A NEURAL NETWORKS BASED METHOD WITH GENETIC DATA ANALYSIS OF COMPLEX DISEASES

By

Jinghang Lin

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Statistics — Doctor of Philosophy

2021

ABSTRACT

A NEURAL NETWORKS BASED METHOD WITH GENETIC DATA ANALYSIS OF COMPLEX DISEASES

By

Jinghang Lin

The genetic etiologies of common diseases are highly complex and heterogeneous. Classic statistical methods, such as linear regression, have successfully identified numerous genetic variants associated with complex diseases. Nonetheless, for most complex diseases, the identified variants only account for a small proportion of heritability. Challenges remain to discover additional variants contributing to complex diseases. In this dissertation, we developed an expectile neural network (ENN) method and applied the method to genetic data analysis. ENN provides a comprehensive view of relationships between genetic variants and disease phenotypes and can be used to discover genetic variants predisposing to subpopulations (e.g., high-risk groups). We integrate the idea of neural networks into ENN, making it capable of capturing non-linear and non-additive genetic effects (e.g., gene-gene interactions). Through simulations, we showed that the proposed method outperformed an existing expectile regression when there exist complex relationships between genetic variants and disease phenotypes. We also applied the proposed method to the genetic data from the Study of Addiction: Genetics and Environment(SAGE), investigating the relationships of candidate genes with smoking quantity. Neural networks have been widely used in applications. However, few studies have been focused on the statistical properties of neural networks. We further investigate the Asymptotic properties of ENN (e.g., consistency). Simulations have been conducted to test the validity of the theory.

I dedicate this dissertation to my parents, Xianghua Chen and Xilong Lin for their endless love and support.

ACKNOWLEDGMENTS

There are many people who helped me along the way on this journey. Without their help, I could not complete this dissertation. I want to take a moment to thank them.

First of all, I would like to express my sincere gratitude to my advisors Dr. Qing Lu and Dr. Yuehua Cui for their invaluable advice, continuous support, and patience during my PhD study. Their immense knowledge and plentiful experience have encouraged me in all the time of my academic research. They lead me into the area of statistical genetics and train me to be an independent researcher.

I would also like to extend my sincere thanks to my dissertation committee members, Dr. Hyokyoung (Grace) Hong and Dr. Haolei Weng. Their comments and suggestions are beneficial to my research. My special thanks to Dr. Guowei Wei for his help in my job search. I am deeply grateful to Dr. Xiaoxi Shen and Dr. Xiaoran Tong for their insight in theory and computational support.

During my PhD study, I made a lot of friends. My special thanks to my friends: Steven Gagnon, Tengfei Ma, Peide Li, Zihuan Liu, Dr. Cheuk (Ken) Lee for their constant help in my life and study. I would like to express my sincere gratitude to group members in Dr. Qing Lu's group: Shan Zhang, Chang Jiang, Yuan Zhou, Tingting Hou, Mingsheng Tang for creating a positive research atmosphere. My special thanks to my girlfriend Dr. Liping Sun for her love and accompany.

Last but not least, I would like to express my sincere thanks to my parents for their support and endless love for me. I would never make this journey without them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Overview

With the development of biotechnology, especially next-generation sequencing technologies (NGS), it is easy to sequence an entire human genome. New technologies arising from the Human Genome Project and HapMap Project have generated a surge of methodological development for unsolved problems in human genetics. To find genetic variations associated with a particular disease, a genome-wide association study (GWAS) that involves rapidly scanning markers across the complete sets of DNA, or genomes, can be adopted[1]. GWAS investigates the entire genome and identify SNPs and other variants in DNA associated with a disease, but they cannot infer which genes are causal. Once new genetic associations are identified, researchers can use the information to understand, treat and prevent the disease[2]. Successful GWAS has been conducted to identify genetic variations that contribute to the risk of type 2 diabetes, Parkinson's disease, heart disorders, obesity, Crohn's disease and prostate cancer, as well as genetic variations that influence response to anti-depressant medications[5; 6]. Such research lays the groundwork for personalized medicine.

Based on prior knowledge of a gene's biological function on the trait or disease, candidate genes are most often studied in risk prediction research[3]. In risk prediction research, we are interested in developing a new genetic risk prediction model to identify the high-risk individuals for certain diseases. If we could predict high-risk individuals at the early stage, targeted screening and appropriate intervention methods can be used to reduce mortality and morbidity[4]. However, there are tremendous analytic and computational challenges when we implement a risk prediction model. Genetic data is high-dimensional. For example, there are millions of single nucleotide polymorphisms (SNP), and the signal-to-noise ratio of genetic data is quite low, which makes us hard to capture underlying genetic effects. Moreover, the study sample is massive (e.g., a million samples in the UK Biobank), which brings the computational issue.

In this chapter, we will first review some basic knowledge of human genetics in section 1.2. In section 1.3, we will briefly introduce the neural network and its application in healthcare. We give the overall organization of this dissertation in section 1.4.

1.2 A review of basic human genetics

In the human genome, the genetic material is stored on chromosomes in the nucleus of the cell. There are 23 pairs of chromosomes in the human genome: 22 pairs of them are autosomal and the 23rd pair is the sex chromosomes. For the sex chromosomes, males have one X and Y, while females have two non-identical copies of the X chromosome. Each chromosome is composed of long strands of deoxyribonucleic acid (DNA), which determines how proteins are manufactured in the human body.

Genes are segments of DNA that code for specific proteins that function in one or more types of cells in the body. These proteins control how our body grows and works; they are also responsible for many of our characteristics, such as our eye color, blood type or height. Genes are the basic physical units of inheritance, which are passed from parents to offspring and contain the information needed to specify traits. Most parts of DNA are the same in all people, but a small proportion of DNA (less than 1 percent of the total DNA) are different between people. These differences contribute to each person's unique physical features. An allele is one of two or more versions of a gene. An individual inherits two alleles, one from each parent.

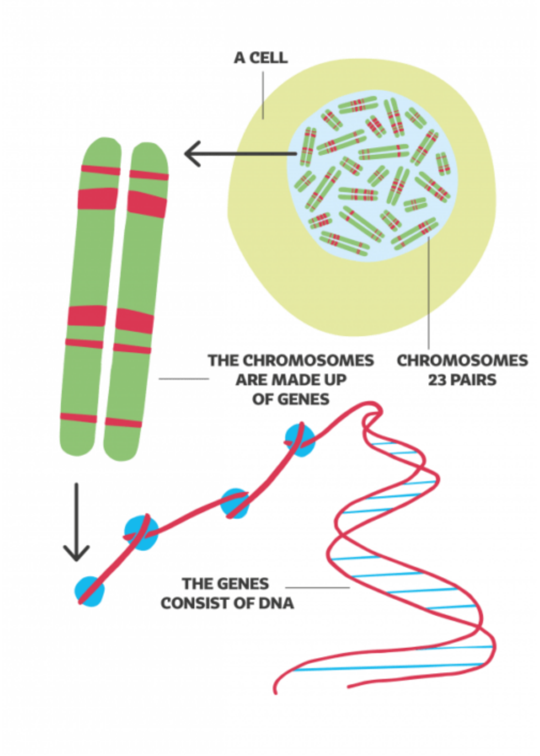

Figure 1.1: A graphical representation of Chromosome, DNA and gene. Credit to Genetic Alliance UK

SNPs are the most common type of genetic variation among people, which are typically coded as the number of minor frequent alleles (e.g., AA=2, Aa=1, aa=0). A trait is any gene-determined characteristic and is often determined by more than one gene. The genotypes for the traits are often not observable and should be inferred from linked markers. In statistical genetics, we intend to construct a statistical model that connects genotypes and phenotypes[7].

1.3 Statistical learning

We give a brief introduction of the statistical learning framework. Suppose $\mathcal{X}$ stands for the vector space of input and $\mathcal{Y}$ for the vector space of output. In statistical learning, we assume that there is an underlying unknown probability distribution over the product space $Z = X \times Y$. The training set $\mathcal{D} = \{(\mathbf{x_1}, y_1), ..., (\mathbf{x_n}, y_n)\}$ comprise of n samples from the probability distribution. The goal of statistical learning is to find the unknown function $f : \mathcal{X} \to \mathcal{Y}$ from the data $\mathcal{D}$.

We start with a set of candidate hypotheses $\mathcal{H} = \{h_1, h_2..., \}$, which are likely to represent f. The hypothesis space is the space of functions that the algorithm will search through. We want to select a hypothesis f from $\mathcal{H}$. The way we do this is called a learning algorithm. Let $L(f(\mathbf{x}, y)$ be the loss function that is a metric of the difference between the predicted value $f(\mathbf{x})$ and the observed value y. The problem of statistical learning is to minimize the expected risk:

$$R(f) = \int L(f(\mathbf{x}, y) dF(\mathbf{x}, y).$$

Since the probability distribution $F(\mathbf{x}, y)$ is unknown, a proxy measure for the expected risk must be used. We try to minimize empirical risk:

$$R_{emp}(f) = \frac{1}{n} \sum_{i=1}^{n} L(f(\mathbf{x_i}, y_i).$$

1.3.1 Neural network

The basis of the biological neural networks is the nerve cells, which is composed of a cell body, a dendrite and an axon. At the high-level view, incoming stimuli are transmitted to the cell body via dendrites. Outputs generated after operations in the cell body are transmitted to other nerve cells via axons. In the neural network model, it imitates the functioning of the human brain. The biological nervous system in the human body consists of a three-layered structure that includes receiving data, interpreting them, and making decisions. A neuron model is composed of three layers: input layer, hidden layer and output layer.

Here we give a graphical representation of similarity between biological and artificial neural networks with one hidden layer in Figure 1.2[58]. $x_1, ..., x_m$ are input units, which mimic the dendrites of a neuron. Σ is a computation unit, which is involved in the same role in the cell body. The computation unit is the most important part in neural networks, which is the linear combination of inputs units and bias and then apply the activation function. The number of computation units and the type of activation function are crucial in building models.

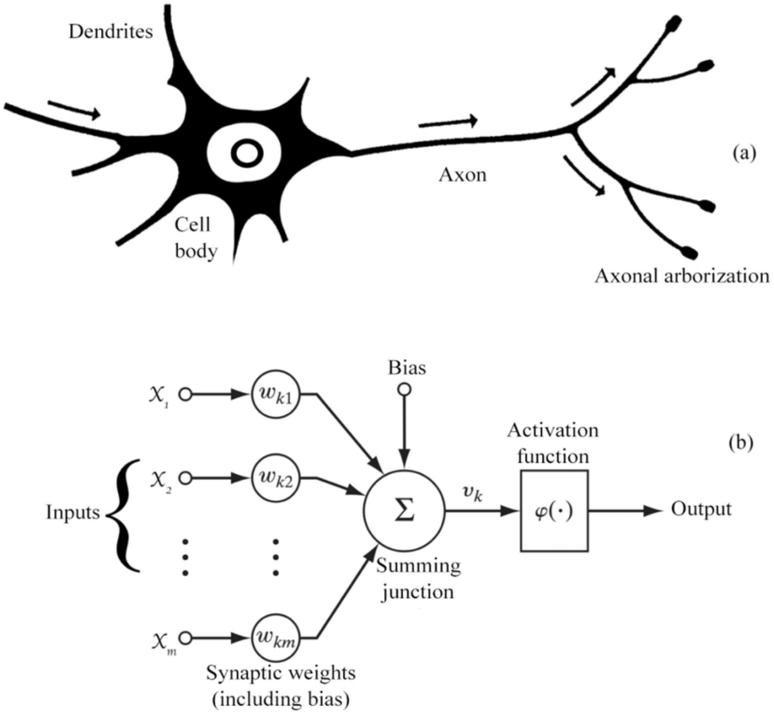

Figure 1.2: Similarity between biological and artificial neural networks

Common activation functions of neural networks used in perceptrons and neural networks are

- Rectified Linear Unit (ReLU):

$$\sigma(x) = x_+ = \max\{x, 0\},$$

- Standard Sigmoid:

$$\sigma(x) = (1 + e^{-x})^{-1},$$

- Hyperbolic Tangent (Tanh):

$$\sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

The output layer consists of a single layer, where the generated data are transmitted to the outside world. This is analogous to the axon of a neuron.

Neural networks with multiple hidden layers are called deep neural networks. Deep neural networks contain multiple non-linear hidden layers and this enables them learn very complicated relationships between inputs and outputs. For most data sets, neural networks with one hidden layer are enough to build a decent model. While various theoretical perspectives have been developed to explain why deep learning is successful, the general consensus of the community is to attribute the success to the joint forces of straightforward neural modeling, simple learning techniques, the availability of big data and the hardware revolution in high-performance computing[40]. Deep neural networks are a powerful tool in analyzing a large dataset. However, overfitting is a serious issue in deep neural networks. Dropout is a technique to address the overfitting problem. Dropout randomly drops units (along with their connections) from the neural network during training[39]. By reducing the number of parameters, the model performance of a deep neural network can be improved.

Deep learning has been implemented in many software frameworks, such as Tensorflow and Pytorch[62; 63]. Those frameworks offer building blocks for designing, training and validating deep neural networks, through a high-level programming language, like Python. They also provide a clear and concise way to simplify the implementation of complex and large-scale deep learning models by using a collection of pre-built and optimized components.

It is worthwhile to mention a well-known result of neural networks: the universal approximation theorem. A neural network with one hidden layer could approximate any continuous function[61].

Theorem 1.3.1 (Universal Approximation Theorem)**.** *For every continuous function* $f : [a, b]^d \to \mathbb{R}$ *and for every* $\epsilon > 0$*, there exists a neural network with one hidden layer* $\psi(\mathbf{x})$ *such that*

$$\sup_{\mathbf{x}\in[a,b]^d} |f(\mathbf{x}) - \psi(\mathbf{x})| < \epsilon.$$

1.3.2 Artificial intelligence in healthcare

Deep learning or AI has been applied to many applications, such as natural language processing and computer vision. AI also holds great promise for healthcare. With the development of biotechnology, healthcare data has a large size and complexity that traditional data management tools cannot store or process it efficiently. Many successful AI applications in healthcare have been conducted. For example, AI can be used to optimize the care trajectory of chronic disease patients, suggest precision therapies for complex illnesses, reduce medical errors, and improve subject enrollment into clinical trials[8]. Fakoor et al. showed that how unsupervised feature learning can be used for cancer detection and cancer type analysis from gene expression data[9]. Krittanawong et al. gave a glimpse of AI's application in cardiovascular clinical care and discussed its potential role in facilitating precision cardiovascular medicine[12]. Pham et al used a deep learning approach to read medical records, store previous illness history, infer current illness states and predict future medical outcomes[59]. Plis et al. applied deep learning methods to learn physiologically important representations and detect latent relations in neuroimaging data[60]. There is a great promise that the applications of AI can provide substantial improvement in all areas of healthcare from diagnostics to treatments. Although there are many instances in which AI can perform healthcare tasks better than humans, implementation will prevent large-scale automation of healthcare professional jobs for a considerable period[76]. However, AI will not take over the jobs which require unique human skills such as empathy and persuasion.

1.4 Organization

The dissertation is organized as follows. In chapter 2, we develop a neural-network-based method called expectile neural networks. In chapter 3, The asymptotic properties of ENN are discussed. In chapter 4, we summarize this dissertation and discuss some potential future work.

Chapter 2

Expectile Neural Networks for Genetic Data Analysis of Complex Diseases

2.1 Overview

The genetic etiologies of common diseases are highly complex and heterogeneous. Classic statistical methods, such as linear regression, have successfully identified numerous genetic variants associated with complex diseases. Nonetheless, for most complex diseases, the identified variants only account for a small proportion of heritability. Challenges remain to discover additional variants contributing to complex diseases. Expectile regression is a generalization of linear regression and provides completed information on the conditional distribution of a phenotype of interest. While expectile regression has many nice properties and holds great promise for genetic data analyses (e.g., investigating genetic variants predisposing to a high-risk population), it has been rarely used in genetic research. In this chapter, we develop an expectile neural network (ENN) method for genetic data analyses of complex diseases. Similar to expectile regression, ENN provides a comprehensive view of relationships between genetic variants and disease phenotypes and can be used to discover genetic variants predisposing to sub-populations (e.g., high-risk groups). We further integrate the idea of neural networks into ENN, making it capable of capturing non-linear and non-additive genetic effects (e.g., gene-gene interactions). Through simulations, we showed that the proposed method outperformed an existing expectile regression when there exist complex relationships between genetic variants and disease phenotypes. We also applied the proposed method to the genetic data from the Study of Addiction: Genetics and Environment(SAGE), investigating the relationships of candidate genes with smoking quantity.

2.2 Introduction

Converging evidence suggests that the genetic etiologies of complex diseases are highly heterogeneous [13; 14] and various genetic factors and environmental determinants could play different roles in subgroups of the population. Linear regression has been commonly used in genetic studies to investigate the effects of genetic variants on the mean of a continuous phenotype. However, if we are interested in a complete view of genetic effects across the entire distribution of phenotypes or are interested in investigating genetic contribution to a sub-population(e.g., a high-risk population), quantile regression and expectile regression are great alternative choices [15; 16]. Quantile regression generalizes median regression and has been widely used in fields such as economics [17], medicine [18; 19] and environmental science [20] to study entire conditional distributions of responses given covariates. While quantile regression has many good properties (e.g., being robust to distribution assumption and outlies), as pointed out by Newey and Powell [16], quantile regression has several limitations. First, quantile regression uses the check function with the absolute least error as loss function, which is not continuously differentiable and is computationally difficult for parameter estimation. Second, quantile regression is relatively inefficient for error distributions that are close to Gaussian or have low densities at the corresponding percentile. Third, it is challenging to estimate the density function values of quantile regression.

To address these issues, Newey and Powell [16] proposes expectile regression, which uses the sum of asymmetric residual squares as the loss function. Since the loss function is convex and differentiable, expectile regression has a computational advantage over quantile regression. Similar to quantile regression, expectile regression makes no assumption on error distribution (e.g., homoscedasticity) and can be used to study the entire distribution of the responses. Expectile regression can be viewed as a generalization of linear regression. A typical expectile regression assumes a linear relationship between the expectile and the covariates, which may not be suitable for genetic data analysis as genetic variants likely influence phenotypes in a complicated manner (e.g., through interactions) [21]. Simply considering linear and additive genetic effects can't fully take this complexity into account.

In this chapter, we integrate the idea of neural networks into expectile regression and develop an expectile neural network (ENN) method to model the complex relationship between genotypes and phenotypes. While several methods have been developed to integrate neural networks into quantile regression[22; 23; 24], few studies have been focused on investigating nonlinear expectile regressions, especially using neural networks. Compared to quantile regression neural networks(QRNN), ENN has several advantages. The empirical loss function in ENN is differentiable everywhere. Moreover, ENN can detect the heteroscedasticity in the data since ENN is more sensitive to extreme values than QRNN[25; 26; 27; 28; 29].

The rest of the chapter is organized as follows: in Section 2, we review expectile regression and propose an ENN method. We then give an inequality that bounds the integrated squared error of an expectile function estimator in terms of risk functions. The proof of inequality is detailed in the Appendix. Simulations were conducted in Section 3 to evaluate the performance of the new method. In Section 4, we applied ENN to the SAGE data, studying genetic contribution to smoking quantity. We provide the summary and concluding remarks in Section 5.

2.3 Method

In this section, we briefly introduce expectile regression and then propose an expectile neural network. Suppose we have n samples,$\{(\mathbf{x_i}, y_i), i = 1, ..., n\}$, where $\mathbf{x_i} = (1, x_{i,1}, ..., x_{i,p})^T$ and y_i denote a $p-$dimensional covarites and the response for the ith sample, respectively. In this chapter, the covariates are primarily genetic variants, such as single nucleotide polymorphisms (SNPs), which are typically coded as the number of minor frequent allele (e.g., AA=2, Aa=1, aa=0). The covariates $\mathbf{x_i}$ can also include personal characteristics (e.g., gender) and environmental determinants. The response y_i is the set of observable characteristics of an individual in genetics. For example, y_i could be the type of diabetes, or the height of an individual. By building models between $\mathbf{x_i}$ and y_i, we tend to explore the relationship of candidate genes and certain disease.

2.3.1 Expectile regression

Given the data, linear regression is commonly used to model the relationship between the covariates and the mean response. However, if we want to explore a complete relationship between the covarites and the response (e.g., genetic contribution to a high-risk population), an expectile regression can be used. To simplify the notation, we denote expectile regression as ER. The expectile regression for the $\tau-$expectile can be expressed as,

$$\text{Expectile}(\tau) = \mathbf{x}^T\hat{\beta}, \tag{2.1}$$

where $\hat{\beta}$ is the estimator of coefficients $\beta = (\beta_0, \beta_1, ..., \beta_p)^T$. The expectile is also closely related to two commonly used measures in mathematical finance, value at risk and expected shortfall. The regression parameters, $\hat{\beta}$, can be obtained by minimizing an asymmetric L_2 loss function,

$$\mathcal{R}_{L_\tau}(\beta;\tau) = \frac{1}{n}\sum_{i=1}^{n} L_\tau(y_i, \mathbf{x_i}^T\beta), 0 < \tau < 1, \tag{2.2}$$

where $L_\tau(\cdot)$ is asymmetric squared loss with convex form

$$L(y_i, \mathbf{x_i}^T\beta) = \begin{cases} (1-\tau)(y_i - \mathbf{x_i}^T\beta)^2, & if\ y_i < \mathbf{x_i}^T\beta \\ \tau(y_i - \mathbf{x_i}^T\beta)^2, & if\ y_i \geq \mathbf{x_i}^T\beta. \end{cases} \tag{2.3}$$

Minimizing asymmetically weighted sums of squared errors yields the the expectiles. If we minimize sums of asymmetrically weighted absolute errors, the estimators are quantiles. In contrast to the quantiles, expectiles have a more global dependence on the form of the distribution. Shifting mass in the lower tail of a distribution has no impact on the quantiles of the upper tail, but it will affect all expectiles. We cite the Figure 2.1 to show the relationship between quantiles and expectiles[54].

For a model with a large p, a penalty term can be added to the risk function to reduce the model complexity,

$$\mathcal{R}_{L_\tau}(\beta;\tau) = \frac{1}{n}\sum_{i=1}^{n} L_\tau(y_i - \mathbf{x_i}^T\beta) + \lambda\sum_{i=1}^{p}\beta_i^2. \tag{2.4}$$

τ is a hyperparameter between 0 and 1. By tuning τ, we could get different conditional distributions of responses which is similiar to quantile regression. However, quantile regression uses asymmetric absolute value function. When $\tau = 0.5$, the corresponding expectile regression degenerates to a standard linear regression. Therefore, expectile regression can also be viewed as a generalization of linear regression. Quantile regression can be seen as a generalization of median regression, expectiles as alternative are a generalized form of mean regression.

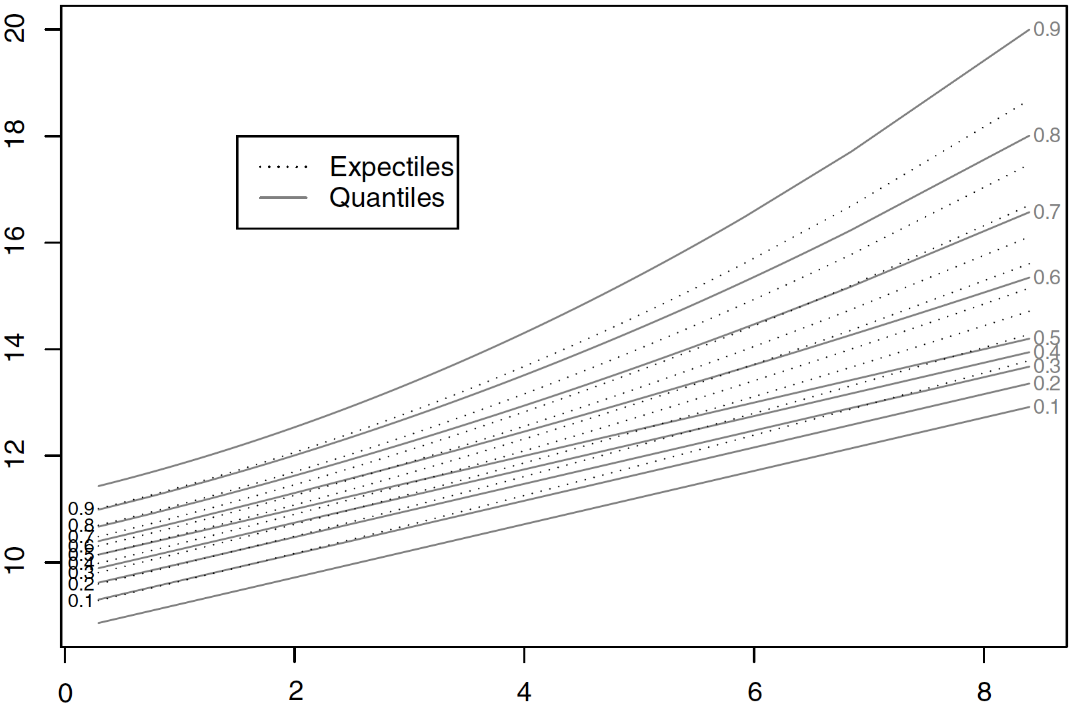

Figure 2.1: Quantiles and expectiles.

2.3.2 Expectile neural network

A typical expectile regression model focuses on linear relationships between covariates and responses. In reality, the underlying relationship could be non-linear and involve complicated interactions among covariates. In order to model complex relationships between covariates and responses, we integrate the idea of neural networks into expectile regression and propose an ENN method. Neural network is a powerful nonlinear approximator. For every continuous function, neural network with one hidden layer could approximate it well[33]. We don't assume a particular functional form of covariates and use neural networks to approximate the underlying expectile regression function. ENN can be considered as a nonparametric expectile regression or neural networks with asymmetric L_2 loss function, We illustrate ENN with one hidden layer. The method can be easily extended to an expectile regression deep neural network with multiple layers.

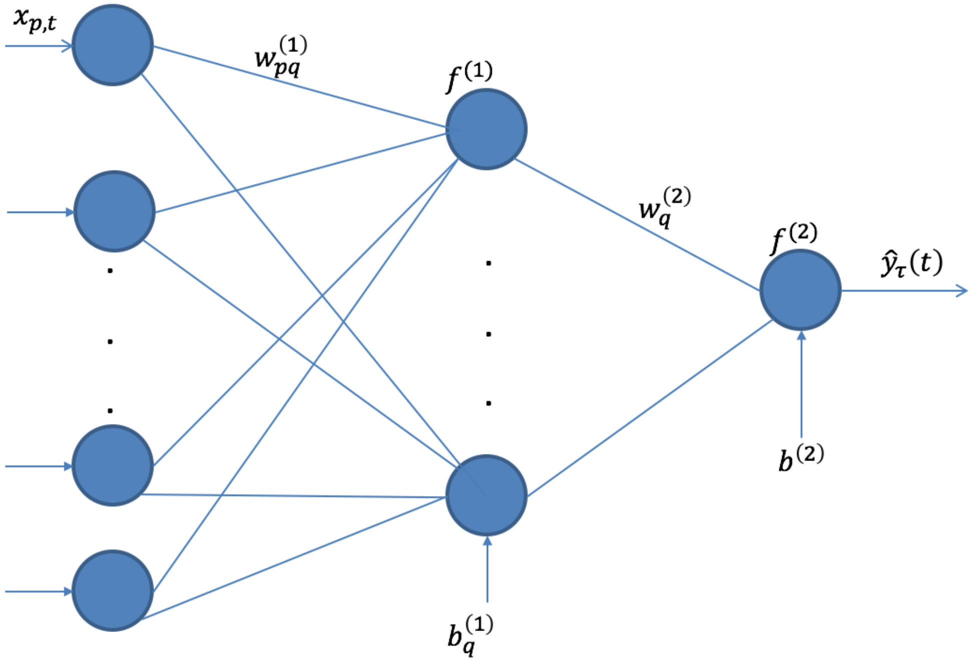

Figure 2.2: A graphical representation of expectile neural network

Given the covariates $\mathbf{x}_t$, we first build the hidden nodes $h_{q,t}$,

$$h_{q,t} = f^{(1)}(\sum_{p=1}^{P} x_{p,t} w_{pq}^{(1)} + b_q^{(1)}), q = 1, ..., Q, t = 1, ..., n, \tag{2.5}$$

where Q is the number of nodes in the first hidden layer, w_{pq} denotes weights and b_q denotes the bias; $f^{(1)}$ is the activation function for the hidden layer that can be a sigmoid function, a hyperbolic tangent function, or a rectified linear units(ReLU) function. Similar to hidden nodes in neural networks, the hidden nodes in ENN can learn complex features from covariates $\mathbf{x}$, which makes ENN capable of modelling non-linear and non-additive effects. Based on these hidden nodes, we can model the conditional τ-expectile, $\hat{y}_{\mathcal{T}}(t)$,

$$\hat{y}_{\mathcal{T}}(t) = f^{(2)}(\sum_{q=1}^{Q} h_{q,t} w_q^{(2)} + b^{(2)}), \tag{2.6}$$

where $f^{(2)}$, $w_q^{(2)}$, and $b^{(2)}$ are the activation function, weights, and bias in the output layer, respectively. $f^{(2)}$ can be an identity function, a sigmoid function, or a rectified linear units(ReLU) function. To illustrate the structure of ENN, a graphical representation of ENN is given in Figure 2.1.

From equations (2.5) and (2.6), we can have the ENN model:

$$\hat{y}_{\mathcal{T}}(t) = f^{(2)}(\sum_{q=1}^{Q} f^{(1)}(\sum_{p=1}^{P} x_{p,t} w_{pq}^{(1)} + b_q^{(1)}) w_q^{(2)} + b^{(2)}). \tag{2.7}$$

If we choose $\tau = 0$, $f^{(1)}$ and $f^{(2)}$ as identity function, ENN is reduced to linear regression. To estimate $w_{pq}^{(1)}, b_q^{(1)}, w_q^{(2)}, b^{(2)}$, we minimize the empirical risk function

$$\mathcal{R}(\tau) = \frac{1}{n}\sum_{i=1}^{n} L_\tau(y_i, f(\mathbf{x_i})), \tag{2.8}$$

where

$$L_\tau(y_i, f(\mathbf{x_i})) = \begin{cases} (1-\tau)(y_i - f(\mathbf{x_i}))^2, & if\ y_i < f(\mathbf{x_i}) \\ \tau(y_i - f(\mathbf{x_i})))^2, & if\ y_i \geq f(\mathbf{x_i}). \end{cases} \tag{2.9}$$

The model tends to be overfitted with the increasing number of covariates. To address the overfitting issue, a L_2 penalty is added to the risk function,

$$\mathcal{R}(\tau) = \frac{1}{n}\sum_{i=1}^{n} L_\tau(y_i, f(\mathbf{x_i})) + \lambda \sum_{p=1}^{P}\sum_{q=1}^{Q}(w_{pq}^{(1)})^2 + (w_q^{(2)})^2. \tag{2.10}$$

The loss function for ENN is differentiable everywhere which gives us computation advantage. Even though ENN is differentiable, it is not easy to get exact estimator like linear regression because of the existence of indicator function. We can obtain the estimator of ENN by using gradient-based optimization algorithms (e.g., quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm). In numerical optimization, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm is an iterative method for solving unconstrained nonlinear optimization problems[34].

2.3.3 Theoretical result

Intuitively, if we fix τ, the upper and lower bound of $\tau-$expectile is related to risk function. To illustrate well, some notations are changed. We give one theoretical result which shows that upper bound and lower bound of error of $\tau-$expectile are bounded by risk function $\mathcal{R}_{L_\tau,P}(f)$. In ENN, $\tau-$expectiles $f^*_{L_\tau,P}$ can be estimated by minimizing the asymmetric least squares (ALS) loss,

$$\mathcal{R}^*_{L_\tau,P} = inf\{\mathcal{R}_{L_\tau,P}(f) = \int_{X\times Y} L_\tau(y, f(x))dP(x,y)|f : X \to \mathbb{R} \text{ measurable}\},$$

where P is the distribution on $X \times Y$ and $f : X \to \mathbb{R}$ is some predictor. The following theorem describe the upper bound and lower bound of error of $f^*_{L_\tau,P}$.

Theorem 2.3.1. *Let L_τ be the ALS loss function and P be the distribution on $X \times Y$. We further assume that $f^*_{L_\tau,P} < \infty$ is the $\tau-$expectile for fixed $\tau \in (0,1)$. Then, for an arbitrary neural network function f, we have*

$$C_\tau^{-1/2}(\mathcal{R}_{L_\tau,P}(f) - \mathcal{R}^*_{L_\tau,P})^{1/2} \leq ||f - f^*_{L_\tau,P}||_{L_2(P_\mathbf{X})} \leq c_\tau^{-1/2}(\mathcal{R}_{L_\tau,P}(f) - \mathcal{R}^*_{L_\tau,P})^{1/2},$$

where $c_\tau = min\{\tau, 1-\tau\}$, $C_\tau = max\{\tau, 1-\tau\}$.

Proof of this theorem can be found in the appendix of the chapter.

2.4 Simulation

Simulation studies were conducted to compare the performance of ENN and ER under different settings. The genetic data used in the simulation is the real sequencing data from the 1000 Genomes Project, located on Chromosome $17 : 7344328 - 8344327$ [30]. Totally 1000 replicates were simulated for each simulation setting. In each replicate, we randomly selected a number of samples and SNPs from the 1000 Genomes Project based on the simulation settings. Given the genotypes, we further simulated the phenotype by using different linear/non-linear functions or by assuming different types of interactions among SNPs or genes.

We divided the samples into training, validation, and testing sets with the ratio 3: 1: 1. ENN and ER were applied to the training set to build models. While a variety of activation functions can be used in ENN, we choose ReLU due to its performance and computational advantage[10]. Since the loss function of ENN is differentiable, we use the quasi-Newton BFGS optimization algorithm to estimate the parameters in ENN. We chose the starting point carefully to avoid the local minimum. To select a proper starting point, we generated a set of initial values from $U[-1, 1]$, ran the algorithm for a few steps, and chose the initial values achieving the smallest loss as the initial values. Based on the initial values, the quasi-Newton BFGS optimization algorithm is implemented to iteratively estimate the parameters until the convergence criterion is satisfied. The models built on the training set were then applied to the validation set to choose the most parsimonious model with the optimal tuning parameter (i.e., λ). To choose the best λ, we use the grid search with different values of 0,0.1,1,10,100. This final model was then evaluated on the testing set by using the mean squared error (MSE). We chose the number of hidden nodes with smallest MSE value by doing simulation. We simplify those terms: expectile neural network, expectile regression, training data and testing data as ENN, ER, TR, TS in three simulations.

2.4.1 Simulation I - nonlinear relationship

In simulation I, we varied the relationships between genotypes and phenotypes. Since the existence of hyperparameter τ, we compared the performances of ENN with ER. If we wanted to compare with other model, we need to fix τ. The existence of τ gived us a complete view of genetic effects across the entire distribution of phenotypes, like quantile regression. If τ is close to 0 or 1, we could investigate genetic contribution to high-risk individuals. Specially, we considered the following four nonlinear functions as true functions to simulate the relationship between genotypes and phenotypes. For comparison purpose, we also include a linear function. We compare ENN with ENN under four different nonlinear functions: hyperbolic function, mixed function, quadratic function, cubic function.

1. linear function:

$$y = \alpha + \epsilon, \alpha = \mathbf{x}^T \beta,$$

2. Hyperbolic function:

$$y = \frac{|\alpha|}{(1 + |\alpha|)} + \epsilon, \alpha = \mathbf{x}^T \beta,$$

3. Mixed function:

$$y = sin(\alpha) + 2 * exp(-16\alpha^2) + \epsilon, \alpha = \mathbf{x}^T \beta,$$

4. Quadratic function:

$$y = \alpha^2 + \epsilon, \alpha = \mathbf{x}^T \beta,$$

5. Cubic function:

$$y = \alpha^3 + \epsilon, \alpha = \mathbf{x}^T \beta,$$

where $\mathbf{x}$ is the vector of SNPs (coded as 0, 1 or 2), β represents the genetic effects generated from the uniform distribution of $U(-1, 1)$, and $\epsilon \sim N(0, 1)$. Totally 1000 replicates were simulated by setting ϵ with different seed. For each replicate, We randomly choose 500 samples and 50 SNPs from the 1000 Genomes Project. For each nonlinear function, we choose five different value τ of 0.1, 0.25, 0.5, 0.75, 0.9 in order to get different expectiles. To have better readability, the columns of validation data are not shown.

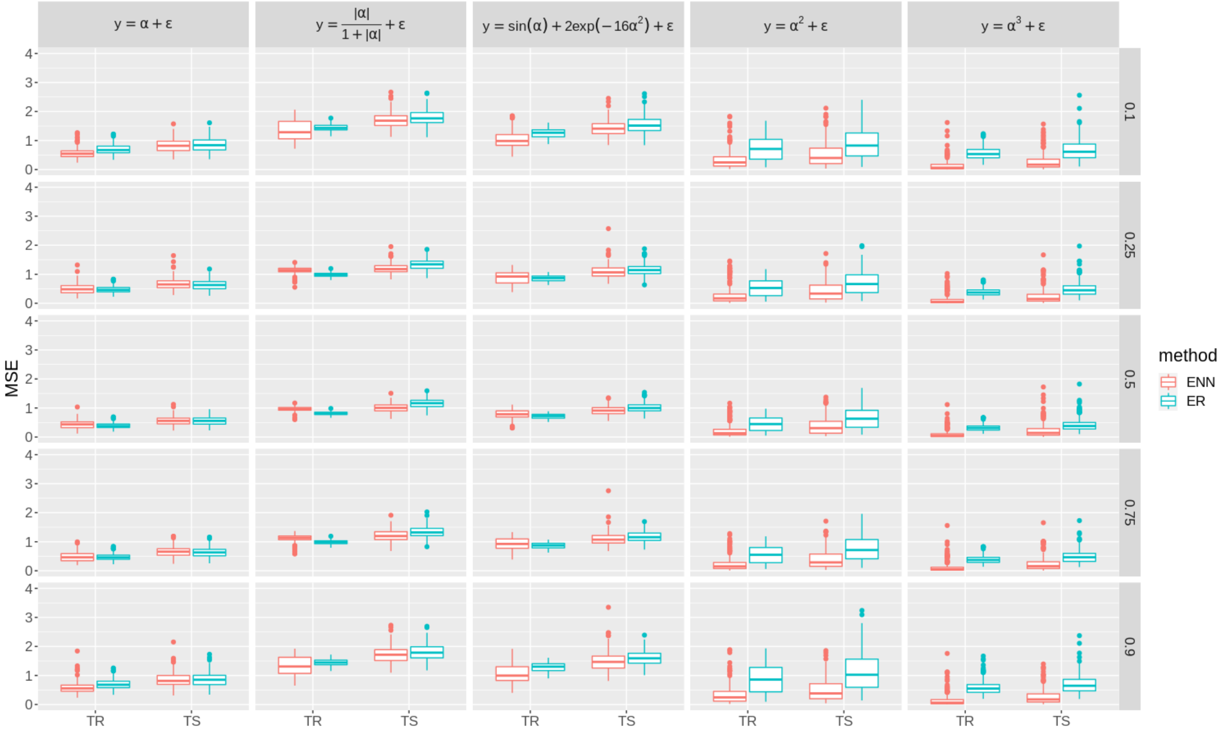

Figure 2.3: Performance comparison between ENN and ER under various relationships between genotypes and phenotypes and different expectiles (i.e., 0.1, 0.25, 0.5, 0.75, and 0.9)

The results from the simulation I are summarized in Figure 2.3. ENN outperforms ER in terms of MSE under four different nonlinear relationships, and has comparable performance with ER when the underlying relationship is linear. The pattern is consistent across different expectiles (i.e., 0.1, 0.25, 0.5, 0.75, and 0.9). While ENN outperforms ER for all four non-linear cases, ENN attains its best performance relative to ER when the underlying relationship is a high-order polynomial function (i.e., a cubic function). From the simulation result, ENN has advantages to explore the underlying nonliner relationship between genetic variants and certain disease. By fixing τ as 0.1 or 0.9, we could apply ENN into real data to identify high-risk individuals.

2.4.2 Simulation II - interactions among SNPs

Increasing empirical evidence from model organisms and human studies suggests that interactions among loci contribute broadly to complex traits[36; 37; 38]. In simulation II, we considered three different interactions scenarios that attempt to mimic simple biological mechanisms. Those three types of interactions included a two-way multiplicative interaction, a two-way threshold interaction, and a three-way interactions [14]. Similar to simulation I, we simulated 1000 replicates for each type of interaction. We use the same structure of ENN like simulatino II. For each replicate, 500 samples and 50 SNPs were chosen from the 1000 Genomes Project. Among the 50 SNPs, we randomly selected 20% of SNPs and simulated different types of interactions among the selected SNPs. Based on the simulated data, we compared MSEs of ENN and ER. For the comparison purpose, we also included a baseline model without any interaction. Only training and testing data are shown.

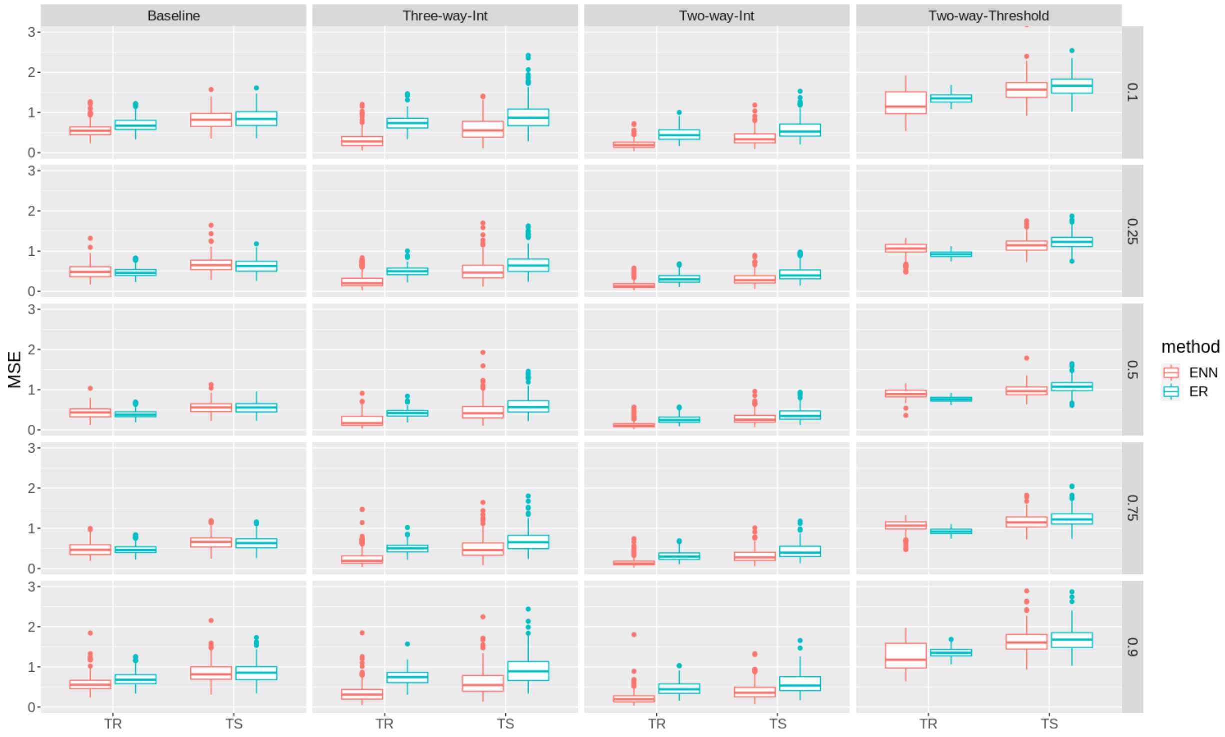

Figure 2.4: Performance comparison between ENN and ER for different types of interactions and different expectiles (i.e., 0.1, 0.25, 0.5, 0.75, and 0.9)

The results of the simulation II are summarized in Figure 2.4. Overall, ENN outperforms ER under all three interaction scenarios due to its ability of taking interactions into account. Among all interaction models, ENN attains its best performance relative to ER when there are three-way interactions. ENN also has more advantage over ER at the upper and lower expectiles (e.g., 0.1 and 0.9). When there is no interaction, ENN has comparable performance with ER.

2.4.3 Simulation III - interactions between genes

Following the identification of several disease-associated polymorphisms by whole genome association analysis, investigating interactions among two or more than two genes is often interested in genetic studies[35]. Detecting gene-gene interaction will allow us to elucidate the biological and biochemical pathways underpinning disease.

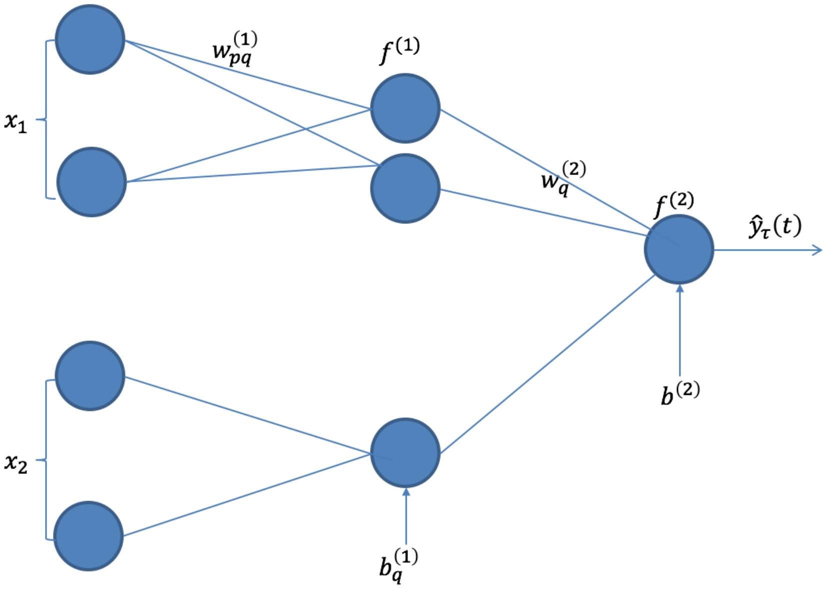

Figure 2.5: An alternative architecture for gene-gene interaction analyses

While a fully connected neural network can be built on all SNPs in the genes of interest, a neural network with a simpler architecture reflecting the underlying genetic data structure can be used to reduce the model's complexity and improve the model's performance. In this simulation, We illustrate the idea by modeling interactions between two genes with a non-fully connected architecture. In the non-fully connected architecture, the hidden units are only locally connected to SNPs in one gene (Figure 2.5). By using this simple architecture, we can reduce the number of parameters and build "gene-specific" hidden units to capture abstract features of a specific gene. To evaluate the performance of such an architecture, we simply simulated four SNPs for each gene, considered a two-way multiplicative interaction between two genes, and compared ENN with the non-fully connected architecture to ENN with a fully connected architecture.

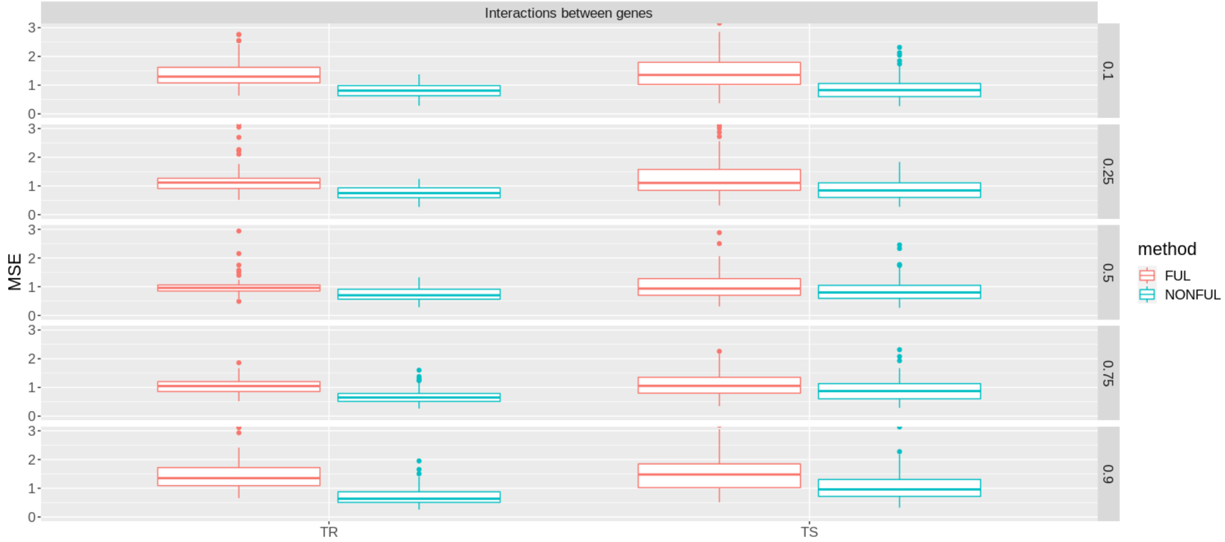

Figure 2.6: Performance comparison between ENN with a fully connected architecture and ENN with a non-fully connected architecture for gene-gene interaction analyses

Figure 2.6 summarizes the results from simulation III. The results show that ENN with the non-fully connected architecture attains lower MSE than ENN with the fully-connected architecture. As expected, the non-fully connected architecture requires fewer parameters and more reflects the underlying genetic data structure (i.e., genes are separate functional units), and therefore attains better performance than the fully-connected architecture. By reducing the number of parameters, we have more computational advantage.

2.5 Real data applications

Tobacco use is the leading cause of preventable disease and death in the United States. In 2019, nearly 34 million adults currently smoked cigarettes. More than 16 million Americans are related to a disease caused by smoking. More than 300 billion a year are spent in direct medical care for adults or in lost productivity due to premature death and exposure to secondhand smoke in United States. More than 7 million deaths per year are caused by tobacco use in the world(`https://www.cdc.gov/tobacco/data_statistics/index.htm`). Predicting high-risk individuals at early stage so that appropriate prevention methods can be used to reduce mortality and morbidity.

In this section, we applied ENN into analyzing two real data set. The first one is to explore genetic effects on nicotine dependence. In the second real data analysis, we take gene-gene interactions into consideration. Since the existence of hyperparameter τ, we choose ER as baseline. Five different τ values $0.1, 0.25, 0.5, 0.75, 0.9$ are chosen. We use mean square error(MSE) as metrics to measure the performance of ENN and ER.

Table 2.1: The accuracy performance of two models built by ENN and ER based on 149 candidate SNPs and 3 covariates

	ENN		**ER**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	409.612	678.331	504.215	694.809
0.25	346.118	579.164	394.836	588.759
0.5	358.783	502.752	342.144	535.925
0.75	344.399	604.969	421.955	613.676
0.9	570.994	809.733	699.654	882.781

2.5.1 The relationship between candidate SNPs with smoking quantities

We applied both ENN and ER to the genetic data from the Study of Addiction: Genetics and Environment(SAGE). The participants of the SAGE are selected from three large and complementary studies: the Family Study of Cocaine Dependence(FSCD), the Collaborative Study on the Genetics of Alcoholism(COGA), and the Collaborative Genetic Study of Nicotine Dependence(COGEND). In this application, we selected 155 SNPs, which were previously shown to have a potential role in nicotine dependence. After quality control, 149 SNPs remained for the analysis. There are a total of 3897 samples in the SAGE data from different ethnic groups. We only included 3888 Caucasian and African American samples due to the small sample size of other ethnic groups. Our interest is to use ENN and ER to build models on 149 SNPs, 3 covariates (i.e., sex, age, and race), and smoking quantities, which is measured by the largest number of cigarettes smoked in 24 hours. We divided the whole sample into the training, validation and test samples in the ratio of 3:1:1 to build the models, select the turning parameter, and evaluate the models, respectively.

Table 2.1 summarizes MSE of the models built by ENN and ER for five expectile levels (i.e., τ= 0.1, 0.25, 0.5, 0.75, and 0.9). For readability, MSE of validation data is omitted.

Table 2.1 shows that ENN outperforms ER, indicating the possibility of non-linear or non-additive effects among candidate SNPs and covariates.

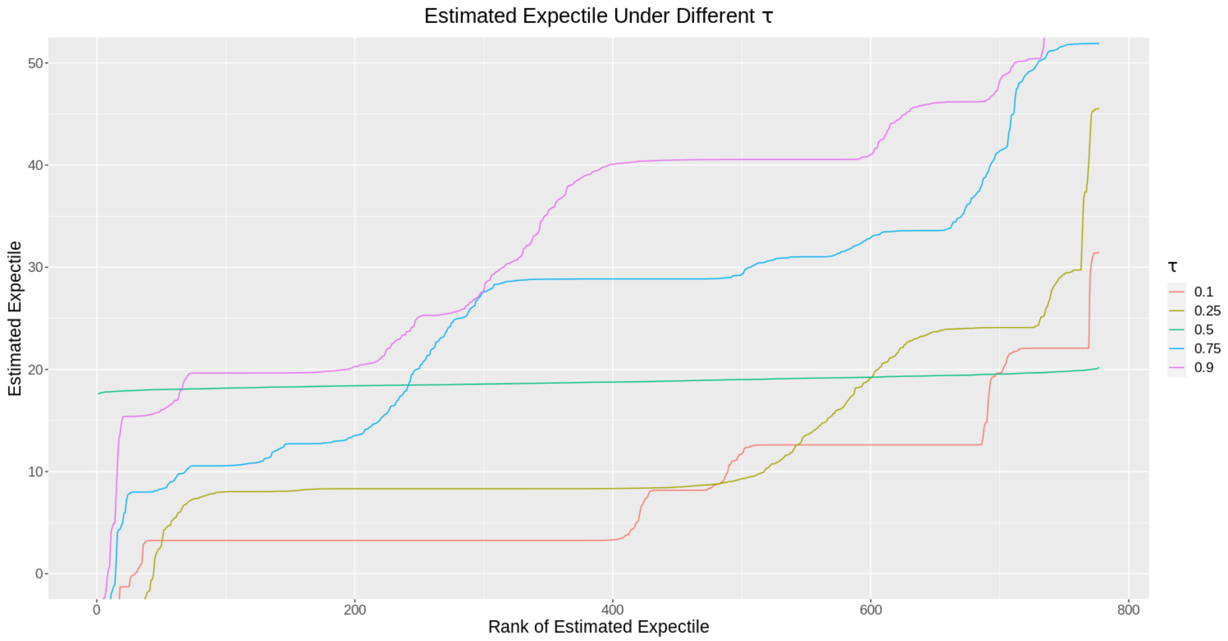

Figure 2.7: A comprehesive view of the conditional distribution of smoking quantity for five expectile levels (i.e., 0.1, 0.25, 0.5, 0.75, and 0.9)

To provide a comprehensive view of the conditional distribution of smoking quantity, we ordered the expectiles estimated from ENN from lowest to highest and plotted their values for all five expectile levels. Figure 2.7 shows that the distributions of estimated expectiles are different across five expectile levels. Under different expectile levels, different expectiles are predicted. When τ= 0.5, ENN models the mean response, in which the estimated expectiles are similar for all individuals. Nonetheless, for high expectile levels (e.g., τ= 0.9), the estimated expectiles vary among individuals and high-ranked individuals have much higher expectiles than low-ranked individuals. ENN gives us more information compared to linear regression which only shows predicted value with $\tau = 0.5$.

2.5.2 Gene-gene interactions between the CHRNA5-CHRNA3-CHRNB4 gene cluster

Based on previous genome-wide association studies, variants in the CHRNA5-CHRNA3-CHRNB4 gene cluster on chromosome 15 that encode the $\alpha5$, $\alpha3$ and $\beta4$ subunits of the nicotinic acetylcholine receptor (nAChRs) are associated with nicotine dependence (ND) in European Americans (EAs) or others of European origin[31]. In the second data analysis, we focused on the CHRNA5-CHRNA3-CHRNB4 gene cluster, and evaluated potential interactions by using ENN and ER. We consider three pairwise interactions between CHRNA5 and CHRNA3, CHRNA5 and CHRNB4, CHRNA3 and CHRNB4. The phenotype of interest in this analysis is the number of cigarettes smoked per day (CPD), which has been popularly used in the genetic study of nicotine dependence.

Table 2.2: Evaluating a pairwise interaction between CHRNA5 and CHRNA3 by using ENN and ER

	ENN		**ER**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	1.106	2.022	1.183	2.036
0.25	0.994	1.699	1.027	1.737
0.5	0.896	1.266	0.908	1.304
0.75	1.148	1.045	1.136	1.066
0.9	2.015	1.335	2.069	1.357

Table 2.3: Evaluating a pairwise interaction between CHRNA5 and CHRNB4 by using ENN and ER

	ENN		**ER**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	1.139	2.020	1.186	2.049
0.25	0.980	1.701	1.029	1.735
0.5	0.901	1.277	0.908	1.305
0.75	1.149	1.047	1.136	1.071
0.9	2.054	1.318	2.070	1.351

Tables 2.2-2.4 summarize MSE of the interaction models built by using ENN and ER for five expectile levels. For all 3 scenarios, expectile neural network outperforms expectile regression in terms of MSE slightly because the signal-to-noise ratio of genetic data is low.

To graphically view the conditional distribution of CPD, we ranked the expectiles estimated from ENN and plotted the values against the estimated expectiles (Figures 2.8-2.10).

Table 2.4: Evaluating a pairwise interaction between CHRNA3 and CHRNB4 by using ENN and ER

	ENN		**ER**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	1.133	2.019	1.183	2.035
0.25	0.979	1.683	1.020	1.696
0.5	0.892	1.278	0.896	1.279
0.75	1.150	1.048	1.128	1.081
0.9	2.020	1.342	2.040	1.386

Overall, the estimated expectiles tends to be similar when τ= 0.5 (i.e., mean), while they are quite different for high expectile levels (e.g., τ= 0.9). This suggest that the gene-gene interactions may play a more important role in models with high expectiles than the mean models.

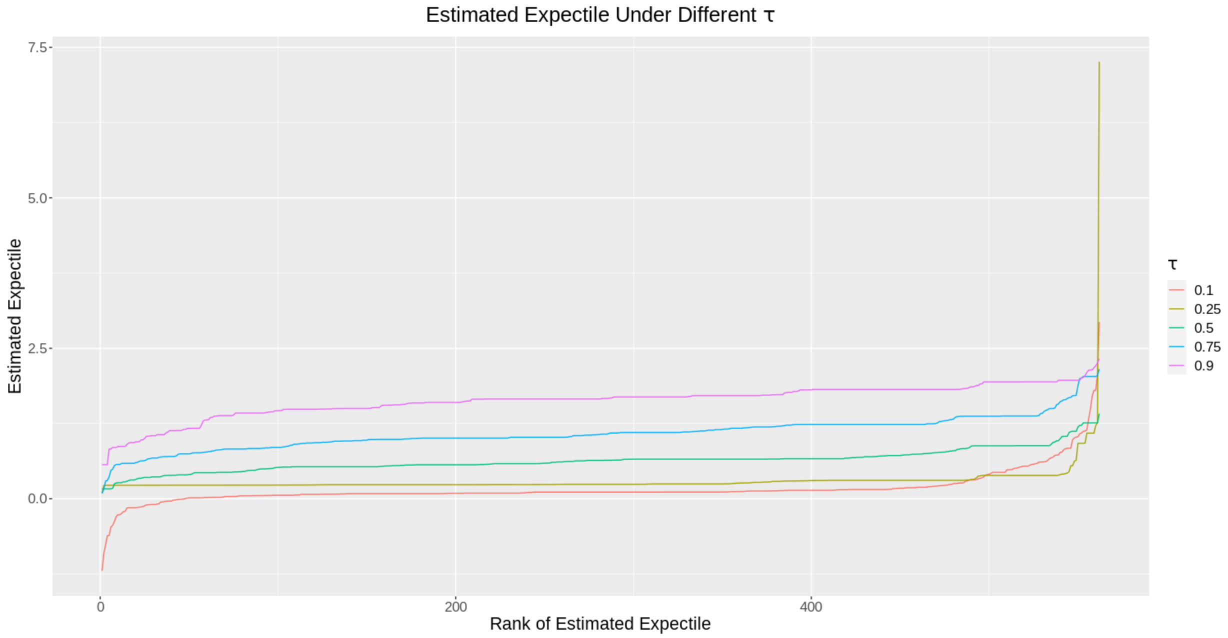

Figure 2.8: The conditional distribution of CPD considering the interaction between CHRNA5 and CHRNA3

2.6 Summary and discussion

In this chapter, we develop an ENN method, which inherits advantages from both neural networks and expectile regression. Using the hierarchical structure from neural networks, ENN can learn complex and abstract features from genotypes, making it suitable for modeling the complex relationship between genotypes and phenotype. Similar to ER, ENN can also explore the conditional distribution and provide a comprehensive view of the genotype-phenotype relationship.

Through simulations and a real data application, we demonstrate that ENN outperforms ER when there are non-additive and non-linear effects. Evidence also suggests that ENN has more advantages than ER when the model involves high-order interaction effects or non-linear effects. This may suggest ENN has improved performance when the underlying genotype-phenotype relationships become more complicated. The real data analysis shows

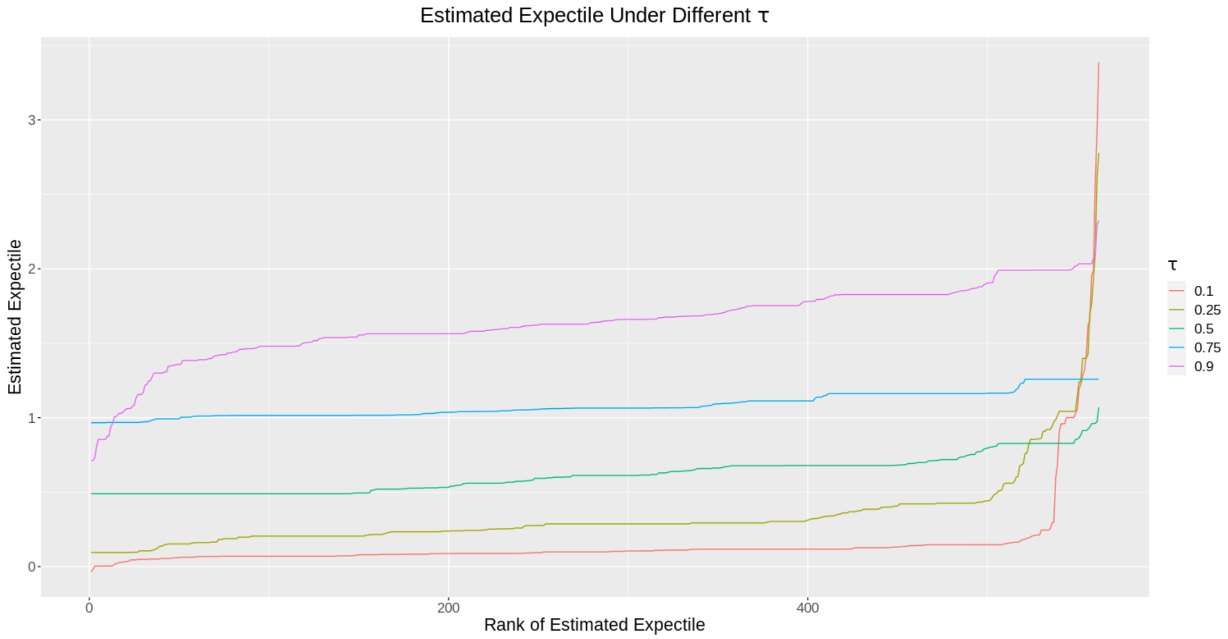

Figure 2.9: The conditional distribution of CPD considering the interaction between CHRNA5 and CHRNB4

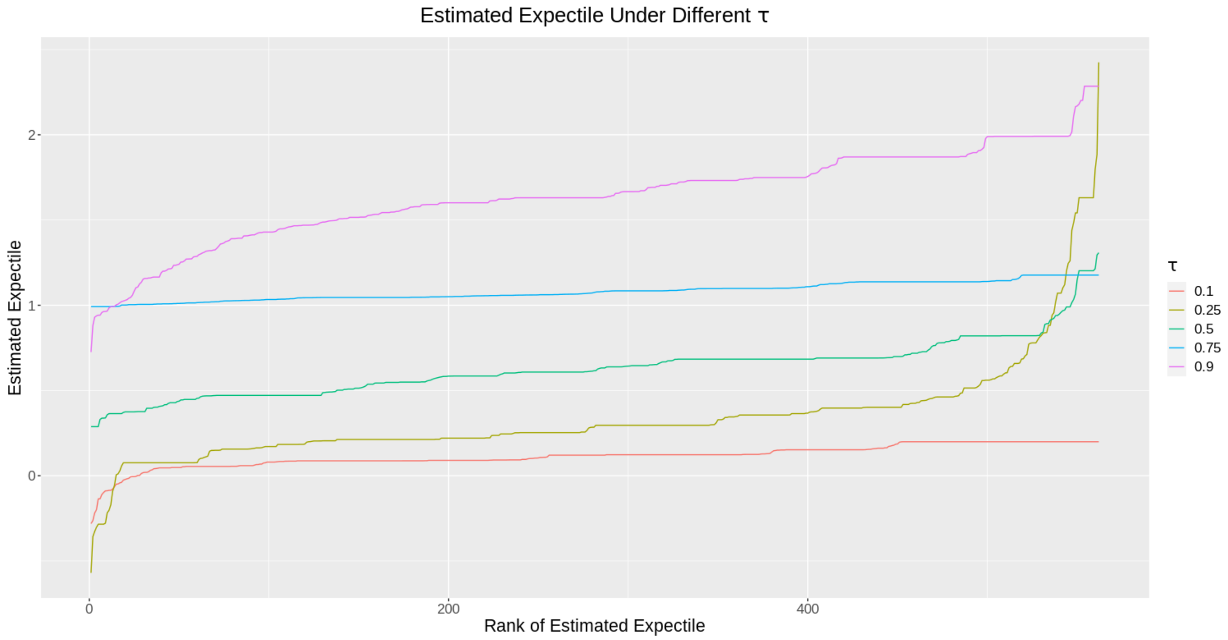

Figure 2.10: The conditional distribution of CPD considering the interaction between CHRNB4 and CHRNA3

that genetic effects can vary among different expertiles. Compared to the classical linear regression, ENN provides us more information about the genotype-phenotype relationship via the conditional distributions for different expectile levels.

While regularization has been incorporated into ENN to avoid overfitting, ENN can still be subject to overfitting when the number of SNPs becomes extremely large (e.g., one million). To deal with such a large number of SNPs, we can model the overall genetic effect as a random effect and extend ENN, which is an interesting topic for future work.

Chapter 3

Asymptotic Theory of Expectile Neural Networks

In the previous chapter, we focus on introducing the ENN model and providing an inequality that bounds the integrated squared error of an expectile function estimator. Statistical properties of ENN (e.g., consistency) are also important topics that worth further investigation. In this chapter, we study the asymptotic properties of expectile neural networks, including consistency and normality. In ENN, we use the asymmetric square loss as the loss function. When the size of parameters is too large, the standard maximum likelihood procedures may not work. Therefore, we use the sieve method to constrain the parameter space of ENN, and prove the consistency and normality under the nonparametric regression framework.

3.1 Introduction

Neural networks have been widely used in industry and academy. However, the theoretical properties of neural networks have not been thoroughly studied. For a typical artificial neural network, we use the squared loss function to estimate parameters. A general result for the asymptotic normality of squared loss function could be find[52]. By the universal approximation theorem, a neural network with one hidden layer can approximate any continuous functions[43]. In this chapter, we use the asymmetric squared loss function, which gives us a comprehensive view of conditional distribution and computation advantage. In statistics, fitting a neural network can be considered as a parametric nonlinear regression problem,

$$y_i = \alpha_0 + \sum_{j=1}^{r} \alpha_j \sigma(\gamma_j^T \mathbf{x}_i + \gamma_{0,j}),$$

where $\epsilon_1, \ldots, \epsilon_n$ are i.i.d. random errors with $\mathbb{E}[\epsilon] = 0$ and $\mathbb{E}[\epsilon^2] = \sigma^2 < \infty$ and $\sigma(z) = 1/(1+e^{-z})$. However, it is impractical to fix the number of hidden units r,. If we do not fix r, the parameter in unidentifiable. Fukumizu (1996)[55] and Fukumizu et al. (2003) [56] provided an example to illustrate the unidentifiable issue. If the true function is $f_0(x) = \alpha\sigma(\gamma x)$ with one hidden unit, we fit the model using a neural network with two hidden units. Then, any parameter $\Theta = [\alpha_0, \alpha_1, \ldots, \alpha_r, \gamma_{0,1}, \ldots, \gamma_{0,r}, \gamma_1^T, \ldots, \gamma_r^T]^T$ in the following set

$$\{\Theta : \gamma_1 = \gamma, \alpha_1 = \alpha, \gamma_{0,1} = \gamma_{0,2} = \alpha_2 = \alpha_0 = 0\} \cup$$
$$\{\Theta : \gamma_1 = \gamma_2 = \gamma, \gamma_{0,1} = \gamma_{0,2} = \alpha_0 = 0, \alpha_1 + \alpha_2 = \alpha\}$$

realizes the true function $f_0(x)$. Therefore, when the number of hidden units is unknown, the parameters in this parametric nonlinear regression problem are unidentifiable.

To address this issue, we can consider the neural network in the nonparametric setting. We assume that the true nonparametric regression model is as follows:

$$y_i = f_0(\mathbf{x_i}) + \epsilon_i,$$

where $\epsilon_1, ..., \epsilon_n$ are *i.i.d* random variables defined on $(\Omega, \mathcal{A}, \mathbb{P})$ with $E(\epsilon) = 0$ and $E(\epsilon^2) = \sigma^2 < \infty$. $f_0 \in \mathcal{F}$ is an unknown function, where $\mathcal{F}$ is the class of continuous function with compact support. However, if the complexity of $\mathcal{F}$ is large, the estimator may be inconsistent[48]. The standard and penalized maximum likelihood procedures may be inefficient, whereas the method of sieves may be able to overcome this difficulty[52]. The method of sieves provides one way to tackle such difficulties by optimizing an empirical criterion over a sequence of approximating parameter spaces (i.e., sieves). The sieves are less complex but are dense in the original space, and the resulting optimization problem becomes well-posed. To address this issue, we constrain the class of $\mathcal{F}$ and use method of sieves to prove the normality of ENN.

3.2 Method of sieves

Sieve is a sequence of increasing functions that can be used to reduce the number of parameters. Sieve plays an important role in infinite-dimensional unknown parameter, such as in a nonparametric or semiparametric model. When the method of sieves is implemented, a nonparametric or semiparametric estimation problem is often reduced to a parametric one. However, to obtain the desired theoretical properties of the estimator, it is necessary that the number of parameters increases slowly with the sample size[42]. We consider a sequence of function classes,

$$\mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \cdots \subseteq \mathcal{F}_n \subseteq \mathcal{F}_{n+1} \subseteq \cdots \subseteq \mathcal{F},$$

approximating $\mathcal{F}$ in the sense that $\bigcup_{n=1}^{\infty} \mathcal{F}_n$ is dense in $\mathcal{F}$, that is for each $f \in \mathcal{F}$, there exists $\pi_n f \in \mathcal{F}_n$ such that $d(f, \pi_n f) \to 0$ as $n \to \infty$, where $d(\cdot, \cdot)$ is some pseudo-metric defined on $\mathcal{F}$.

The method of sieves consists of two key ingredients: a loss function and sieve parameter spaces (a sequence of approximating spaces). Both loss function and the sieve parameter spaces are flexible. Almost all of the classical loss functions, so long as they allow for identification, can be used as loss functions in the method of sieve estimation. Therefore, the main challenge is the choice of sieve parameter spaces. In this chapter, we focus on the sieve of neural networks with one hidden layer and the sigmoid activation function.

$$\begin{aligned}\mathcal{F}_{r_n} = \{\alpha_0 + \sum_{j=1}^{r_n} \alpha_j \sigma\left(\gamma_j^T \mathbf{x} + \gamma_{0,j}\right) : \gamma_j \in \mathbb{R}^d, \alpha_j, \gamma_{0,j} \in \mathbb{R}, \sum_{j=0}^{r_n} |\alpha_j| \leq V_n \\ \text{for some } V_n \geq 4 \text{ and } \max_{1 \leq j \leq r_n} \sum_{i=0}^{d} |\gamma_{i,j}| \leq M_n \text{ for some } M_n > 0\},\end{aligned} \tag{3.1}$$

where $r_n, V_n, M_n \to \infty$ as $n \to \infty$. $\mathcal{F}_{r_n}$ has some important properties. For example, $\mathcal{F}_{r_n}$ is dense in $\mathcal{F}$ and $f \in \mathcal{F}_{r_n}$ has upper bound. When we consider the asymptotic properties of the sieve estimators, we use the pseudo-norm $\|f\|_n^2 = n^{-1} \sum_{i=1}^n f^2(\boldsymbol{x}_i)$.

With some abuse of notation, an approximate sieve estimator $\hat{f}_n$ is defined to be

$$\mathbb{Q}_n(\hat{f}_n) \leq \inf_{f \in \mathcal{F}_n} \mathbb{Q}_n(f) + \mathcal{O}_p(\eta_n), \tag{3.2}$$

where $\eta_n \to 0$ as $n \to \infty$.

We refer the reader to Chen for more details in the method of sieves [42]. Since we use the asymmetric loss function, we establish the upper bounds for the empirical risk and the sample complexity based on the covering number and the Vapnik-Chervonenkis dimension [41]. The estimator of expectile neural networks can also be regarded as M-estimator[50].

3.3 Existence

Before we study the consistency and normality of ENN, it is crucial to ask if the sieve estimator based on neural networks exists. In this chapter, we focus on $\mathcal{F}_{r_n}$ as sieve estimator.

First, we show that any function in $\mathcal{F}_{r_n}$ has an upper bound.

Lemma 3.3.1. *For each fixed n,*

$$\sup_{f \in \mathcal{F}_{r_n}} \|f\|_\infty \leq V_n.$$

Proof. For any $f \in \mathcal{F}_{r_n}$ with a fixed n and $\boldsymbol{x} \in \mathcal{X}$, we have

$$\begin{aligned} |f(\boldsymbol{x})| &= \left| \alpha_0 + \sum_{j=1}^{r_n} \alpha_j \sigma \left(\boldsymbol{\gamma}_j^T \boldsymbol{x} + \gamma_{0,j} \right) \right| \\ &\leq |\alpha_0| + \sum_{j=1}^{r_n} |\alpha_j| \sigma \left(\boldsymbol{\gamma}_j^T \boldsymbol{x} + \gamma_{0,j} \right) \leq \sum_{j=0}^{r_n} |\alpha_j| \leq V_n. \end{aligned}$$

Since the right-hand side does not depend on $\boldsymbol{x}$ and f, we have

$$\sup_{f \in \mathcal{F}_{r_n}} \|f\|_\infty = \sup_{f \in \mathcal{F}_{r_n}} \sup_{\boldsymbol{x} \in \mathcal{X}} |f(\boldsymbol{x})| \leq V_n.$$

□

Lemma 3.3.2. *Let χ be a compact subset of $\mathbb{R}^d$, then for each fixed n, $\mathcal{F}_{r_n}$ is a compact set.*

The proof of this lemma is in the appendix. This lemma tells us that $\mathcal{F}_{r_n}$ is compact in $\mathcal{C}(\mathcal{X})$, which is the set of all continuous functions. We use the theorem 3.3.1 to show the existence of estimator of ENN[47].

Theorem 3.3.1. *Let $(\Omega, \mathcal{F}, P)$ be a complete probability space and (Θ, ρ) be a metric space. Let $\{\Theta_n\}$ be a sequence of compact subsets of Θ and $\mathbb{Q}_n : \Omega \times \Theta_n \to \overline{\mathbb{R}}$ be measurable $\mathcal{F} \times \mathcal{B}(\Theta_n)/\overline{\mathcal{B}}$. Assuming that for each ω in Ω, $\mathbb{Q}_n(\omega, \cdot)$ is lower semicontinuous on $\Theta_n, n = 1, 2, ...,$ then for each $n = 1, 2, ...,$ there exists $\hat{\theta}_n : \Omega \to \Theta_n$ measurable $\mathcal{F}/\mathcal{B}(\Theta_n)$ such that* *for each* ω *in* Ω, $\mathbb{Q}_n(\omega, \hat{\theta}_n(\omega)) = \inf_{\theta \in \Theta_n} Q_n(\omega, \Theta)$.

Proof.

$$\begin{aligned}
\mathbb{Q}_n(f) &= \frac{1}{n}\sum_{i=1}^{n}\left(|\tau - \mathbb{1}_{\{y_i < f(x_i)\}}|(y_i - f(\mathbf{x}_i))^2\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(|\tau - \mathbb{1}_{\{f_0(\mathbf{x}_i)+\epsilon_i < f(x_i)\}}|(f_0(\mathbf{x}_i) + \epsilon_i - f(\mathbf{x}_i))^2\right).
\end{aligned}$$

Since $\mathbb{Q}_n$ is measurable and lower semicontinuous and $\mathcal{F}_{r_n}$ is compact, we could get the existence of sieve estimator for ENN. □

3.4 Consistency

In this section, we are interested in proving the consistency of the neural network sieve estimator under asymmetric squared loss function. If we choose $\tau = 0.5$, the proof of consistency of the neural network sieve estimator can be found in [51]. In ENN, we minimize the following empirical risk,

$$\begin{aligned}
\hat{f}_n &= \mathrm{argmin}_{f \in \mathcal{F}_{r_n}} \mathbb{Q}_n(f) \\
&= \mathrm{argmin}_{f \in \mathcal{F}_{r_n}} \frac{1}{n}\sum_{i=1}^{n} \tau \left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i) \geq 0\}} + (1-\tau)\left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i) < 0\}}.
\end{aligned}$$

One important step in proving consistency is to show that the empirical risk is uniformly over $\mathcal{F}_{r_n}$ close to the expected risk. More specifically, we need to show that

$$\frac{1}{n}\sum_{i=1}^{n}\{\tau\left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i)\geq 0\}} + (1-\tau)\left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i)<0\}}$$
$$- E\left(\tau\left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i)\geq 0\}} + (1-\tau)\left(Y_i - f(X_i)\right)^2 \mathbb{1}_{\{Y_i - f(X_i)<0\}}\right)\}$$
$$= \sup_{f\in\mathcal{F}_{r_n}} \frac{1}{n}\sum_{i=1}^{n}\tau[g_1(Z_i) - E(g_1(Z_i))] + (1-\tau)[g_2(Z_i) - E(g_2(Z_i))] \to 0 \text{ a.s., as } n\to\infty,$$

where $Z_i = (X_i, Y_i), i = 1, ..., n,$ $g_1(x,y) = |y - f(x)|^2\mathbb{1}_{\{y-f(x)\geq 0\}}$, $g_2(x,y) = |y - f(x)|^2\mathbb{1}_{\{y-f(x)<0\}}$ for $f \in \mathcal{F}_n$, $\mathcal{G}_{n,1} = \{|y - f(x)|^2\mathbb{1}_{\{y-f(x)\geq 0\}} : f \in \mathcal{F}_{r_n}\}$ and $\mathcal{G}_{n,2} = \{|y - f(x)|^2\mathbb{1}_{\{y-f(x)<0\}} : f \in \mathcal{F}_{r_n}\}$.

In order to bound the distance between an average and its expectation uniformly over $\mathcal{F}_{r_n}$, we introduce the concept of covering number with respect to the supremum norm.

Definition 3.4.1. *Let $\epsilon > 0$ and $\mathcal{G}$ be a set of functions $\mathcal{R}^d \to \mathcal{R}$. Each finite collection of functions $g_1, ..., g_N : \mathcal{R}^d \to \mathcal{R}$ has the following property. For every $g \in \mathcal{G}$, there is a $j = j(g) \in \{1, ..., N\}$ such that*

$$||g - g_j|| = \sup_x |g(x) - g_j(x)| < \epsilon$$

is called an $\epsilon-$cover of $\mathcal{G}$ with respect to $||\cdot||_\infty$.

Definition 3.4.2. *let $\epsilon > 0$, $\mathcal{G}$ be a set of functions $\mathcal{R}^d \to \mathcal{R}$, and $\mathcal{N}(\epsilon, \mathcal{G}, ||\cdot||_\infty)$ be the size of the smallest $\epsilon-$ cover of $\mathcal{G}$ w.r.t. $||\cdot||_\infty$. By taking $\mathcal{N}(\epsilon, \mathcal{G}, ||\cdot||_\infty) = \infty$ if no finite $\epsilon-$ cover exists, $\mathcal{N}(\epsilon, \mathcal{G}, ||\cdot||_\infty)$ is called an $\epsilon-$covering number of $\mathcal{G}$.*

To prove the uniform law of large numbers, we need to introduce the lemma 3.4.1 [44].

Lemma 3.4.1. *For $n \in \mathcal{N}$, Assuming $\mathcal{G}_n$ be a set of functions $g : \mathcal{R}^d \to [0, B]$ and $\epsilon > 0$, we have*

$$\mathbf{P}\left\{\sup_{g\in\mathcal{G}_n} \Big|\frac{1}{n}\sum_{i=1}^{n} g(Z_i) - Eg(Z)\Big| > \epsilon\right\} \leq 2\mathcal{N}(\epsilon/3, \mathcal{G}_n)e^{-\frac{2n\epsilon^2}{9B^2}}.$$

Theorem 3.4.1 (The uniform law of large numbers)**.** *Assuming $Z_i = (X_i, Y_i), i = 1, ..., n$. If $[r_n(d+2)+1]\, log\, [r_n(d+2)+1] = o(n)$. We can get*

$$\sup_{f\in\mathcal{F}_n} \Big|\frac{1}{n}\sum_{i=1}^{n} \tau[g_1(Z_i) - E(g_1(Z_i))] + (1-\tau)[g_2(Z_i) - E(g_2(Z_i))]\Big| \to 0 \;\; a.s, n \to \infty. \tag{3.3}$$

Proof. Since the loss function has two parts, the empirical risk can also be considered as two parts.

$$\begin{aligned} &\sup_{f\in\mathcal{F}_n} \Big|\frac{1}{n}\sum_{i=1}^{n} \tau[g_1(Z_i) - E(g_1(Z_i))] + (1-\tau)[g_2(Z_i) - E(g_2(Z_i))]\Big| \\ &\leq \sup_{g_1\in\mathcal{G}_{n,1}} \tau\Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i))\Big| + \sup_{g_2\in\mathcal{G}_{n,2}} (1-\tau)\Big|\frac{1}{n}\sum_{i=1}^{n} g_2(Z_i) - E(g_2(Z_i))\Big|. \end{aligned} \tag{3.4}$$

We focus on the first part since the proof of second part can be derived in the same manner.

$$\sup_{g_1\in\mathcal{G}_{n,1}} \tau\Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i))\Big| \to 0. \tag{3.5}$$

For $B > 0$, let $G(x) = \sup_{g_1\in\mathcal{G}_{n,1}} |g_1(x)|$,$\mathcal{G}_B = \{g_1 \mathbb{1}\{G < B\} : g_1 \in \mathcal{G}_{n,1}\}$.

If $g_1 \in \mathcal{G}_{n,1}$,

$$\begin{aligned}
&\Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i))\Big| \\
&\leq \Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - g_1(Z_i)\mathbb{1}_{\{G(Z_i)\leq B\}}\Big| + \Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i)\mathbb{1}_{\{G(Z_i)\leq B\}} - E(g_1(Z_i))\mathbb{1}_{\{G(Z)\leq B\}}\Big| \\
&+ \Big|\frac{1}{n}\sum_{i=1}^{n} E(g_1(Z_i))\mathbb{1}_{\{G(Z)\leq B\}} - E(g_1(Z_i))\Big| \\
&\leq \frac{1}{n}\sum_{i=1}^{n} G(Z_i)\mathbb{1}_{\{G(Z_i>B)\}} + \Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i)\mathbb{1}_{\{G(Z_i)\leq B\}} - E(g_1(Z_i))\mathbb{1}_{\{G(Z)\leq B\}}\Big| \\
&+ E(G(Z)\mathbb{1}_{\{G(Z)>B\}}).
\end{aligned} \tag{3.6}$$

This implies

$$\begin{aligned}
&\sup_{g_1\in\mathcal{G}_{n,1}} \Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i))\Big| \\
&\leq \sup_{g_1\in\mathcal{G}_B} \Big|\frac{1}{n}\sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i))\Big| + \frac{1}{n}\sum_{i=1}^{n} G(Z_i)\mathbb{1}_{\{G(Z_i>B)\}} \\
&+ E(G(Z)\mathbb{1}_{\{G(Z)>B\}}).
\end{aligned} \tag{3.7}$$

Based on $E(G(Z)) < \infty$ and the strong law of large numbers, we get

$$\frac{1}{n}\sum_{i=1}^{n} G(Z_i)\mathbb{1}_{\{G(Z_i>B)\}} \to E(G(Z)\mathbb{1}_{\{G(Z)>B\}}) \text{ a.s. } \text{ if } n\to\infty$$

If $B \to \infty$,

$$E(G(Z)\mathbb{1}_{\{G(Z)>B\}}) \to 0.$$

Therefore, we only need to consider,

$$\sup_{g_1 \in \mathcal{G}_B} \tau \Big| \frac{1}{n} \sum_{i=1}^{n} g_1(Z_i) - E(g_1(Z_i)) \Big| \to 0.$$

Recall that if g is a function $g : \mathcal{R} \to [0, B]$, then by Hoeffding's inequality

$$\mathbf{P} \left\{ \Big| \frac{1}{n} \sum_{j=1}^{n} g(Z_j) - E(g(Z)) \Big| > \epsilon \right\} \le 2e^{-\frac{2n\epsilon^2}{B^2}}. \tag{3.8}$$

By lemma 3.4.1, we have

$$\mathbf{P} \left\{ \sup_{g_1 \in \mathcal{G}_{n,1}} \Big| \frac{1}{n} \sum_{j=1}^{n} g(Z_j) - E(g(Z)) \Big| > \epsilon \right\} \le 2\mathcal{N}(\epsilon/3, \mathcal{G}_{n,1}, \| \cdot \|_\infty) e^{-\frac{2n\epsilon^2}{B^2}}. \tag{3.9}$$

We use the the upper bound covering number result from the Theorem 14.5 in Anthony and Gartlett,

$$\mathcal{N}(\epsilon/3, \mathcal{F}_{r_n}, \| \cdot \|_\infty) \le \left(\frac{12e\,[r_n(d+2)+1]\,(\frac{1}{4}V)^2}{\epsilon(\frac{1}{4}V - 1)} \right)^{(r_n(d+2)+1)}. \tag{3.10}$$

Recall the definition of covering number, $\mathcal{N}(\epsilon/3, \mathcal{F}_{r_n}, \| \cdot \|_\infty) = N$, is minimum number such that there exist functions $f_1, ..., f_N$ with the property that for every $f \in \mathcal{F}_{r_n}$ there is a $j = j(f) \in 1, ..., N$ such that

$$\sup_x |f(x) - f_j(x)| < \epsilon.$$

Since $f(x)$ and$f_j(x)$ is close enough, $y - f(x)$ and $y - f_j(x)$ are either negative or positive in the following situation,

$$\begin{aligned}
&\sup_x |(y-f(x))^2 \mathbb{1}_{\{y-f(x)\geq 0\}} - (y-f_j(x))^2 \mathbb{1}_{\{y-f_j(x)\geq 0\}}| \\
&\leq \sup_x |(y-f(x))^2 - (y-f_j(x))^2| \\
&= \sup_x |2y(f_j-f) + (f-f_j)(f+f_j)| \\
&< 2(M_1+M_2)\epsilon.
\end{aligned} \tag{3.11}$$

Since $y \in \mathcal{G}_B$ and any functions in $\mathcal{F}_{r_n}$ are bounded, there exist M_1 and M_2 such that $|y| < M_1$ and $|f| < M_2$. So $\mathcal{N}(\epsilon/3, \mathcal{G}_{n,1}, \|\cdot\|_\infty) \leq \mathcal{N}(\epsilon/3, \mathcal{F}_{r_n}, \|\cdot\|_\infty)$.

If $[r_n(d+2)+1]\, log\, [r_n(d+2)+1] = 0(n)$, then

$$\sum_{n=1}^{\infty} \exp\left\{ [r_n(d+2)+1]\, log \left(\frac{12e\, [r_n(d+2)+1]\, (\frac{1}{4}V)^2}{\epsilon(\frac{1}{4}V-1)} \right) \right\} \cdot e^{-\frac{2n\epsilon^2}{B^2}} < \infty. \tag{3.12}$$

(3.5) follows by using the Borel-Cantelli lemma. □

Since we have proven the uniform laws of large numbers, we use it to show the consistency of the neural networks. We rewrite the population loss criterion function:

$$Q_n(f) = \frac{1}{n} \sum_{i=1}^{n} E\left[\left(|\tau - \mathbb{1}_{\{y_i < f(x_i)\}}| (y_i - f(\mathbf{x}_i))^2 \right) \right] \tag{3.13}$$

In order to prove the consistency of ENN, we use the Theorem 3.4.2 that is the corollary 2.6 in White and Wooldridge[47], and check the condition of the corollary.

Theorem 3.4.2. *Let the condition of Theorem 3.3.1 holds. Suppose there exists a function*

$\overline{Q} : \Theta \to \overline{\mathbb{R}}$ *such that* $\overline{Q}$ *is continuous at* θ_0 *in* Θ, $\overline{Q}(\theta_0) < \infty$. *For any* $\epsilon > 0$,

$$P(\omega : \sup_{\theta \in \Theta_n} |Q_n(\omega, \theta) - \overline{Q}(\theta)| > \epsilon) \to 0 \text{ as } n \to \infty.$$

and

$$\inf_{\theta \in \eta^c(\theta_0, \epsilon)} \overline{Q}(\theta) - \overline{Q}(\theta_0) > 0.$$

If $\{\Theta_n\}$ *is an increasing sequence and* $\cup_n \Theta_n$ *is dense in* Θ, *then*

$$\rho(\hat{\theta}_n, \theta_0) \xrightarrow{P} 0.$$

Lemma 3.4.2. *Suppose*$\frac{\frac{1}{n}\sum_{i=1}^{n}(f_0(x_i) - f(x_i))^2}{\sigma^2} > \frac{2\tau - 1}{1 - \tau}$ *for* $\frac{1}{2} < \tau < 1$, *then* $\inf_{f : \|f - f_0\|_n \geq \epsilon} Q_n(f) - Q_n(f_0) > 0$.

Proof.

$$\begin{aligned}
Q_n(f) &= \frac{1}{n}\sum_{i=1}^{n} E\left[\left(|\tau - \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}|(y_i - f(\mathbf{x}_i))^2\right)\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f(x_i))^2\right] - \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f(x_i))^2 \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}\right] \\
&+ (1 - \tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f(x_i))^2 \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f(x_i))^2\right] + (1 - 2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f(x_i))^2 \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}\right]
\end{aligned} \tag{3.14}$$

$$Q_n(f_0) == \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f_0(x_i))^2\right] + (1 - 2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f_0(x_i))^2 \mathbb{1}_{\{y_i < f_0(\mathbf{x_i})\}}\right] \tag{3.15}$$

$$
\begin{aligned}
Q_n(f) - Q_n(f_0) &= \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f(x_i))^2\right] + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f(x_i))^2 \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}\right] \\
&- \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(y_i - f_0(x_i))^2\right] + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f_0(x_i))^2 \mathbb{1}_{\{y_i < f_0(\mathbf{x_i})\}}\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \tau \left(f_0(x_i) - f(x_i)\right)^2 + \sigma^2 \\
&= +(1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(y_i - f(x_i))^2 \mathbb{1}_{\{y_i < f(\mathbf{x_i})\}}\right] \\
&- \frac{\tau}{n}\sum_{i=1}^{n} \sigma^2 - (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[\epsilon_i^2 \mathbb{1}_{\{\epsilon_i < 0\}}\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \tau E\left[(f_0(x_i) - f(x_i))^2\right] + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(\epsilon_i)^2 \mathbb{1}_{\{\epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] \\
&+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} 2\left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right) E\left[\epsilon_i \mathbb{1}_{\{\epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] \\
&+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} \left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right)^2 P(\epsilon < f_0(\mathbf{x_i}) - f((x_i))) \\
&- (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[\epsilon_i^2 \mathbb{1}_{\{\epsilon_i < 0\}}\right] \\
&= \frac{1}{n}\sum_{i=1}^{n} \tau \left(f_0(x_i) - f(x_i)\right)^2 + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(\epsilon_i)^2 \mathbb{1}_{\{0 \le \epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] \\
&+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} 2\left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right) E\left[\epsilon_i \mathbb{1}_{\{\epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] \\
&+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} \left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right)^2 P(\epsilon_i < f_0(\mathbf{x_i}) - f((x_i)))
\end{aligned}
\tag{3.16}
$$

If $\epsilon_i < f_0(\mathbf{x_i}) - f((x_i)$, then $E\left[\epsilon_i \mathbb{1}_{\{\epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] = E\left[\epsilon_i\right] = 0$.

If $\epsilon_i \ge f_0(\mathbf{x_i}) - f((x_i)$, then $E\left[\epsilon_i \mathbb{1}_{\{\epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] = 0$.

We could simplify

$$
\begin{aligned}
Q_n(f) - Q_n(f_0) = \frac{1}{n}\sum_{i=1}^{n} \tau \left(f_0(x_i) - f(x_i)\right)^2 + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(\epsilon_i)^2 \, \mathbb{1}_{\{0 \le \epsilon_i < f(\mathbf{x_i}) - f_0(\mathbf{x_i})\}}\right] \\
+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} \left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right)^2 P(\epsilon_i < f_0(\mathbf{x_i}) - f((x_i)))
\end{aligned}
\tag{3.17}
$$

If $\tau \le \frac{1}{2}$,

$$
Q_n(f) - Q_n(f_0) \ge \frac{1}{n}\sum_{i=1}^{n} \tau \left(f_0(x_i) - f(x_i)\right)^2 > 0 \tag{3.18}
$$

If $\tau > \frac{1}{2}$,

$$
\begin{aligned}
Q_n(f) - Q_n(f_0) &\ge \frac{1}{n}\sum_{i=1}^{n} \tau \left(f_0(x_i) - f(x_i)\right)^2 + (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} E\left[(\epsilon_i)^2\right] \\
&+ (1-2\tau)\frac{1}{n}\sum_{i=1}^{n} \left(f_0(\mathbf{x_i}) - f(\mathbf{x_i})\right)^2 \\
&= (1-\tau)\frac{1}{n}\sum_{i=1}^{n} \left(f_0(x_i) - f(x_i)\right)^2 + (1-2\tau)\sigma^2 \\
&> 0, \text{ if } \frac{\frac{1}{n}\sum_{i=1}^{n} \left(f_0(x_i) - f(x_i)\right)^2}{\sigma^2} > \frac{2\tau - 1}{1-\tau}
\end{aligned}
\tag{3.19}
$$

Therefore,

$$
\inf_{f : \|f - f_0\|_n \ge \epsilon} Q_n(f) - Q_n(f_0) > 0. \tag{3.20}
$$

□

Since the conditions of the corollary 2.6 satisfy, we have the consistency of ENN sieve estimator.

Theorem 3.4.3. *Under the notation given above, if* $\frac{\frac{1}{n}\sum_{i=1}^{n}(f_0(x_i)-f(x_i))^2}{\sigma^2} > \frac{2\tau-1}{1-\tau}$ *for* $\frac{1}{2} < \tau < 1$ *and* $[r_n(d+2)+1]\, log\, [r_n(d+2)+1] = o(n)$*, then*

$$\|\hat{f}_n - f_0\|_n \xrightarrow{P} 0.$$

Proof. By using the Theorem 3.4.2, Theorem 3.4.1, lemma 3.4.2 and lemma 3.3.2, we have

$$\|\hat{f}_n - f_0\|_n \xrightarrow{P} 0.$$

□

3.5 Normality

We use the following theorem to prove the normality of the ENN sieve estimator[48].

Theorem 3.5.1. *Suppose that* $\mathcal{F}$ *is a* $P-$*Donsker class of measurable function and* $\hat{f}_n$ *is a sequence of random functions that take their values in* $\mathcal{F}$ *such that*

$$\int \left(\hat{f}_n(x) - f_0(x)\right)^2 dP(x) \xrightarrow{P} 0.$$

For some $f_0 \in L_2(P)$*, we have*

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \left((\hat{f}_n - f_0)(X_i) - P(\hat{f}_n - f_0)\right) \xrightarrow{P} 0,$$

and

$$\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\hat{f}_n(X_i) - P\hat{f}_n \sim N(0, Pf_0^2 - (Pf_0)^2).$$

From theorem 3.5.1, We need to check two conditions: $\mathcal{F}_{r_n}$ is $P-$Donsker class and $\int \left(\hat{f}_n(x) - f_0(x)\right)^2 dP(x) \xrightarrow{P} 0$.

Next, we give the definition of the Donsker class. In short, if the sequence of processes $\sqrt{n}(\mathcal{P}f - Pf)$ converges in distribution to a tight limit process, then a class $\mathcal{F}$ of measurable functions f is called Donsker. We review the formal definition of the Donsker class.

Let $(\mathcal{X}, \mathcal{A}, \mathbb{P})$ be a probability space and G_p be a Gaussian process with zero mean and covariance $E[G_p(f)G_p(g)] = P(fg) - Pf \cdot Pg$. We define a class $\mathcal{F} \subset L_2(\mathcal{X}, \mathcal{A}, \mathbb{P})$ as a G_pBUC class if and only if the process $G_p(f, \omega)$ can be chosen so that for all ω, the sample functions $f \longmapsto G_p(f, \omega), f \in \mathcal{F}$ are bounded and continuous for ρ_p.

Definition 3.5.1 (Donsker Class). *A class $\mathcal{F} \subset L_2(\mathcal{X}, \mathcal{A}, \mathcal{P})$ is called a Donsker class if and only if it is a G_pBUC class. There are processes $Y_j(f, \omega), f \in \mathcal{F}, \omega \in \Omega$, where Y_j are independent copies of G_p with $f \longmapsto Y_j(f, \omega)$ bounded and ρ_P-uniformly continuous on $\mathcal{F}$ for each j, such that for every $\epsilon > 0$,*

$$\mathbb{P}^*\left(n^{-1/2}\max_{m\leq n}\sup_{f\in\mathcal{F}}\|\sum_{j=1}^{m} f(X_j) - Pf - Y_j(f)\| > \epsilon\right) \to 0 \text{ as } n \to \infty.$$

It is not convenient to check if one class of functions is the Donsker class by definition. A sufficient condition for a class to be Donsker is that they do not grow too fast. The speed can be measured by bracketing integral

$$J_{[]}(\delta, \mathcal{F}, L_2(P)) = \int_0^\delta \sqrt{logN_{[]}(\epsilon, \mathcal{F}, L_2(P))d\delta},$$

where $N_{[]}(\epsilon, \mathcal{F}, L_2(P)$ is the bracketing number. If this integral is finite, then the class $\mathcal{F}$ is a Donsker class.

Theorem 3.5.2. *$\mathcal{F}_{r_n}$ is a Donsker class.*

Proof. By using the result of the uniform covering number for deep neural neural networks, we have

$$N(\epsilon, \mathcal{F}_{r_n}, ||\cdot||_{sup}) \leq \left(\frac{4e[r_n(d+2)+1](\frac{1}{4}V_n)^2}{\epsilon(\frac{1}{4}V_n - 1)} \right).$$

By using the relationship between packing number and covering number, for a small enough ϵ, we have

$$\begin{aligned} logN_{[\,]}(2\epsilon, \mathcal{F}_{r_n}, ||\cdot||_\infty) &\leq log\left(2N(\frac{\epsilon}{2}, \mathcal{F}_{r_n}, ||\cdot||_{sup})\right) \\ &\leq 2log\left(N(\frac{\epsilon}{2}, \mathcal{F}_{r_n}, ||\cdot||_{sup})\right) \\ &\leq 2[r_n(d+2)+1]\left(log\tilde{A}_{r_n,V_n,d} + log\frac{1}{\epsilon}\right), \end{aligned}$$

where $\tilde{A}_{r_n,V_n,d} = \frac{2eV_n^2[r_n(d+2)+1]}{V_n-4}$. By letting

$$\begin{aligned} A_{r_n,V_n,d} &= [r_n(d+2)+1]log\tilde{A}_{r_n,V_n,d} - [r_n(d+2)+1] \\ &= [r_n(d+2)+1]\left(log\frac{2eV_n^2[r_n(d+2)+1]}{V_n-4} - 1\right) \\ &= [r_n(d+2)+1]\left(log\frac{2V_n^2[r_n(d+2)+1]}{V_n-4}\right), \end{aligned}$$

and $V_n^2 - eV_n + 4e \geq 0$ for all V_n, we have

$$log\frac{2V_n^2[r_n(d+2)+1]}{V_n-4} \geq log\frac{V_n^2}{V_n-4} \geq log\frac{e(V_n-4)}{V_n-4} = 1.$$

Then

$$\begin{aligned}
logN_{[\,]}(2\epsilon, \mathcal{F}_{r_n}, ||\cdot||_\infty) &\leq 2[r_n(d+2)+1]\left(log\tilde{A}_{r_n,V_n,d} + log\frac{1}{\epsilon}\right) \\
&\leq 2\left(\tilde{A}_{r_n,V_n,d} + 2[r_n(d+2)+1](\frac{1}{\epsilon}+1)\right) \\
&\leq 2\tilde{A}_{r_n,V_n,d} + [r_n(d+2)+1]\frac{1}{\epsilon} \text{ (since } logx \leq x-1 \text{ for all } x>0) \\
&\leq 2\tilde{A}_{r_n,V_n,d}\left(1+\frac{1}{\epsilon}\right).
\end{aligned}$$

Since $\mathcal{F}_{r_n}$ is uniformly bounded by V_n, it is clear that $N_{[\,]}(2\epsilon, \mathcal{F}_{r_n}, ||\cdot||_\infty) = 1$ for all $\epsilon \geq V_n$. Therefore, for each fixed n, we have

$$\begin{aligned}
\int_0^\infty \left(logN_{[\,]}(2\epsilon, \mathcal{F}_{r_n}, ||\cdot||_\infty)\right)^{1/2} &\lesssim \int_0^{V_n}\left(1+\frac{2}{\epsilon}\right)^{1/2} d\epsilon \\
&< \infty.
\end{aligned}$$

Then $\mathcal{F}_{r_n}$ is a Donsker class. □

More details about the Donsker class can be found in Van der Vaart and A.W., Wellner [49]. It is easy to check that the sigmoid activation function is squashing function since $\sigma(x) = \frac{1}{1+e^{-x}}$ is nondecreasing ($\lim_{x\to\infty}\sigma(x) = 1$ and $\lim_{x\to-\infty}\sigma(-x) = 0$). We use the theorem 3.5.3 to check $\int\left(\hat{f}_n(x) - f_0(x)\right)^2 dP(x) \xrightarrow{P} 0$.

Theorem 3.5.3. *Let σ be a squashing function. For each probability measure μ on $\mathcal{R}^d$, each measurable $f : \mathcal{R}^d \to \mathcal{R}$ with $\int |f(x)|^2\mu(dx) < \infty$, and each $\epsilon > 0$, there exists a neural network $h(x)$ in*

$$h(x) = \{\sum_{i=1}^{k} c_i\sigma(a_i^T x + b_i) + c_0 : k \in \mathbf{N}, a_i \in \mathcal{R}^d, b_i, c_i \in \mathcal{R}\},$$

such that

$$\int |f(x) - h(x)|^2 \mu(dx) < \epsilon.$$

Next, we establish the asymptotic normality of ENN. We assume that $f_0 \in \mathcal{F}$, where $\mathcal{F}$ is the class of continuous functions with compact supports. f_0 is a function needed to be estimated.

Theorem 3.5.4. *Suppose $\hat{f}_n(x) \in \mathcal{F}$ is a sequence of random functions and $\int |f_0(x)|^2 dP(x) < \infty$. If conditions in consistency exist, we can get*

$$\int \left(\hat{f}_n(x) - f_0(x)\right)^2 dP(x) \xrightarrow{P} 0.$$

For some $f_0 \in L_2(P)$, we have

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \left((\hat{f}_n - f_0)(X_i) - P(\hat{f}_n - f_0)\right) \xrightarrow{P} 0,$$

and

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \hat{f}_n(X_i) - P\hat{f}_n \sim N(0, Pf_0^2 - (Pf_0)^2).$$

Proof. Assuming $\pi_{r_n} f_0 \in \mathcal{F}_{r_n}$, we have

$$\|\hat{f}_n(x) - f_0(x)\|^2 \leq \|\hat{f}_n - \pi_n f_0\|^2 + \|\pi_n f_0 - f_0\|^2. \tag{3.21}$$

Using the result of consistency of ENN, we have

$$\|\hat{f}_n - \pi_n f_0\|^2 \xrightarrow{p} 0.$$

From the theorem 3.5.3,

$$\|\pi_n f_0 - f_0\|^2 < \epsilon.$$

Therefore, we can get

$$\int \left(\hat{f}_n(x) - f_0(x)\right)^2 dP(x) \xrightarrow{P} 0.$$

Based on the theorem 3.5.1, we can obtain the result for the normality,

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \left((\hat{f}_n - f_0)(X_i) - P(\hat{f}_n - f_0)\right) \xrightarrow{P} 0,$$

and

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \hat{f}_n(X_i) - P\hat{f}_n \sim N(0, Pf_0^2 - (Pf_0)^2).$$

□

3.6 Simulation

To validate the theoretical properties of ENN, we ran simulations on the consistency and normality of ENN. We obtained the estimator of ENN by using the gradient-based optimization algorithms (e.g., quasi-Newton Broyde-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm). The response was simulated through the following equation:

$$y_i = f_0(x_i) + \epsilon_i, i = 1, .., n, \tag{3.22}$$

where $x_1, .., x_n \sim \mathcal{N}(0, 1), \epsilon_1, ..., \epsilon_n \overset{i.i.d}{\sim} \mathcal{N}(0, 0.1^2)$. For the true function f_0, we consider three different nonlinear functions:

1. a neural network with one single hidden layer and two hidden units,

2. a polynomial function:

$$f_0 = x^3 + 1,$$

3. a complex nonlinear function:

$$f_0 = sin(x) + 2exp((-16)x^2).$$

3.6.1 Consistency

In this section, we used simulations to check the validity of consistency result in Section 4. Since τ was between 0 and 1. For ENN with $0.5 < \tau < 1$, ENN had one more condition than ENN with $0 < \tau < 0.5$. we mainly considered ENN with $\tau = 0.5, 0.75$. For ENN with $\tau = 0.75$, we made σ^2 smaller(e.g., $\sigma^2 = 0.01$) to satify the condition: $\frac{\frac{1}{n}\sum_{i=1}^{n}(f_0(x_i)-f(x_i))^2}{\sigma^2} > \frac{2\tau-1}{1-\tau}$ for $\frac{1}{2} < \tau < 1$.

3.6.1.1 Simulation results of consistency with $\tau = 0.5$

We chose five different sample sizes: 50, 100, 200, 500 and 1000. From Figure 3.1 to Figure 3.3, the fitted curve is closer to the true function as the sample increases.

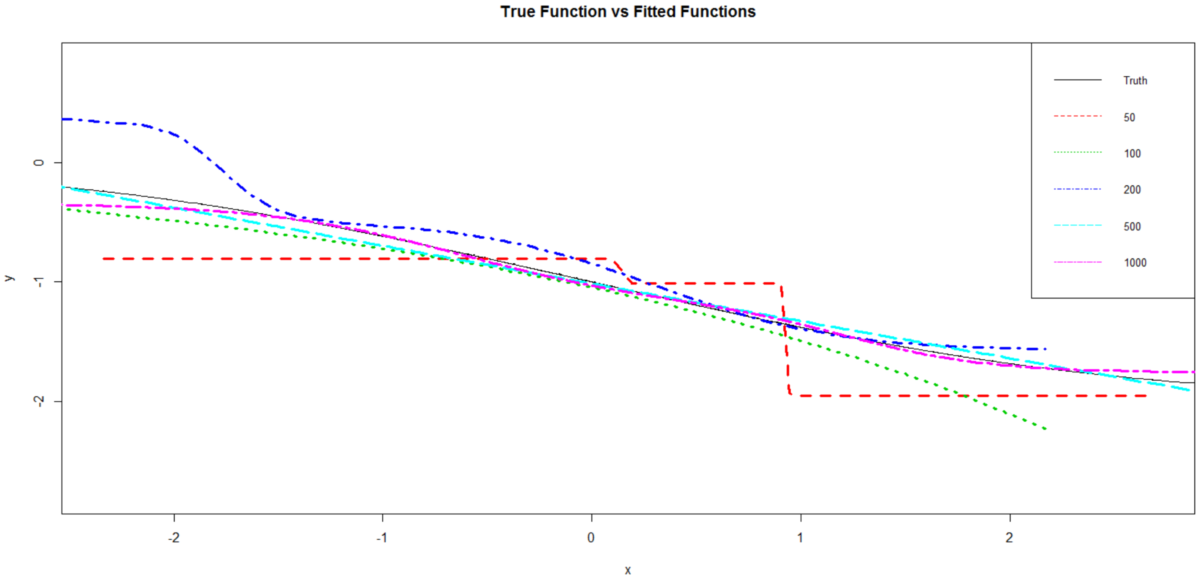

Figure 3.1: Comparison between the true function f_0 and fitted functions under different sample sizes, where f_0 is a neural network with one single hidden layer and two hidden units $\tau = 0.5$.

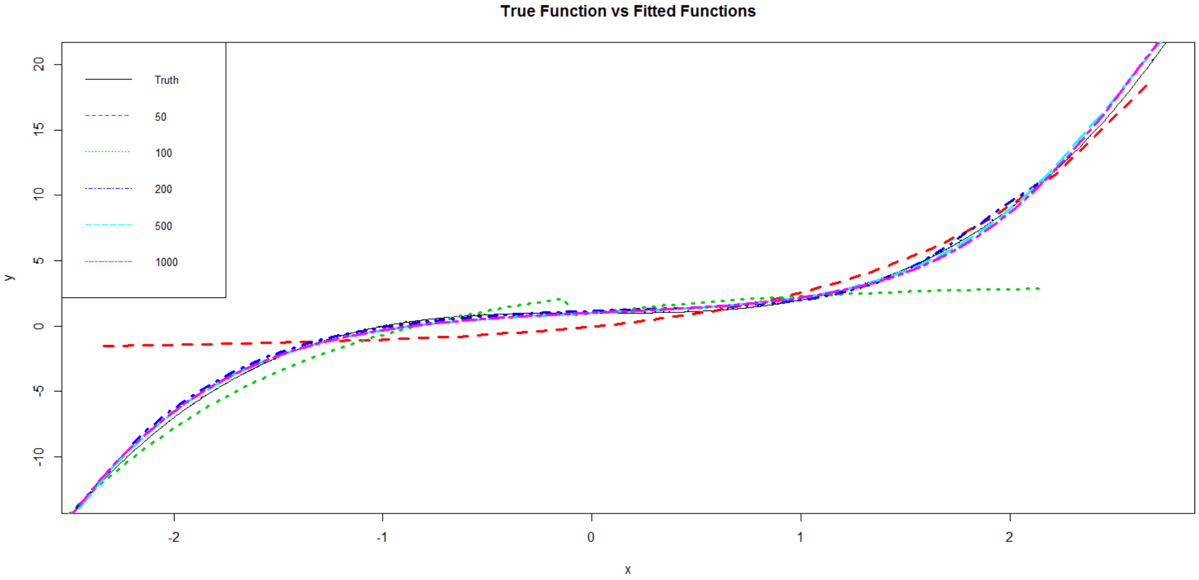

Figure 3.2: Comparison between the true function $f_0 = x^3 + 1$ and fitted functions under different sample sizes with $\tau = 0.5$.

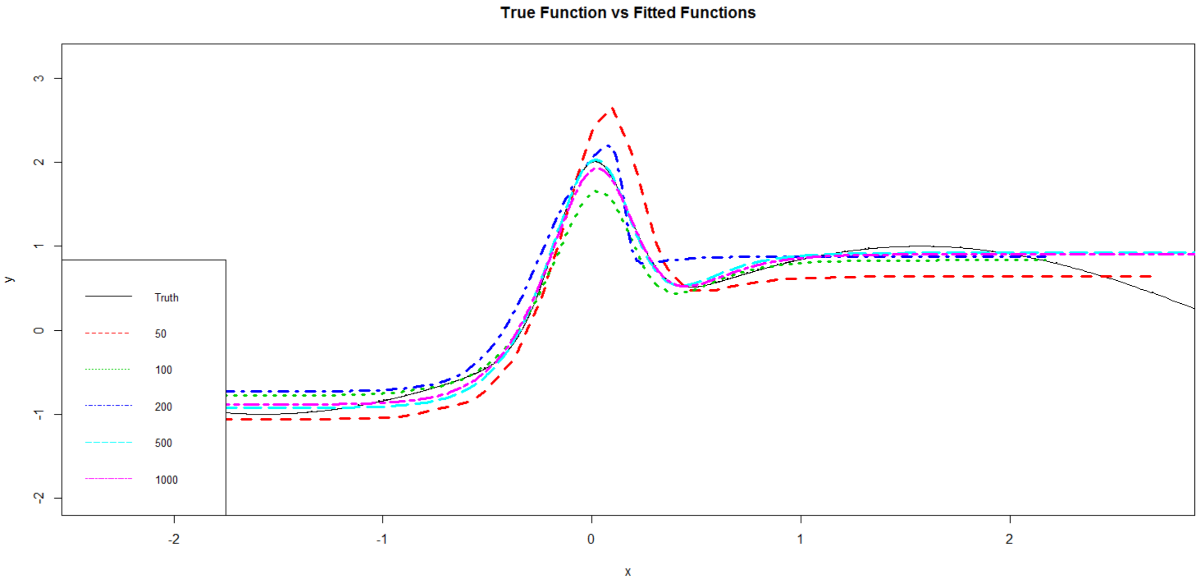

Figure 3.3: Comparison between the true function $f_0 = sin(x) + 2exp((-16)x^2)$ and fitted functions under different sample sizes with $\tau = 0.5$.

3.6.1.2 Simulation results of consistency with $\tau = 0.75$

We also chose five different sample sizes: 50, 100, 200, 500 and 1000. From Figure 3.4 to Figure 3.6, the fitted curve is closer to the true function as the sample increases. Overall, the simulation results are consistent with the theoretical finding.

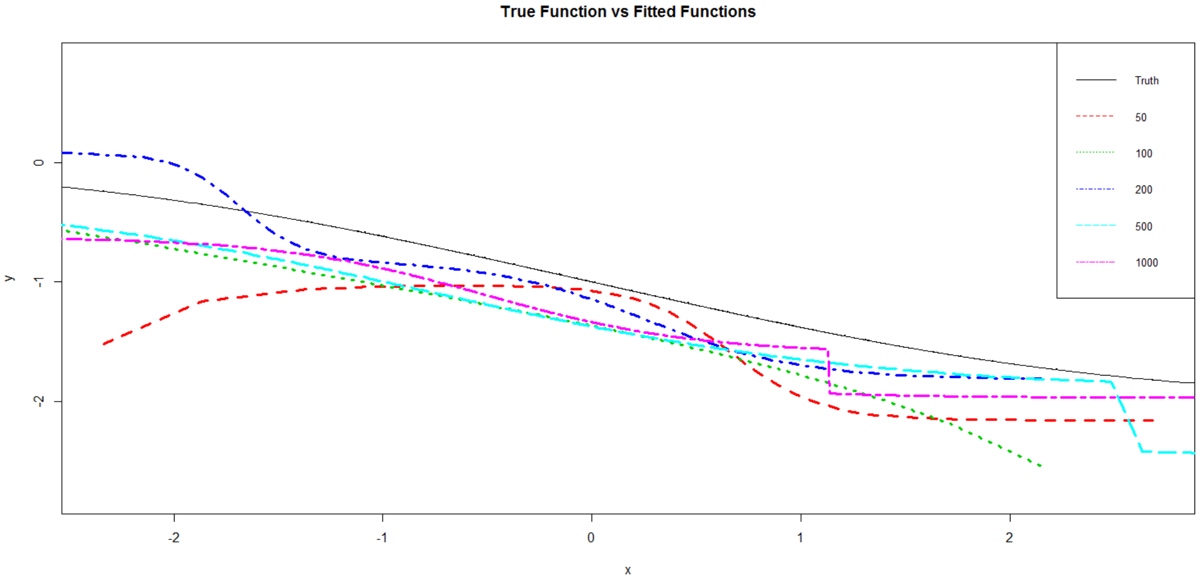

Figure 3.4: Comparison between the true function f_0 and fitted functions under different sample sizes, where f_0 is a neural network with one single hidden layer and two hidden units with $\tau = 0.75$.

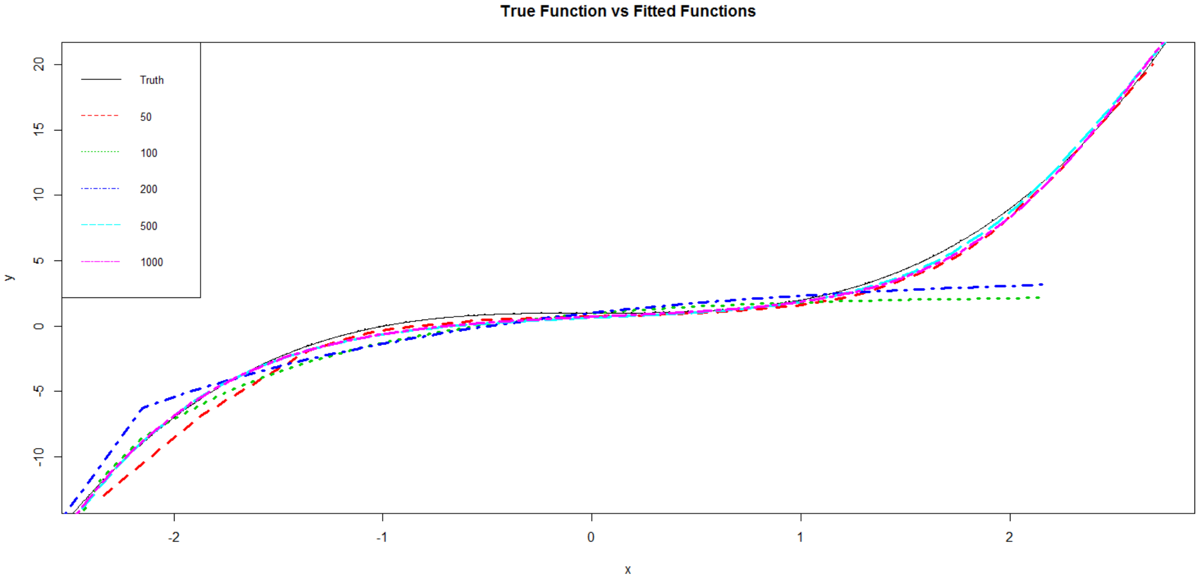

Figure 3.5: Comparison between the true function $f_0 = x^3 + 1$ and fitted functions under different sample sizes with $\tau = 0.75$.

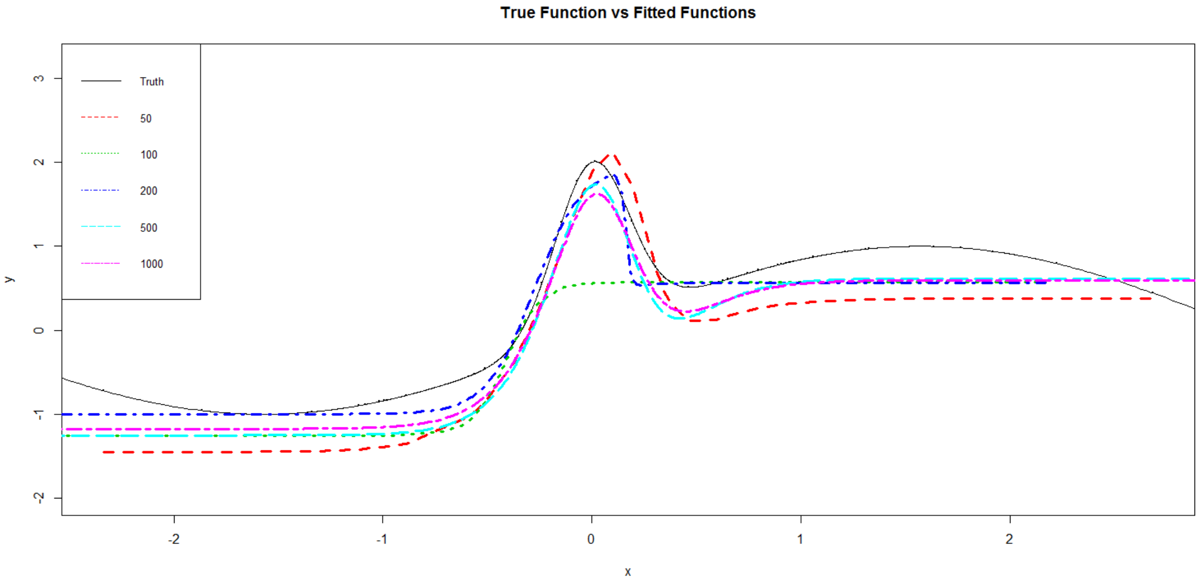

Figure 3.6: Comparison between the true function $f_0 = sin(x) + 2exp((-16)x^2)$ and fitted functions under different sample sizes with $\tau = 0.75$.

3.6.2 Normality

In this section, we demonstrated our asymptotic normality derived in theorem 3.5.4. The same true function were used but the random errors were sampled from standard normal distribution. We used

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \Big(\hat{f}_n(x_i) - f_0(x_i) \Big) \tag{3.23}$$

as test statistic to draw the Q-Q plots. We varied sample sizes (i.e., 50, 100, 200, 300, 400, and 500) when evaluating three nonlinear functions.

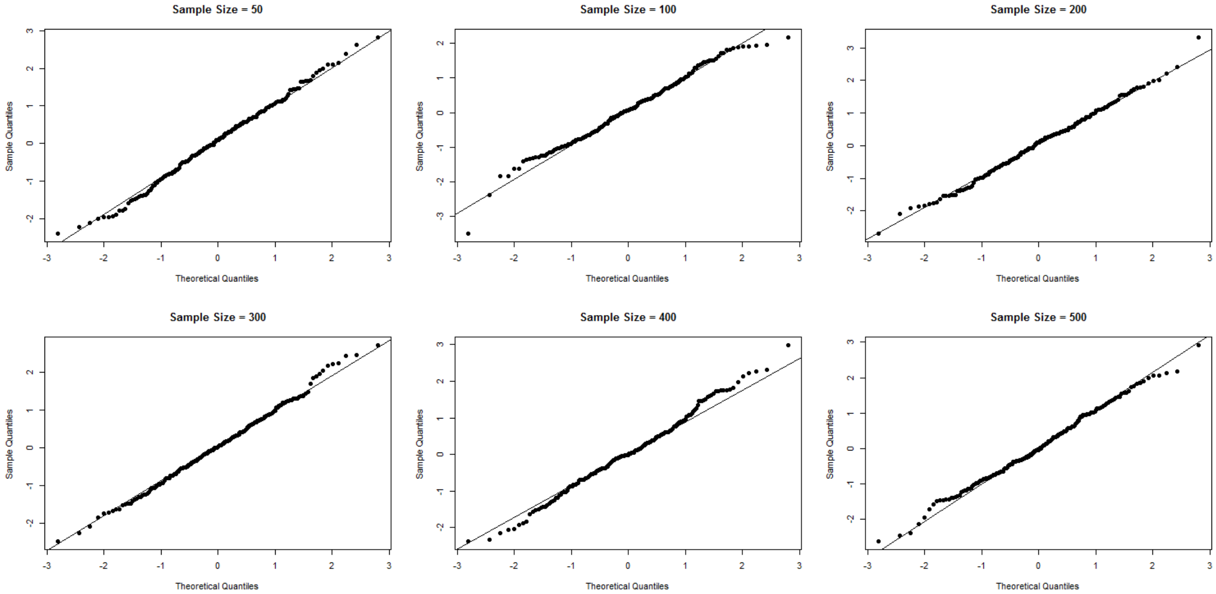

Figure 3.7: Q-Q plot with different sample sizes, where the true function f_0 is a neural network with one single hidden layer and two hidden units with $\tau = 0.5$

3.6.2.1 Simulation result of normality with $\tau = 0.5$

From Figure 3.7 to Figure 3.9, data points appear as roughly a straight line. The test statistic

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \left(\hat{f}_n(x_i) - f_0(x_i) \right) \tag{3.24}$$

fits the normal distribution pretty well.

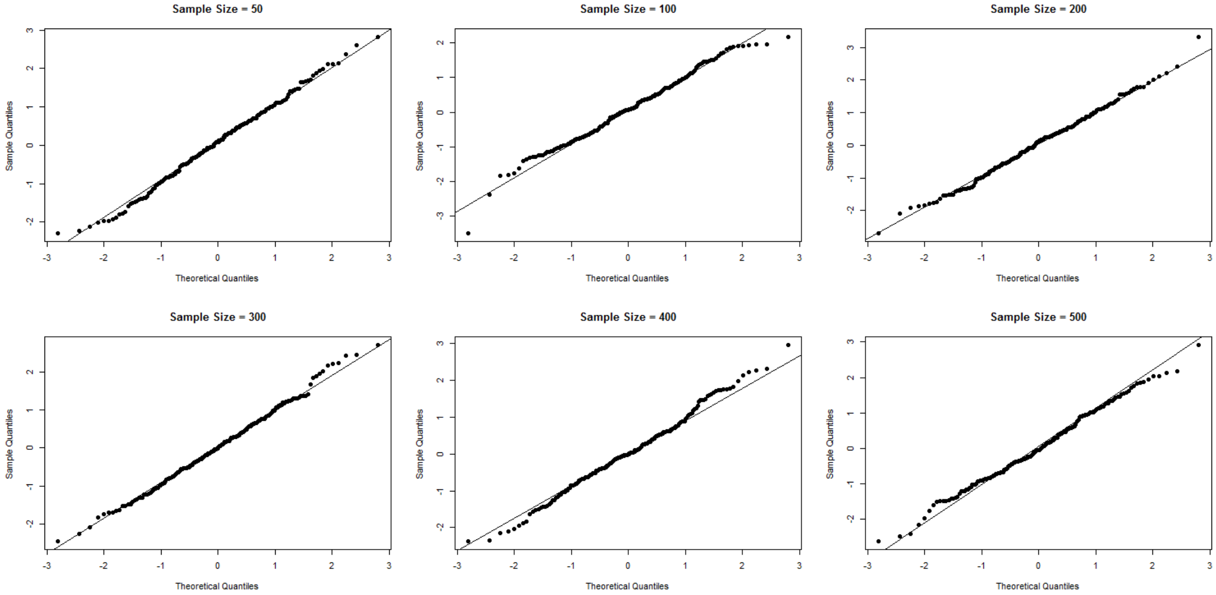

Figure 3.8: Q-Q plot with different sample sizes, where the true function is $f_0 = x^3 + 1$ with $\tau = 0.5$.

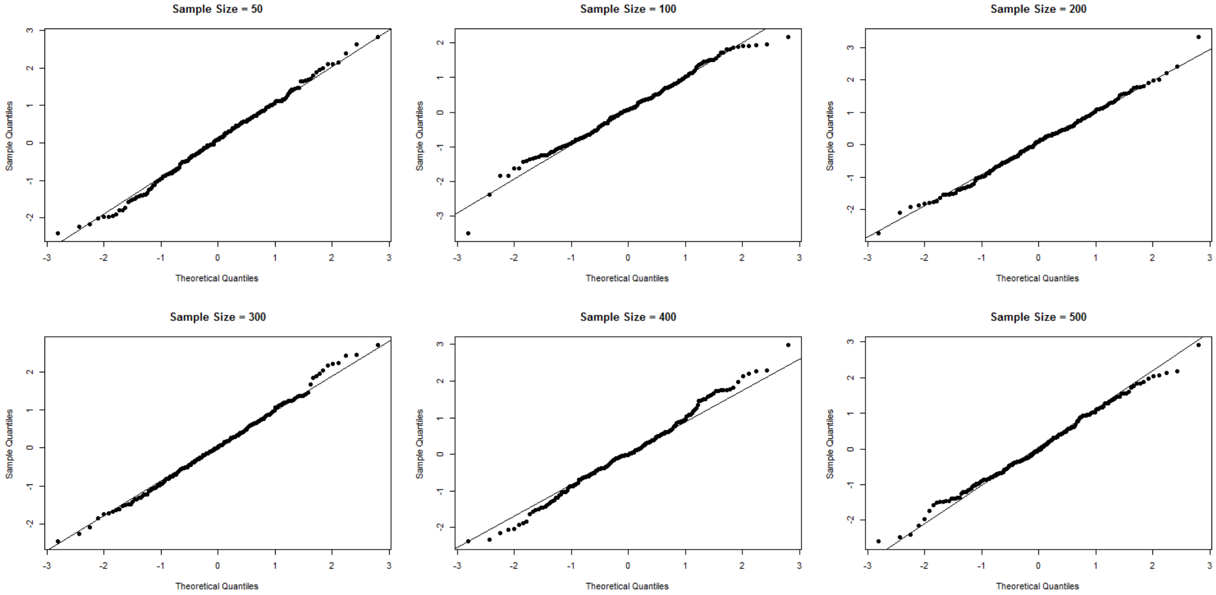

Figure 3.9: Q-Q plot with different sample sizes, where the true function is $f_0 = sin(x) + 2exp((-16)x^2)$ with $\tau = 0.5$.

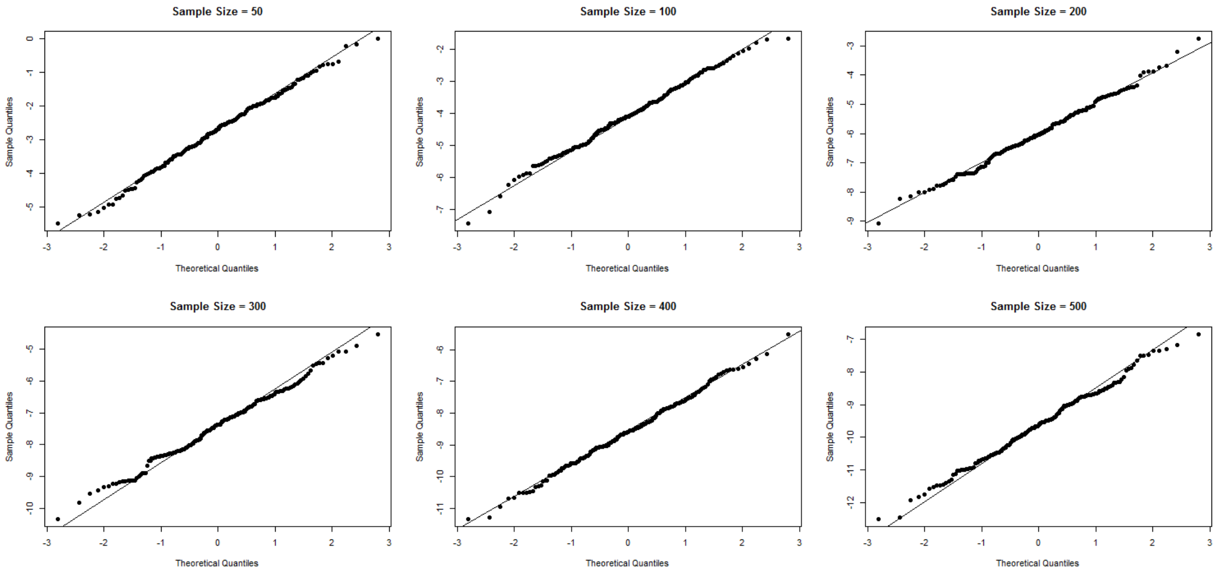

Figure 3.10: Q-Q plot with different sample sizes, where the true function f_0 is a neural network with one single hidden layer and two hidden units with $\tau = 0.75$.

3.6.2.2 Simulation result of normality with $\tau = 0.75$

From Figure 3.10 to Figure 3.12,we used the same test statistic and data points appeared as roughly a straight line. Based on the simulation results, we demonstrated the validity of normality of ENN.

3.7 Summary and discussion

In this section, we study the consistency and normality of ENN sieve estimators with one hidden layer. To overcome the issue of unidentifiability, we use the method of sieve to narrow down the choice of parametric space. The covering numbers is used to find an approximation to a rich class $\mathcal{F}_{r_n}$. By establishing an upper bound for the covering number of $\mathcal{F}_{r_n}$, we prove the consistency and normality of ENN. To check the validity of theoretical results, we also ran simulations based on the theorem conditions. If we choose τ as 0.5, then ENN becomes the traditional neural network. The ENN method inherits advantages from both neural networks

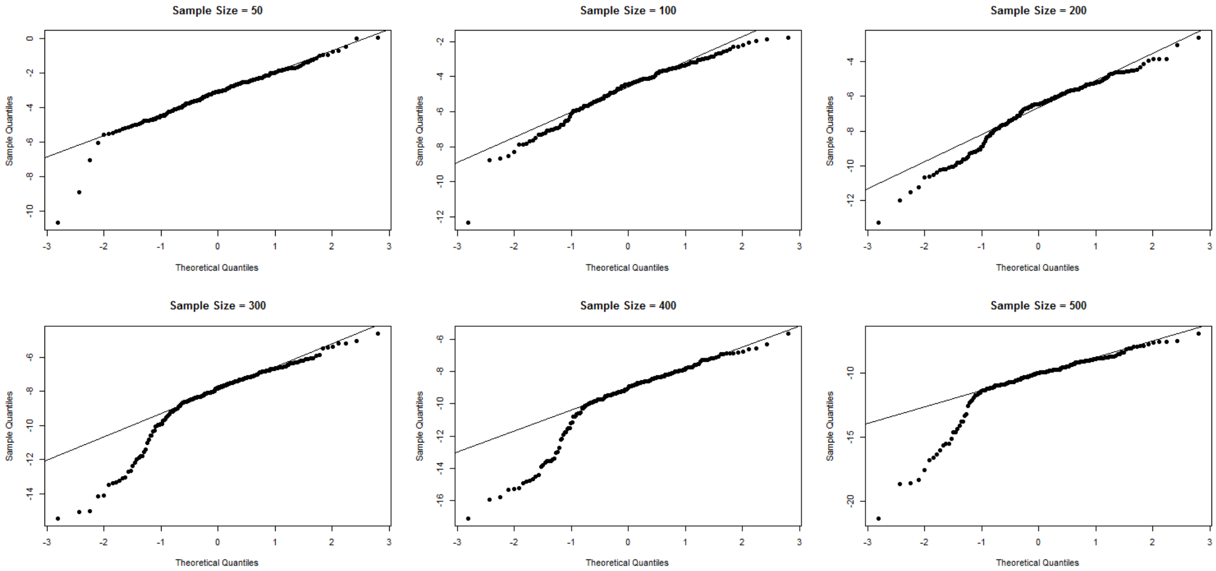
Sample Size = 50
Sample Size = 100
Sample Size = 200
Sample Size = 300
Sample Size = 400
Sample Size = 500

Figure 3.11: Q-Q plot with different sample sizes, where the true function is $f_0 = x^3 + 1$ with $\tau = 0.75$.

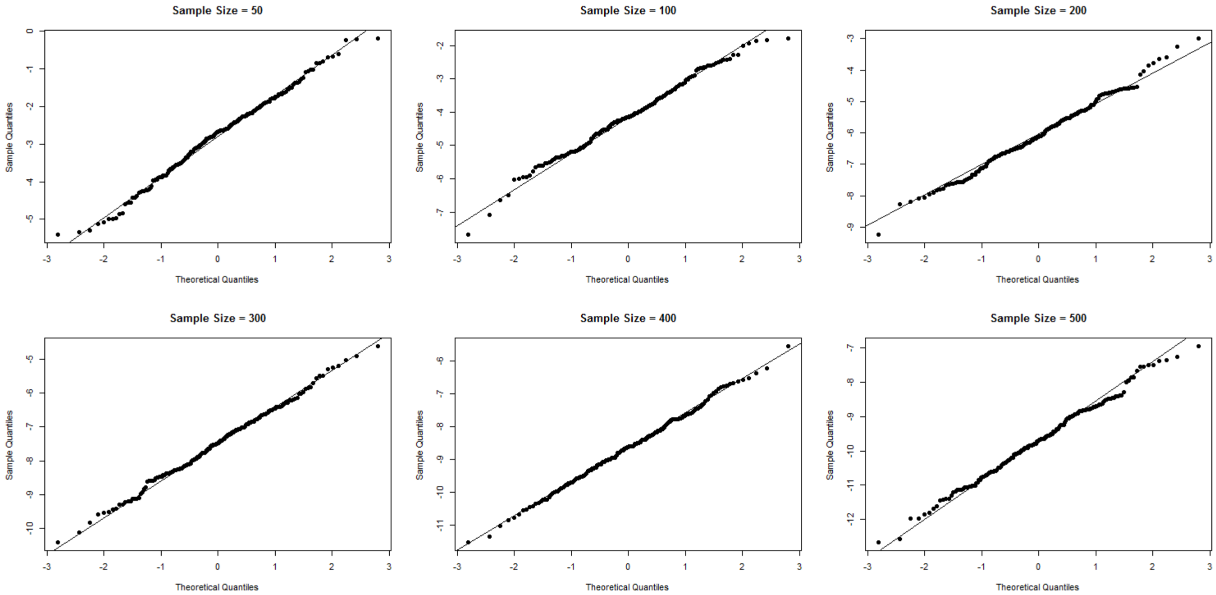
Sample Size = 50
Sample Size = 100
Sample Size = 200
Sample Size = 300
Sample Size = 400
Sample Size = 500

Figure 3.12: Q-Q plot with different sample sizes, where the true function is $f_0 = sin(x) + 2exp((-16)x^2)$ with $\tau = 0.75$.

and expectile regression. Using the hierarchical structure from neural networks, ENN can learn complex and abstract features from covariates, making it suitable for modeling the complex relationship between covariates and response by tuning hyperparameter τ.

Although we focus on one hidden layer neural network sieve estimators with sigmoid activation function in this chapter, the results of this chapter can be extended to other neural networks and activation functions. For instance, it can be potentially extended to other popular activation functions (e.g., the rectified linear unit). Deep neural network structures are commonly used in convolutional neural networks and recurrent neural networks. Therefore, it is worthwhile to investigate the asymptotic theory of different neural network architectures.

It may be also worthwhile to consider the regularization of neural networks into consideration. Since the number of parameters in deep neural networks is large, the overfitting issue is common in practice. Dropout is an approach of regularization in neural networks, which reduces the number of hidden units [53]. In statistics, we also add a penalty term as a regularization approach. To avoid overfitting, it is common to add a penalty term. Establishing the asymptotic theory of neural networks with regularization is crucial when we apply neural networks into real data analysis. We will consider this problem in the future.

Chapter 4

Summary and Discussion

This dissertation focuses mainly on developing a neural-network-based method, ENN, with application in risk prediction of genetic data. We also study the statistical properties of ENN, including consistency and normality.

In chapter 2, we develop a neural-network-based method called ENN. To demonstrate the performance of ENN, we run three different simulation settings: nonlinear, interactions among SNPs and interactions between genes. If there are nonlinear or high-order interaction effects in genetic data, ENN outperforms ER. To model the more complex relationship between genotypes and phenotypes, we change the architecture of ENN to non-fully connected architecture. By tuning the hyperparameter τ, ENN can provide a comprehensive view of the genotype-phenotype relationship for different expectile levels. Different expectile levels could also help us to identify high-risk individuals for certain disease, especially at the low expectile level($\tau = 0.1$) and the high expectile level($\tau = 0.9$).

Through two real data applications, we also demonstrate that ENN outperforms ER when the underlying genotype-phenotype relationships become complicated. For different expectiles, genetic effects vary, which provides us more information about the genotype-phenotype relationship via the conditional distributions. By studying different expectile levels, it may help us to predict high-risk individuals since genetic variations can have large effects on a particular disease.

In chapter 3, we study the consistency and normality of ENN sieve estimators with one hidden layer. We consider neural networks as a nonparametric regression problem to avoid the issue of unidentifiability. The method of sieve is used to narrow down the choice of parametric space. To measure the complexity of neural networks, we use covering number as measurement. By establishing an upper bound for the covering number of $\mathcal{F}_{r_n}$, we first prove the uniform law of large numbers of ENN. With some regularity conditions, we also prove the consistency and normality of ENN. Simulations have also been conducted to test the validity of theoretical results.

Most complex diseases are not only explained by genetic effects but also can be influenced by environmental determinants, which can be physical, chemical, biological, behavior patterns or life events. A small difference in one person's genes can cause them to respond differently to the same environmental exposure to another person. As a result, some people may develop the disease after being exposed to the environment while others may not. Therefore, it is worthwhile to take environmental determinants into consideration. In the future, we could apply ENN to study a disease with a potential gene-environment interaction component. By doing this, we could gain a better understanding of the disease and increase prediction accuracy.

Many researchers focus on improving the prediction accuracy of neural network estimators, while the statistical inference based on neural network estimator is not fully studied. By establishing the asymptotic properties of ENN, it is worthwhile to investigate the statistical inference of ENN. We could also incorporate other machine learning techniques into ENN. For many genetic datasets, Caucasian samples are larger than African samples. Due to the limitation of African samples, we could first train ENN on Caucasian samples and get the estimator, which can be used to improve prediction accuracy in African samples. We also apply ENN into real data with transfer learning, which is described in the appendix. By using this technique, we could improve the performance of ENN.

APPENDICES

Appendix A

Technical Details of Chapter 2

Proof of theorem 2.3.1

Theorem A.0.1. *Let $L_\tau : Y \times \mathbb{R} \to [0, \infty)$ be the asymmetric least square loss function and Q be a distribution on $Y = [-M, M]$. Then, the inner $L_\tau - risks$ of Q could be defined as*

$$\mathcal{C}_{\tau,Q}(t) = \int_Y L_\tau(y,t)dQ(y), t = f(\mathbf{x_i}) \in \mathbb{R},$$

and the minimal inner $L_\tau - risk$ is

$$\mathcal{C}^*_{L_\tau,Q} = inf_{t \in \mathcal{R}} \mathcal{C}_{L_\tau,Q}(t).$$

Lemma A.0.1. *Let L_τ be the asymmetric least square loss function and Q be a distribution on $\mathbb{R}$ with $\mathcal{C}^*_{L_\tau,Q} < \infty$. For a fixed $\tau \in (0,1)$ and for all $t \in \mathbb{R}$, we have*

$$c_\tau(t - t^*)^2 \leq \mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}^*_{L_\tau,Q} \leq C_\tau(t - t^*)^2,$$

where $c_\tau = min\{\tau, 1 - \tau\}$ and $C_\tau = max\{\tau, 1 - \tau\}$, t^* is $\tau-$expectile .

Proof. Let us fix $\tau \in (0,1)$. We use the result obtained in Newey and Powell [16]. For a distribution Q on $\mathcal{R}$ satisfies $\mathcal{C}^*_{L_\tau,Q} < \infty$, the $\tau-$expectile t^* is the only solution of

$$\tau \int_{y \geq t^*} (y - t^*) dQ(y) = (1 - \tau) \int_{y < t^*} (t^* - y) dQ(y). \tag{A.1}$$

First, We consider the lower bound.

To obtain the inner $L_\tau-$risks of Q, we consider two cases: $t \geq t^*$ and $t < t^*$.

When $t \geq t^*$, we have

$$\begin{aligned}
\int_{y<t} (y-t)^2 dQ(y) &= \int_{y<t} (y - t^* + t^* - t)^2 dQ(y) \\
&= \int_{y<t} (y-t^*)^2 dQ(y) + 2(t^* - t) \int_{y<t} (y - t^*) dQ(y) \\
&+ (t^* - t)^2 Q((-\infty, t)) \\
&= \int_{y<t^*} (y-t^*)^2 dQ(y) + \int_{t^* \leq y < t} (y-t^*)^2 dQ(y) + (t^* - t)^2 Q((-\infty, t)) \\
&+ 2(t^* - t) \int_{y<t^*} (y - t^*) dQ(y) + 2(t^* - t) \int_{t^* \leq y < t} (y - t^*) dQ(y),
\end{aligned}$$

and

$$\begin{aligned}
\int_{y \geq t} (y-t)^2 dQ(y) = \int_{y \geq t^*} (y-t^*)^2 dQ &- \int_{t^* \leq y < t} (y-t^*)^2 dQ(y) + (t^* - t)^2 Q([t, \infty)) \\
&+ 2(t^* - t) \int_{y \geq t^*} (y - t^*) dQ(y) - 2(t^* - t) \int_{t^* \leq y < t} (y - t^*) dQ(y).
\end{aligned}$$

By definition and (13), we have

$$
\begin{aligned}
\mathcal{C}_{L_\tau,Q}(t) &= (1-\tau)\int_{y<t}(y-t)^2dQ(y) + \tau\int_{y\geq t}(y-t)^2dQ(y) \\
&= (1-\tau)\int_{y<t^*}(y-t^*)^2dQ(y) + \tau\int_{y\geq t^*}(y-t^*)dQ(y) \\
&+ 2(t^*-t)((1-\tau)\int_{y<t^*}(y-t^*)dQ(y) + \tau\int_{y\geq t^*}(y-t^*)dQ(y)) \\
&+ (t^*-t)^2(1-\tau)Q((-\infty,t)) + (t^*-t)^2\tau Q([t,\infty)) \\
&+ (1-2\tau)\int_{t^*\leq y<t}(y-t^*)^2dQ(y) + 2(1-2\tau)(t^*-t)\int_{t^*\leq y<t}(y-t^*)dQ(y) \\
&= \mathcal{C}_{L_\tau,Q}(t^*) + (t^*-t)^2(1-\tau)Q((-\infty,t)) + (t^*-t)^2\tau Q([t,\infty)) \\
&+ (1-2\tau)\int_{t^*\leq y<t}(y-t^*)^2 + 2(t^*-t)(y-t^*)dQ(y)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}_{L_\tau,Q}(t^*)\\
&= (t^*-t)^2(1-\tau)Q((-\infty,t^*)) + (t^*-t)^2(1-\tau)Q([t^*,t)) + (t^*-t)^2\tau Q([t,\infty))\\
&+ (1-2\tau)\int_{t^*\le y<t}(y-t^*)^2 + 2(t^*-t)(y-t^*)dQ(y)\\
&= (t^*-t)^2((1-\tau)Q((-\infty,t^*)) + \tau Q([t,\infty)))\\
&- \tau\int_{t^*\le y<t}(y-t^*)^2 + 2(t^*-t)(y-t^*)dQ(y)\\
&+ (t^*-t)^2(1-\tau)Q([t^*,t)) + (1-\tau)\int_{t^*\le t<t}(y-t^*)^2 + 2(t^*-t)(y-t^*)dQ(y)\\
&= (t^*-t)^2((1-\tau)Q((-\infty,t^*)) + \tau Q([t,\infty))) - \tau\int_{t^*\le y<t}(y-t^*)(y+t^*-2t)dQ(y)\\
&+ (1-\tau)\int_{t^*\le y<t}(y-t^*)^2 + 2(t^*-t)(y-t^*) + (t^*-t)^2dQ(y)\\
&= (t^*-t)^2((1-\tau)Q(-\infty,t^*)) + \tau Q([t,\infty))) + \tau\int_{t^*\le y<t}(y-t^*)(2t-t^*-y)dQ(y)\\
&(1-\tau)\int_{t^*\le y<t}(y-t)^2dQ(y). \qquad \text{(A.2)}
\end{aligned}
$$

This leads to the lower bound of inner $L_\tau-$risk when $t \ge t^*$,

$$
\begin{aligned}
&\mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}_{L_\tau,Q}(t^*)\\
&\ge c_\tau(t^*-t)^2(Q((-\infty,t^*)) + Q([t,\infty))) + c_\tau\int_{t^*\le y\le t}(y-t^*)(2t-t^*-y) + (y-t)^2dQ(y)\\
&= c_\tau(t^*-t)^2(Q((-\infty,t^*)) + Q([t,\infty))) + c_\tau\int_{t^*\le y\le t}(t^*)^2 - 2tt^* + t^2dQ(y)\\
&= c_\tau(t^*-t)^2(Q((-\infty,t^*)) + Q([t,\infty))) + c_\tau(t^*-t)^2Q([t^*,t))\\
&= c_\tau(t^*-t)^2.
\end{aligned}
$$

When $t < t^*$, using similar arguments, we have

$$\begin{aligned}\mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}_{L_\tau,Q}(t^*) &= (t^* - t)^2((1-\tau)Q((-\infty, t)) + \tau \int_{t \le y < t^*} (y-t)^2 dQ(y) \\ &+ (1-\tau)\int_{t \le y < t^*} (t^* - y)(y + t^* - 2t) dQ(y) + +\tau Q([t^*, \infty))) \\ &\ge c_\tau (t^* - t)^2.\end{aligned}$$

Therefore, we summarize them into one inequality

$$\mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}_{L_\tau,Q}(t^*) \ge c_\tau (t^* - t)^2.$$

Next, we consider the upper bound. Similarly, when $t \ge t^*$,

$$\begin{aligned}&\mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}_{L_\tau,Q}(t^*) \\ &\le C_\tau (t^* - t)^2 (Q((-\infty, t^*)) + Q([t, \infty))) \\ &+ C_\tau \int_{t^* \ge y < t} ((y - t^*)(2t - t^* - y) + (y-t)^2) dQ(y) \\ &= C_\tau (t^* - t)^2. \end{aligned} \tag{A.3}$$

For the case of $t < t^*$, the inequality still holds. Combining these two inequality, we have

$$c_\tau (t - t^*)^2 \le \mathcal{C}_{L_\tau,Q}(t) - \mathcal{C}^*_{L_\tau,Q} \le C_\tau (t - t^*)^2.$$

□

Based on the Lemma A.1.1, we can prove Theorem 2.3.1[32].

Proof. If $x \in X$, we define $t = f(x)$ and $t^* = f^*_{L_\tau,P}(x)$. By Lemma 1, for $Q = P(\cdot|x)$, we can get the following result

$$C_\tau^{-1}\left(\mathcal{C}_{L_\tau,P(\cdot|x)}(f(x))) - \mathcal{C}^*_{L_\tau,P(\cdot|x)}\right) \leq |f(x) - f^*_{L_\tau,P}(x)|^2$$

and

$$|f(x) - f^*_{L_\tau,P}(x)|^2 \leq c_\tau^{-1}\left(\mathcal{C}_{L_\tau,P(\cdot|x)}(f(x)) - \mathcal{C}^*_{L_\tau,P(\cdot|x)}\right).$$

If we integrate it with respect to P_X and take the square root, we can get the final result. □

Appendix B

Technical Details of Chapter 3

Proof of lemma 3.3.2

Proof. For each fixed n, let $\boldsymbol{\theta}_n = [\alpha_0, \ldots, \alpha_{r_n}, \boldsymbol{\gamma}_{0,1}, \ldots, \gamma_{0,r_n}, \boldsymbol{\gamma}_1^T, \ldots, \boldsymbol{\gamma}_{r_n}^T]^T$ belong to $[-V_n, V_n]^{r_n+1} \times [-M_n, M_n]^{r_n(d+1)} := \Theta_n$. For n fixed, Θ_n is a bounded closed set and hence it is a compact set in $\mathbb{R}^{r_n(d+2)+1}$. Consider a map

$$\begin{aligned} H : (\Theta_n, \|\cdot\|_2) &\to (\mathcal{F}_{r_n}, \|\cdot\|_n) \\ \boldsymbol{\theta}_n &\mapsto H(\boldsymbol{\theta}_n) = \alpha_0 + \sum_{j=1}^{r_n} \alpha_j \sigma\left(\boldsymbol{\gamma}_j^T \boldsymbol{x} + \gamma_{0,j}\right) \end{aligned}$$

Note that $\mathcal{F}_{r_n} = H(\Theta_n)$. Therefore, to show that $\mathcal{F}_{r_n}$ is a compact set, it suffices to show that H is a continuous map due to the compactness of Θ_n. Let $\boldsymbol{\theta}_{1,n}, \boldsymbol{\theta}_{2,n} \in \Theta_n$, then

$$\begin{aligned}
&\|H(\boldsymbol{\theta}_{1,n}) - H(\boldsymbol{\theta}_{2,n})\|_n^2 \\
=&\frac{1}{n}\sum_{i=1}^{n}\left[\alpha_0^{(1)} + \sum_{j=1}^{r_n}\alpha_j^{(1)}\sigma\left({\boldsymbol{\gamma}_j^{(1)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(1)}\right) - \alpha_0^{(2)} - \sum_{j=1}^{r_n}\alpha_j^{(2)}\sigma\left({\boldsymbol{\gamma}_j^{(2)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(2)}\right)\right]^2 \\
\leq&\frac{1}{n}\sum_{i=1}^{n}\left[\left|\alpha_0^{(1)} - \alpha_0^{(2)}\right| + \sum_{j=1}^{r_n}\left|\alpha_j^{(1)}\sigma\left({\boldsymbol{\gamma}_j^{(1)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(1)}\right) - \alpha_j^{(2)}\sigma\left({\boldsymbol{\gamma}_j^{(2)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(2)}\right)\right|\right]^2 \\
=&\frac{1}{n}\sum_{i=1}^{n}\left[\left|\alpha_0^{(1)} - \alpha_0^{(2)}\right| + \sum_{j=1}^{r_n}|\alpha_j^{(1)}|\left|\sigma\left({\boldsymbol{\gamma}_j^{(1)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(1)}\right) - \sigma\left({\boldsymbol{\gamma}_j^{(2)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(2)}\right)\right| + \right. \\
&\qquad\qquad\qquad\qquad\qquad\qquad \left.|\alpha_j^{(1)} - \alpha_j^{(2)}|\sigma\left({\boldsymbol{\gamma}_j^{(2)}}^T\boldsymbol{x}_i + \gamma_{0,j}^{(2)}\right)\right]^2 \\
\leq&\frac{1}{n}\sum_{i=1}^{n}\left[\sum_{j=0}^{r_n}|\alpha_j^{(1)} - \alpha_j^{(2)}| + \frac{V_n}{4}\sum_{j=1}^{r_n}\left|\left(\boldsymbol{\gamma}_j^{(1)} - \boldsymbol{\gamma}_j^{(2)}\right)^T\boldsymbol{x}_i\right| + \left|\gamma_{0,j}^{(1)} - \gamma_{0,j}^{(2)}\right|\right]^2 \\
\leq&\left[\sum_{j=0}^{r_n}|\alpha_j^{(1)} - \alpha_j^{(2)}| + \frac{V_n}{4}(1 \vee \|\boldsymbol{x}\|_\infty)\sum_{j=1}^{r_n}\left\|\boldsymbol{\gamma}_j^{(1)} - \boldsymbol{\gamma}_j^{(2)}\right\|_1 + \left|\gamma_{0,j}^{(1)} - \gamma_{0,j}^{(2)}\right|\right]^2 \\
\leq&\left(\frac{V_n}{4}(1 \vee \|\boldsymbol{x}\|_\infty)\right)^2[r_n(d+1)]\|\boldsymbol{\theta}_{1,n} - \boldsymbol{\theta}_{2,n}\|_2^2.
\end{aligned}$$

Hence, for any $\epsilon > 0$, by choosing $\delta = \epsilon / \left(\frac{V_n}{4}(1 \vee \|\boldsymbol{x}\|_\infty)\sqrt{r_n(d+1)}\right)$, we observe that when $\|\boldsymbol{\theta}_{1,n} - \boldsymbol{\theta}_{2,n}\|_2 < \delta$, we have

$$\|H(\boldsymbol{\theta}_{1,n}) - H(\boldsymbol{\theta}_{2,n})\|_n < \epsilon,$$

which implies that H is a continuous map and hence $\mathcal{F}_{r_n}$ is a compact set for each fixed n. □

Appendix C

Supplementary Materials

Expectile neural networks with transfer learning

Normally, machine learning models focus on one single and specific task. If we have two related tasks, one task could inherit some information from the other task. We call this technique transfer learning. Transfer learning focuses on storing knowledge gained by solving one problem and applying the knowledge to a different but related problem. It is easier to transfer knowledge if tasks are more related. Transfer learning has been implemented in a wide area, like natural language processing (NLP)[69], medical image[66].

Transfer learning could be applied in both classification and regression scenarios. For example, Syed proposes seeded transfer learning in a regression context to improve prediction performance in target domain[71]. Many approaches could be implemented in transfer learning. Yosinski et al. show how lower layers in neural networks act as conventional computer-vision feature extractors, such as edge detectors, while the final layer works toward task-specific features[65]. Rosenstein uses naive Bayes classification algorithm to detect, perhaps implicitly, that the inductive bias learned from the auxiliary tasks will actually hurt performance on the target task [68]. In this chapter, we focused on applying the transfer learning technique into expectile neural networks. We focus on parameter transfer or instance reweighting. This approach works on the assumption that the models for related tasks share some parameters. There are some advantages of doing these. First, if the initial task and target task are relevant, we could improve our result. Second, since we inherit information from the initial task, the number of parameters in target task is reduced, which gives us some computational advantages, especially in large datasets.

Real data application

In this section, we integrate expectile regression and transfer learning to improve prediction performance. To verify if transfer learning works, we run two real data sets to compare the performance of ENN with transfer learning and ENN without transfer learning.

First real data application

Intuitively, participants in this study tend to be addicted to drinking who have the nicotine addiction. We applied ENN to the genetic data from the Study of Addiction: Genetics and Environment(SAGE). The participants of the SAGE are selected from three large, complementary studies: the Family Study of Cocaine Dependence(FSCD), the Collaborative study on the Genetics of Alcoholism(COGA), and the Collaborative Genetic Study of Nicotine Dependence(COGEND).

We choose max_cigs as smoking quantity, which is measured by the largest number of cigarettes smoked in 24 hours, ranged from 0-240. We choose max_drinks as drinking quantity, which is measured by the largest number of alcoholic drinks consumed in 24 hours, range from 0-258. To have better performance, we transfer smoking-related information to drinking-related information. We use the following algorithm.

First, we choose max_cigs as phenotype, and get the estimator of the expectile neural network. Second, we get the estimator obtained from the first step as the initial value(transfer

Table C.1: Real data application result of CHRNA5

	ENN.tsf		**ENN**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	551.83	605.79	546.90	672.44
0.25	325.84	439.18	321.94	473.10
0.5	282.57	433.058	275.83	444.16
0.75	304.81	484.60	297.81	487.44
0.9	347.17	544.24	339.79	549.08

Table C.2: Real data application result of CHRNA3

	ENN.tsf		**ENN**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	554.11	605.10	533.04	753.96
0.25	325.71	441.47	311.85	517.05
0.5	281.20	439.40	260.45	491.62
0.75	304.60	486.80	292.63	502.86
0.9	350.01	558.95	335.89	573.92

learning part). Third, we choose max_drinks as a new phenotype and keep the parameter from the input layer to the hidden layer and then train the expectile neural network again. Finally, we compare two models: ENN with transfer learning and ENN without transfer learning.

We divide the data into three parts: training(60%), validation(20%), testing(20%). We get the following results.

Table C.3: Real data application result of CHRNB4

	ENN.tsf		**ENN**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	558.39	622.18	564.57	673.97
0.25	327.63	448.63	325.48	473.34
0.5	283.28	435.11	270.50	453.76
0.75	306.05	488.15	303.02	489.71
0.9	349.24	544.85	343.52	553.41

Table C.1-C.3 summarize the MSE of ENN with transfer learning and ENN without transfer learning for five different expertiles (i.e., 0.1, 0.25, 0.5, 0.75, and 0.9). From those three tables, we show the expectile neural networks with transfer learning outperform expecilt neural networks without transfer learning.

Second real data application

In this real data application, we apply our method to the Alzheimer's Disease Neuroimaging Initiative(ADNI), which is a multisite study that aims to improve clinical trials for the prevention and treatment of Alzheimer's disease. APOE allele is the most important genetic risk factor for Alzheimer's disease[67]. We focus our ENN model on APOE gene. After quality control, 168 SNPs remained for the analysis. We only included 699 Caucasian and African American individuals due to the small sample size of other ethnic groups. To improve the performance of ENN, we also included 3 covariates: sex(male=1, female=2), age, and education in the analysis.

Hippocampus is the part of the brain area associated with memories. Alzheimer's disease usually first damages hippocampus, leading to memory loss and disorientation. Study shows that hippocampal volume and ratio was reduced by 25% in Alzheimer's disease[72]. The Mini-Mental State Examination (MMSE) is a 30-point questionnaire that is used extensively in clinical and research settings to measure cognitive impairment. For more information, refer to https://www.ncbi.nlm.nih.gov/projects/gap/cgi-bin/GetPdf.cgi?id=phd001525.1. We transfer Hippocampus_bl to MMSE.

To have stable performance, we randomly split the dataset 50 times and average the result.

From table C.4, expectile neural network with transfer learning outperforms expectile

Table C.4: Real data application result of ADNI

	ENN.tsf		**ENN**	
τ	**Train**	**Test**	**Train**	**Test**
0.1	8.21	8.40	8.65	9.50
0.25	5.00	5.17	5.30	6.78
0.5	4.11	4.31	4.30	4.82
0.75	4.67	4.88	4.85	6.86
0.9	5.87	6.10	5.99	6.69

regression without transfer learning under different τ.

Summary and discussion

From these two real data application, transfer learning improves performance of expectile neural networks. However, transfer learning relies on data heavily based on our experience. If the data does not fit the model, the negative transfer happens where the transfer of knowledge from the source to the target does not lead to any improvement, but rather causes a drop in the overall performance of the target task.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Genome-Wide Association Studies. National Human Genome Research Institute. https://www.genome.gov/about-genomics/fact-sheets/Genome-Wide-Association-Studies-Fact-Sheet.

[2] Manolio TA. Genome wide association studies and assessment of the risk of disease. The New England Journal of Medicine, 363 (2): 166–76, 2010.

[3] Kwon JM, Goate AM. The candidate gene approach. Alcohol Research & Health, 24 (3): 164–8, 2000.

[4] Xuexia Wang, Michael J Oldani, Xingwang Zhao, Xiaohui Huang, Dajun Qian. A Review of Cancer Risk Prediction Models with Genetic Variants. Cancer Inform, 13(2): 19–28, 2014.

[5] The Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature, 447, 661–678, 2007.

[6] Scott LJ, Mohlke KL, Bonnycastle LL, Willer CJ, Li Y, Duren WL, Erdos MR, Stringham HM, Chines PS, Jackson AU, Prokunina-Olsson L, Ding CJ, Swift AJ, Narisu N, Hu T, Pruim R, Xiao R, Li XY, Conneely KN, Riebow NL, Sprau AG, Tong M, White PP, Hetrick KN, Barnhart MW, Bark CW, Goldstein JL, Watkins L, Xiang F, Saramies J, Buchanan TA, Watanabe RM, Valle TT, Kinnunen L, Abecasis GR, Pugh EW, Doheny KF, Bergman RN, Tuomilehto J, Collins FS, Boehnke M. A genome-wide association study of type 2 diabetes in Finns detects multiple susceptibility variants. Science, 316(5829):1341-5, 2007.

[7] Nan M. Laird, Christoph Lange. The Fundamentals of Modern Statistical Genetics. Springer-Verlag, 2011.

[8] Miller DD, Brown EW. Artificial Intelligence in Medical Practice: The Question to the Answer? Am J Med, 131(2):129-133, 2018.

[9] Fakoor R, Ladhak F, Nazi A, Huber M. Using deep learning to enhance cancer diagnosis and classification. Proceedings of the 30th International Conference on Machine Learning, 2013.

[10] Goodfellow I, Bengio Y, Courville A. Deep Learning. The MIT Press, 96-161, 2016.

[11] Le Cun Y, Bengio Y, Hinton G. Deep learning. Nature, 521:436- 444, 2015.

[12] Krittanawong C, Zhang H, Wang Z, Aydar M, Kitai T. Artificial Intelligence in Precision Cardiovascular Medicine. J Am Coll Cardiol, 69(21):2657-2664, 2017.

[13] McClellan J, King MC. Genetic heterogeneity in human disease. Cell, 141(2):210-7, 2010.

[14] Marchini J, Donnelly P, Cardon LR. Genome-wide strategies for detecting multiple loci that influence complex diseases. Nat Genet, 37(4):413-7, 2005.

[15] R. Koenker, G.W. Bassett Jr. Regression quantiles. Econometrica, 46(1):33-50, 1978.

[16] W. Newey, J. Powell. Asymmetric least squares estimation and testing. Econometrica, 55(4):819-847, 1987.

[17] Moshe Buchinsky. Quantile regression, Box-Cox transformation model, and the U.S. wage structure, 1963–1987. Journal of Econometrics, 65(1):109-154, 1995.

[18] John Crowley, Marie Hu. Covariance Analysis of Heart Transplant Survival Data. Journal of the American Statistical Association, 72-357, 1977.

[19] Stuart R. Lipsitz Garrett M. Fitzmaurice Geert Molenberghs Lue Ping Zhao. Quantile Regression Methods for Longitudinal Data with Drop-outs: Application to CD4 Cell Counts of Patients Infected with the Human Immunodeficiency Virus. Jornal of the Royal Statistical Society: Applied Statistics Series C, 46(4):463-476, 1997.

[20] G.R.PandeyaV, T.V.Nguyenb. A comparative study of regression based methods in regional flood frequency analysis. Journal of Hydrology, 225:92-101, 1999.

[21] H. J. Cordell. Detecting gene-gene interactions that underlie human diseases, Nat. Rev. Genet. 10:392–404, 2009.

[22] A. Cannon. Non-crossing nonlinear regression quantiles by monotone composite quantile regression neural network, with application to rainfall extremes. A.J. Stoch Environ Res Risk Assess, 32:3207, 2018.

[23] A. Cannon. Quantile regression neural networks: Implementation in R and application to precipitation downscaling. Computers & Geosciences, 37:1277-1284, 2011.

[24] J. Taylor. A quantile regression neural network approach to estimating the conditional density of multiperiod returns. Journal of Forecasting, 19:299-311, 2000.

[25] C. Jiang, M. Jiang, Q. Xu, X. Huang. Expectile regression neural network model with applications. Neurocomputing, 247:73-86, 2017.

[26] L. Liao, C. Park, H. Choi. Penalized expectile regression: an alternative to penalized quantile regression. Ann Inst Stat, 71:409–438, 2018.

[27] L. Waltrup, F. Sobotka, T. Kneib, G. Kauermann. Expectile and quantile regression-David and Goliath? Statistical Modelling, 15(5): 433–456, 2015.

[28] M. Kim, S. Lee. Nonlinear expectile regression with application to Value-at-Risk and expected shortfall estimation. Computational Statistics and Data Analysis, 94:1-19, 2016.

[29] Q. Yao, H. Tong. Asymmetric least squares regression estimation: a nonparametric approach. Journal of Nonparametric Statistics, 6:2-3, 1996.

[30] Durbin, R., Altshuler, D., Durbin, R. et al. A map of human genome variation from population-scale sequencing. Nature, 467:1061–1073, 2010.

[31] Li MD, Xu Q, Lou XY, Payne TJ, Niu T, Ma JZ. Association and interaction analysis of variants in CHRNA5/CHRNA3/CHRNB4 gene cluster with nicotine dependence in African and European Americans. Am J Med Genet B Neuropsychiatr Genet, 153B(3):745–756, 2010.

[32] M. Farooq, I. Steinwart. Learning rate for kernel-based expectile regression. Mach Learning, 108: 203–227, 2019.

[33] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. Neural Networks, 251-257, 1991.

[34] Fletcher, Roger, Practical methods of optimization(2nd ed.), New York: John Wiley & Sons, 1987.

[35] Heather J. Cordell, Detecting gene-gene interactions that underlie human diseases. Nat Rev Genet, 10(6):392–404, 2009.

[36] Mackay, T.F. Quantitative trait loci in Drosophila. Nat. Rev. Genet, 2:11–20, 2001.

[37] Routman EJ, Cheverud JM. Gene effects on a quantitative trait: Two-locus epistatic effects measured at microsatellite markers and at estimated QTL. Evolution, 51: 1654–1662, 1997.

[38] Zerba, K.E., Ferrell, R.E. & Sing, C.F. Complex adaptive systems and human health: the influence of common genotypes of the apolipoprotein E (ApoE) gene polymorphism and age on the relational order within a field of lipid metabolism traits. Hum. Genet, 107: 466–475, 2000.

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan SalakhutdinovDropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15: 1929-1958, 2014.

[40] Chengxi Ye, Yezhou Yang, Cornelia Fermuller, Yiannis Aloimonos. On the Importance of Consistency in Training Deep Neural Networks, arXiv:1708.00631, 2017.

[41] Anthony, M. and Bartlett, P.L., Neural network learning: Theoretical foundations, Cambridge university press, 2009.

[42] X Chen. Large sample sieve estimation of semi-nonparametric models. Handbook of econometrics, 2007.

[43] Kurt Hornik, Maxwell Stinchcombe, Halbert White. Multilayer feedforward networks are universal approximators. Neural newtorks, 2(5):359-366, 1989.

[44] László Györfi. A Distribution-Free Theory of Nonparametric Regression. Springer New York, 2006.

[45] Jinghang Lin, Xiaoran Tong, Chenxi Li, Qing Lu. Expectile Neural Networks for Genetic Data Analysis of Complex Diseases, arXiv:2010.13898, 2020.

[46] Grenander. Abstract Inference. Wily, New York, 1981.

[47] White, H. and Wooldridge, J. Some results on sieve estimation with dependent observations. In Nonparametric and Semiparametric Methods in Economics (W. A. Barnett, J. Powell and G. Tauchen, eds.) 459-493. Cambridge University Press New York. 1991.

[48] Van der Vaart. Asymptotic Statistics, Cambridge University Press, 1998.

[49] Van der Vaart, Jon A. Wellner. Weak convergence and empirical processes. Springer, 1996.

[50] Van de Geer. Empirical Processes in M-estimation. Cambridge university press, 2020.

[51] Xiaoxi Shen, Chang Jiang, Lyudmila Sakhanenko, Qing Lu. Asymptotic Properties of Neural Network Sieve Estimators, arXiv:1906.00875, 2019.

[52] Xiaotong Shen, On Methods of sieves and penalization. The Annals of Statistics, 25(6):2555-2591, 1997.

[53] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Advances in neural information processing systems, 2012.

[54] Koenker, Roger. Quantile regression. Cambridge University Press, 2005.

[55] Kenji Fukumizu. A regularity condition of the information matrix of a multilayer perceptron network. Neural networks, 9(5):871–879, 1996.

[56] Kenji Fukumizu et al. Likelihood ratio of unidentifiable models and multilayer neural networks. The Annals of Statistics, 31(3):833–851, 2003.

[57] Hongtu Zhu and Heping Zhang. Asymptotics for estimation and testing procedures under loss of identifiability. Journal of Multivariate Analysis, 97(1):19–45, 2006.

[58] Ergün Akgün, Metin Demir. Modeling Course Achievements of Elementary Education Teacher Candidates with Artificial Neural Networks. International Journal of Assessment Tools in Education, 2018.

[59] T. Pham, T. Tran, D. Phung, S. Venkatesh. Predicting healthcare trajectories from medical records: a deep learning approach. J Biomed Inform, 69:218-229, 2017.

[60] Plis, Sergey M. and Hjelm, Devon R. and Salakhutdinov, Ruslan and Allen, Elena A. and Bockholt, Henry J. and Long, Jeffrey D. and Johnson, Hans J. and Paulsen, Jane S. and Turner, Jessica A. and Calhoun, Vince D. Deep learning for neuroimaging: a validation study. Front. Neurosci, 229: 8, 2014.

[61] Devroye, Luc and Györfi, László and Lugosi, Gábor. A probabilistic theory of pattern recognition,Springer Science & Business Media, 2013.

[62] Martin Abadi and Paul Barham and Jianmin Chen and Zhifeng Chen and Andy Davis and Jeffrey Dean and Matthieu Devin and Sanjay Ghemawat and Geoffrey Irving and Michael Isard and Manjunath Kudlur and Josh Levenberg and Rajat Monga and Sherry Moore and Derek G. Murray and Benoit Steiner and Paul Tucker and Vijay Vasudevan and Pete Warden and Martin Wicke and Yuan Yu and Xiaoqiang Zheng. TensorFlow: A system for large-scale machine learning. 12th USENIX Symposium on Operating Systems Design and Implementation, 2016.

[63] Adam Paszke and S. Gross and Francisco Massa and A. Lerer and James Bradbury and Gregory Chanan and Trevor Killeen and Z. Lin and N. Gimelshein and L. Antiga and Alban Desmaison and Andreas Köpf and Edward Yang and Zach DeVito and Martin Raison and Alykhan Tejani and Sasank Chilamkurthy and Benoit Steiner and Lu Fang and Junjie Bai and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS, 2019.

[64] Jindong Wang, Yiqiang Chen, Han Yu, Meiyu Huang, and Qiang Yang. Easy Transfer Learning By Exploiting Intra-domain Structures. arXiv preprint arXiv:1904.01376, 2019.

[65] J Yosinski, J Clune, Y Bengio, H Lipson. How transferable are features in deep neural networks? Advances in neural information processing systems, 3320-3328, 2014.

[66] Amin Khatami, Morteza Babaie, H.R. Tizhoosh, Abbas Khosravi, Thanh Nguyen, Saeid Nahavandi. A sequential search-space shrinking using CNN transfer learning and a

radon projection pool for medical image retrieval. Expert Systems with Applications, 100:224–233, 2018.

[67] Liu Y, Tan L, Wang HF, Liu Y, Hao XK, Tan CC, Jiang T, Liu B, Zhang DQ, Yu JT; Alzheimer's Disease Neuroimaging Initiative. Multiple Effect of APOE Genotype on Clinical and Neuroimaging Biomarkers Across Alzheimer's Disease Spectrum. Mol Neurobiol, 53(7):4539-47, 2016.

[68] M.T. Rosenstein, Z. Marx and L.P. Kaelbling, To Transfer or Not to Transfer. Neural Information Processing Systems, Workshop Inductive Transfer: 10 Years Later, 2005.

[69] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1-67, 2020.

[70] S. J. Pan, Q. Yang. A Survey on Transfer Learning, IEEE Transactions on Knowledge and Data Engineering, 22(10): 1345-1359, 2010.

[71] S.M. Salaken, A. Khosravi, T. Nguyen, S. Nahavandi. Seeded transfer learning for regression problems with deep learning. Expert Syst. Appl, 115:565-577, 2019.

[72] Vijayakumar A . Comparison of hippocampal volume in dementia subtypes. ISRN Radiol, 2012.

[73] Yoshua Bengio. Deep Learning of Representations for Unsupervised and Transfer Learning. Proceedings of ICML Workshop on Unsupervised and Transfer Learning, JMLR Workshop and Conference Proceedings, 27:17-36, 2012.

[74] Ye Jia, Yu Zhang, Ron J Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Igna-cio Lopez Moreno et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. Conference on Neural Information Processing Systems, 2018.

[75] Zhilin Yang, Ruslan Salakhutdinov, William W Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks.ICLR, 2017.

[76] Davenport T, Kalakota R. The potential for artificial intelligence in healthcare. Future Healthc J. 6(2):94-98, 2019.

[77] Chien-Fu Wu. Asymptotic theory of nonlinear least squares estimation. The Annals of Statistics, 501–513, 1981.

[78] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. The elements of statistical learning, volume 1. Springer series in statistics New York, 2001.