DebiNet: Debiasing Linear Models with Nonlinear Overparameterized Neural Networks

Shiyun Xu[⋆] **Zhiqi Bu**[⋆]
Department of Applied Mathematics and Computational Science
University of Pennsylvania

Abstract

Recent years have witnessed strong empirical performance of over-parameterized neural networks on various tasks and many advances in the theory, e.g. the universal approximation and provable convergence to global minimum. In this paper, we incorporate over-parameterized neural networks into semi-parametric models to bridge the gap between inference and prediction, especially in the high dimensional linear problem. By doing so, we can exploit a wide class of networks to approximate the nuisance functions and to estimate the parameters of interest consistently. Therefore, we may offer the best of two worlds: the universal approximation ability from neural networks and the interpretability from classic ordinary linear model, leading to both valid inference and accurate prediction. We show the theoretical foundations that make this possible and demonstrate with numerical experiments. Furthermore, we propose a framework, DebiNet, in which we plug-in arbitrary feature selection methods to our semi-parametric neural network. DebiNet can debias the regularized estimators (e.g. Lasso) and perform well, in terms of the post-selection inference and the generalization error.

1 Introduction

In the studies including signal inverse problem, genome-wide association studies, criminal justice and economic forecasts, linear models may be preferred over non-parametric models such as neural networks, due to their simplicity and interpretability. Especially, the ordinary least squares (OLS) estimator is known to be consistent and thus capable of offering both valid inference as well as prediction. In high dimension though, the unique OLS is not available and certain structural assumptions such as the sparsity have to be introduced, e.g. via the regularization. However, such regularization induces bias to the estimators and many efforts have been devoted to correcting or debiasing this bias for post-selection inference or selective inference [45, 5, 29, 48]. Yet the methods are usually specific to a type of estimator and hard to compute efficiently. On the other hand, the black-box neural networks commonly have much stronger prediction performance over linear models, by adapting to the features automatically without requiring many assumptions. Nevertheless, it can be difficult to explain the prediction of a neural network and therefore to trust it without the inference guarantee.

The need to bridge the gap between the inference and the prediction motivates the partially linear model (PLM) [17, 33, 8, 22], a semi-parametric model that combines the strengths of two lines of researches. We leverage the approximation power of the non-parametric component to efficiently and consistently estimate the significant regressors in the linear component, thus allowing the regularized linear models to infer. Additionally, the non-parametric component trades its interpretability off and contributes to the prediction power of the PLM.

Mathematically, PLM is a generalized problem which includes the linear problem as a sub-case:

$$\mathbf{y} = \mathbf{D}\boldsymbol{\beta} + f(\mathbf{Z}) + \boldsymbol{\epsilon}. \tag{1.1}$$

Here $\mathbf{D}$ and $\mathbf{Z}$ are two data matrices, $\mathbf{y}$ is the label, $\boldsymbol{\beta}$ is the parameters of interest, f is a (possibly) non-linear function and $\boldsymbol{\epsilon}$ is the noise. The linear component $\mathbf{D}\boldsymbol{\beta}$ is parametric and the coefficients $\boldsymbol{\beta}$ can be estimated consistently [22, 40], provided that the non-linear component f, which is non-parametric, can be accurately approximated. In other words, the effects of estimation and approximation in PLM are separated into different

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. ⋆ **Equal contribution.**

components.

The key element to the strong PLM performance is the approximation power of the non-parametric component. Traditionally, kernel estimators such as Nadaraya-Watson (NW) kernel [40] or Local Linear (LL) regression [20] are applied for the approximation and have been proven to be consistent. Recently, Double/Debiased Machine Learning (DML) is proposed in [10] that uses any qualified machine learning models to replace the kernels. In this work, we propose to apply a particular choice of models, namely the over-parameterized neural network, to extend the DML framework both theoretically and algorithmically. To be more specific, our contribution is three-fold:

1. We propose a PLM that employs arbitrary over-parameterized neural networks to approximate nuisance functions, based on the partialling-out technique. We illustrate its advantages over the computational efficiency, the flexibility, the robustness and the performance.
2. We utilize the over-parameterized neural network theories, such as the Neural Tangent Kernel (NTK) [26], to guarantee the goodness of approximation and the consistency of $\boldsymbol{\beta}$ estimators.
3. We further develop a framework, DebiNet, that uses our PLM to debias arbitrary feature selection methods and demonstrates promising results on both the inference and the prediction.

Moreover, while our PLM can be applied to real-world datasets, as demonstrated in Section 7, we focus on synthetic data for Table 1 and Table 2 as we need the access to true $\boldsymbol{\beta}$ to illustrate the statistical consistency, which is our main goal of debiasing.

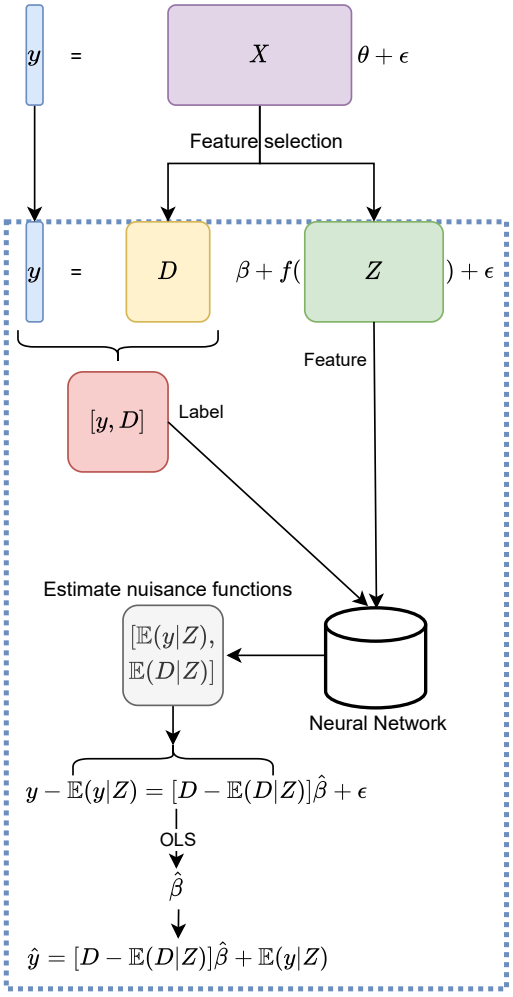

Figure 1: The architecture of DebiNet. The DebiNet applies arbitrary feature selection method and then PLM-NN in the blue dashed frame to estimate the parameters of interest as well as to predict.

1.1 Related Work

Feature selection and post-selection inference In the high dimensional data analysis, many feature selection methods such as the forward-backward selection [16], Lasso [47], adaptive Lasso [52], elastic net [53], (sparse) group lasso [41, 18] and SLOPE [6] have been proposed to select important variables. However, these methods may incur large bias and make valid inference difficult [37, 21] if not impossible. A long list of researches have developed methods to debias these models, especially the Lasso [29, 9, 46], by separating the procedure of variable selection and coefficient estimation. Two well-known yet quite different approaches are the OLS post-Lasso [4] and the debiased Lasso (or the desparsified Lasso) [51, 49, 32]. Interestingly, both methods are deeply related to PLM.

Partially linear models PLMs are commonly used in multiple fields of study to partly debias the estimators. Kernel smoothing [42], local polynomial regression [20], spline-based regression [23] and related techniques have been applied to PLM and proven consistent under various conditions. To take one step further, Double/Debiased Machine Learning (DML) [10] proposed to use sample-splitting and any qualifying machine learning model to learn PLM, but the conditions under which a model qualifies can be case-specific and hence hard to verify. We note that the Lasso, decision trees, random forests and an under-parameterized two-neuron neural network are investigated in [10], yet the over-parameterized neural networks are not covered. We emphasize that the choice of model is critical and our proposal is efficient in learning with theoretical support.

Over-parameterized neural networks To solve complicated machine learning tasks, state-of-the-art

neural networks are heavily over-parameterized. Such neural networks may enjoy many nice properties. For example, the universal approximation property [43, 19, 50, 24, 30] allows neural networks to learn arbitrary target functions. Recently, NTK has been explored to study the generalization behavior and the convergence of over-parameterized neural networks [2, 1, 15, 28, 14]. In this work, we follow this approach to show that multivariate-output neural networks converge to the global minimum exponentially fast.

2 Setup and Notation

Suppose the data matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ with i.i.d. observations, true parameters $\boldsymbol{\theta} \in \mathbb{R}^p$, label $\mathbf{y} \in \mathbb{R}^n$ and $\boldsymbol{\epsilon}$ is the random noise. We consider the linear model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon} \tag{2.1}$$

and apply a feature selection method which generates an estimator $\hat{\boldsymbol{\theta}}$ together with an active set of $[p]$ defined as $S := \{j : \hat{\theta}_j \neq 0\}$. We rewrite $\mathbf{X} = [\mathbf{D}, \mathbf{Z}]$ by grouping the selected features into the submatrix $\mathbf{D} := \mathbf{X}_S \in \mathbb{R}^{n\times p_L}$ and the unselected ones into $\mathbf{Z} := \mathbf{X}_S^C \in \mathbb{R}^{n\times p_N}$. We can write $\boldsymbol{\theta}^\top = [\boldsymbol{\beta}^\top, \boldsymbol{\gamma}^\top]$ according to whether a feature is selected or not. Then, the original linear model (2.1) is equivalent to

$$\mathbf{y} = \mathbf{D}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon} \tag{2.2}$$

in which we assume $\mathbb{E}(\boldsymbol{\epsilon}|\mathbf{D}, \mathbf{Z}) = 0$ and $\mathrm{Var}(\boldsymbol{\epsilon}) = \sigma_\epsilon^2\mathbf{I}$, same as in [40] and [10, Example 1.1]. Here we can consider **arbitrary feature selection methods** including the arguably most well-known ℓ_1 regularized linear model, Lasso, as our main example:

$$\hat{\boldsymbol{\theta}}_{\text{Lasso}} = \mathrm{argmin}_{\boldsymbol{\theta}} \frac{1}{2}\|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|^2 + \lambda\|\boldsymbol{\theta}\|_1$$

We generalize the model (2.2) to a partially linear model as in (1.1), in which $\mathbf{D}\boldsymbol{\beta}$ is the parametric component and $f(\mathbf{Z})$ is the non-parametric component with $f : \mathbb{R}^{p_N} \to \mathbb{R}$.

On one hand, this formulation enables us to incorporate non-parametric tools in estimating the parameters of interest $\boldsymbol{\beta}$ consistently. Notice that $f(\mathbf{Z}) = \mathbf{Z}\boldsymbol{\gamma}$ is only estimated as a whole without estimating $\boldsymbol{\gamma}$. In other words, we trade off the consistency of estimating $\boldsymbol{\gamma}$ to gain the consistency of estimating $\boldsymbol{\beta}$. When the feature selection method is effective, most of the significant features are selected, together with some noisy features. In this case, the partially linear model with carefully chosen nuisance functions and tuned hyperparameters, is expected to estimate the coefficients of the selected features consistently.

The trick of obtaining consistent estimator is to apply the conditional expectation on $\mathbf{Z}$ to orthogonalize or '**partial out**' the non-linear component:

$$\mathbf{y} = \mathbf{D}\boldsymbol{\beta} + f(\mathbf{Z}) + \boldsymbol{\epsilon} \Longrightarrow \mathbb{E}(\mathbf{y}|\mathbf{Z}) = \mathbb{E}(\mathbf{D}|\mathbf{Z})\boldsymbol{\beta} + f(\mathbf{Z})$$

Taking the difference between the two equations leaves an ordinary linear model without intercept,

$$\mathbf{y} - \mathbb{E}(\mathbf{y}|\mathbf{Z}) = (\mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z}))\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.3}$$

Therefore, as long as we can accurately estimate the conditional expectation (at least asymptotically), $\hat{\boldsymbol{\beta}}$ is consistent by the theory of ordinary least squares. On the other hand, PLM can accurately predict on future data, if the conditional expectation is well-approximated:

$$\hat{\mathbf{y}} = \mathbf{D}\hat{\boldsymbol{\beta}} + \hat{f}(\mathbf{Z}). \tag{2.4}$$

Here $\hat{f}(\mathbf{Z})$ is an approximation of the unknown $f(\mathbf{Z})$.

3 Partially Linear Model with Neural Network

We propose an efficient PLM in Algorithm 1, using the neural network to universally approximate the mapping $\mathcal{M}(\mathbf{Z}) := \mathbb{E}([\mathbf{y}, \mathbf{D}]|\mathbf{Z})$ in a single fitting, without approximating f explicitly. Comparing with other PLMs, our method enjoys several advantages over the computational efficiency (fewer model fittings and less memory burden), the flexibility with multivariate output and network architectures, as well as the provable global convergence at linear rate.

Algorithm 1 Partially Linear Model with Neural Networks (PLM-NN)

Input: Data matrix $[\mathbf{D}, \mathbf{Z}]$, label $\mathbf{y}$
Estimation of $\boldsymbol{\beta}$:
1. fit $[\mathbf{y}, \mathbf{D}] \sim \mathbf{Z}$ via over-parameterized neural network to derive $\mathbb{E}([\mathbf{y}, \mathbf{D}]|\mathbf{Z})$;
2. fit $\mathbf{y} - \mathbb{E}(\mathbf{y}|\mathbf{Z}) \sim \mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z})$ via OLS to derive $\hat{\boldsymbol{\beta}}$;
Prediction of y and Estimation of f:
3. define $\hat{\mathbf{y}} := \mathbb{E}(\mathbf{y}|\mathbf{Z}) + (\mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z}))\hat{\boldsymbol{\beta}}$ and $\hat{f}(\mathbf{Z}) := \mathbb{E}(\mathbf{y}|\mathbf{Z}) - \mathbb{E}(\mathbf{D}|\mathbf{Z})\hat{\boldsymbol{\beta}}$

Historical researches and recent advances have been made towards the fast and accurate estimation of PLMs. Traditional PLMs (see Algorithm 4 in Appendix) employ kernel methods (e.g. the NW kernel [40] and the LL estimator [20]) to estimate the nuisance functions $m_y(\mathbf{Z}) := \mathbb{E}(\mathbf{y}|\mathbf{Z})$ and $m_D(\mathbf{Z}) := \mathbb{E}(\mathbf{D}|\mathbf{Z})$ separately, before fitting the OLS to obtain $\hat{\boldsymbol{\beta}}$. Kernels with proper bandwidths has been rigorously shown to consistently estimate the nuisance functions [40, 22, 31], under mild conditions, including the case that $\mathbf{D}$ is multivariate.

However, additional models may need to be fitted to estimate $\boldsymbol{\beta}$ and to predict $\hat{\mathbf{y}}$. As a toy example, consider $\mathbf{y} = \mathbf{D}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$ and $\mathbf{D}$ is independent of $\mathbf{Z}$. Due to the dependence conditions, different bandwidths

h_D, h_y are used in the weight functions: h_y is finite for approximating m_y but h_D should be set to infinity for approximating m_D. As a consequence, another possibly different bandwidth h_f is needed for approximating f to predict $\hat{\mathbf{y}}$, resulting in a total of three kernel regressions and one OLS regression. In addition, kernel regressions require memorizing the entire dataset and incur severe storage issues on big data.

Recently, DML [10] extends PLMs, by replacing the kernel regression with a rich class of qualified machine learning methods and by employing sample-splitting to remove the bias from overfitting nuisance functions. Unfortunately, checking the qualification of models may be difficult and the sample-splitting adds to the computational cost. Furthermore, $\mathbf{D}$ being multivariate renders many popular methods invalid or inefficient to learn m_D, including the vanilla Lasso and decision trees. DML suggests to fit each feature of $\mathbf{D}$ to $\mathbf{Z}$ separately so as to only learn univariate output functions. We note that such operation can be costly in high dimension: with K-fold sample-splitting, one needs $K(p_L+1)$ model fittings to learn (m_y, m_D).

PLMs	Estimation MSE	Train MSE	Test MSE
PLM-NN	1.59×10^{-5}	10.48	10.36
PLM-NW	0.11×10^{-5}	16.33	16.08
DML Lasso	0.60×10^{-5}	15.79	15.79
DML DT	65.1×10^{-5}	336.00	342.68
DML RF	0.58×10^{-5}	44.00	44.87

Table 1: Comparison of PLMs in 50 independent runs. Here PLM-NW denotes the PLM using NW kernel, DT denotes decision trees at depth of 2 and RF denotes random forests over 100 trees. See Appendix D.1 for experiment details.

Instead, we propose to use over-parameterized neural networks as strong candidates compared to other methods. For instance, neural networks may not suffer as much as kernel regressions from the curse of dimensionality [3], a significant decrease of performance when the dimension increases. We further extend the DML framework by significantly reducing the computational cost as follows. First, we choose the nuisance function $\mathcal{M}: \mathbb{R}^{p_N} \to \mathbb{R}^{p_L+1}$ instead of (m_y, m_D) to handle the multivariate output directly. We will show that our nuisance function $\mathcal{M}$ is learnable to neural networks. Second, we do not need sample-splitting to deal with overfitting, thanks to the universal approximating property of over-parameterized neural networks. In practice, we adopt the early stopping to enhance the generalization performance of our estimator. As a result, the total number of model fittings to approximate nuisance functions reduces from $K(p_L+1)$ to 1.

We empirically illustrate that our method is comparable to the traditional PLM with kernels and to the original DML with K-fold cross-fitting in Table 1. Here the estimation MSE is $\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}\|_2^2/p_L$ (only available in synthetic data) and the train/test MSE is $\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2/n$. Notice that throughout this paper, we train the two-layer, fully-connected, ReLU activated neural networks with an appropriately wide hidden layer, for its simplicity in proof. However, the DebiNet can work with any neural networks such as deep fully-connected and convolutional neural networks. We optimize over MSE loss with the gradient descent at a sufficiently small learning rate.

4 Gradient Descent Optimizes Over-parameterized Multivariate Networks

With the universal approximation capability for any continuous function, the over-parameterized neural networks with optimal parameters can successfully learn $\mathbb{E}(\mathbf{y}|\mathbf{Z})$ and $\mathbb{E}(\mathbf{D}|\mathbf{Z})$. Many works have studied the generalization behavior of the over-parameterized neural networks [35, 2, 1, 7, 34, 36]. We empirically demonstrate that, in our setting of Table 1, the generalization indeed benefits from the over-parameterization.

The next question is how to efficiently find the optimal parameters and we address this by extending the NTK approach in [15].

To demonstrate the trainability of wide neural networks, we consider a two-layer fully-connected neural networks with rectified linear unit (ReLU) activation. Denoting $\mathbf{W} \in \mathbb{R}^{p_N \times m}, \mathbf{A} \in \mathbb{R}^{m \times (p_L+1)}$ as the weights in first and second layers respectively, we can write the neural network as

$$F(\mathbf{W}, \mathbf{A}, \mathbf{z}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} \mathbf{A}_r \sigma\left(\mathbf{w}_r^\top \mathbf{z}\right) \tag{4.1}$$

where $F: \mathbb{R}^{p_N} \to \mathbb{R}^{p_L+1}$, $\mathbf{z} \in \mathbb{R}^{p_N}$ is the input, $\mathbf{w}_r$ and $\mathbf{A}_r$ are the weights corresponding to the r-th neuron in the hidden layer and $\sigma(\cdot)$ is the ReLU activation function.

Given the dataset $\{(\mathbf{Z}_i, \mathbf{M}_i)\}_{i=1}^n$ with the multivariate response $\mathbf{M} := [\mathbf{y}, \mathbf{D}]$. We aim to minimize

$$L(\mathbf{W}, \mathbf{A}) = \sum_{i=1}^{n} \frac{1}{2} \left\|F(\mathbf{W}, \mathbf{A}, \mathbf{Z}_i) - \mathbf{M}_i\right\|^2. \tag{4.2}$$

Adopting the same strategy as [15], we fix the second layer and apply the gradient descent to optimize the

first layer, via the gradient flow defined as

$$\frac{d\mathbf{w}_r(t)}{dt} = -\frac{\partial L(\mathbf{W}(t), \mathbf{A})}{\partial \mathbf{w}_r(t)}$$
$$= \sum_{h=1}^{(1+p_L)} \sum_{j=1}^{n} (M_{jh} - F_h(\mathbf{W}, \mathbf{A}, \mathbf{Z}_j)) \frac{\partial F_h(\mathbf{W}, \mathbf{A}, \mathbf{Z}_j)}{\partial \mathbf{w}_r}.$$

Now we quote an important fact that justifies our main theorem.

Fact 4.1 (Assumption 3.1 and Theorem 3.1 in [15]). *If for any $i \neq j$, $\mathbf{Z}_i \nparallel \mathbf{Z}_j$, then the least eigenvalue $\lambda_0 := \lambda_{\min}(\mathbf{H}^\infty) > 0$, where matrix $\mathbf{H}^\infty \in \mathbb{R}^{n\times n}$ with $(\mathbf{H})_{ij}^\infty = \mathbb{E}_{\mathbf{w}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\mathbf{Z}_i^\top \mathbf{Z}_j \mathbb{I}\left\{\mathbf{w}^\top \mathbf{Z}_i \geq 0, \mathbf{w}^\top \mathbf{Z}_j \geq 0\right\}\right]$.*

Our main theorem shows that, if the least eigenvalue of $\mathbf{H}_{sh}(t)$, which will be defined in (4.4), is always lower bounded, then the loss converges to 0 at a linear rate.

Theorem 1. *Suppose the condition of Fact 4.1 holds and for all $i \in [n]$, $\|\mathbf{Z}_i\|_2 = 1$ and $\|\mathbf{M}_i\| \leq C$ for some constant C. Then if we set the number of hidden neurons $m = \Omega\left(\frac{(1+p_L)^5 n^6}{\delta^3 \lambda_0^4}\right)$ and we i.i.d. initialize $\mathbf{w}_r \sim N(\mathbf{0}, \mathbf{I})$, $A_{rs} \sim \text{unif}\{-1, 1\}$ for $r \in [m], s \in [1 + p_L]$, then with probability at least $1 - \delta$ over the initialization, we have*

$$\|\mathbf{F}_s(t) - \mathbf{M}_s\|_2^2 \leq \exp(-\lambda_0 t)\|\mathbf{F}_s(0) - \mathbf{M}_s\|_2^2.$$

Proof of Theorem 1 (sketch). The full proof can be found in Appendix B. Here we sketch the proof at high level. The conditions of our theorem is to guarantee that $\mathbf{W}(t)$ and consequently $\mathbf{H}_{sh}(t)$, although being time-dependent, stay close to their initializations. Such phenomenon is commonly observed and known as the 'lazy training' for over-parameterized neural networks. Then a careful analysis on the initialization $\mathbf{H}_{sh}(0)$ shows that it is close to $\mathbf{H}^\infty$ and that the NTK is positive definiteness, which leads to the exponentially fast convergence. Interestingly, we note that the NTK is close to a block diagonal matrix $\text{diag}(\mathbf{H}^\infty, \cdots, \mathbf{H}^\infty)$, i.e. each dimension of the output evolves under the same dynamics.

More formally, we derive the dynamics of the output by the chain rule and the gradient flow,

$$\begin{aligned}\frac{d}{dt}F_{is}(t) &= \sum_{r=1}^{m} \langle \frac{\partial F_s(\mathbf{W}(t), \mathbf{A}, \mathbf{Z}_i)}{\partial \mathbf{w}_r(t)}, \frac{d\mathbf{w}_r(t)}{dt} \rangle \\ &= \sum_{h=1}^{(1+p_L)} \sum_{j=1}^{n} (M_{jh} - u_{jh})(\mathbf{H}_{sh})_{ij}(t)\end{aligned} \tag{4.3}$$

in which $\mathbf{H}_{sh}(t)$ is an $n \times n$ matrix:

$$(\mathbf{H}_{sh})_{ij}(t) = \sum_{r=1}^{m} \langle \frac{\partial F_s(\mathbf{W}, \mathbf{A}, \mathbf{Z}_i)}{\partial \mathbf{w}_r}, \frac{\partial F_h(\mathbf{W}, \mathbf{A}, \mathbf{Z}_j)}{\partial \mathbf{w}_r} \rangle =$$
$$\frac{1}{m}\mathbf{Z}_i^\top \mathbf{Z}_j \sum_{r=1}^{m} A_{rs}A_{rh}\mathbb{I}\left\{\mathbf{Z}_i^\top \mathbf{w}_r(t) \geq 0, \mathbf{Z}_j^\top \mathbf{w}_r(t) \geq 0\right\}. \tag{4.4}$$

Vectorizing (4.3) gives

$$\frac{d}{dt}\mathbf{F}_s(t) = \sum_{h=1}^{(1+p_L)} \mathbf{H}_{sh}(t)(\mathbf{M}_s - \mathbf{F}_s(t))$$

which leads to

$$\frac{d}{dt}(\mathbf{M}_s - \mathbf{F}_s(t)) = -\sum_{h=1}^{(1+p_L)} \mathbf{H}_{sh}(t)(\mathbf{M}_s - \mathbf{F}_s(t))$$

This matrix ordinary differential equation has a solution which decays exponentially fast (see Figure 2), provided that $\mathbf{H}_{sh}(t)$ is positive definite with the least eigenvalue bounded away from 0. □

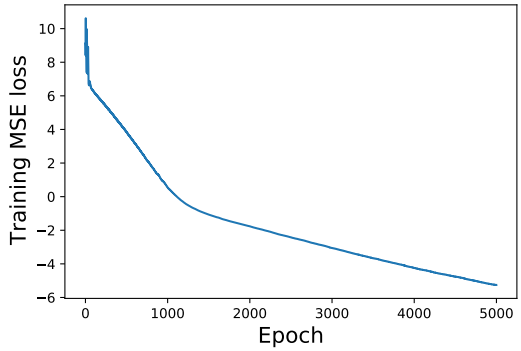

Figure 2: Same setting as Table 1 except $n = 100$ and $\mathbf{Z}$ is normalized. The loss is in logarithmic scale.

Similar to the analysis in [15], we remark that our analysis can be easily generalized from the continuous time analysis to the discrete time one, as well as to training both layers jointly. To see this, we train both layers of the same network as in Table 1 with the gradient descent. The linearity of the log training loss in Figure 2 confirms the exponential convergence rate.

5 Consistency of $\hat{\boldsymbol{\beta}}$

Denote $\mathcal{X} := \mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z})$ and $\mathcal{Y} := \mathbf{y} - \mathbb{E}(\mathbf{y}|\mathbf{Z})$. If the nuisance function $\mathcal{M}$ is consistently estimated, then the standard OLS theory states that the OLS estimator $\hat{\boldsymbol{\beta}} = (\mathcal{X}^\top \mathcal{X})^{-1}\mathcal{X}^\top \mathcal{Y}$ is $\sqrt{n}$-consistent to $\boldsymbol{\beta}$, as $\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})$ converges in probability. Though in reality, the errors in approximating the conditional expectation, incurred when learning $\mathcal{M}$, are unavoidable for any model including the neural networks. Thus they may

cause $\hat{\boldsymbol{\beta}}$ to be inconsistent and the bias needs careful adjustment by the measurement error model (or errors-in-variables) theory as follows.

Suppose instead of $(\mathcal{X}, \mathcal{Y})$, we only observe data $(\widetilde{\mathcal{X}}, \widetilde{\mathcal{Y}}) := (\mathcal{X} + \boldsymbol{\epsilon}_X, \mathcal{Y} + \boldsymbol{\epsilon}_Y)$ which are measured with independent errors $\boldsymbol{\epsilon}_X, \boldsymbol{\epsilon}_Y$. Then the estimator is

$$\begin{aligned}\widetilde{\boldsymbol{\beta}} =& (\widetilde{\mathcal{X}}^\top \widetilde{\mathcal{X}})^{-1} \widetilde{\mathcal{X}}^\top \widetilde{\mathcal{Y}} \\ =& \left(1 - (\widetilde{\mathcal{X}}^\top \widetilde{\mathcal{X}})^{-1} \widetilde{\mathcal{X}}^\top \boldsymbol{\epsilon}_X\right) \boldsymbol{\beta} + (\widetilde{\mathcal{X}}^\top \widetilde{\mathcal{X}})^{-1} \widetilde{\mathcal{X}}^\top (\boldsymbol{\epsilon}_Y + \boldsymbol{\epsilon}).\end{aligned} \tag{5.1}$$

With a careful error analysis in Appendix C, we show that $\widetilde{\boldsymbol{\beta}}$ is consistent if m_D is consistently approximated (meaning the MSE $\frac{\boldsymbol{\epsilon}_X^\top \boldsymbol{\epsilon}_X}{n} \to \sigma_X^2 = 0$). Otherwise, we give a correction based on $\widetilde{\boldsymbol{\beta}}$.

Theorem 2. *Under the assumption of additive measurement errors and if*

$$\begin{aligned}\mathbb{E}(\boldsymbol{\epsilon}_X) = \mathbb{E}(\boldsymbol{\epsilon}_Y) = 0 \\ \mathrm{Var}(\boldsymbol{\epsilon}_X) = \sigma_X^2 \mathbf{I}, \quad \mathrm{Var}(\boldsymbol{\epsilon}_Y) = \sigma_Y^2 \mathbf{I},\end{aligned}$$

then $(\mathbf{I} - \mathbf{R})^{-1} \widetilde{\boldsymbol{\beta}}$ *is* $\sqrt{n}$*-consistent and so is* $\widetilde{\boldsymbol{\beta}}$ *if and only if* $\sigma_X^2 = 0$*, with* $\mathbf{R} = \sigma_X^2 \left(\mathrm{plim} \frac{\mathcal{X}^\top \mathcal{X}}{n} + \sigma_X^2 \mathbf{I}\right)^{-1}$.

Furthermore, suppose the errors are Gaussian: $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_\epsilon^2 \mathbf{I}), \boldsymbol{\epsilon}_X \sim \mathcal{N}(0, \sigma_X^2 \mathbf{I}), \boldsymbol{\epsilon}_Y \sim \mathcal{N}(0, \sigma_Y^2 \mathbf{I})$*, then we have the asymptotic normality*

$$\sqrt{n}\left(\widetilde{\boldsymbol{\beta}} - (\mathbf{I} - \mathbf{R})\boldsymbol{\beta}\right) \xrightarrow{\mathcal{D}} \mathcal{N}\left(0, \frac{\sigma_\epsilon^2 + \sigma_Y^2}{\sigma_X^2} \mathbf{R}\right).$$

6 DebiNet: Debiasing Neural Network

Before introducing our debiasing network, DebiNet, we revisit two broadly used approaches to debias Lasso. The first method is OLS post-Lasso [4], which uses the features selected by the Lasso, i.e. $\mathbf{D} = \mathbf{X}_S$, to fit an OLS with $\mathbf{y}$ as the response variable and obtains $\hat{\boldsymbol{\beta}}_{\mathrm{OLS}}$. The final estimator substitutes the non-zero entries in the Lasso estimator $\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}$ by the OLS estimator:

$$[\hat{\boldsymbol{\theta}}_{\text{OLS post-Lasso}}]_j = \begin{cases} 0 & \text{if } [\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}]_j = 0 \\ [\hat{\boldsymbol{\beta}}_{\mathrm{OLS}}]_j & \text{if } [\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}]_j \neq 0 \end{cases} \tag{6.1}$$

The second approach is the debiased (or desparsified) Lasso [51, 49],

$$\hat{\boldsymbol{\theta}}_{\text{debiased Lasso}} = \hat{\boldsymbol{\theta}}_{\mathrm{Lasso}} + \hat{\Sigma}^{-1}_{\mathrm{Lasso}} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}) \tag{6.2}$$

where $\hat{\Sigma}^{-1}_{\mathrm{Lasso}}$ is a pseudo-inverse of $\mathbf{X}^\top \mathbf{X}$, obtained by nodewise Lasso regressions.

Notice that OLS post-Lasso only partly debiases $\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}$ while the debiased Lasso debiases all elements in $\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}$. The main difference between these approaches is the information they exploit: OLS post-Lasso requires only the information of $\mathbf{X}_S$ and the indicator $\mathbb{I}(\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}} \neq 0)$, and the debiased Lasso uses the entire $\mathbf{X}$ and $\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}$. Nevertheless, both methods actually connect to PLM: one can view the OLS post-Lasso as a special case of PLM with $f = 0$ and the nodewise Lasso regression indeed learns m_D in a semi-parametric regime.

Now we are ready to present our debiasing model.

Algorithm 2 DebiNet

Input: Data matrix $\mathbf{X}$, label $\mathbf{y}$

1. fit $\mathbf{y} \sim \mathbf{X}$ via any feature selection model to obtain $\mathbf{D} := \mathbf{X}_S$ and $\mathbf{Z} := \mathbf{X}_S^C$;

Estimation of $\boldsymbol{\beta}$:

2. fit $[\mathbf{y}, \mathbf{D}] \sim \mathbf{Z}$ via over-parameterized neural network to derive $\mathbb{E}([\mathbf{y}, \mathbf{D}]|\mathbf{Z})$;
3. fit $\mathbf{y} - \mathbb{E}(\mathbf{y}|\mathbf{Z}) \sim \mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z})$ via OLS to derive $\hat{\boldsymbol{\beta}}$;

Prediction of y and Estimation of f:

4. define $\hat{\mathbf{y}} := \mathbb{E}(\mathbf{y}|\mathbf{Z}) + (\mathbf{D} - \mathbb{E}(\mathbf{D}|\mathbf{Z}))\hat{\boldsymbol{\beta}}$ and $\hat{f}(\mathbf{Z}) := \mathbb{E}(\mathbf{y}|\mathbf{Z}) - \mathbb{E}(\mathbf{D}|\mathbf{Z})\hat{\boldsymbol{\beta}}$.

Similar to the debiased Lasso, DebiNet also exploits the unselected features and thus avoids the model mis-specification error in OLS post-Lasso. For example, suppose that the sparsity of $\boldsymbol{\beta}$ is k out of p (i.e. there are k important features) and that Lasso fails to select all them, then it is impossible for the OLS post-Lasso model ($\hat{\mathbf{y}}_{\text{OLS post-Lasso}} = \mathbf{D}\hat{\boldsymbol{\beta}}_{\mathrm{OLS}}$) to learn the true model nor to be consistent. However, under the debiased Lasso ($\hat{\mathbf{y}}_{\text{debiased Lasso}} = \mathbf{X}\hat{\boldsymbol{\theta}}_{\text{debiased Lasso}}$) and DebiNet, the true model is learnable. We highlight that the mis-specification (when true positive rate is not 1) is common in high dimension and high sparsity [44, 11, 13, 12], known as the Donoho-Tanner phase transition. Therefore, it is vital to remedy the failure of variable selection by using the signals left in $\mathbf{Z}$ (see Table 2).

On the other hand, DebiNet also shares some similarity with the OLS post-Lasso as both methods only need the information in $\mathbb{I}(\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}} \neq 0)$, both debias the parameters of interest $\boldsymbol{\beta}$ and both make use of the OLS which requires $p_L < n$. It is remarkable that DebiNet only fits three models and the OLS post-Lasso fits two models, while the debiased Lasso needs p nodewise regression to construct $\hat{\Sigma}^{-1}_{\mathrm{Lasso}}$. The computation cost can be immense in high dimension.

In Table 2 and Figure 3, we compare different debiasing methods under various metrics over the active set $\{j : [\hat{\boldsymbol{\theta}}_{\mathrm{Lasso}}]_j \neq 0\}$. We emphasize that, since this work focuses on the consistency of estimation and the effect of debiasing, we must experiment on synthetic

Methods	Estimation MSE	Train MSE	Test MSE	95% Coverage	sec/run
$n = 1000$; High dimension $p = 3000$; High sparsity $k = 300$					
Lasso	0.536 (±0.058)	286.218 (±9.766)	296.112 (±25.615)	NA	-
OLS post-Lasso	1.402 (±0.309)	233.363 (±11.050)	308.440 (±25.241)	0.558	-
debiased Lasso	2.019 (±0.290)	4998.75 (±364.669)	1100.68 (±111.53)	0.000	154
DebiNet	**0.421** (±0.418)	**136.376** (±54.195)	**278.625** (±30.932)	**0.830**	7(60)
NW post-Lasso	1.398 (±0.307)	229.193 (±10.860)	308.399 (±25.301)	0.560	-
$n = 1000$; High dimension $p = 3000$; Low sparsity $k = 10$					
Lasso	0.264 (±0.027)	3.531 (±0.131)	3.634 (±0.411)	NA	-
OLS post-Lasso	**0.001** (± 0.000)	**0.992** (±0.057)	**1.009** (±0.086)	**0.958**	-
debiased Lasso	0.004 (±0.002)	55.024 (±3.295)	14.000 (±1.410)	0.702	150
DebiNet	**0.001** (±0.000)	**0.989** (±0.056)	**1.012** (±0.086)	**0.956**	3(21)
NW post-Lasso	**0.001** (±0.000)	**0.975** (±0.056)	**1.009** (±0.084)	**0.960**	-
$n = 1000$; Low dimension $p = 500$; High sparsity $k = 300$					
OLS	**0.003** (±0.000)	**0.378** (±0.031)	**2.625** (±0.313)	**0.955**	-
Lasso	0.458 (±0.021)	175.803 (±3.846)	245.660 (±25.020)	NA	-
OLS post-Lasso	0.175 (±0.025)	88.959 (±6.314)	211.952 (±21.839)	0.927	-
debiased Lasso	0.299 (±0.036)	122.515 (±12.222)	122.204 (±13.268)	0.123	4
DebiNet	0.175 (±0.025)	88.953 (±6.324)	211.877 (±21.926)	0.928	2(6)
NW post-Lasso	0.174 (±0.025)	87.132 (±6.195)	211.007 (±21.821)	0.927	-

Table 2: Comparison of debiasing methods in 50 independent runs. See Appendix D.2 for data generation details. In the 'sec/run' column, '-' means < 1 second. Here we record two time for DebiNet, one for GPU acceleration and the other in bracket for CPU. NW post-Lasso applies PLM-NW, instead of PLM-NN, after Lasso.

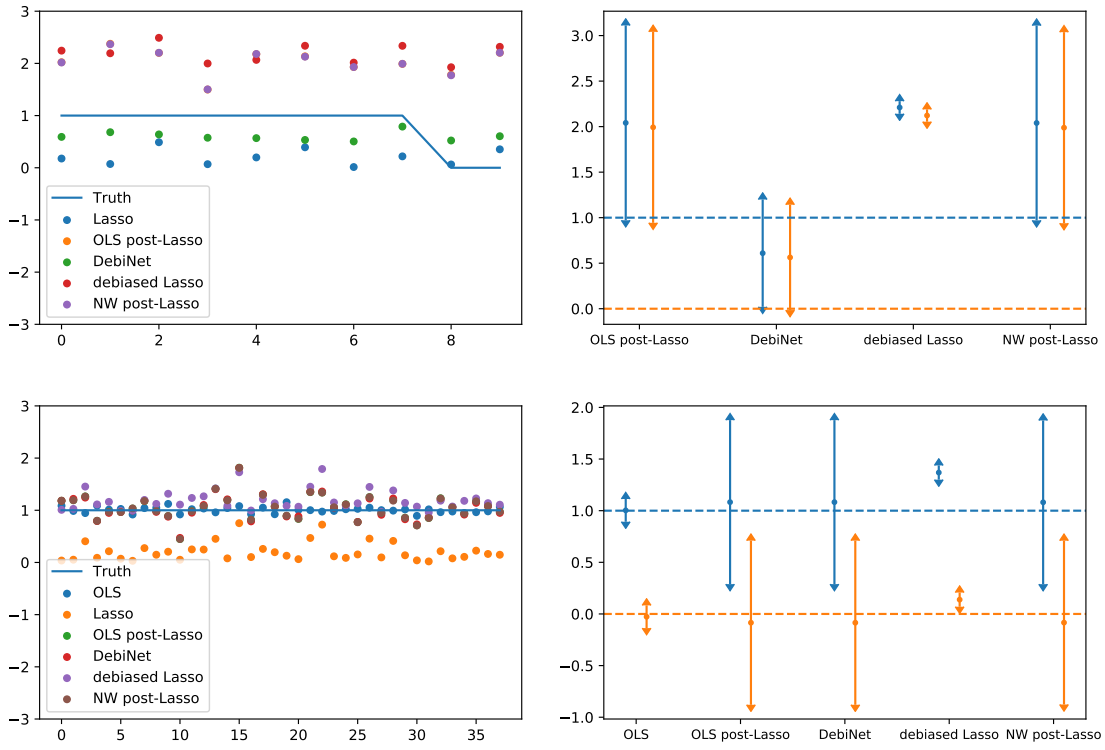

Figure 3: Comparison of debiasing methods on Lasso. $\mathbf{X} \in \mathbb{R}^{1000\times p}$, $\mathrm{Var}(\epsilon) = 1$. Top plots: $p = 3000, \lambda = 2, k = 300$. Bottom plots: $p = 500, \lambda = 1$; left $k = 100$; right $k = 300$. $\boldsymbol{\beta}$ is binary and we plot the 95% confidence intervals according to $\beta_j = 1$ (blue) and $\beta_j = 0$ (orange).

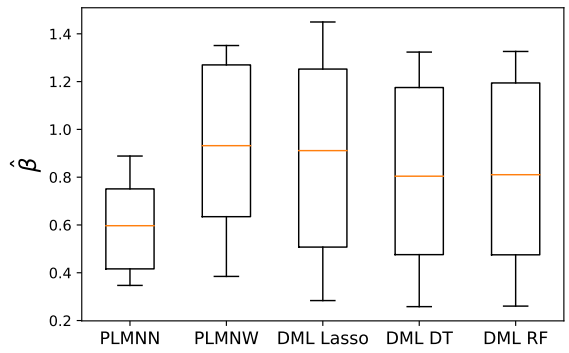

Figure 4: Estimation of $\hat{\boldsymbol{\beta}}$ by different PLMs.

data where the truth $\boldsymbol{\beta}$ is known, in order to compute the estimation MSE and the statistical coverge. In all settings, DebiNet outperforms other methods (except the golden rule, OLS, in the low dimension) and demonstrates its robustness against high dimension and high sparsity. We note that the debiased Lasso exhibits strange behaviors in very high dimension as its working assumption requires the sparsity $k = o(\sqrt{n}/\log p)$ which is roughly 9. In our high sparsity setting, $k = 300 \gg 9$ and the debiased Lasso, giving narrow confidence intervals, may not perform well. In addition, the objective of the debiased Lasso is to reduce the estimation error, instead of to minimize the prediction error. These two goals are unified in low dimension but not necessarily in high dimension. Moreover, the coverage is computed over the active set of Lasso, while the debiased Lasso may offer higher coverage outside the active set [32]. Lastly, we remark that arbitrary variable selection methods can be plugged into DebiNet, as long as the true model is linear and the number of selected variables is less than the sample size: the feature selection only affects the choice of $[\mathbf{D}, \mathbf{Z}]$, but not the subsequent convergence and consistency analyses.

7 A Study of Treatment Effect on 401(k) Data

In this section, we analyze the 1991 Survey of Income and Program Participation data in [10], to study the average treatment effect (ATE) which estimates the impact of household income (our treatment variable $\mathbf{D} \in \mathbb{R}^{9915\times 1}$) on the net financial assets (our response variable $\mathbf{y} \in \mathbb{R}^{9915}$), after accounting for the confounding factors $\mathbf{Z} \in \mathbb{R}^{9915\times 9}$ such as age, family size, years of education, etc. The ATE, or $\beta \in \mathbb{R}$, is expected to be positive according to [38, 39]: when the 401(k) plan program started, people did not base job decisions on the retirement offers but rather on the income.

Methods	Mean $\hat{\beta}$	Median $\hat{\beta}$	SE $\hat{\beta}$
PLM-NN	0.590	0.596	0.186
PLM-NW	0.912	0.931	0.351
DML Lasso	0.887	0.911	0.415
DML DT	0.805	0.804	0.385
DML RF	0.809	0.810	0.388

Table 3: Comparison of treatment effects by different PLMs on the 401(k) dataset.

Reassuringly, the results obtained from different PLMs are relatively consistent with each other across different sub-samplings. We note that the PLM theory indeed supports such consistency and thus over-parameterized neural networks can fit in DML, even without sample-splitting. We also observe that PLM-NN has much smaller standard error (SE) than other methods, suggesting that it can be more efficient in estimation.

8 Discussion

In this paper, we propose to use over-parameterized neural networks in the partially linear model (PLM-NN). We show that PLM-NN is computationally efficient and provably converges to global minimum exponentially fast. We demonstrate its robustness to high dimension and high sparsity, and its strong performance in terms of the estimation and prediction errors. Based on this new PLM, we design DebiNet, a debiasing framework for arbitrary feature selection method and illustrate its potential to infer and to predict accurately (e.g. in adversarial attack in Appendix D.4). Now we discuss some future directions.

As also discussed in [15, 14], our framework can easily extend to deep over-parameterized neural networks: for L-hidden-layer neural network with equally wide l-th layer $\{\boldsymbol{w}_r^{(l)}\}_{r\in[m]}$, it has been shown that the last hidden layer's kernel (similar to the notation in (4.4)): $\sum_{r=1}^m \langle \frac{\partial F_s(\mathbf{W},\mathbf{A},\mathbf{Z}_i)}{\partial \mathbf{w}_r^{(L)}}, \frac{\partial F_h(\mathbf{W},\mathbf{A},\mathbf{Z}_j)}{\partial \mathbf{w}_r^{(L)}} \rangle$ is positive definite, and so is the overall NTK with respect to all weights. Hence we can guarantee linear convergence for deep over-parameterized neural networks. In addition, the width requirement of the hidden layer can be tightened by more advanced matrix perturbation theory. Based on our experiments in Appendix E, we may also use other activation functions, optimizers (e.g. Adam [27]) and losses (e.g. Huber loss [25]) and still observe the linear convergence. We note that PLM-NN does not perform well when $\mathbf{y}$ or $\mathbf{D}$ is discrete, or when $\mathbf{y}$ and $\mathbf{D}$ have different scales. It would be desirable to robustify the model against the input distributional assumption, e.g. using the generalized partially linear model within DebiNet.

Algorithm 3 Double/Debiased Machine Learning (DML)

Input: Data matrix $[\mathbf{D}, \mathbf{Z}]$, label $\mathbf{y}$
for $j \in [K]$ **do**
1. fit $\mathbf{y}^C_{I_j} \sim \mathbf{Z}^C_{I_j}$ via some machine learning method to learn $\mathbb{E}(\mathbf{y}|\mathbf{Z})$;
2. fit $\mathbf{D}^C_{I_j} \sim \mathbf{Z}^C_{I_j}$ via some machine learning method to learn $\mathbb{E}(\mathbf{D}|\mathbf{Z})$;
3. fit $\mathbf{y}_{I_j} - \mathbb{E}(\mathbf{y}_{I_j}|\mathbf{Z}_{I_j}) \sim \mathbf{D}_{I_j} - \mathbb{E}(\mathbf{D}_{I_j}|\mathbf{Z}_{I_j})$ via OLS to derive $\hat{\boldsymbol{\beta}}$, denoted as $\hat{\boldsymbol{\beta}}^{(j)}$;
end for
4. aggregate the estimators: $\hat{\boldsymbol{\beta}} = \sum_j \hat{\boldsymbol{\beta}}^{(j)}/K$.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. "Learning and generalization in overparameterized neural networks, going beyond two layers". In: *Advances in neural information processing systems*. 2019, pp. 6155–6166.

[2] Sanjeev Arora et al. "Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks". In: *arXiv preprint arXiv:1901.08584* (2019).

[3] Francis Bach. "Breaking the curse of dimensionality with convex neural networks". In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 629–681.

[4] Alexandre Belloni, Victor Chernozhukov, et al. "Least squares after model selection in high-dimensional sparse models". In: *Bernoulli* 19.2 (2013), pp. 521–547.

[5] Richard Berk et al. "Valid post-selection inference". In: *The Annals of Statistics* 41.2 (2013), pp. 802–837.

[6] Małgorzata Bogdan et al. "SLOPE—adaptive variable selection via convex optimization". In: *The annals of applied statistics* 9.3 (2015), p. 1103.

[7] Yuan Cao and Quanquan Gu. "Generalization bounds of stochastic gradient descent for wide and deep neural networks". In: *Advances in Neural Information Processing Systems*. 2019, pp. 10835–10845.

[8] Hung Chen et al. "Convergence rates for parametric components in a partly linear model". In: *The Annals of Statistics* 16.1 (1988), pp. 136–146.

[9] Victor Chernozhukov, Christian Hansen, and Martin Spindler. "Post-selection and post-regularization inference in linear models with many controls and instruments". In: *American Economic Review* 105.5 (2015), pp. 486–90.

[10] Victor Chernozhukov et al. *Double/debiased machine learning for treatment and structural parameters*. 2018.

[11] David Donoho and Jared Tanner. "Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 367.1906 (2009), pp. 4273–4293.

[12] David L Donoho. "High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension". In: *Discrete & Computational Geometry* 35.4 (2006), pp. 617–652.

[13] David L Donoho. "Neighborly polytopes and sparse solutions of underdetermined linear equations". In: (2005).

[14] Simon S Du et al. "Gradient descent finds global minima of deep neural networks". In: *arXiv preprint arXiv:1811.03804* (2018).

[15] Simon S Du et al. "Gradient descent provably optimizes over-parameterized neural networks". In: *arXiv preprint arXiv:1810.02054* (2018).

[16] MA Efroymson. "Multiple regression analysis". In: *Mathematical methods for digital computers* (1960), pp. 191–203.

[17] Robert F Engle et al. "Semiparametric estimates of the relation between weather and electricity sales". In: *Journal of the American statistical Association* 81.394 (1986), pp. 310–320.

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. "A note on the group lasso and a sparse group lasso". In: *arXiv preprint arXiv:1001.0736* (2010).

[19] G Gybenko. "Approximation by superposition of sigmoidal functions". In: *Mathematics of Control, Signals and Systems* 2.4 (1989), pp. 303–314.

[20] Scott A Hamilton and Young K Truong. "Local linear estimation in partly linear models". In: *Journal of Multivariate Analysis* 60.1 (1997), pp. 1–19.

[21] Chris Hans. "Bayesian lasso regression". In: *Biometrika* 96.4 (2009), pp. 835–845.

[22] Wolfgang Härdle, Hua Liang, and Jiti Gao. *Partially linear models*. Springer Science & Business Media, 2012.

[23] Nancy E Heckman. "Spline smoothing in a partly linear model". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 48.2 (1986), pp. 244–248.

[24] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural networks* 4.2 (1991), pp. 251–257.

[25] Peter J Huber. "Robust estimation of a location parameter". In: *Breakthroughs in statistics*. Springer, 1992, pp. 492–518.

[26] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems*. 2018, pp. 8571–8580.

[27] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[28] Jaehoon Lee et al. "Wide neural networks of any depth evolve as linear models under gradient descent". In: *Advances in neural information processing systems*. 2019, pp. 8570–8581.

[29] Jason D Lee et al. "Exact post-selection inference, with application to the lasso". In: *The Annals of Statistics* 44.3 (2016), pp. 907–927.

[30] Moshe Leshno et al. "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural networks* 6.6 (1993), pp. 861–867.

[31] Qi Li. "On the root-n-consistent semiparametric estimation of partially linear models". In: *Economics Letters* 51.3 (1996), pp. 277–285.

[32] Sai Li. "Debiasing the Debiased Lasso with Bootstrap". In: *arXiv preprint arXiv:1711.03613* (2017).

[33] Hua Liang, Wolfgang Härdle, Raymond J Carroll, et al. "Estimation in a semiparametric partially linear errors-in-variables model". In: *The Annals of Statistics* 27.5 (1999), pp. 1519–1535.

[34] Behnam Neyshabur et al. "The role of overparametrization in generalization of neural networks". In: *7th International Conference on Learning Representations, ICLR 2019*. 2019.

[35] Roman Novak et al. "Sensitivity and generalization in neural networks: an empirical study". In: *arXiv preprint arXiv:1802.08760* (2018).

[36] Daniel S Park et al. "The effect of network width on stochastic gradient descent and generalization: an empirical study". In: *arXiv preprint arXiv:1905.03776* (2019).

[37] Trevor Park and George Casella. "The bayesian lasso". In: *Journal of the American Statistical Association* 103.482 (2008), pp. 681–686.

[38] James M Poterba and Steven F Venti. "401 (k) plans and tax-deferred saving". In: *Studies in the Economics of Aging*. University of Chicago Press, 1994, pp. 105–142.

[39] James M Poterba, Steven F Venti, and David A Wise. "Do 401 (k) contributions crowd out other personal saving?" In: *Journal of Public Economics* 58.1 (1995), pp. 1–32.

[40] Peter M Robinson. "Root-N-consistent semiparametric regression". In: *Econometrica: Journal of the Econometric Society* (1988), pp. 931–954.

[41] Noah Simon et al. "A sparse-group lasso". In: *Journal of computational and graphical statistics* 22.2 (2013), pp. 231–245.

[42] Paul Speckman. "Kernel smoothing in partial linear models". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 50.3 (1988), pp. 413–436.

[43] Maxwell Stinchcombe and Halbert White. "Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions". In: *IJCNN International Joint Conference on Neural Networks*. 1989.

[44] Weijie Su, Małgorzata Bogdan, Emmanuel Candes, et al. "False discoveries occur early on the lasso path". In: *The Annals of statistics* 45.5 (2017), pp. 2133–2150.

[45] Jonathan Taylor and Robert J Tibshirani. "Statistical learning and selective inference". In: *Proceedings of the National Academy of Sciences* 112.25 (2015), pp. 7629–7634.

[46] Jonathan Taylor et al. "Post-selection adaptive inference for least angle regression and the lasso". In: *arXiv preprint arXiv:1401.3889* 354 (2014).

[47] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288.

[48] Ryan J Tibshirani et al. "Exact post-selection inference for sequential regression procedures". In: *Journal of the American Statistical Association* 111.514 (2016), pp. 600–620.

[49] Sara Van de Geer et al. "On asymptotically optimal confidence regions and tests for high-dimensional models". In: *The Annals of Statistics* 42.3 (2014), pp. 1166–1202.

[50] Halbert White. "Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings". In: *Neural networks* 3.5 (1990), pp. 535–549.

[51] Cun-Hui Zhang and Stephanie S Zhang. "Confidence intervals for low dimensional parameters in high dimensional linear models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76.1 (2014), pp. 217–242.

[52] Hui Zou. "The adaptive lasso and its oracle properties". In: *Journal of the American statistical association* 101.476 (2006), pp. 1418–1429.

[53] Hui Zou and Trevor Hastie. "Regularization and variable selection via the elastic net". In: *Journal of the royal statistical society: series B (statistical methodology)* 67.2 (2005), pp. 301–320.