# Readings on ML Following Double Debiased ML Cites

Manu Navjeevan

June 25, 2020

## Contents

# 1 Generalized Random Forests; ***Susan Athey, Julie Tibshirani, Setfan Wager (AOS, 2018)***

This paper can be found on ArXiv here.

## 1.1 Introduction

- Random Forests first introduced by Breiman (2001)
- Used for conditional mean estimation. Given a data generating distribution for $(X_i, Y_i) \in \mathcal{X} \times \mathbb{R}$, want to estimate

$$\mu(x) = \mathbb{E}[Y|X_i = x] \tag{1}$$

- Paper extends this to a flexible method for estimating any quantity $\theta(x)$ defined via local moment conditions. Specifically, given data $(X_i, O_i) \in \mathcal{X} \times \mathcal{O}$, we want forest based estimates of $\theta(x)$ defined by a local moment condition of the form

$$\mathbb{E}[\psi_{\theta(x),\nu(x)}(O_i)|X_i = x] = 0, \quad \text{for all } x \in \mathcal{X} \tag{2}$$

  where $\psi(\cdot)$ is a score function and $\nu(\cdot)$ is an optional nuisance parameter.

  - For example, if we model the distribution of $O_i$ conditional on $X_i$ to have a density $f_{\theta(x),\nu(x)}(\cdot)$ then the moment condition one with $\psi = \nabla \log f_{\theta(x),\nu(x)}(\cdot)$ ideintifies the local maximum likelihood
  - Substantive application involved heterogeneous treatment effect estimation with IV

- Aim is to build a famly of non-parametric estimators that inherit desirable empirical properties of regression forests: stability, ease of use, flexible adaptation to different functional forms
- Regression forests typically understood as ensemble methods

$$\hat{\mu}(x) = B^{-1} \sum_{b=1}^{B} \hat{\mu}_b(x)$$

  because individual trees have low bias but high variance, this averaging stabilizes predictions.

  This method may not work as well when we are given moment conditions as in 2. Noisy solutions to moment equations are generally biased and averaging would do nothing to alleviate the bias.

- Cast forests as a type of adaptive locally weighted estimatior that first uses a forest to calculate a weighted set of neighbors for each test point $x$ and then solves a plug-in version of 2 using these neighbors.
  - Previously advocated by Hotheorn et. al (2004) in the context of survival analysis and by Meinshausen (2006) for quantile regression
  - For conditonal mean estimation the averaging and weighting views of forests are equivalent, for moment conditions the weighting based perspective proves more effective
- Bulk of this paper is devoted to theoretical analysis of generalized random forests

### 1.1.1 Related Work

- Idea of local maximum likelihood has a long history. Core idea: when estimating parameters at a particular value of covariates, a kernel weighting function is used to place more weight on nearby observations in the covariate space. Paper replaces the kernel weighting function with forest based weights.
  - Weights dervied from the fraction of trees in which an observation appears in the same leag as the target value of the covariate vector.

- If the covariate space has more than a few dimensions kernel methods can suffer from curse of dimensionality.

## 1.2 Generalized Random Forests

- In the standard classification or regresion forests proposed by Breiman (2001), prediction for a particular point $x$ is determined by averaging predictions across an ensemble of different trees.

Suppose that we have $n$ independent and identically distributed samples, indeced $i = 1, \ldots, n$. For each sample, access to an observable quantity $O_i$ that encodes information relevant to estimation $\theta(\cdot)$, along with a set of auxilary covariates $X_i$.

- In the case of NPR; $O_i = \{Y_i\}, Y_i \in \mathbb{R}$, though in general it may contain richer information.
  - In the case of treatment effect estimation weith exogeneous treatment assignment, $O_i = \{Y_i, W_i\}$ where $W_i$ represents the treatment assignment.

Given this type of data, the goal is to estimation solutions to local estimation equations of the form $\mathbb{E}[\psi_{\theta(x),\nu x}(O_i)|X_i = x] = 0$ (Eq. 2), for all $x \in \mathcal{X}$. We care about $\theta(x)$ and $\nu(x)$ is a nuisance parameter.

One approach: Define some similarity weights $\alpha_i(x)$ that measure the relevance of the $i$-th training example to fitting $\theta(\cdot)$ at $x$ and then fit the target of interest via an empirical version of the estimation equation

$$\left(\hat{\theta}(x), \hat{\nu}(x)\right) \in \arg\min_{\theta,\nu} \left\{ \left\| \sum_{i=1}^{n} \alpha_i(x)\psi_{\theta,\nu}(O_i) \right\|_2 \right\} \tag{3}$$

If the expresion has a unique root we can say that the estimators "solve" eq. 3. Weights used in the above equations are traditionally obtained via a deterministic kernel function, perhaps with an adaptively chosen bandwith parameter. This method of choosing weights suffers from curse of dimensionality. This paper uses forest-based algorithms to adaptively learn better, problem specifc, weights, $\alpha_i(x)$) that van be used in conjunction with eq. 3.

1. Grow a set of $B$ trees indicated by $b = 1, \ldots, B$ and, for each such tree, define $L_b(x)$ as the set of training examples falling in the same "leaf" as $x$.
2. Define the weights as the frequency with with the $i$-th training example falls into the same leaf as $x$:

$$\alpha_{bi}(x) = \frac{\mathbf{1}\{X_i \in L_b(x)\}}{|L_b(x)|} \tag{4}$$

   These weights sum to 1 and define the forest based adaptive neighborhood of $x$.

Construction of the trees and the "neighbor" sets $L_b(x)$ require some subtleties. In particular, construction will rely on both subsampling and specific form of sample splitting to achieve consistency.

- For the special case of regression trees, the weighting based definition of a random forest is equivalent to the standard "average of trees" perspective taken in Breiman (2001)

### 1.2.1 Splitting to Maximize Heterogeneity

Seek trees that, when combined into a forest, induce weights $\alpha_i(x)$ that lead to good estimates of $\theta(x)$. Random forests use recursive partitioning on subsamples to generate these weights $\alpha_i(x)$. Algorithm considered in the paper mimics Breiman (2001) as closely as possible, while tailoring splitting to focus on heterogeneity in $\theta(x)$.

Use a greedy algorithm to look for splits. Each split starts with a parent node $P \subset \mathcal{X}$. Given a sample $\mathcal{J}$, define $(\hat{\theta}_P, \hat{\nu}_p)(\mathcal{J})$ as

$$\left(\hat{\theta}_P, \hat{\nu}_P\right) \in \arg\min_{\theta,\nu} \left\{ \left\| \sum_{\{i \in \mathcal{J}, X_i \in P\}} \psi_{\theta,\nu}(O_i) \right\|_2 \right\}^1 \tag{5}$$

This contrasts to (4) because there is no weighting. Would like to divide $P$ into two children, $C_1, C_2 \subset \mathcal{X}$ using an axis-aligned cut[2] to improve the accuracy of our $\theta$ estimates as much as possible. Formally, this means seeking to minimize

$$\operatorname{err}(C_1, C_2) = \sum_{j=1,2} \mathbb{P}\left[X \in C_j | X \in P\right] \mathbb{E}\left[\left(\hat{\theta}_{C_j} - \theta(X)\right)^2 | X \in C_j\right]$$

where $\hat{\theta}_{C_j}(\mathcal{J})$ are fit over children $C_j$ as in eq. 5. Expectations are taken over both the randomness in $\hat{\theta}_{C_j}(\mathcal{J})$ and a new test point $X$. This is to say, the err function is the "true" function we want to minimize.

Many standard regression tree implementations choose splits by minimizing prediction error of the node. This corresponds to $\operatorname{err}(C_1, C_2)$ with plug in estimators from the training sample. Athey and Imbens (2016) study sample-splitting trees to estimate a treatment effect. They propose an unbiased, model-free (nonparametric) estimate of $\operatorname{err}(C_1, C_2)$ using an overfitting penalty as in Mallows (1973). In the general moment condition setting as defined by 2 this may not work. If $\theta(x)$ is defined only by a moment condition, then we do not in general have access to an unbiased, model free estimate of the criterion $\operatorname{err}(C_1, C_2)$. The following proposition tries to address this.

**Proposition 1.** *Suppose that the basic assumption detaled later in Section 3 hold, and that the parent node $P$ has a radius smaller than $r > 0$. We write $n_P = |\{i \in \mathcal{J} : X_i \in P\}|$ for the number of observations in the parent and $n_{C_j}$ for the number of observations in each child and define*

$$\Delta(C_1, C_2) := n_{C_1} n_{C_2} / n_P^2 \left(\hat{\theta}_{C_1}(\mathcal{J}) - \hat{\theta}_{C_2}(\mathcal{J})\right)^2 \tag{6}$$

*where $\hat{\theta}_{C_1}, \hat{\theta_{C_2}}$ are the solutions to the estimating equation computer in the children, following eq. 5. Then, treating the child notes $C_1, C_2$ as well as the corresponding counts $n_{C_1}, n_{C_2}$ as fized, and assuming that $n_{C_i} \gg r^{-2}$ we have that*

$$\operatorname{err}(C_1, C_2) = K(P) - \mathbb{E}[\Delta(C_1, C_2)] + o(r^2)$$

*where $K(P)$ is a deterministic term that measures the purity of the parent node that does not depend on how the parent is split, and the o-term incorporates terms that depend on sampling variance.*

Motivated by this observation, paper considers splits that make the above $\Delta$-criterion in eq. 6 large.

### 1.2.2 The Gradient Tree Algorithm

Above discussion provides conceptual guidance on how to pick good splits. But actually optimizing the criterion $\Delta(C_1, C_2)$ over all possible axis-aligned cuts while also solving for $(\hat{\theta}, \hat{\nu})$ at each leaf can be computationally expensive. To avoid the issue, paper proposes optimizing an approximate criterion $\tilde{\Delta}(C_1, C_2)$ using gradient based approximations for $(\hat{\theta}_{C_1}, \hat{\theta}_{C_2})$. For each child $C$, use $\tilde{\theta}_C \approx \hat{\theta}_C$ as follows: First, compute $A_P$ as any consistent estimate for the gradient of the expectation of $\psi$ function; i.e, $A_P \to \nabla \mathbb{E}[\psi_{\hat{\theta}_P, \hat{\nu}_P}(O_i)]$. Then, set

$$\tilde{\theta} = \hat{\theta} - \frac{1}{|\{i : X_i \in C\}|} \sum_{\{i : X_i \in C\}} \xi^T A_P^{-1} \psi_{\hat{\theta}_P, \hat{\nu}_P}(O_i) \tag{7}$$

[1] Minimize the $L_2$ norm because we want the moment condition to be as close to zero as possible

[2] Axis-aligned means that the cut considers only one variable at a time. See link for a visual representation

$\hat{\theta}_P$ and $\hat{\nu}_P$ are obtained by solving eq. 5 once in the parent node and $\xi$ is a vector that picks out the $\theta$ coordinate from the vector $(\theta, \nu)$. When $\psi$ is itself continuously differentiable we use

$$A_P = \frac{1}{|\{i : X_i \in P\}|} \sum_{\{i:X_i \in P\}} \nabla \psi_{\hat{\theta},\hat{\nu}}(O_i) \tag{8}$$

Algorithim's recursive partitioning scheme reduces to alternatively applying the following two steps. First, in a **labeling step**, compute $\hat{\theta}_P, \hat{\nu}_P$ and the derivative matrix $A_P^{-1}$ on the parent data as in eq. 5, and use them to get the psuedo-outcomes

$$\rho_i = -\xi^T A_P^{-1} \psi_{\hat{\theta}_P,\hat{\nu}_P}(O_i) \in \mathbb{R} \tag{9}$$

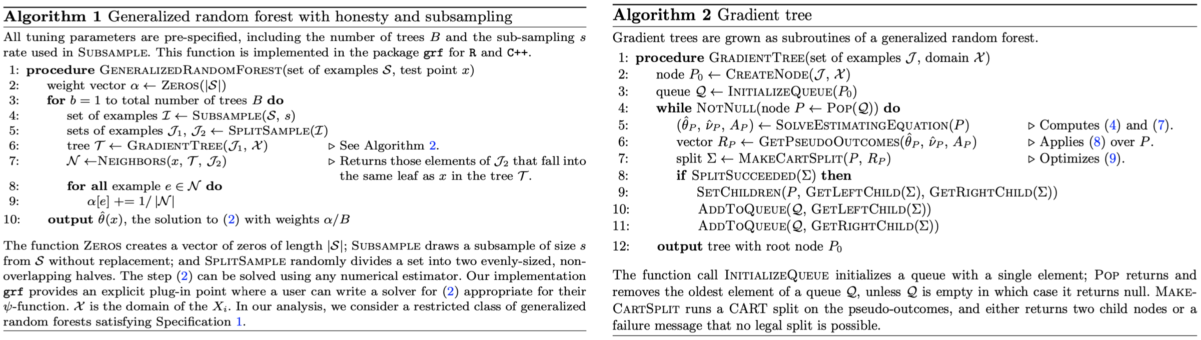
Next in a **regression step**, run a standard CART regression split on the outcome $\rho_i$. Specifically, we split $P$ into two axis-aligned children $C_1$ and $C_2$ such as to maximize the criterion

$$\tilde{\Delta}(C_1, C_2) = \sum_{j=1}^{2} \frac{1}{|\{i : X_i \in C_j\}|} \left( \sum_{\{i:X_i \in C_j\}} \rho_i \right)^2 \tag{10}$$

Once the regression step has been executed, relabel observations in each child by solving the estimating equation, and continue on recursively. [3]

- In the simplest case of least square regression (mean regression), with $\psi_{\theta(x)}(Y) = Y - \theta(x)$ the labeling step in eq. 9 doesn't change anything. The second step in maximizing eq. 10 corresponds to the usual way of making splite in Breiman (2001).
    - Special structure of the problem considered in this paper is encoded into eq. 9.

This approach is expected to provide more consistent computational performance than optimizing 6 at each step. Computation in growing a tree is typicall dominated by the split-selection step, so it is critical for this step to be implemented as effeciently as possible. Conversely the labeling step is only solved once per node, so is less performance sensitive. The algorithms for doing this are specified below:



**Algorithm 1** Generalized random forest with honesty and subsampling

All tuning parameters are pre-specified, including the number of trees $B$ and the sub-sampling $s$ rate used in SUBSAMPLE. This function is implemented in the package **grf** for R and C++.

```
procedure GeneralizedRandomForest(set of examples $\mathcal{S}$, test point $x$)
    weight vector $\alpha \leftarrow$ Zeros($|\mathcal{S}|$)
    for $b = 1$ to total number of trees $B$ do
        set of examples $\mathcal{I} \leftarrow$ Subsample($\mathcal{S}$, $s$)
        sets of examples $\mathcal{J}_1$, $\mathcal{J}_2 \leftarrow$ SplitSample($\mathcal{I}$)
        tree $\mathcal{T} \leftarrow$ GradientTree($\mathcal{J}_1$, $\mathcal{X}$)        ▷ See Algorithm 2.
        $\mathcal{N} \leftarrow$Neighbors($x$, $\mathcal{T}$, $\mathcal{J}_2$)        ▷ Returns those elements of $\mathcal{J}_2$ that fall into
                                                          the same leaf as $x$ in the tree $\mathcal{T}$.
        for all example $e \in \mathcal{N}$ do
            $\alpha[e]$ += $1/|\mathcal{N}|$
    output $\hat{\theta}(x)$, the solution to (2) with weights $\alpha/B$
```

The function ZEROS creates a vector of zeros of length $|\mathcal{S}|$; SUBSAMPLE draws a subsample of size $s$ from $\mathcal{S}$ without replacement; and SPLITSAMPLE randomly divides a set into two evenly-sized, non-overlapping halves. The step (2) can be solved using any numerical estimator. Our implementation **grf** provides an explicit plug-in point where a user can write a solver for (2) appropriate for their $\psi$-function. $\mathcal{X}$ is the domain of the $X_i$. In our analysis, we consider a restricted class of generalized random forests satisfying Specification 1.

(a) Algorithm 1

**Algorithm 2** Gradient tree

Gradient trees are grown as subroutines of a generalized random forest.

```
procedure GradientTree(set of examples $\mathcal{J}$, domain $\mathcal{X}$)
    node $P_0 \leftarrow$ CreateNode($\mathcal{J}$, $\mathcal{X}$)
    queue $\mathcal{Q} \leftarrow$ InitializeQueue($P_0$)
    while NotNull(node $P \leftarrow$ Pop($\mathcal{Q}$)) do
        ($\hat{\theta}_P$, $\hat{\nu}_P$, $A_P$) $\leftarrow$ SolveEstimatingEquation($P$)        ▷ Computes (4) and (7).
        vector $R_P \leftarrow$ GetPseudoOutcomes($\hat{\theta}_P$, $\hat{\nu}_P$, $A_P$)        ▷ Applies (8) over $P$.
        split $\Sigma \leftarrow$ MakeCartSplit($P$, $R_P$)        ▷ Optimizes (9).
        if SplitSucceeded($\Sigma$) then
            SetChildren($P$, GetLeftChild($\Sigma$), GetRightChild($\Sigma$))
            AddToQueue($\mathcal{Q}$, GetLeftChild($\Sigma$))
            AddToQueue($\mathcal{Q}$, GetRightChild($\Sigma$))
    output tree with root node $P_0$
```

The function call INITIALIZEQUEUE initializes a queue with a single element; POP returns and removes the oldest element of a queue $\mathcal{Q}$, unless $\mathcal{Q}$ is empty in which case it returns null. MAKECARTSPLIT runs a CART split on the pseudo-outcomes, and either returns two child nodes or a failure message that no legal split is possible.

(b) Algorithm 2

Figure 1: Algorithms for growing generalized random forests

In contrast to using a regression splitting criterion as in 10, which only requres a single pass over the data in the parent node, directly optimizing the original criterion in eq. 6 may require optimizing at every possible candidate split. This sort of gradient based approximation also underlies other popular statistical algorithms, including gradient boosting (Friedman, 2001) and model based recursive partitioning algorithm of Zeileis, Hothorn, and Hornik (2008).

Paper can verify that the error from using the approcimate criterion $\tilde{\Delta}$ instead of the exact $\Delta$-criterion is within the tolerance used to motivate the $\Delta$-criterion in Proposition 1, thus suggesting that use of it may

[3] This whole section is really going over the recursive step of the algorithm

not result in too much ineffeciency. Consistent estimates of $A_P$ can, in general, be derived directly, without relying on the proposition below

**Proposition 2.** *Under the conditions of Proposition 1, if* $|A_P - \nabla\mathbb{E}[\psi_{\hat{\theta}_P,\hat{\nu}_P}(O_i)|X_i \in P]| \to_P 0$*, then* $\Delta(C_1, C_2)$ *and* $\tilde{\Delta}(C_1, C_2)$ *are approximately equivalent in that*

$$\tilde{\Delta}(C_1, C_2) = \Delta(C_1, C_2) + o_P\left(\max\{r^2, 1/n_{C_1}, 1/n_{C_2}\}\right)$$

Now, given a practical splitting scheme for growing individual trees, we want to grow a forest that allows for consistent estimation of $\theta(x)$ using 5 using the forest weights in eq. 4. Each tree will provide small, relevant neighborhoods for $x$ that will lead to noisy estimates of $\theta(x)$; then we may hope that forest based aggregation will provide a single larger but still relevant neighborhood of $x$ that yields stable estimates $\hat{\theta}(x)$. Rely on two conceptual ideas that have proven to be succesful in the literature on forest-based least-squares regression. Training trees on subsamples of the data nd a subsampling splitting technique called “honesty”.

### 1.3 Asymptotic Analysis

Aim of this section is to establish asymptotic Gaussianity of the $\hat{\theta}(x)$ and of providing tools for statistical inference about $\theta(x)$. The covariate space and the parameter space are both subsets of Euclidean space. Specifically $\mathcal{X} = [0,1]^p$ and $(\theta, \nu) \in \mathcal{B} \subset \mathbb{R}^k$ for some $p, k > 0$ and $\mathcal{B}$ is a compact subset.[4] Moreover, we assume that $X$ has a density that is bounded away from 0 and from above. This is a weaker requirement in the forest prediction space since trees and forests are invariant to monotone rescaling of the features.

Some practically interesting cases, such as quantile regression involve discontinuous score functions $\psi$, which complicates analysis. Here we assume that the spected score function

$$M_{\theta,\nu}(x) := \mathbb{E}[\psi_{\theta,\nu}(O)|X = x] \tag{11}$$

varies smoothly in the parameters, even though $\psi$ itself may be discontinuous. For example, with quantile regression $\psi_\theta(Y) = \mathbf{1}(\{Y > \theta\}) - (1-q)$ is discontinuous in $q$ and $Y$, but $M_\theta(x) = \mathbb{P}[Y > \theta|X = x] - (1-q)$ is smooth whenever $Y|X = x$ has a smooth density. We add the following assumptions

**Assumption 1.** (Lipschitz $x$-signal) For fixed valued of $(\theta, \nu)$ we assume that $M_{\theta,\nu}(x)$ is Lipschitz continuous in $x$.

**Assumption 2.** (Smooth identification) When $x$ is fixed, assume that the $M$-function is twice continuously differentiable in $(\theta, \nu)$ with a uniformly bounded second derivative, and that $V(x) := V_{\theta(x),\nu(x)}(x)$ is invertible for $x \in \mathcal{X}$, with $V_{\theta,\nu}(x) := \frac{\partial}{\partial(\theta,\nu)} M_{\theta,\nu}(x)\Big|_{\theta(x),\nu(x)}$.

**Assumption 3.** (Lipschitz $(\theta, \nu)$-variogram) The score functions $\psi_{\theta,\nu}(O_i)$ have a continuous covariance structure. Writing $\gamma$ for the worst-case variogram and $\|\cdot\|_F$ for the Frobenius norm, then for some $L > 0$

$$\gamma\left(\begin{pmatrix}\theta \\ \nu\end{pmatrix}, \begin{pmatrix}\theta' \\ \nu'\end{pmatrix}\right) \le L \left\|\begin{pmatrix}\theta \\ \nu\end{pmatrix} - \begin{pmatrix}\theta' \\ \nu'\end{pmatrix}\right\|_2$$

$$\gamma\left(\begin{pmatrix}\theta \\ \nu\end{pmatrix}, \begin{pmatrix}\theta' \\ \nu'\end{pmatrix}\right) := \sup_{x \in \mathcal{X}}\{\|\text{Var}[\psi_{\theta,\nu}(O_i) - \psi_{\theta',\nu'}(O_i)|X_i = x]\|_F\}$$

**Assumption 4.** (Regularity of $\psi$) The $\psi$-fucntions can be written as $\psi_{\theta,\nu}(O) = \lambda(\theta, \nu; O_i) + \zeta_{\theta,\nu}(g(O_i))$ such that $\lambda$ is Lipschitz-continuous in $\theta, \nu$ and $g : O_i \to \mathbb{R}$ is a univariate summary of $O_i$, and $\zeta_{\theta,\nu} : \mathbb{R} \to \mathbb{R}$ is any family of monotone and bounded functions

[4] This seems to restrict $\theta$ to be semiparametric. I don’t think that is the right interpretation though. $\theta(x)$ can still be an arbitrary function taking values on a the real line.

**Assumption 5.** (Existence of solutions) We assume that, for any weights $\alpha_i$ with $\sum \alpha_i = 1$, the estimating equation returns a minimizer $(\hat{\theta}, \hat{\nu})$ that at least approximately solves the estimating equation: $\|\sum_{i=1}^{n} \alpha_i \psi_{\hat{\theta},\hat{\nu}}(O_i)\|_2 \leq C \max\{\alpha_i\}$ for some constant $C \geq 0$.

**Assumption 6.** (Convexity) The score function $\psi_{\theta,\nu}(O_i)$ is a negative sub-gradient of a convex function, and the expected score $M_{\theta,\nu}(X_i)$ is the negative gradient of a strongly function.

Assumption 3 holds trivially if $\psi$ is Lipschitz in the parameters. Assumption 4 is used to show that a certain empirical process is Donsker. The first 5 assumptions deal with local properties of the estimating equation and can be used to control the behavior of $(\hat{\theta}(x), \hat{\nu}(x))$ in neighborhoods of the population parameter value $(\theta(x), \nu(x))$. The 6th assumption garuntees consistency.

Consistency and Gaussianity results require using some specific settings for the trees from Algorithm 1. In particular, require that all trees are honest and regular in the sense of Wager and Athey (2018), as follows. In order to satisfy the minimum split probability condition below, our implementation relies on the device of Denil, Matheson and De Freitas (2014), whereby the number splitting variables considered at each step of the algorithm is random. Specifically, try $\min\{\max\{\text{Poisson}(m), 1\}, p\}$ variables at each step, where $m > 0$ is a tuning parameter.

**Specification 1.** All trees are symmetric in that their output is invariant to permuting the indices of training examples; make balanced splits in the sense that every split puts at least a fraction $\omega$ of the observations in the parent node into each child, for some $\omega > 0$; and are randomized in such a way that, at every split, the probability that the tree splits on the $j$-th feature. is bounded from below by $\pi > 0$. The forest is honest and built with subsample size satisfying $s/n \to 0$ and $s \to \infty$.

These assumptions hold trivially under some weak assumptions for least squares and quantile regression.

### 1.3.1 A Central Limit Theorem for Generalized Random Forests

Now ready for asymptotic results. Note that regression forests are averages of regression trees grown over sub-samples and were thus be analyzed as $U$-statistics (Hoeffding, 1948). Unlike regression forest predictions, however, the parameter estimates $\hat{\theta}(x)$ from generalized random forests are not averages of estimates made by different trees. Instead, we obtain $\hat{\theta}$ by solving a single weighted moment equation as in eq. 3. So existing proof strategies do not apply in thi setting.

Tackle this problem using method of influence functions as described by Hampel (1974). In particular, we are motivated by the analysis of Newey (1994a). Core idea is to derive a sharp, linearized approzimation to the local estimatior, and then to analyze the linear approximation instead. Let $\rho_i^*(x)$ denote the influence function on the $i$-th observation with respect to the true parameter value, $\theta(x)$

$$\rho_i^*(x) := -\xi^T V(x)^{-1} \psi_{\theta(x),\nu(x)}(O_i)$$

Then, given any set of forest weights $\alpha_i(x)$ used to define the generalized random forest estimate $\hat{\theta}(x)$ by solving (3) define a pseudo-forest

$$\tilde{\theta}^*(x) := \theta(x) + \sum_{i=1}^{n} \alpha_i(x) \rho_i^*(x) \tag{12}$$

used to approximate $\hat{\theta}(x)$. $\tilde{\theta}^*(x)$ is the output of an infeasible regression forest with weights $\alpha_i(x)$ and outcomes $\theta(x) + \rho_i^*(x)$. The upshot is that this is a $U$-statistic, which we know how to analyze. Because $\tilde{\theta}^*(x)$ is a linear function of the pseudo outcomes $\rho_i^*(x)$, it can be written as an average of pseudo-tree predictions $\tilde{\theta}^*(x) = \frac{1}{B}\sum_{b=1}^{B} \tilde{\theta}_{b^*}(x)$ where $\tilde{\theta}_{b^*}(x) = \sum_{i=1}^{n} \alpha_{ib}(x)(\theta(x) + \rho_i^*(x))$. Then, because each individual pseudo-tree prediction $\tilde{\theta}_{b^*}$ is trained on a size $s$ usbsample of the training data, drawn without replacement, $\tilde{\theta}^*(x)$ is an infinite order $U$-statistic whose order corresponds to the subsample size.

- Arguments of Mentch and Hooker (2016) and Wager and Athey (2018) can be used to study the averaged estimator $\tilde{\theta}^*(x)$ using results on U-statistics from Hoeffding (1948) and Efron and Stein

(1981)[5]

Difficulty in this proof stratefy is showing that $\tilde{\theta}^*(x)$ is a good approximation for $\tilde{\theta}(x)$. Following theorem establishes this. This is the only point where $\phi$ being the negative gradient of a convex loss function is used.

**Theorem 1.** *Under Assumptions 1-6, estumatees $\hat{\theta}(x), \hat{\nu}(x)$ converge in probability to $\theta(x), \nu(x)$.*

Seperating the analysis of moment estimators into a local approcumation argument that hinges on consistency and a seperate result that estabilishes consistency is standard; see chapter 5.3 of Van Der Vaart (2000)[6]

The remainder of analysis assumes that trees are grown on subsamples of size $s$ scalig as $s = n^\beta$ for some $\beta_{\min} < \beta < 1$ with

$$\beta_{\min} := 1 - \left(1 + \pi^{-1}\left(\log(\omega^{-1})\right)\right)^{-1} \tag{13}$$

where $\pi$ and $\omega$ are as in Specification 1. Scaling garuntees errors of forests are varaicne-dominated.

**Lemma 1.** *Given Assumptions 1-5 and a forest trained according to Specification 1 with condition 13 holding, suppose that the generalized random forest estimator $\hat{\theta}$ is consistent for $\theta(x)$. Then $\hat{\theta}(x)$ and $\tilde{\theta}^*(x)$ are coupled at the following rate*

$$\sqrt{\frac{n}{s}}\left(\theta^*(x) - \hat{\theta}(x)\right) = \mathcal{O}_P\left(\max\left\{s^{-\frac{\pi \log\left((1-\omega)^{-1}\right)}{2\log\left(\omega^{-1}\right)}}, \left(\frac{s}{n}\right)^{\frac{1}{6}}\right\}\right) \tag{14}$$

*where $s$, $\omega$ and $\pi$ are as in Specification 1.*

Given this coupling result, it now remains to study the asymptotics of $\tilde{\theta}^*(x)$. In doing so, important to know that $\tilde{\theta}^*(x)$ is exactly the output of an infeasible regression forest trained on outcomes $\theta(x) + \rho_i^*(x)$. So can apply results of Wager and Athey (2018) to this object. With this approach, authors show that, given 13m $\tilde{\theta}^*(x)$ and $\hat{\theta}(x)$ are both asymptotically normal. Extending the argument can also so this for nuisance parameters, but noting that since tree is not trained to optimize nuisance, may not work well in finite samples.

**Theorem 2.** *Suppose Assumptions 1-6 hold and a forest is trained according to Specification 1 with trees grown on subsamples of size $s = n^\beta$ satisfying 13. Finally, suppose that $\mathrm{Var}[\rho_i^*(x)|X = x] > 0$. Then, there is a sequence $\sigma_n(x)$ for which $(\hat{\theta}_n(x) - \theta(x))/\sigma_n(x) \to \mathcal{N}(0,1)$ and $\sigma_n^2(x) = \mathrm{polylog}(n/s)^{-1}s/n$, where $\mathrm{polylog}(n/s)$ is a function that is bounded away from 0 and increases at most polynomially with the log-inverse sampling ratio $\log(n/s)$.*

### 1.4 Confidence Intervals via the Delta Method

Theorem 2 can be used for statistical inference about $\theta(x)$. Given a consistent estimator $\hat{\sigma}_n(x)$ for $\sigma_n(x)$, Theorem 2 can be paired with Sltutsky's lemma to verify

$$\lim_{n\to\infty} \mathbb{P}\left[\theta(x) \in \left(\hat{\theta}(x) \pm \Psi^{-1}(1-\alpha/2)\hat{\sigma}_n(x)\right)\right] = \alpha$$

So to build asymptotically valid pointwise confidence intervales, it suffices to derive an estimator for $\sigma_n(x)$. Doing so requires leveraging coupling with the approximate pseudo-forest $\tilde{\theta}^*(x)$. Moreover, from the defini-

[5] The definition of U-statistic from Hoeffding (1948), via Wikipedia. Let $f : \mathbb{R}^r \to \mathbb{R}$ be a real-valued or complex-valued function of $r$ variables. For each $n \geq r$, the associated $U$-statistic $f_n : \mathbb{R}^n \to \mathbb{R}$ is equal to the average over ordered samples $\varphi(1), \dots, \varphi(r)$ or size $r$ of the sample values $f(x_\varphi)$. In otherwords $f_n(x_1, \dots, x_n) = \mathrm{ave} f(x_{\varphi(1),\dots,x_{\varphi(r)}})$. By neccesity, each U-statistic is a symmetric function.

[6] Textbook is *Asymptotic Statistics* and it can be found in the google drive

tion of $\tilde{\theta}^*(x)$, we directly see that

$$\operatorname{Var}\left[\tilde{\theta}^*(x)\right] = \xi^T V(x)^{-1} H_n\left(x; \theta(x), \nu(x)\right)\left(V(x)^{-1}\right)^T \xi \tag{15}$$

where $H_n(x;\theta,\nu) = \operatorname{Var}[\sum_{i=1}^n \alpha_i(x)\psi_{\theta,\nu}(O_i)]$. Authors then propose building confidence intervals via

$$\hat{\sigma}_n^2 := \xi^T \hat{V}_n(x)^{-1} \hat{H}_n(x)(\hat{V}_n(x)^{-1})^T \xi \tag{16}$$

Coming up with consistent estimators of $V(x)$ is well studied and not so complex, according to the authors. Estimating $H$, however, can be difficult since it depends on the true forest score $\Psi(\theta(x),\nu(x)) = \sum_{i=1}^n \alpha_i(x)\psi_{\theta(x),\nu(x)}(O_i)$. To estimate this, they use a variant of the bootstrap of little bags algorithm (noisy bootstrap) proposed by Sexton and Laake (2009). They obtain the first consitency garuntees for this method for any type of forest, including regression forests. Notes about this are breifly given below

### 1.4.1 Consistency of the Bootstrap of Little Bags

## 2 Deep Learning in NPR *Benedikt Bauer and Michael Kohler (AOS, 2019)*

Full paper title is "On Deep Learning As A Remedy for the Curse of Dimensionality in Nonparametric Regression" and can be found via the AoS website here.

### 2.1 Introduction

In regression analysis, a random vector $(X, Y)$ with values in $\mathbb{R}^d \times \mathbb{R}$ satisfying $\mathbf{E}Y^2 < \infty$ is considered, and an estimation of the relationship between $X$ and $Y$ is attempted. Generally the aim is to minimize the MSE or $L_2$r risk. So the construction of a measurable function $m^* : \mathbb{R}^d \to \mathbb{R}$ satisfying

$$m^* = \underset{f:\mathbb{R}^d \to \mathbb{R}}{\arg\min}\, \mathbf{E}\left\{\left|Y - f(X)\right|^2\right\}$$

is of interest. In the following, let $m : \mathbb{R}^d \to \mathbb{R}$, $m(x) = \mathbf{E}\{Y|X = x\}$ denote the "regression function". It is true that for any $f$:

$$\mathbf{E}\left[\left|Y - f(X)\right|^2\right] = \mathbf{E}\left[\left|Y - m(X)\right|^2\right] + \int \left|f(x) - m(x)\right|^2 \mathbf{P}_X(dx)$$

it is the optimal predictor $m^*$. Moreover, a good estimate $f : \mathbb{R}^d \to \mathbb{R}$ (in the $L_2$ risk minimization sense) has to keep the "$L_2$" error small

$$\int \left|f(x) - m(x)\right|^2 \mathbf{P}_X(dx)$$

In applications, the distribution of $(X, Y)$ and $m$ are (typically) unkown, but the statistician does have access to a set of data

$$\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$$

Goal is typically to create estimates of $m$, $m_n$ to minimize the $L_2$ error. In non-parametric regression estimation of the regression function does not reduce to estimation of finitely many parameters. Gyordi et al. (2002) provide a systematic overview of different approaches and nonparametric estimation results.

#### 2.1.1 Rate of Convergence

Well known that one has to restric the class of regression functions one considers to obtain useful results for the rate of convergence. Following definiton of $(p, C)$-smoothness is to that end

**Definition 1.** ($(p, C)$-smooth) Let $p = q + s$ for some $q \in \mathbb{N}_0$ and $0 < s \leq 1$. A function $m : \mathbb{R}^d \to \mathbb{R}$ is called $(p, C)$-smooth if, for every $\alpha = (\alpha, \ldots, \alpha_d) \in \mathbb{N}_0^d$ with $\sum_{j=1}^d \alpha_j = q$, the partial derivatives below exist and satisfy

$$\left|\frac{\partial^q m}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(x) - \frac{\partial^q m}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(z)\right| \leq C\|x - z\|^s$$

for all $x, z \in \mathbb{R}^d$, where $\|\cdot\|$ denotes the Euclidean norm.[a]

[a] This is similar to the Holder condition we went over with Zhipeng

Stone (1982) determined the optimal minimax rate of convergence in nonparametric regresion for $(p, C)$-smooth functions. A sequence of eventually positive numbers $(a_n)_{n\in\mathbb{N}}$ is called a *lower minimax rate of convergence* for the class of distributions $\mathcal{D}$ if

$$\liminf_{n\to\infty} \inf_{m_n} \sup_{(X,Y)\in\mathcal{D}} \frac{\mathbf{E}\int |m_n(x) - m(x)|^2 \mathbf{P}_X(dx)}{a_n} = C_1 > 0$$

Sequence is said to be an *achievable rate of convergence* for the class of distributions $\mathcal{D}$ if

$$\limsup_{n\to\infty} \sup_{(X,Y)\in\mathcal{D}} \frac{\mathbf{E}\int|m_n(x)-m(x)|^2\mathbf{P}_X(dx)}{a_n} = C_2 > 0^1$$

Sequence is called an *optimal minimax rate of convergence* if it both a lower minimiax and achievable rate of convergence. Stone (1982) shows that the optimal rate of convergence for the estimation of a $(p, C)$-smooth regression function is $n^{-\frac{2p}{2p+d}}$

### 2.1.2 Curse of dimensionality

Optimal rate $n^{-\frac{2p}{2p+d}}$ suffers if $d$ is relatively large compared with $p$. Phenomenon is well known and called the curse of dimensionality. Unfortunately, in many applications, the problems are high dimensional and hence very hard to solve. Only way around this is to impose additional assumptions on the regression function to derive better rates of convergence. For example, under additive seperability of the regression function, Stone (1985) shows that the optimal minimax rate of convergence is $n^{-2p/(2p+1)}$.

Paper focuses on applications in connection with complex technical systems, constructed in a modular form. In this case, modeling the outcome of the system as a function of the results of its modular parts seems reasonable, where each modular part computes a function depending only on a few of the components of the high-dimensional input. Modularity can be extremely complex and deep. So, a recursive application of the described relation makes sense and leads to the following assumption of $m$, introduced by Kohler and Kryzak (2017).

**Definition 2.** Let $d \in \mathbb{N}, d^* \in \{1, \dots, d\}$ and $m : \mathbb{R}^d \to \mathbb{R}$. Then:

1. We say that $m$ satisfies a *generalized hierarchical interaction of order $d^*$ and level 0* if there exist $a_1, \dots a_{d^*} \in \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$ such that

$$m(x) = f(a_1^T x, \dots, a_{d^*}^T x) \quad \text{for all } x \in \mathbb{R}^d$$

2. We say that $m$ satisfies a *generalized hierarchical model of order $d^*$ and level $l+1$*, if there exist $K \in \mathbb{N}, g_k : \mathbb{R}^{d^*} \to \mathbb{R}$ for $k = 1, \dots, K$, and $f_{1,k}, \dots, f_{d^*,k} : \mathbb{R} \to \mathbb{R}$ for $k = 1, \dots, K$ such that all $f_{1,k}, \dots, f_{d^*,k}$ satisfy a generalized hierarchical interaction model of order $d^*$ at level $l$ and

$$m(x) = \sum_{k=1}^{K} g_k\left(f_{1,k}(x), \dots, f_{d^*,k}(x)\right) \quad \text{for all } x \in \mathbb{R}^d$$

3. We say that the *generalized hierarchical interaction model* defined above is $(p, C)$-smooth if all functions occuring in its definition are $(p, C)$-smooth.

To better understand the above definition, we consider the additive model from the beggining of this section as an example. Notate $\mathrm{id} : \mathbb{R} \to \mathbb{R}$ for the identity function and $e_i$ for the $i$th unit vector. Can then rewrite the additive model as

$$\sum_{i=1}^{d} m_i(x^{(i)}) = \sum_{i=1}^{d} m_i(\mathrm{id}(e_i^T x)) = \sum_{i=1}^{K} g_i(f_{1,i}(a_i^T x))$$

where $K = d, g_i = m_i, f_{1,i} = \mathrm{id}$ and $a_i = e_i$. This corresponds to the definition of a gneralized hierarchichal interaction model of order 1 and level 1.

### 2.1.3 Neural Networks

Use of neural networks has been most promising approaches in connection with applications related to approxumation and estimation of multivariate functions. Recently, focus is on multilayer neural networks,

[1] Achievable in the sense that it is the minimax rate of convergence for at least one estimator $m_n$

which use many hidden layers and corresponding techniques.

Multilayer feedforward neural networks with a sigmoidal function $\sigma : \mathbb{R} \to [0,1]$ can be defined recursively as follows. A multilayer feedforward neural network with $l$ hidden layers, which has $K_1, \ldots, K_l \in \mathbb{N}$ neurons in the first, second, through $l$-th layer, respectively, and uses the activation function $\sigma$ is a real valued function defined on $\mathbb{R}^d$ of the form

$$f(x) = \sum_{i=1}^{K_l} c_i^{(l)} \cdot f_i^{(l)} + c_0^{(l)}, [2] \tag{1}$$

for some $c_0^{(l)}, \ldots, c_{K_l}^{(l)} \in \mathbb{R}$ and for $f_i^{(l)}$ recursively defined by

$$f_i^{(r)}(x) = \sigma\left(\sum_{j=1}^{K_{r-1}} c_{i,j}^{(r-1)} \cdot f_j^{(r-1)}(x) + c_{i,0}^{(r-1)}\right), [3] \tag{2}$$

for some $c_{i,0}^{(r-1)}, \ldots, c_{i,K_{r-1}}^{(r-1)} \in \mathbb{R}$ and $r = 1, \ldots, l$ and

$$f_i^{(1)}(x) = \sigma\left(\sum_{j=1}^{d} c_{i,j}^{(0)} \cdot x^{(j)} + c_{i,0}^{(0)}\right), [4] \tag{3}$$

for some $c_{i,0}^{(0)}, \ldots, c_{i,d}^{(0)} \in \mathbb{R}$. Neural network estimates often use an activation function $\sigma : \mathbb{R} \to [0,1]$ that is nondecreasing and satisfies

$$\lim_{z \to -\infty} \sigma(z) = 0 \quad \text{and} \quad \lim_{z \to \infty} \sigma(z) = 1$$

for example, the so-called sigmoidal or logistic squasher

$$\sigma(z) = \frac{1}{1 + \exp(-z)}, \forall z \in \mathbb{R}$$

Most existing theoretical results concerining neural networks consider neural networks using only one hidden layer, that is functions of the form

$$f(x) = \sum_{j=1}^{K} c_j \cdot \sigma\left(\sum_{k=1}^{d} c_{j,k} \cdot x^{(k)} + c_{j,0}\right) + c_0 \tag{4}$$

Consistency of neural network regression estimates is studied by Meilnichzuk and Tyrcha (1993) and Lugosi and Zeger (1995). The rate of convergence has been analyzed by Barron (1991, 1993, 1993), McCaffery and Gallant (1994) and Kohler and Krzyzak (2005, 2017). For the $L_2$ error of a single hidden layer neural network, Barron (1994) proves a dimensionless rate of $n^{-1/2}$, provided the Fourier transform has a finite first moment. McCaffery and Gallant (1994) show a rate of $n^{-\frac{2p}{2p+d+5}+\epsilon}$ for the $L_2$ error of a suitably defined single hidden layer neural network estimate for $(p, C)$-smooth functions, but their study was restricted to the use of a certain cosine squasher as the activation function.

Kohler and Krzyzak (2017) extends convergence results to $(p, C)$-smooth generalized hierarchical interaction models of the order $d^*$. It is shown that for such models suitable defined multilayer neural networks achieve the rate of convergence $n^{-2p/(2p+d^*)}$ in case $p \leq 1$. Nevertheless this result cannot generate extremely good rates of convergenve because, even in case of $p = 1$ and $d^* = 5$, it leads to $n^{-2/7}$.

[2] We can think about this as a linear regression of the outcome against equations from the final layer

[3] Apply a sigmoid function to a linear combination of the outputs from the prior round. To clarify some notation: $f_j^{(r-1)}$ is the output from the $j$-th neuron in the $(r-1)$-th layer, $c_{i,j}^{(r-1)}$ is the weight given at neuron $i$ in the $r$-th layer to the output of the $j$-th neuron in the $(r-1)$-th layer. There are $K_r$ neurons at each layer $r$, so that each neuron in layer $r$ has to "pick" appropriate weights for all $K_{r-1}$ outputs of neurons in layer $(r-1)$.

[4] $x^{(j)}$ is the $j$-th "feature", "variable", "column", what have you.

Given the succeful application of multilayer feedforward neural networks, the current focus in the theoretical analysis of approximation properties of neural networks is also on a a possible theoretical advantage of multilayer feedforward neural networks in contrast to neural networks with only one hidden layer.

### 2.1.4 Main Results

This article analyzes the rate of convergence of suitable multilayer neural network regression estimates when the regression function satisfies a $(p, C)$-smooth generalized hierarchical interaction model of order $d^*$ and given level $l$. Unlike Kohler and Kryzak (2005, 2017) also allow the case $p > 1$, this leads to far better rates of convergence. Define sets of multilayer feedforward nueral netwroks that correspond to such a generalized a generalized hierarchical interaction model and define our regression estimates based on this class of neural networks. Main finding is that the $L_2$ errors of these least squares neural network regression estimates achieve the rate of convergence

$$n^{-\frac{2p}{2p+d^*}}$$

up to some logarithmic factor which does not depend on $d$. Similar rates have been obrained in the literature but with much more stringent assumptions on the functional class the regression function belongs too. So this article considerably generalizes the previos results in this regard.

After the original versin of this paper, a relating arXiv article was uploaded by Schmidt-Heiber (2017). Therein a similar result is proven using a particular unbounded activation function in the neural networks Available Here

### 2.1.5 Notation

Let $A \subset \mathbb{R}^d$ and $\mathcal{F}$ be a set of functions $f : \mathbb{R}^d \to \mathbb{R}$ and let $\epsilon > 0$. A finite collection $f_1, \ldots, f_N$ is called an $\epsilon$-$\|\cdot\|_{\infty,A}$-cover of $\mathcal{F}$ if for any $f \in \mathcal{F}$ there exists $i \in \{1, \ldots, N\}$ such that

$$\|f - f_i\|_{\infty,A} = \sup_{x \in A} |f(x) - f_i(x)| < \epsilon$$

The $\epsilon$-$\|\cdot\|_{\infty,A}$-covering number of $\mathcal{F}$ is the size $N$ of the smallest $\epsilon$-$\|\cdot\|_{\infty,A}$-cover of $\mathcal{F}$ and is denoted by $\mathcal{N}(\epsilon, \mathcal{F}, \epsilon\text{-}\|\cdot\|_{\infty,A})$[5].

## 2.2 Nonparametric Regression Estimation by Multilayer Feedforward Neural Networks

Motivated by the generalized hierarchical interaction models, define spaces of hierarchical neural networks with parameters $K, M^*, D^*, d$ and level $l$ as follows. Parameter $M^*$ is introduced for technical reasons and originates from the composition of several smaller networks in the later proof of approximation results. $M^*$ controls the accuracy of the approximation and the ideal value will depend on certain properties of the estimated function. For $M^* \in \mathbb{N}, d \in \mathbb{N}, d^* \in [d]$ and $\alpha > 0$, denote the set of all functions $f : \mathbb{R}^d \to \mathbb{R}$ that satisfy

$$f(x) = \sum_{i=1}^{M^*} \mu_i \cdot \sigma\left(\sum_{j=1}^{4d^*} \lambda_{i,j} \cdot \sigma\left(\sum_{v=1}^{d} \theta_{i,j,v} \cdot x^{(v)} + \theta_{i,j,0}\right) + \lambda_{i,0}\right) + \mu_0$$

for $x \in \mathbb{R}^d$ and some $\mu_i, \lambda_{i,j}, \theta_{i,j,v} \in \mathbb{R}$ where

$$|\mu_i| \leq \alpha, \ |\lambda_{i,j}| \leq \alpha, \ |\theta_{i,j,v}| \leq \alpha$$

[5] These are covered in Van derVaart and are important in the Donsker Theorems.

for all $i \in \{0, 1, \ldots, M^*\}, j \in \{0, \ldots, 4d^*\}, v \in 0, \ldots, d$ by $\mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}$ In the first and second hidden layer, we use $4 \cdot d^* \cdot M^*$ and $M^*$ neurons respectively. However, the neural network has only

$$\begin{aligned} W(\mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}) &= M^* + 1 + M^* \cdot (4d^* + 1) + M^* \cdot 4d^* \cdot (d+1) \\ &= M^* \cdot (4d^* \cdot (d+2) + 2) + 1 \end{aligned} \tag{5}$$

weights because the first and second hidden layer of the neural network are not fully connected. Instead, each neuron in te second hidden layer is connected with $4d^*$ neurons in the first hidden layer, and this is done in such a way that each neuron in the first hidden layer is connected with exactly one neural network in the second hidden layer. This is illustrated below in Figure 1.

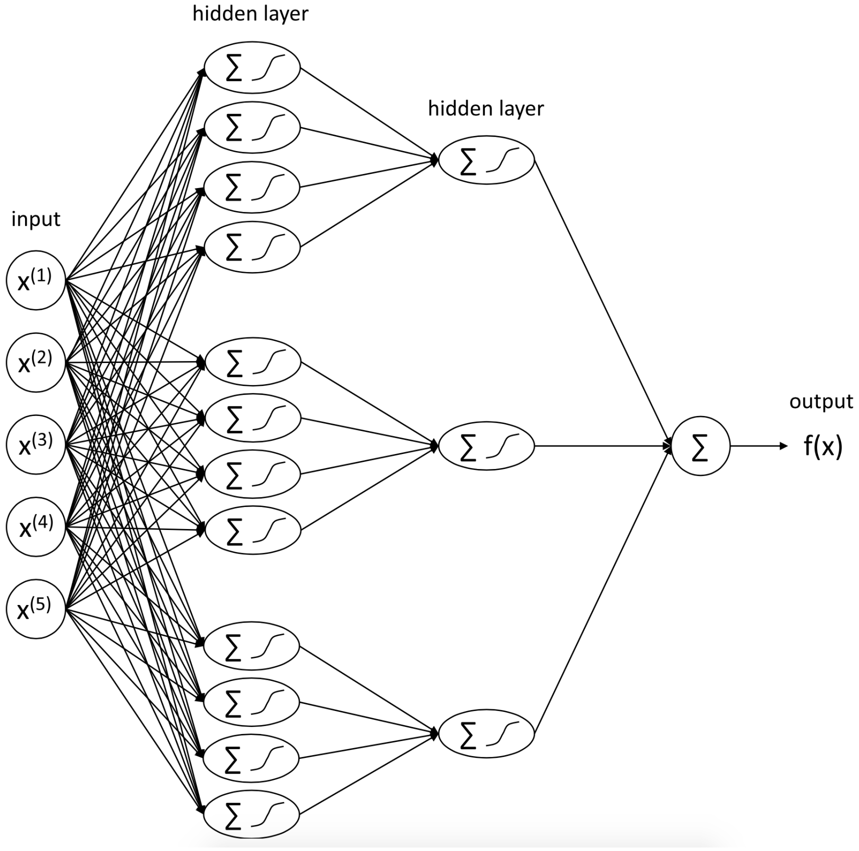


Figure 1: A not completely connected neural network $f : \mathbb{R}^5 \to \mathbb{R}$ from $\mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}$ with the structure $f(x) = \sum_{i=1}^{3} \mu_i \cdot \sigma(\sum_{j=1}^{4} \lambda_{i,j} \cdot \sigma(\sum_{v=1}^{5} \theta_{i,j,v} \cdot x^{(v)}))$ (all weights with an index including zero neglected for a clear illustration). [Lifted from the paper]

For $l = 0$, we define our space of hierarchical neural networks by

$$\mathcal{H}^{(0)} = \mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}$$

For $l > 0$ we define recursively

$$\mathcal{H}^{(l)} = \left\{ h : \mathbb{R}^d \to \mathbb{R} : h(x) = \sum_{k=1}^{K} g_k(f_{1,k}(x), \ldots, f_{d^*,k}(x)) \text{ for some } g_k \in \mathcal{H}^{(0)} \text{ and } f_{j,k} \in \mathcal{H}^{(l-1)} \right\} \tag{6}$$

The class $\mathcal{H}^{(0)}$ is a set of neural networks with two hidden layers and a number of weights given by (5). From this, one can recursively conclude that for $l > 0$, the class $\mathcal{H}^{(l)}$ is a set of neural networks with $2 \cdot l + 2$ hidden layers. This is illustrated below in Figure 2 Furthermore, let $N(\mathcal{H}^{(l)})$ denote the number of linked two-layered networks from $\mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}$ that define the functions from $\mathcal{H}^{(l)}$. Then the following recursion holds:

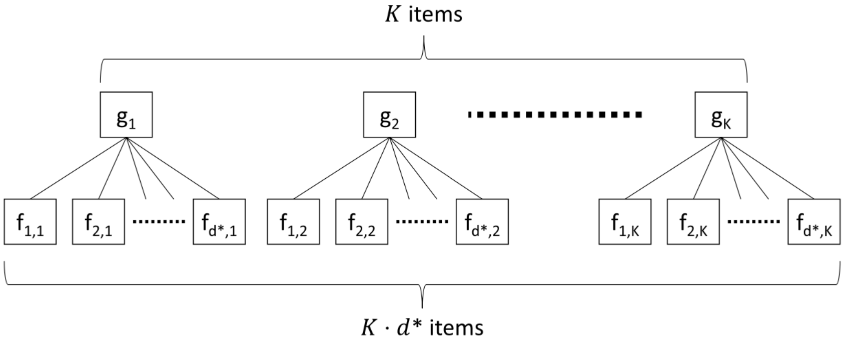


Figure 2: Illustration of the components of a function from $\mathcal{H}^{(l)}$ [Lifted from the paper]

$$\begin{aligned} N(\mathcal{H}^{(0)}) &= 1, \\ N(\mathcal{H}^{(l)}) &= K + K \cdot d^* \cdot N(\mathcal{H}^{(l-1)}), \qquad l \in \mathbb{N} \end{aligned}$$

which can be retraced following Figure 2. Above functions $g_1, \ldots, g_K$ correspond to $K$ networks from $\mathcal{H}^{(0)} = \mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}$ and the $K \cdot d^*$ inner functions $f_{1,1}, \ldots, f_{d^*,K}$ originate from $\mathcal{H}^{(l-1)}$, which leads to $K \cdot d^* \cdot N(\mathcal{H}^{(l-1)})$ additional networks.

Recursuve consideration yields

$$N(\mathcal{H}^{(l)}) = \sum_{t=1}^{l} {d^*}^{t-1} \cdot K^t + (d^* \cdot K)^l \tag{7}$$

Consequently, a function from $\mathcal{H}^{(l)}$ has at most

$$N(\mathcal{H}^{(l)}) \cdot W(\mathcal{F}^{(\text{neural networks})}_{M^*,d^*,d,\alpha}) \tag{8}$$

variable weights. Although this number of weights is exponential in the number of layers $l$, it can be controlled because a typical example of the technical systems which motivated Definition 2 has only a moderate finite $l$. As explained in the definition, all typical assumptions for the regression function in the literature also correspond to a small $l$.

Define $\tilde{m}_n$ as the least squares estimate

$$\tilde{m}_n(\cdot) = \arg \min_{h \in \mathcal{H}^{(l)}} \frac{1}{n} \sum_{i=1}^{n} \left| Y_i - h(X_i) \right|^2 \tag{9}$$

For the result this needs to be truncated. Define the truncation operator $T_\beta$ with level $\beta > 0$ as

$$T_\beta u = \begin{cases} u & \text{if } |u| \leq \beta \\ \beta \cdot \operatorname{sign}(u) & \text{otherwise} \end{cases}$$

Results requre a few additional properties on activation function, which are satisfied by many common activation functions (like the sigmoidal squasher) and they can be checked with arbitrary $N \in \mathbb{N}_0$. Summarized in the next definition

**Definition 3.** A nondecresaing and Lipschitz continuous function $\sigma : \mathbb{R} \to [0,1]$ is called *$N$-admissable* if the following conditions hold

1. The function $\sigma$ is at least $N+1$ times differentiable with bounded derivatives.
2. A point $t_\sigma \in \mathbb{R}$ exists where all derivatives up to the order $N$ of $\sigma$ are different from zero.
3. If $y > 0$, the relation $|\sigma(y) - 1| \leq \frac{1}{y}$ holds. If $y < 0$, the relation $|\sigma(y)| \leq \frac{1}{|y|}$ holds.

**Theorem 1** (Main Result)**.** *Let $\{(X_i, Y_i)\}_{i=1}^n$ be independent and identically distributed random variables in $\mathbb{R}^d \times \mathbb{R}$ such that $\operatorname{supp}(X)$ is bounded and*

$$\mathbf{E}\exp(c_1 \cdot Y^2) < \infty, \tag{10}$$ [a]

*for some constant $c_1 > 0$. Let $m$ be the corresponding regression function, which satisfies a $(p, C)$-smooth generalized hierarchical interaction model of order $d^*$ and finite level $l$ with $p = q + s$ for some $q \in \mathbb{N}_0$ and $s \in (0,1]$. Let $N \in \mathbb{N}_0$ with $N \geq q$. Furthermore, assume that in Definition 2.b all partial derivatives of order less than or equal to $q$ of the functions $g_k, f_{j,k}$ are bounded. That is, assume that easch function $f$ satisfies*

$$\max_{\substack{j_1,\ldots,j_d \in \{0,1,\ldots,q\}, \\ j_1+\cdots+j_d \leq q}} \left\| \frac{\partial^{j_1+\cdots+j_d} f}{\partial^{j_1} x^{(1)} \ldots \partial^{j_d} x^{(d)}} \right\| \leq c_2 \tag{11}$$

*and let all functions $g_k$ be Lipschitz continuous with Lipschitz constant $L > 0$ [which follows from (11 if $q > 0$]. Let $\mathcal{H}^{(l)}$ be defined as in (6) with $K, d, d^*$ as in the definition of $m$, $M^* = \left\lceil c_{56} \cdot n^{d^*} 2p + d^* \right\rceil$. $\alpha = n^{c_{57}}$ for suffeciently large constants $c_{56}, c_{57} > 0$, and using an $N$-admissible $\sigma : \mathbb{R} \to [0,1]$ according to Definition 3.Let $\tilde{m}_n$ be the least squares estimate defined by (9) and define $m_n = T_{c_3 \log n} \tilde{m}_n$. Then*

$$\mathbf{E} \int \left| m_n(x) - m(x) \right|^2 \mathbf{P}_X(dx) \leq c_4 \cdot \log^3(n) \cdot n^{-\frac{2p}{2p+d^*}}$$

*holds for suffeciently large $n$.*

[a] This is basically saying that the moment generating function of $Y^2$ exists in some neighborhood around 0

The authors include the following remarks on this main result

1. For $p \geq 1$ and $C \geq 1$, the class of $(p, C)$-smooth generalized hierarchical interaction models of order $d^*$ satisfying the assumptions of the theorem contains all $(p, C)$-smooth functions, which depend on at most $d^*$ of its input components (because all functions in Def 2 can be chosen as projections). So, the rate of convergence in Theorem 1 is optimal up to some logarithmic factor, according to Stone (1982).

2. Some parameters of the estimate $m_n$, like $l, K$, or $d^*$ can be uknown in practice. They then would have to be chosen in a data dependent way. This has been studied in the literature apparently.

3. Equation (10) in above theorem prevents heavy tails and ensure that the distribution of $Y$ is suffeciently concentrated in order to allow good estimates.

**Corollary 1.** *Suppose $\{(X_i, Y_i)\}_{i=1}^n$ is an i.i.d sample with values in $\mathbb{R}^d \times \mathbb{R}$ such that the support of $X$ is bounded and $\mathbf{E}\exp(c_1 \cdot Y^2) < \infty$ for some constant $c_1 > 0$. Suppose the corresponding regression function $m$ satisfies a $(2, C)$-smooth generalized hierarchical interaction model of order 2 and finite level 0. Further assume that in Definition 2.b all partial derivatives of order $\leq 1$ of $g_k, f_{j,k}$ are bounded. Take $M^* = \left\lceil c_{56} n^{\frac{1}{3}} \right\rceil$. Use $\sigma(z) = \frac{1}{1+\exp(-z)}$ and $\tilde{m}_n$ and $m_n$ as defined in Theorem 1. Then*

$$\mathbf{E} \int \left| m_n(x) - m(x) \right|^2 \mathbf{P}_X(dx) \leq c_4 \cdot \log^3(n) \cdot n^{-\frac{2}{3}},$$ [a]

*holds for sufficiently large $n$.*

[a] Stringent conditions, but that is a wicked rate of convergence

*Proof.* Using notation from Theorem 1, can choose $N = 1 = 1$. The sigmoidal squasher $\sigma$ is 1-admissible. Then the application of Theorem 1 implies the corollary. □

### 2.3 Application to Simulated Data

Section compares the neural net to an adaptive $k$-nearest neighbors appraoch as interpolation with radial basis function (*RBF*). The paramaters $l, K, d^*, M^*$ of the neural network estimate (*neural-x*) defined in Theorem 1. To solve the least squares problem in (9). To solve the least squares problem use the quasi-Newton method of the function *fminunc* in *MATLAB* to approximate a solution.

Also compare this neural network estimate, which is characterized by the data-dependent choice of its structure and not completely connected neurons, to more ordinary fully connected neural networks with predefined numbers of layers but adaptively chosen numbers of neurons per layer.

Estimate outpreforms the other approaches in the three typical examples for generalized hierarchical interaction models. In these cases, the relative improvement of the estimate is larger with a larger sample size, which is an indicator of a better rate of convergence.

In some more extreme cases, this paper's approach is not always the best, though it still preforms will in some situations. In any case though, the results from simulation are promising.

### 2.4 Proofs

Won't be covered in notes, but the proofs are given in section four of the paper and would be a good idea to examine.

## 3 NPR Using Deep Neural Networks *Johannes Schmidt-Hieber (ArXiv, 2017)*

Full paper title is "Nonparametric regression using deep beural networks with ReLU activation function." Paper appeared on ArXiv in 2019 and (I believe) is due to appear in Annals at some point. It can be found here.

### 3.1 Introduction

In nonparametric regression model with random covariates in unit hypercube, observe $n$ i.i.d vectors $\mathbf{X}_i \in [0,1]^d$ and $n$ responses $Y_i \in \mathbb{R}$ from the model $\mathbf{X}$

$$Y_i = f_0(\mathbf{X}_i) + \epsilon_i \tag{1}$$

The noise variables $\epsilon_i$ are assumed to be *i.i.d* standard normal and independent of $\mathbf{X}_i$. Statistical problem is to recover the unkown function $f_0 : [0,1]^d \to \mathbb{R}$ from the sample $(\mathbf{X}, Y_i)_i$. Various methods exist that allow one to estimate the regression function nonparametrically, including kernel regression, smoothing, series estimators/wavelets, and splines. This paper considers fitting a multilayer feedforward artificial neural network to the data. Shown that estimator achieves nearly optimal convergence rates under various constraints on the regression function.

Deep neural networks have been used in practice from some time, but there is not much mathematrical understanding. Problem is that fitting a neural network to data is highly nonlinear in the parameters. Moreover, the function class is non-convex and various regularization methods are combined in practice.

Article inspired by the idea to build a statistical theory that provides some understanding of these procedures. Method is too complex to be theoretically tractible, so some selection of important characteristics must be done in analysis.

To fit a neural network, an activation function $\sigma : \mathbb{R} \to \mathbb{R}$ needs to be chosen. Traditionally, sigmoidal activation functions were employed (as in Secition 2). For deep neural networks, however, there is a clear gain to using the non-sigmoidal rectifier linear unit (ReLU)

$$\sigma(x) = \max(x, 0) = (x)_+$$

In practice, ReLU outperforms other activation functions with regards to performance and computational cost. Statistical analysis for ReLU activation function is quite different from earlier approaches. Viewed as a nonparametric method, ReLU networks have some suprising properties. Deep networks with ReLU activation produce functions that are piecewise linear in the input. Nonparametric methods based on piecewise linear approacimations are typically not able to capture higher order smoothness in the signal and are rate-optimal only up to smoothness index two. Paper shows that ReLU combined with deep network architecture achieves near minimax rates for arbitrary smoothness of the regression function.

Number of hidden layers has been growing, results support this. Further, generally contain many more network parameters than sample size. Paper accounts for this by assuming number of potential nework parameters is much larger than the sample size. For noisy data generated from the nonparameteric regression model, overfitting leads to generalization errors and incorporating regularization becomes esential.

Existing statistical theoery often requires that the size of the network parameteres tends to infinity as the sample size increases. In practice, estimated network weights are, however, rather samll. Paper incorporates this into theory, procing it is suffecient to consider neural networks with all network parameters bounded in absolute value by one.

Still, NPR using deep neural nets has to get around the curse of dimensionality. Paper gets around this by imposing the generalized hierarchical model structure.

### 3.2 Mathematical Definition of Multilayer Neural Networks

**Neural Network** Fitting a nerual network requires the choice of an activatino function $\sigma : \mathbb{R} \to \mathbb{R}$ and the network architecture. Paper studies the ReLU activation funtion

$$\sigma(x) = \max(x, 0)$$

For $\mathbf{v} = (v_1, \ldots, v_r)$ define the shifted activation function $\sigma_{\mathbf{v}} : \mathbb{R}^r \to \mathbb{R}^r$ as

$$\sigma_{\mathbf{v}} \begin{pmatrix} y_1 \\ \vdots \\ y_r \end{pmatrix} = \begin{pmatrix} \sigma(y_1 - v_1) \\ \vdots \\ \sigma(y_r - v_r) \end{pmatrix}$$

The network architecture $(L, \boldsymbol{p})$ consists of a positive integer $L$ called the number of hidden layers or *depth* and a *width vector* $\boldsymbol{p} \in \mathbb{N}^{L+2}$. So a neural network with network architecture $(L, \boldsymbol{p})$ is then any function of the form

$$f : \mathbb{R}^{p_0} \to \mathbb{R}^{P_{L+1}}, \quad \boldsymbol{x} \mapsto f(\boldsymbol{x}) = W_l \sigma_{\boldsymbol{v}_L} W_{L-1} \sigma_{\boldsymbol{v}_{l-1}} \cdots W_1 \sigma_{\boldsymbol{v}_1} W_0 \boldsymbol{x} \tag{2}$$

where $W_i$ is a $p_i \times p_{i+1}$ weight matrix and $\boldsymbol{v}_i \in \mathbb{R}^{p_i}$ is a shift vector. Network functions are therefore built by alternating matrix-vector multiplications with the action of the non-linear activation functions $\sigma$. In (2) it is also possible to omit the shift vectors by considering the input $(\boldsymbol{x}, 1)$ and enlarging the weight matrices by one row and one column with appropriate entries.

In the compsci literature, neural networks are more commonly introduced via their representation as directed acyclic graphs, like in a figure above in section 2.

**Mathematical Modeling of Deep Network Characteristics** Given a network function $f(\boldsymbol{x})$ as defined in (2), the network parameters are the entries of the matricies $(W_j)_{j=0,\ldots,L}$ and the vectors $(\boldsymbol{v}_j)_{j=0,\ldots,L}$. These parameters need to be estimated/learned from the data.

Aim of this article is to consider a framework that incorporates essential features of modern deep network architectures. Allow for large depth $L$ and large number of potential network parameter. Thus, consider high dimensional settings with more parameters that training data.

Another characteristic of trained networks is that the size of the learned network parameters is typically not very large. Common network initialize the weight matrices $W_j$ by a nearly orthogonal random matrix if two succesive layers have the same width. IN practice, the trained network weights are typically not far from the initialized weights. In an orthogonal matrix, all entries are bounded in absolute value by one, this explains that also the trained network weights are not large.

Existing theory requires that the size of the network parameters tends to infinity. If large parameters are allowed, one can easily approximate step functions by ReLU networks. To be more in line with what is observed in practice, consider networks with all parameters bounded by one. Constraint can easily be build into the deep learning algorithm by projecting the network parameters in each iteration onto the interval [-1,1].

If $\|W_i\|_\infty$ denotes the sup-norm of $W_j$, the space of network functions with given network architecture and network parameters bounded by one is

$$\mathcal{F}(L, \boldsymbol{p}) := \left\{ f \text{ of the form (2) } : \max_{j=0,\ldots,L} \|W_j\|_\infty \vee |\boldsymbol{v}_k|_\infty \leq 1 \right\} \tag{3}$$

with the coefficeint that $\boldsymbol{v}_0$ is a vector of coefficients all equal to zero.

In deep learning, sparsity of the neural network is enforced through regularization or specific forms of networks. Dropout, for instance, randomly sets units to 0 and has the effect that each unit will be active only for a small fraction of the data. In the notation of this paper, this means that each of the vectors $\sigma_{\boldsymbol{v}_k} W_{k=1} \cdots W_1 \sigma_{\boldsymbol{v}_1} W_0 \boldsymbol{x}$, $k = 1, \ldots, L$ is zero over a large range of the input space $x \in [0, 1]^d$. Convolutional neural networks filter the input over local neighborhoods. Rewritten in the form (2), this essentially means that the $W_i$ are banded Toeplitz matrices[1]. All network parameters corresponding to higher off-diagonal entries are thus set to zero.

This paper models sparsity by asusming that there are only a few non-zero / active network parameters. If $\|W_j\|_0$ denotes the number of non-zero entries, then the $s$-sparse networks are given by

$$\begin{aligned}\mathcal{F}(L, \boldsymbol{p}, s) &:= \mathcal{F}(L, \boldsymbol{p}, s, F) \\ &:= \left\{ f \in \mathcal{F}(L, \boldsymbol{p}) : \sum_{j=0}^{L} \|W_j\|_0 + |\boldsymbol{v}_j|_0 \leq s, \|f\|_\infty \leq F \right\}\end{aligned} \tag{4}$$

The upper bound on the uniform/sup norm of $f$ is most of the time not needed and omitted in the notation. Consider cases where the number of network parameters $s$ is small compared to the total number of parameters in the network.

To estimate the parameters of the model, it is common to apply variations of stochastic gradient descent combined with other techniques such as dropout to the loss induced by the log-likelihood. For nonparametric regression with normal errors, this coincides with the least-squares loss. The common objective of all reconstruction methods is to find networds $f$ with a small empirical risk $\frac{1}{n}\sum_{i=1}^{n}(Y_i - f(\mathbf{X}_i))^2$. For any estimator $\hat{f}_n$ that returns a network in the class $\mathcal{F}(L, \boldsymbol{p}, s, F)$ define the correspoinding quantity

$$\Delta_n(\hat{f}_n, f_0) := \mathrm{E}_{f_0}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{f}_n(\mathbf{X})_i)^2 - \inf_{f \in \mathcal{F}(L, \boldsymbol{p}, s, F)} \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(\mathbf{X})_i)^2\right] \tag{5}$$

The sequence $\Delta_n(\hat{f}_n, f_0)$ measures the difference between the expected empirical risk of $\hat{f}_n$ and the global minimum over all networks in the class. The subscript $f_0$ indicates that the expectation is taken with respect to the sample generated from the nonparametric regression model with regression function $f_0$. In general $\Delta_n(\hat{f}_n, g_0) \geq 0$ and $\Delta_n(\hat{f}_n, f_0) = 0$ if $\hat{f}_n$ is an empirical risk minimizer. Note here this is just measuring the "distance" between the estimation technique and the global minimumum.

To evaluate the statistical performance of an estimator $\hat{f}_n$, derive bounds for the prediction error

$$R(\hat{f}_n, f_0) := \mathrm{E}_{f_0}\left[\left(\hat{f}_n(\mathbf{X}) - f_0(\mathbf{X})\right)^2\right]$$

The term $\Delta_n(\hat{f}_n, f_0)$ can be related via empirical process theory to a constant times $(R(\hat{f}_n, f_0) - R(\hat{f}_n^{\mathrm{ERM}}, f_0))$ plus a remainder, where $\hat{f}_n^{\mathrm{ERM}}$ being an empirical risk minimizer. So $\Delta_n(\hat{f}_n, f_0)$ in $n$ for commonly employed methods such as stochastic gradient descent is an interesting problem in its own. Only skech a possible proof strategy here:

1. Because of potentially local minima and saddle poins, gradient descent bethods only have a small chance to reach the global minimum w/o getting stuck in a local minimum first
2. Making a link to sperical spin glasses, paper cited provides a heuristic suggesting that the loss of any local minima lies in a band that is lower bounded by the loss of the global minimum
3. Width of the band depends on the width of the network, if the heuristic argument can be made rigorous then the width of the band provides an upper bound for $\Delta_n(\hat{f}_n, f_0)$ for all methods that converge to a local minumum

[1] A Toeplitz matrix is one where each descending diagonal from left to right is constant. For example

$$\begin{pmatrix} a & b & c \\ d & a & b \\ e & d & a \end{pmatrix}$$

is a Toeplitz matrix. An $l$ banded Toeplitz matrix is one such that only the middle $l$ diagonals of the matrix are non-zero. For example, a traditional diagonal matrix is a 1 banded Toeplitz matrix

This would allow study of deep learning without an explicit analysis of the algorithm.

### 3.3 Main Results

Theoretical performance of neural networks dpends on the underlying function class. Classical appraoch in nonparametric statistics is to assume that the regression function is $\beta$-smooth. The minimax estimation rate for the prediction error is then

$$n^{-2\beta/(2\beta+d)}$$

Since the input dimension $d$ in neural network applications is very large, these rates are extremely slow. The huge sample sizes often encountered in these applications are by far not sufficient to compensate the slow rates. With this in mind, consider a function class that is natural for neural nrwtorks and exhibits some low-dimensional structure that leads to input dimension free exponents in the esimation rates.

Assume that the regression function $f_0$ is a composition of several functions, that is,

$$f_0 = g_q \circ g_{q-1} \circ \ldots \circ g_1 \circ g_0 \tag{6}$$

with $g_i : [a_i, b_i]^{d_i} \to [a_{i+1}, b_{i+1}]^{d_{i+1}}$. Denote by $g_i = (g_{ij})^T_{j=1,\ldots,d_{i+1}}$the components of $g_i$ and let $t_i$ be the maximal number of variables on which each of the $g_{ij}$ depends on. Thus, without loss of generality, each $g_{ij}$ is a $t_i$ variate function. As an example, consider the function $f_0(x_1, x_2, x_3) = g_{11}(g_{01}(x_3), g_{02}(x_2))$ for which $d_0 = 3$, $t_0 = 1$, $d_1 = t_1 = 2$, and $d_2 = 1$.

Always must have $t_i \leq d_i$, and for certain constraints, such as in additive models, $t_i$ mught be much smaller than $d_i$. The single components $g_0, \ldots, f_1$ and the pairs $(\beta_i, t_i)$ are clearly not identifiable. As we are only interested in estimation of $f_0$, this causes no problems. Among all possible representations, one should pick the one that leads to the fastest estimation rate.

In the $d$-variate regression model (1), $f_0 : [0,1]^d \to \mathbb{R}$ and thus $d_0 = d$, $a_0 = 0$, $b_0 = 1$, and $d_{q+1} = 1$. One should keep in mind that (6) is an assumption on the regression function that can be made independently of whether neural networks are used to fit the data or not. In particular, the number of layers $L$ in the network need not be the same as $q$.

Conceivable that for many of the problems for which neural networks perform well, a hidden hierarchical input-output relationship of the form (6) is present with small values $t_i$. Slightly more specific function spaces, which alternate between summations and compositions of functions have been considered (the paper in Section 2) is an example of one of these).

A function has Hölder smoothness index $\beta$ if all partial derivatives up to order $\lfloor \beta \rfloor$ exist and are bounded, and the partial derivatives of order $\lfloor \beta \rfloor$ are $\beta - \lfloor \beta \rfloor$ Hölder. The ball of $\beta$-Hölder functions with raduis $K$ is then defined as

$$\mathcal{C}_r^\beta(D, K) = \left\{ f : D \subset \mathbb{R}^r \to \mathbb{R} : \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|<\beta|} \|\partial^{\boldsymbol{\alpha}} f\|_\infty + \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|=\lfloor\beta\rfloor} \sup_{\substack{\boldsymbol{x},\boldsymbol{y}\in D \\ \boldsymbol{x}\neq\boldsymbol{y}}} \right\}$$

Assume that each of the functions $g_{ij}$ has Hölder smoothness $\beta_i$. Since $g_{ij}$ is also $t_i$-variate, $g_{ij} \in \mathcal{C}_{t_i}^{\beta_i}\left([a_i, b_i]^{t_i}, K_i\right)$ and the underlying function space becomes

$$\begin{aligned} \mathcal{G}(q, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta}, K) := \Big\{ f = g_q \circ \ldots \circ g_0 : g_i = (g_{ij})_j : [a_i, b_i]^{d_i} \to [a_{i+1}, b_{i+1}]^{d_{i+1}}, \\ g_{ij} \in \mathcal{C}_{t_i}^{\beta_i}\left([a_i, b_i]^{t_i}, LK\right), \text{ for some } |a_i|, |b_i| \leq K \Big\} \end{aligned}$$

with $\boldsymbol{d} := (d_0, \ldots, d_{q+1})$, $\boldsymbol{t} := (t_0, \ldots, t_q)$, $\boldsymbol{\beta} := (\beta_0, \ldots, \beta_q)$.

For estimation rates in the nonparametric regression model, the crucial quantity is the smoothness of $f$. Imposing smoothness on the functions $g_i$, one must be find the induced smoothness of $f$, for comparasion on the minimax rates, etc. If, for instance, $q = 1, \beta_0, \beta_1 \leq 1$, and $d_0 = d_1 = t_0 = t_1 = 1$ and $f$ has smoothness $\beta_0, \beta_1$, then one should be able to achieve at lest the convergence rate $n^{-2\beta_0\beta_1/(2\beta_0\beta_1+1)}$

For $\beta_1 > 1$, this rate changes. Below will show that the convergence of the network estimator is described by the effective smoothness indices

$$\beta_i^* := \beta_i \prod_{\ell=i+1}^{q} (\beta_\ell \wedge 1)$$

via the rate

$$\phi_i := \max_{i=0,\dots,q} n^{-\frac{2\beta_i^*}{2\beta_i^*+t_i}} \tag{7}$$

Recalling the definition of $\Delta_n(\hat{f}_n, f_0)$ in (5).

**Theorem 1** (Main Result)**.** *Consider the d-variate nonparametric model in (1) for composite regression function (6) in the class $\mathcal{G}(q, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta}, K)$. Let $\hat{f}_n$ be an estimator taking values in the network class $\mathcal{F}(L, (p_i)_{i=0,\dots,L+1}, s, F)$ satisfying*

1. $F \geq \max(K, 1)$
2. $\sum_{i=0}^{q} \log_2(4t_i \vee 4\beta_i) \log_2 n \leq L \lesssim n\phi_n$[a]
3. $n\phi_n \lesssim \min_{i=1,\dots,L} p_i$
4. $s \asymp n\phi_n \log n$

*Then there exist constants $C, C'$ only depending on $q, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta}, F$ such that if $\Delta_n(\hat{f}_m, f_0) \leq C\phi_n L \log^2(n)$ then*

$$R(\hat{f}_n, f_0) \leq C'\phi_n L \log^2 n \tag{8}$$

*and if $\Delta_n(\hat{f}_n, f_0) \geq C\phi_n L \log^2 n$ then*

$$\frac{1}{C'}\Delta_n(\hat{f}_n, f_0) \leq R(\hat{f}_n, f_0) \leq C'\Delta_n(\hat{f}_n, f_0) \tag{9}$$

[a] From what I understand, the notation $\lesssim f_n$ means $o(f_n)$ whereas the notation $\asymp f_n$ means $O(f_n)$

In order to minimize the rate $\phi_n L \log^2 n$ the best choice is to choose $L$ of the order $\log_2 n$. The rate in the regime $\Delta_n(\hat{f}_n, f_0) \leq C'\phi_n \log^3 n$ becomes then

$$R(\hat{f}_n, f_0) \leq C'\phi_n \log^3 n$$

Convergence rate in Theorem 1 depends on $\phi_n$. Below will show that $\phi_n$ is a lower bound for the minimax estimation risk over this class. The term $\Delta_n(\hat{f}_m, f_0)$ is large if $\hat{f}_n$ has a large empirical risk compared to the empirical risk minimizer. Having this term in the convergence rate is unavoidable as it also wppears in the lower bound derived in (9). Since for the empirical risk minimizer the $\delta_n$-term is zero by definition, we have the following direct consequence of the main theorem.

**Corollary 1.** *Let $\tilde{f}_n \in \arg\min_{f\in\mathcal{F}(L,\boldsymbol{p},s,F)} \sum_{i=1}^{n}(Y_i - f(\mathbf{X}_i))^2$ be the empirical risk minimizer. Under the same conditions as for Theorem 1, there exists a constant $C'$ only depending on $q, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta}$ such that*

$$R(\tilde{f}_n, f_0) \leq C'\phi_n L \log^2 n \tag{10}$$

Condition (i) in Theorem 1 is mild and only states that the newtwork functions should have at least the same supremum norm as the regression function. From the other assumptions in Theoream 1 it becomes clear that there is a lot of flexibility in picking a good network architecture as long as the number of active parameters is of the “right” order. To choose a network depth $L$ is is suffecient to have an upper bound on the $t_i \leq d_i$ and the smoothness indices $\beta_i$. Network width can be chosen independent of the smoothness indices by taking, for instance $n \lesssim \min_i p_i$.

Maybe possible to choose the sparsity $s$ adaptibely. From a practical point of view it is concievable but left to future work.

Number of network parameters in a fully connected network is of the order $\sum_{i=0}^{L} p_i p_{i+1}$. This shows that Theorem 1 requires sparse networks (since it requires a condition on $L$). For clearness of exposition, Theorem 1 is stated without explicit constants, the proofs, however, an non-asymptotic. It is well-known that depe learning outperforms other methods only for long sample size. This indicated that the method may be able to adapt to underlying structure in the signal and thereform achieving fast convergence rates but with large constants or remainder terms which spoil the results for small samples.

Proof of the risk bounds in Theorem 1 is based on the following oracle-type inequality

**Theorem 2.** *Consider the d-variate nonparametric regression model specified in (1) with unkown regression function $f_0$ satisfying $\|f_0\|_\infty \leq F$ for some $F \geq 1$. Let $\hat{f}_n$ be any estimator taking values in the class $\mathcal{F}(L, \boldsymbol{p}, s, F)$ and let $\Delta_n(\hat{f}_n, f_0)$ be the quantity defined in (5). For any $\epsilon \in (0, 1]$ there exists a constant $C_\epsilon$ depending only on $\epsilon$ such that with*

$$\tau_{\epsilon,n} := C_\epsilon F^2 \frac{(s+1)\log\left(n(s+1)^L p_0 p_{L+1}\right)}{n}$$

*we have*

$$\begin{aligned}(1-\epsilon)^2 \Delta_n(\hat{f}_n, f_0) - \tau_{\epsilon,n} &\leq R(\hat{f}_n, f_0) \\ &\leq (1+\epsilon)^2 \left( \inf_{f \in \mathcal{F}(L,\boldsymbol{p},s,F)} \|f - f_0\|_\infty^2 + \Delta_n(\hat{f}_n, f_0) \right) + \tau_{\epsilon,n}\end{aligned}$$

A consequence of the oracle inequality is that the upper bounds on the risk become worse as the number of later increases. This is consistent with what has been observed in practice.

An inspection of the proof shows two specific properties of the ReLU function is used. One of the advantages of deep ReLU networks is the projection property

$$\sigma \circ \sigma = \sigma \tag{11}$$

that can be used to pass a signal without change through several layers in the network. This is important since the approximation theory is based on the construction of smaller networks for simpler tasks that may not all have the same network depth. To combine these subnetworks into one needs to synchronzie network depths by adding hidden layers that do not change the output.

Another advantage of the ReLU activation is that all network parameters can be taken to be bounded in absolute value by one. If all network parameters are initialized by a value in [-1,1], this means that each network parameters only need to be varied by at most two during training. It is unclear whether results in the literature for non ReLU acrivation functions hold for bounded network parameters.

The $L\log^2 n$ factor in the convergence rate $\phi_n L \log^2 n$ is likely an artifact of the proof. Nexgt show that $\phi_n$ is a lower bound for the minimax estimation risk over the class $\mathcal{G}(q, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta}, K)$ in the interesting case that $t_i \leq \min(d_0, \ldots, d_{i-1})$ for all $i$. This means that no dimensions are added on deeper abstraction levls in the composition of functions.

**Theorem 3.** *Consider the nonparameteric regression model (1) with $\mathbf{X}_i$ drawn from a distribution with Lebesgue density on $[0,1]^d$ which is lower and upper bounded by positive constants. For any non-negative integer q, any dimension vectors $\boldsymbol{d}$ and $\boldsymbol{t}$ satisfying $t_i \leq \min(d_0, \ldots, d_{i=1})$, any smoothness vector $\boldsymbol{\beta}$ and all sufficiently large constants $K > 0$, there exists a positive constant c such that*

$$\inf_{\hat{f}_n} \sup_{f_0 \in \mathcal{G}(q,\boldsymbol{d},\boldsymbol{t},\boldsymbol{\beta},K)} R(\hat{f}_n, f_0) \geq c\phi_n$$

*where the* inf *is taken over all estimators $\hat{f}_n$*[a]

[a] This shows that $\phi_n$ is a lower bound on minimax rate of convergence for estimators

Proof is in appendix. Main ideas are sketched

1. For simplicity, assume that $t_i = d_i = 1$ for all $i$. In this case, the functions $g_i$ are all univariate and real-valued. Define $i^* \in \arg\min_{i=0,\ldots,q} \beta_i^*/(2\beta_i^*+1)$ as an index for which estimation rate is obtained.
2. For any $\alpha > 0, x^\alpha$ has Hölder smoothness $\alpha$ and for $\alpha = 1$ the function is infintely differentiable and has finite Hölder norm for all smoothness indices.[2].Set $g_\ell(x) = x$ for $\ell < i^*$ and $g_\ell(x) = x^{\beta_\ell \wedge 1}$ for $\ell > i^*$. Then
$$f_0(x) = g_1 \circ g_{q-1} \circ \ldots \circ g_1 \circ g_0(x) = \big(g_{i^*}(x)\big)^{\prod_{\ell=i^*+1}^{q} \beta_\ell \wedge 1}$$
3. Assuming a uniform random design (errors distributed the same), the Kullback-Leibler divergence is $KL(P_f, P_g) = \frac{n}{2}\|g-f\|_2^2$.[3] Take a kernel function $K$ and consider $\tilde{g}(x) = h^{\beta_{i^*}} K(x/h)$. Under standard assumptions on $K$, $\tilde{g}$ has Hölder smoothness index $\beta_{i^*}$.
4. Now can generate two hypotheses $f_{00}(x) = 0$ and $f_{01}(x) = (h^{\beta_{i^*}} K(x/h))^{\prod_{\ell=i^*+1}^{q} \beta_\ell \wedge 1}$ by taking $g_{i^*}(x) = 0$ and $g_{i^*}(x) = \tilde{g}(x)$. Therefore, because our domain bounds $x \in [0,1]$: $|f_{00}(0) - f_{01}(0)| \gtrsim h^{\beta_{i^*}}$ assuming that $K(0) > 0$.
5. For the Kullback-Leibler divergence, find $KL(P_{f_{00}}, P_{f_{01}}) \lesssim nh^{2\beta_{i^*}^*+1}$. Using a theorem 2.2 from the book *Introduction to nonparametric estimation* (Tsybakov 2009), this shows that the poitwise rate of convergence is
$$n^{-2\beta_{i^*}^*/(2\beta_{i^*}^*+1)} = \max_{1=0,\ldots,q} n^{-2\beta_i^*/(2\beta_i^*+1)}$$
which matches with the upper bound since $t_i = 1$ for all $i$. For lower bound on the prediction error, generalize argument to a multiple testing problem.

$L^2$-minimax rate coincides in most regimes with the sup-norm rate obtained for composition of two functions. But unlike the classical nonparameteric regression model, the minimax estimation rates for $L^2$-loss and sup-norm loss differ fot some setups by a polynomial power. There are several results in approximation theory that provide lower bounds on the number of required network weights $s$ such that all functions in a function class can be approximated by a $s$-sparse network up to some prescribed error. Results of this flavor can also be quite easily derived by combining the minimax lower bound with the oracle inequality. Argument is that if the same approximaation rates would hold for networks with fewer parameters, we would obtain rates that are faster than the minimax rates.

**Lemma 1.** *Given $\beta, K > 0, d \in \mathbb{N}$, there exists constants $c_1, c_2$ only depending on $\beta, K, d$ such that if*
$$s \leq c_1 \frac{\epsilon^{-d/\beta}}{L\log(1/\epsilon)}$$
*for some $\epsilon \leq c_2$, then for any width vector $\boldsymbol{p}$ with $p_0 = d$ and $p_{L+1} = 1$*
$$\sup_{f_0 \in \mathcal{C}_d^\beta([0,1]^d, K)} \inf_{f \in \mathcal{F}(L,\boldsymbol{p},s)} \geq \epsilon$$

This, I guess, helps establish some lower bound.

[2] Holder norm with smoothness index $\beta$ over a class of functions defined over $\Omega$ and into $\mathbb{R}$,
$$\|f\|_{1,\beta} = \sup_{x\in\Omega}|f(x)| + \sup_{x\in\Omega}|f'(x)| + \sup_{\substack{x,y\in\Omega \\ x\neq y}} \frac{|f'(x) - f'(y)|}{\|x-y\|^\beta}$$

[3] Kullback-Leibler divergence is a measure of how “far apart” probability distributions are. If $P$ and $Q$ are probability distributions over a set $\mathcal{X}$, and $P$ is absolutely continuous with respect to $Q$, then the Kullback-Leibler divergence from $Q$ to $P$ is defined as
$$KL(P_f, P_g) = \int_{\mathcal{X}} \log\left(\frac{dP}{dQ}\right) dP$$
where $\frac{dP}{dQ}$ is the density (Radon-Nikodym derivative) of $P$ with respect to $Q$.