactions proposed by Algorithm 1, let $\mathcal{I} \in \mathcal{R}$ be a d-way interaction where $d \geq 3$, and let $\mathcal{S}$ be the set of subset $(d-1)$-way interactions of $\mathcal{I}$ where $|\mathcal{S}| = d$. Assume that for any hidden unit j which proposed $s \in \mathcal{S} \cap \mathcal{R}$, $\mathcal{I}$ will also be proposed at the same hidden unit, and $\omega_j(\mathcal{I}) > \frac{1}{d}\omega_j(s)$. Then, one of the following must be true: a) $\exists s \in \mathcal{S} \cap \mathcal{R}$ ranked lower than $\mathcal{I}$, i.e., $\omega(\mathcal{I}) > \omega(s)$, or b) $\exists s \in \mathcal{S}$ where $s \notin \mathcal{R}$.*

Proof. Suppose for the purpose of contradiction that $\mathcal{S} \subseteq \mathcal{R}$ and $\forall s \in \mathcal{S}$, $\omega(s) \geq \omega(\mathcal{I})$. Because $\omega_j(\mathcal{I}) > \frac{1}{d}\omega_j(s)$,

$$\omega(\mathcal{I}) = \sum_{s \in \mathcal{S} \cap \mathcal{R}} \sum_{j \text{ propose } s} z_j \omega_j(\mathcal{I}) > \frac{1}{d} \sum_{s \in \mathcal{S} \cap \mathcal{R}} \sum_{j \text{ propose } s} z_j \omega_j(s) = \frac{1}{d} \sum_{s \in \mathcal{S} \cap \mathcal{R}} \omega(s).$$

Since $\forall s \in \mathcal{S}$, $\omega(s) \geq \omega(\mathcal{I})$,

$$\frac{1}{d} \sum_{s \in \mathcal{S} \cap \mathcal{R}} \omega(s) \geq \frac{1}{d} \sum_{s \in \mathcal{S} \cap \mathcal{R}} \omega(\mathcal{I})$$

Since $\mathcal{S} \subseteq \mathcal{R}$, $|\mathcal{S} \cap \mathcal{R}| = d$. Therefore,

$$\frac{1}{d}\sum_{s\in\mathcal{S}\cap\mathcal{R}}\omega(\mathcal{I}) \geq \frac{1}{d}\omega(\mathcal{I})d \geq \omega(\mathcal{I}),$$

which is a contradiction.

□

Under the noted assumption, the theorem in part a) shows that a d-way interaction will improve over one its $d-1$ subsets in rankings as long as there is no sudden drop from the weight of the $(d-1)$-way to the d-way interaction at the same hidden units. We note that the improvement extends to b) as well, when $d = |\mathcal{S}\cap\mathcal{R}| > 1$.

D Pairwise Interaction Strength via Quadratic Approximation

We provide an interaction strength analysis on a bivariate ReLU function: $\max\{\alpha_1 x_1 + \alpha_2 x_2, 0\}$, where x_1, x_2 are two variables and α_1, α_2 are the weights for this simple network. We quantify the strength of the interaction between x_1 and x_2 with the cross-term coefficient of the best quadratic approximation. That is,

$$\beta_0,\ldots,\beta_5 = \underset{\beta_i, i=0,\ldots,5}{\operatorname{argmin}} \iint_{-1}^{1} \Big[\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2 + \beta_5 x_1 x_2 \\ - \max\{\alpha_1 x_1 + \alpha_2 x_2, 0\}\Big]^2 dx_1\, dx_2.$$

Then for the coefficient of interaction $\{x_1, x_2\}$, β_5, we have that,

$$|\beta_5| = \frac{3}{4}\left(1 - \frac{\min\{\alpha_1^2, \alpha_2^2\}}{5\max\{\alpha_1^2, \alpha_2^2\}}\right)\min\{|\alpha_1|, |\alpha_2|\}. \tag{3}$$

Note that the choice of the region $(-1,1)\times(-1,1)$ is arbitrary: for a larger region $(-c,c)\times(-c,c)$ with $c > 1$, we found that $|\beta_5|$ scales with c^{-1}. Note that the factor before $\min\{|\alpha_1|, |\alpha_2|\}$ in Equation (3) is almost a constant with less than 20% fluctuation. This analysis suggests that the interaction strength of a bivariate ReLU function can be well-modeled by the minimum value between $|\alpha_1|$ and $|\alpha_2|$.

E Spurious Main Effect Approximation

In the synthetic function F_6 (Table 2), the $\{8, 9, 10\}$ interaction, $\sqrt{x_8^2 + x_9^2 + x_{10}^2}$, can be approximated as main effects for each variable x_8, x_9, and x_{10} when at least one of the three variables is close to -1 or 1. Note that in our experiments, these variables were uniformly distributed between -1 and 1.

For example, let $x_{10} = 1$ and $z^2 = x_8^2 + x_9^2$, then by taylor series expansion at $z = 0$,

$$\sqrt{z^2+1} \approx 1 + \frac{1}{2}z^2 = 1 + \frac{1}{2}x_8^2 + \frac{1}{2}x_9^2.$$

By symmetry under the assumed conditions,

$$\sqrt{x_8^2 + x_9^2 + x_{10}^2} \approx c + \frac{1}{2}x_8^2 + \frac{1}{2}x_9^2 + \frac{1}{2}x_{10}^2,$$

where c is a constant.

In Figure 8, we visualize the x_8, x_9, x_{10} univariate networks of a *MLP-M* (Figure 1) that is trained on F_6. The plots confirm our hypothesis that the *MLP-M* models the {8,9,10} interaction as spurious main effects with parabolas scaled by $\frac{1}{2}$.

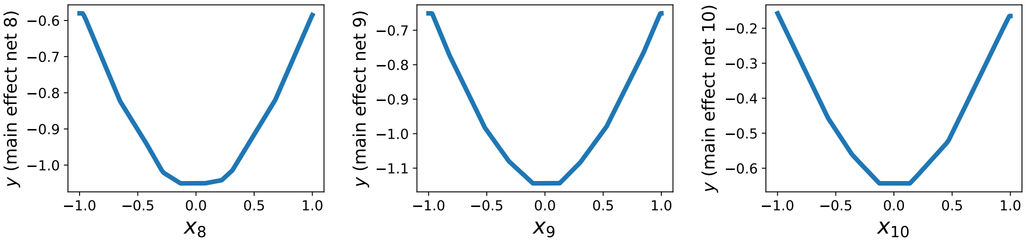


Figure 8: Response plots of an *MLP-M*'s univariate networks corresponding to variables x_8, x_9, and x_{10}. The *MLP-M* was trained on data generated from synthetic function F_6 (Table 2). Note that the plots are subject to different levels of bias from the *MLP-M*'s main multivariate network.