이학석사 학위논문

Spatio-Temporal Prediction Using Deep Neural Networks

딥러닝을 활용한 시공간 데이터 예측

2023 년 8 월

서울대학교 대학원

통계학과

박 태 준

Spatio-Temporal Prediction Using Deep Neural Networks

딥러닝을 활용한 시공간 데이터 예측

지도교수 임 채 영

이 논문을 이학석사 학위논문으로 제출함

2023 년 4 월

서울대학교 대학원

통계학과

박 태 준

박 태 준의 이학석사 학위논문을 인준함

2023 년 7 월

Abstract

Spatio-Temporal Prediction Using Deep Neural Networks

Park Tae Jun
Department of Statistics
The Graduate School
Seoul National University

Kriging provides the Best Linear Unbiased Predictor (BLUP) for a spatial data or spatio-temporal data. This is a method of interpolation used to predict spatial process or spatio-temporal process at unobserved locations. However, for complex data, Kriging, the linear predictor, may not be optimal. Nowadays, Deep learning using Deep neural networks (DNNs) is being used in many fields. Deep feedforward networks can be used for regression, so I propose a novel prediction method using DNN structure in this study. This method may learn more complex spatio-temporal dependencies. Next, I study the traditional Kriging and my method in terms of statistical learning theory. Finally, I apply my method to Korea fine dust data to evaluate the performance. Here, the K-fold Cross Validation method for spatio-temporal data is used.

Keywords: Spatio-temporal data, Kriging, Deep learning, Deep Neural Networks, Deep feedforward networks, Statistical learning theory
Student Number: 2021-24984

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Among many types of data today, spatial data refers to data including spatial location information. The spatio-temporal data includes information indexed by time stamps to spatial data. For example, temperature data by time and location and precipitation data by time and location, etc. Since spatio-temporal data has spatio-temporal dependence, it is necessary to analyze it by applying a spatio-temporal statistical methodology.

In the spatio-temporal statistics, prediction (i.e., interpolation) in unobserved location and time is one of the main purposes (Wikle et al., 2019). Spatio-temporal predictor can be obtained by minimizing the certain optimization criterion. The Best Linear Unbiased Predictor (BLUP) that minimizing the Mean Square Prediction Error (MSPE) is called Kriging predictor (Cressie et al., 2011). There are different types of Kriging depending on the additional assumptions. So when the data do not follow the such assumption, Kriging method no more provide the optimal predictor in a sense of minimum MSPE. We need an interpolation method for more general cases.

Deep learning architectures using Deep Neural Networks (DNNs) have outperformed the state-of-the-art in various fields in modern society. It is widely used for supervised learning, a common form of machine learning (Yann LeCun et al., 2015). So this algorithm can be applied for a regression that estimates the relationships between a label and features. The usage of DNNs in regression allows deriving more complex and abstract structures (Floriän Kastner et al., 2018). I apply these properties to regression for complex spatio-temporal data.

Deep learning has also made many advances in dependent data. This method has been very successful in image recognition or natural language processing. In the same context, it is used for spatial and spatio-temporal data which have dependence (Wikle et al., 2022). The study of Chen et al.(2021) is one of the works on it and this is most related to my work. DeepKriging spatial predictor Chen et al. (2021) proposed has multiple advantages over Kriging for non-Gaussian and non-stationary spatial data by using simple DNNs. I take a step further by using an idea similar to Chen et al. (2021). I study the prediction in spatio-temporal data with added temporal domains. In addition, considering the structure of the spatio-temporal random process model, I make a modification to the DNN structure.

In this work, I propose a spatio-temporal predictor using a DNNs. This method overcomes the aforementioned limitations of traditional kriging and learns the latent spatio-temporal process better. I introduce the spatio-temporal model and spatio-temporal (universal) kriging first to understand the structure of the methodology. And I introduce Deep feedforward networks, a simple structure of DNNs in Preliminaries. Next, I propose a spatio-temporal prediction method using DNNs in detail and theoretical properties. Finally, I apply the method to predict the value of $PM_{2.5}$ concentrations in Korea. Here, the detailed structure set in the practical step is described and the cross-validation for evaluating its performance is considered.

Chapter 2

Preliminaries

In this study, the prediction methodology using DNNs is established based on a spatio-temporal model. So in this chapter, I introduce the spatio-temporal model, spatio-temporal universal kriging predictor, and Deep feedforward networks, also called multilayer perceptrons (MLPs). MLPs are the typical simple DNNs model.

2.1 Spatio-Temporal Model

The most of notations and basic concepts of spatio-temporal statistics follow Cressie et al., (2011), Montero et al., (2015), and Wikle et al., (2019). Let $D_s \times D_t$ be the space-time domain where $D_s \subset \mathbb{R}^d$ is the spatial domain for the spatial dimension d and $D_t \subset \mathbb{R}$ is the temporal domain. Consider the N observed spatio-temporal data given by

$$\mathbf{Z} = \{Z(\mathbf{s}_1; t_1), \cdots, Z(\mathbf{s}_N; t_N)\}^\top$$

where $(\mathbf{s}_i; t_i) \in D_s \times D_t,\ i = 1, \cdots, N$. Suppose we have m observations at each time $t_j,\ j = 1, \cdots, T$. Then we have

$$\mathbf{Z} = \{Z(\mathbf{s}_{1,1}; t_1), \cdots, Z(\mathbf{s}_{m,1}; t_1), \cdots, Z(\mathbf{s}_{1,T}; t_T), \cdots, Z(\mathbf{s}_{m,T}; t_T)\}^\top.$$

Assume our observations arose from a random spatio-temporal process,

$$\{Y(\mathbf{s}; t) : \mathbf{s} \in D_s,\ t \in D_t\},$$

with *iid* measurement error, ε, that has $\mathbb{E}[\varepsilon] = 0, \mathrm{Var}(\varepsilon) = \sigma_\varepsilon^2$ and is independent of Y. That is, $\mathbf{Z}$ is the realization of the process $Y(\mathbf{s};t)$. So the statistical model for our spatio-temporal data is

$$Z(\mathbf{s}_{ij};t_j) = Y(\mathbf{s}_{ij};t_j) + \varepsilon(\mathbf{s}_{ij};t_j), \tag{2.1}$$

for $i = 1, \cdots, N$ and $j = 1, \cdots, T$. We now suppose that the spatio-temporal process follows the model

$$Y(\mathbf{s};t) = \mu(\mathbf{s};t) + \eta(\mathbf{s};t),$$

for all $(\mathbf{s};t) \in D_s \times D_t$, where $\mu(\mathbf{s};t)$ is a deterministic mean and $\eta(\mathbf{s};t)$ is a mean-zero random effect capturing the spatio-temporal dependence.

In the data model (2.1), we attempt to find the best predictor $\hat{Y}$ of the true process Y. This is supervised learning (Hastie et al., 2009). We predict $Y(\mathbf{s}_0;t_0)$ at an unobserved point $(\mathbf{s}_0;t_0)$ (i.e., $\mathbf{s}_0 \notin \{\mathbf{s}_{1,1}, \cdots, \mathbf{s}_{m,1}, \cdots, \mathbf{s}_{1,T}, \cdots, \mathbf{s}_{m,T}\}$ and $t_0 \notin \{t_1, \cdots, t_T\}$) by minimizing the MSPE between $Y(\mathbf{s}_0;t_0)$ and $\hat{Y}(\mathbf{s}_0;t_0)$. MSPE is the most commonly used interpolation criterion. In this sense, if we further consider the best linear unbiased predictor, it is kriging. We attempt to $\mu(\mathbf{s};t)$ consists of p covariates, that is $\mu(\mathbf{s};t) = \mathbf{x}^1(\mathbf{s};t)^\top \boldsymbol{\beta}$ where $\mathbf{x}^1(\mathbf{s};t) = (1, \mathbf{x}(\mathbf{s};t)^\top)^\top \in \mathbb{R}^{p+1}$ is a vector of $p+1$ known covariates including 1 and $\boldsymbol{\beta}$ is a vector of coefficients. This results in spatio-temporal universal kriging.

2.2 Spatio-Temporal Universal Kriging

Consider the spatio-temporal universal Kriging, which derives the best linear unbiased prediction from the above process model. The spatio-temporal process

$Y(\mathbf{s};t)$ is modeled by

$$\begin{aligned} Y(\mathbf{s};t) &= \mu(\mathbf{s};t) + \eta(\mathbf{s};t) \\ &= \mathbf{x}^1(\mathbf{s};t)^\top \boldsymbol{\beta} + \eta(\mathbf{s};t). \end{aligned} \tag{2.2}$$

We assume that the random process is a Gaussian process as usual assumptions, denoted

$$Y(\mathbf{s};t) \sim GP\left(\mu(\mathbf{s};t), c(\cdot;\cdot)\right),$$

determined by a mean function $\mu(\mathbf{s};t) = \mathbb{E}[Y(\mathbf{s};t)] = \mathbf{x}^1(\mathbf{s};t)^\top \boldsymbol{\beta}$ and a covariance function $c\left(\mathbf{s}, \mathbf{s}'; t, t\right) = \mathrm{Cov}\left(Y(\mathbf{s};t), Y(\mathbf{s}';t')\right)$ which is valid (i.e., non-negative-definite) for all $\{(\mathbf{s};t), (\mathbf{s}';t')\} \in D_s \times D_t$. In practice, the process is assumed to be second-order stationary, which implies that $\mu(\mathbf{s};t)$ is constant and the covariance function can be expressed in $c(\mathbf{s}, \mathbf{s}'; t, t') = c(\mathbf{s}' - \mathbf{s}; t' - t)$. Under the Gaussian distributed measurement error, we can consider the joint Gaussian distribution

$$\begin{bmatrix} Y(\mathbf{s}_0;t_0) \\ \mathbf{Z} \end{bmatrix} \sim N\left(\begin{bmatrix} \mathbf{x}^1(\mathbf{s}_0;t_0)^\top \\ \mathbf{X} \end{bmatrix} \boldsymbol{\beta}, \begin{bmatrix} c_{0,0} & \mathbf{c}_0^\top \\ \mathbf{c}_0 & \mathbf{C}_z \end{bmatrix}\right).$$

where $\mathbf{c}_0^\top = \mathrm{Cov}(Y(\mathbf{s}_0, \mathbf{Z}))$, $c_{0,0} = \mathrm{Var}(Y(\mathbf{s}_0;t_0))$, $\mathbf{C}_z = \mathrm{Cov}(\mathbf{Z}) = \mathbf{C}_y + \mathbf{C}_\varepsilon$, $\mathbf{C}_y = \mathrm{Cov}(\mathbf{Y}) = \mathrm{Cov}((Y(\mathbf{s}_{11};t_1), \cdots, Y(\mathbf{s}_{mT};t_T))^\top)$, $\mathbf{C}_\varepsilon = \mathrm{Cov}(\boldsymbol{\varepsilon}) = \mathrm{Cov}((\varepsilon(\mathbf{s}_{11};t_1), \cdots, \varepsilon(\mathbf{s}_{mT};t_T))^\top)$, and $\mathbf{X}$ is the $mT \times (p+1)$ covariate matrix $\mathbf{X} = \left[\mathbf{x}^1(\mathbf{s}_{ij};t_j)^\top : i = 1, \cdots, m; j = 1, \cdots, T\right]$.

Then we can obtain the conditional distribution,

$$Y(\mathbf{s}_0;t_0)|\mathbf{Z} \sim N(\mathbf{x}^1(\mathbf{s};t_0)^\top \boldsymbol{\beta} + \mathbf{c}_0^\top \mathbf{C}_z^{-1}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta}), c_{0,0} - \mathbf{c}_0^\top \mathbf{C}_z^{-1} \mathbf{c}_0),$$

by using Gaussian process assumption and results for conditional distributions from a joint multivariate Gaussian distribution. Here, the conditional expected value $\mathbb{E}[Y(\mathbf{s}_0;t_0)|\mathbf{Z}] = \mathbf{x}^1(\mathbf{s}_0;t_0)^\top \boldsymbol{\beta} + \mathbf{c}_0^\top \mathbf{C}_z^{-1}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})$ becomes Best Linear

Unbiased Predictor (BLUP) of $Y(\mathbf{s}_0; t_0)$. Since we may not know $\boldsymbol{\beta}$ in most case, the spatio-temporal universal kriging predictor of $Y(\mathbf{s}_0; t_0)$ is

$$\hat{Y}(\mathbf{s}_0; t_0) = \mathbf{x}^1(\mathbf{s}_0; t_0)^\top \hat{\boldsymbol{\beta}} + \mathbf{c}_0^\top \mathbf{C}_z^{-1}(\mathbf{Z} - \mathbf{X}\hat{\boldsymbol{\beta}}), \tag{2.3}$$

where $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{C}_z^{-1}\mathbf{X})^{-1} X^\top \mathbf{C}_z^{-1}\mathbf{Z}$ which is the generalized least squares estimator of $\boldsymbol{\beta}$.

When considering the function g that minimizes the MSPE $\mathbb{E}[(Y(\mathbf{s}_0; t_0) - g(\mathbf{Z}))^2]$, g is given by $g(\mathbf{Z}) = \mathbf{x}^1(\mathbf{s}_0; t_0)^\top \boldsymbol{\beta} + \mathbf{c}_0^\top \mathbf{C}_z^{-1}(\mathbf{Z} - \mathbf{X}\boldsymbol{\beta})$. That is,

$$\begin{aligned} \hat{Y}(\mathbf{s}_0; t_0) &= \underset{\hat{Y}}{\operatorname{argmin}}\, \mathbb{E}[L(\hat{Y}(\mathbf{s}_0; t_0), Y(\mathbf{s}_0; t_0))] \\ &= \mathbb{E}[Y(\mathbf{s}_0; t_0) | \mathbf{Z}] \end{aligned} .$$

under the Mean Squared Error (MSE) loss function L. So universal Kriging predictor is also best predictor of $Y(\mathbf{s}_0; t_0)$. In most case, Kriging need to estimate the covariance or Variogram function which characterizes dependencies in space and time.

2.3 Deep Feedforward Networks

The most of notations and basic concepts of Deep feedforward networks follow Hastie et al., (2009) and Goodfellow et al., (2016). Deep feedforward networks, also called Multi-Layer Perceptrons (MLPs), which are the basic and typical structure of neural networks are used to predict spatio-temporal data. The goal of the MLP structure is to approximate some function f^*. Take Regression as an example, and suppose that there is a function $y = f^*(\mathbf{x})$ from an input $\mathbf{x}$ to an output $y \in \mathbb{R}$. The MLP structure is defined as $y = f(\mathbf{x}; \boldsymbol{\theta})$ and derives the most optimal approximation function by learning the value of the parameter $\boldsymbol{\theta}$. Thus, we finally derive $f(\mathbf{x}; \hat{\boldsymbol{\theta}})$ where $\hat{\boldsymbol{\theta}}$ is the learned parameter corresponding to the final target function $f^*(\mathbf{x})$. The parameter $\boldsymbol{\theta}$ is learned in the direction of minimizing the loss of $f(\mathbf{x};\boldsymbol{\theta})$ and $f^*(\mathbf{x})$.

Consider single hidden layer feed-forward neural networks which have inputs x_j, $j = 1, \cdots, p$ in the input layer, outputs y_k, $k = 1, \cdots, K$ in the output layer, and hidden units z_m, $m = 1, \cdots, M$ in the hidden layer. Note that typically $K = 1$ for regression. The above MLP model is written by

$$\begin{aligned} y_k &= g_k\left(\beta_{0k} + \sum_{m=1}^{M} \beta_{mk}\sigma\Big(\alpha_{0m} + \sum_{j=1}^{p} \alpha_{jm}x_j\Big)\right) \\ &= g_k\left(\beta_{0k} + \sum_{m=1}^{M} \beta_{mk}z_m\right). \end{aligned}$$

Here, σ is the activation function, g_k is the output function, and $\{(\alpha_{jm}), (\beta_{mk}), j = 1, \cdots, p, m = 1, \cdots, M\}$ are weights. The activation function σ is often set to be the sigmoid $\sigma(v) = 1/(1 + e^{-v})$ or ReLU $\sigma(v) = \max\{v, 0\}$, the output function is set to be the linear $g_k(v) = v$ for the regression or the softmax for the classification that is determined according to the purpose. The weights, the parameters of the neural networks, is fitted by minimizing of the loss function L. It is straightforward to increase the number of hidden layers within such a structure.

The classic theoretical basis for approximating some functions is the Universal approximation theorem (Cybenko G., 1989). There are many versions of this theorem and Kidger et al., (2020) show the version of networks of bounded width and arbitrary depth. Before describing the theorem, we first define the notations used. Use the same definitions and notations as in the paper.

Definition 2.3.1 (Notation for the function space of MLPs)**.** *Let $\rho : \mathbb{R} \to \mathbb{R}$ and $n, m, k \in \mathbb{N}$. Define $\mathcal{NN}^{\rho}_{n,m,k}$ as the function space of functions from $\mathbb{R}^n$ to $\mathbb{R}^m$ described by MLPs with n input neurons, m output neurons, and an* *arbitrary number of hidden layers, each with k neurons and activation function ρ. All output neurons have an identity activation function.*

Main result of Patrick et al., (2020) is the following theorem. Write $C(K; \mathbb{R}^m)$ for the set of continuous functions from K to $\mathbb{R}^m$.

Theorem 2.3.1 (Universal approximation theorem, Patrick et al., 2020)**.** *Let $K \subset \mathbb{R}^n$ be compact set. Let $\rho : \mathbb{R} \to \mathbb{R}$ be any nonaffine continuous function. Assume that ρ is continuously differentiable at at least one point, with nonzero derivative at that point. Then, ${}^\forall\varepsilon > 0$, ${}^\forall f \in C(K; \mathbb{R}^m)$, ${}^\exists\hat{f} \in \mathcal{NN}^{\rho}_{n,m;n+m+2}$ s.t.*

$$\sup_{x \in K} |\hat{f}(x) - f(x)| < \varepsilon.$$

That is, $\mathcal{NN}^{\rho}_{n,m;n+m+2}$ is dense in $C(K; \mathbb{R}^m)$ with respect to the uniform norm. This theorem is used to study the setting of methodological structures and the theoretical properties of methodology in this paper. The methodology is expected to be able to learn the general spatio-temporal dependence structure.

Chapter 3

The Methodology

In this paper, we propose a novel method for predicting spatio-temporal data. For this, we study the expansion basis property of the processes. Finally, we describe the DNN structure that finally derives the final output in this chapter.

3.1 Decomposition of the Spatio-Temporal Process

The easiest way to think for including spatio-temporal information is to add space and time axes to the input as

$$(\mathbf{x}(\mathbf{s};t)^\top, \mathbf{s}, t)^\top.$$

We expect to learn $\eta(\mathbf{s};t)$ from this input. However, in reality, the dimension of the input is not very extended in most situations where the space-time dimension is less than or equal to 3. We go through the process of enlarging the dimension of spatio-temporal space using the properties of spatio-temporal stochastic processes. By Cressie et al., (2019), rewrite the model (2.2) as

$$\begin{aligned} Y(\mathbf{s};t) &= \mathbf{x}(\mathbf{s};t)^\top \boldsymbol{\beta} + \eta(\mathbf{s};t) \\ &= \mathbf{x}(\mathbf{s};t)^\top \boldsymbol{\beta} + \sum_{i=1}^{n_\alpha} \phi_i(\mathbf{s};t)\alpha_i + \nu(\mathbf{s};t), \end{aligned} \tag{3.1}$$

where $\{\phi_i(\mathbf{s};t) : i = 1, \cdots, n_\alpha\}$ are spatio-temporal basis functions at the spatio-temporal location $(\mathbf{s};t)$, $\{\alpha_i\}$ are random effects and $\nu(\mathbf{s};t)$ is small-scale spatio-temporal random effects. Ensure that the number of basis functions are large enough to represent the true spatio-temporal dependence in Y. The basis functions act like covariates if the coefficient is unknown and may be estimated.

Using the above fact, Approximate $\eta(\mathbf{s};t)$ for a sufficiently large $K \in \mathbb{Z}$ as

$$\eta(\mathbf{s};t) \approx \sum_{k=1}^{K} w_k \phi_k(\mathbf{s};t),$$

for the weights $w_k, k = 1, \cdots, K$. So, the approximation of (3.1) can be expressed as

$$\begin{aligned} Y(\mathbf{s};t) &= \mathbf{x}(\mathbf{s};t)^\top \boldsymbol{\beta} + \eta(\mathbf{s};t) \\ &\approx \mathbf{x}(\mathbf{s};t)^\top \boldsymbol{\beta} + \sum_{k=1}^{K} w_k \phi_k(\mathbf{s};t). \end{aligned}$$

That is, an extended input of length $p+K$ can be obtained as shown in

$$\mathbf{x}^*(\mathbf{s};t) := \Big(\mathbf{x}(\mathbf{s};t)^\top, \phi_1(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t)\Big)^\top \in \mathbb{R}^{p+K}. \tag{3.2}$$

This helps to increase the dimension of the input. Thus, it is expected that neural networks will learn the spatio-temporal structure better. A similar idea was used in the paper by Chen et al. (2021).

3.2 Deep Neural Networks Structure

We learn the input separately by dividing $\big(\mathbf{x}(\mathbf{s};t)^\top, \phi_1(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t)\big)^\top$ into two, and then derive the final output of the ensemble form. Input is divided into two: the first input is $\mathbf{x}_1(\mathbf{s};t) = \mathbf{x}(\mathbf{s};t) \in \mathbb{R}^p$, and the second is $\mathbf{x}_2(\mathbf{s};t) = (\phi_1(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t))^\top \in \mathbb{R}^K$.

Specify a DNN structure with L layers for $\mathbf{x}_i(\mathbf{s};t),\ i=1,2$ as

$$\begin{aligned}
\mathbf{z}_1(\mathbf{s};t) &= \sigma(\mathbf{x}_i(\mathbf{s};t)^\top \mathbf{w}_1 + \mathbf{b}_1);\\
\mathbf{z}_2(\mathbf{s};t) &= \sigma(\mathbf{z}_1(\mathbf{s};t)^\top \mathbf{w}_2 + \mathbf{b}_2);\\
\mathbf{z}_3(\mathbf{s};t) &= \sigma(\mathbf{z}_2(\mathbf{s};t)^\top \mathbf{w}_3 + \mathbf{b}_3);\\
&\vdots\\
\mathbf{z}_{L-2}(\mathbf{s};t) &= \sigma(\mathbf{z}_{L-3}(\mathbf{s};t)^\top \mathbf{w}_{L-2} + \mathbf{b}_{L-2});\\
f_i(\mathbf{s};t) &= g(\mathbf{z}_{L-2}(\mathbf{s};t)^\top \mathbf{w}_{L-1} + b_{L-1}).
\end{aligned} \tag{3.3}$$

where d_i is the length of a vector $\mathbf{x}_i(\mathbf{s};t)$, and $\mathbf{w}_1 \in \mathbb{R}^{d_i\times(d_i+3)},\ \mathbf{w}_2,\cdots,\mathbf{w}_{L-2} \in \mathbb{R}^{(d_i+3)\times(d_i+3)}, \mathbf{w}_{L-1} \in \mathbb{R}^{(d_i+3)\times 1}, \mathbf{b}_1,\cdots,\mathbf{b}_{L-2} \in \mathbb{R}^{1\times(d_i+3)},\ b_{L-1} \in \mathbb{R}$ are the parameters. σ is the activation function and g is the output function. The number of units of the hidden layer, d_i+3, was set as above based on Theorem 2.3.1 described above.

We then obtain the final output $f(\mathbf{s};t)$ by an ensemble form using an MLP structure with the new inputs $f_1(\mathbf{s};f),\ f_2(\mathbf{s};t)$ which is the outputs of (3.3). For $\mathbf{f}(\mathbf{s};t) = (f_1(\mathbf{s};t), f_2(\mathbf{s};t))^\top \in \mathbb{R}^2$, specify a DNN structure with L layers as follows.

$$\begin{aligned}
\mathbf{z}_1(\mathbf{s};t) &= \sigma(\mathbf{f}(\mathbf{s};t)^\top \mathbf{w}_1 + \mathbf{b}_1);\\
\mathbf{z}_2(\mathbf{s};t) &= \sigma(\mathbf{z}_1(\mathbf{s};t)^\top \mathbf{w}_2 + \mathbf{b}_2);\\
\mathbf{z}_3(\mathbf{s};t) &= \sigma(\mathbf{z}_2(\mathbf{s};t)^\top \mathbf{w}_3 + \mathbf{b}_3);\\
&\vdots\\
\mathbf{z}_{L-2}(\mathbf{s};t) &= \sigma(\mathbf{z}_{L-3}(\mathbf{s};t)^\top \mathbf{w}_{L-2} + \mathbf{b}_{L-2});\\
f(\mathbf{s};t) &= g(\mathbf{z}_{L-2}(\mathbf{s};t)^\top \mathbf{w}_{L-1} + b_{L-1}).
\end{aligned} \tag{3.4}$$

where $\mathbf{w}_1 \in \mathbb{R}^{2\times 5},\ \mathbf{w}_2,\cdots,\mathbf{w}_{L-2} \in \mathbb{R}^{5\times 5},\ \mathbf{w}_{L-1} \in \mathbb{R}^{5\times 1}, \mathbf{b}_1,\cdots,\mathbf{b}_{L-2} \in \mathbb{R}^{1\times 5},\ b_{L-1} \in \mathbb{R}$. are the parameters. A schematic diagram of this structure is shown in Figure 3.1.

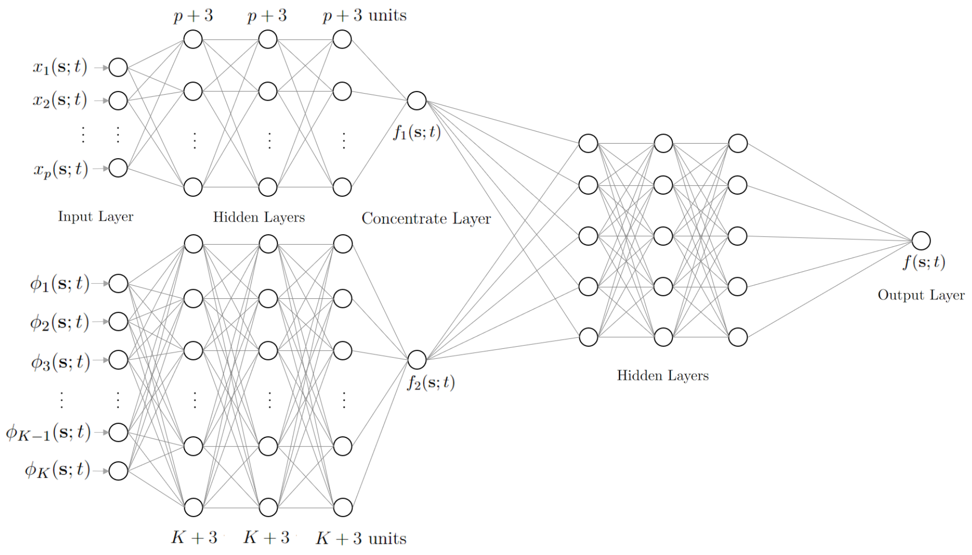

Figure 3.1: Visualization of DNN Structure.

Let $\boldsymbol{\theta}$ be the vector of all unknown parameters (i.e., weights and biases) of the structure. Then we can rewrite $f(\mathbf{s};t) = f(\mathbf{s},t;\boldsymbol{\theta})$ since $f(\mathbf{s};t)$ is the function with the parameter $\boldsymbol{\theta}$. We obtain the estimate $\hat{\boldsymbol{\theta}}$ by

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} L\left(f(\mathbf{s},t;\boldsymbol{\theta}), Z(\mathbf{s};t)\right),$$

with training sample data Z. Therefore, with $\hat{\boldsymbol{\theta}}$, the final prediction at an unobserved spatio-temporal location $(\mathbf{s}_0;t_0)$ is $f(\mathbf{s}_0,t_0;\hat{\boldsymbol{\theta}})$ (Goodfellow et al., 2016). Denote $\hat{Y}_{MLP}(\mathbf{s}_0;t_0) = f(\mathbf{s}_0,t_0;\hat{\boldsymbol{\theta}})$.

Consider spatio-temporal data model (2.1). We assume that the true data is from this model. Note that We make assumptions about the normality and stationarity of the stochastic process for the spatio-temporal universal kriging predictor $\hat{Y}$ in (2.3). Now, we assume that the spatio-temporal process $Y(\mathbf{s};t)$ is not satisfied with normality and is more complex. But unfortunately, we assume that we don't know the process structure exactly. In this case, the kriging predictor in (2.3) is no more optimal (i.e. best). We expect our predictor $\hat{Y}_{MLP}$ is able to learn the more complex and non-Gaussian process Y. So we expect $\hat{Y}_{MLP}$ to show a better performance than $\hat{Y}$ based on the Universal approximation theorem.

Chapter 4

Theoretical Study

Our prediction method using DNNs and traditional Kriging prediction method are supervised learning. So we analyze these two through statistical learning theory in this chapter. In addition, we study the statistical properties of prediction architecture. This chapter may explain why the prediction method is described as Chpater 3.

4.1 Statistical Learning Theory

First, view the spatio-temporal kriging as statistical learning model based on statistical learning theory. The notations and basic concepts of learning theory follow Luxburg et al., (2011). Let $\mathbf{x} = (\mathbf{x}(\mathbf{s};t)^\top, \mathbf{s}, t)^\top$ be the input of the input space and Y be the output of the output space. Training data is $\{(\mathbf{x}_i, y_i),\ i = 1, \cdots, N\}$, assumed to be the random sample from the joint distribution $P(\mathbf{x}, Y)$. Let $\mathcal{C}(\mathbf{x}(\mathbf{s};t))$ be a function space which is a set of all measurable functions with $\mathbf{x}(\mathbf{s};t)$ as a feature. Then there is a some function space $\mathcal{F}_{STUK} \subset \mathcal{C}(\mathbf{x}(\mathbf{s};t))$ such that with squared error loss function L,

$$\begin{aligned}\hat{Y} &= \underset{f \in \mathcal{F}_{STUK}}{\text{argmin}}\ \mathbb{E}_{(\mathbf{x},Y)} L(f(\mathbf{x}), Y) \\ &= \underset{f \in \mathcal{F}_{STUK}}{\text{argmin}}\ \mathbb{E}_{(\mathbf{x},Y)} (f(\mathbf{x}) - Y)^2,\end{aligned}$$

which is a universal kriging predictor. That is, we predict z_0 as $\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}_0)$ for a new input $\mathbf{x}_0$ as

$$\hat{f}_{\mathcal{F}_{STUK}} = \underset{f \in \mathcal{F}_{STUK}}{\operatorname{argmin}} \mathbb{E}_{(\mathbf{x},Y)}(f(\mathbf{x}) - Y)^2,$$

implying $\hat{f}_{\mathcal{F}_{STUK}} = \hat{Y}$. In this way, we consider the traditional kriging method as a statistical learning model and then compare it with our DNNs model.

Define $\mathcal{F}_{all}$ as the function space of all measurable functions from input space to output space. For some function space $\mathcal{F} \subset \mathcal{F}_{all}$, the predictors are

$$\begin{aligned}
\hat{f}_{Bayes} &= \underset{f \in \mathcal{F}_{all}}{\operatorname{argmin}} \mathbb{E}L(Y, f(\mathbf{x})) \\
\hat{f}_{\mathcal{F}} &= \underset{f \in \mathcal{F}}{\operatorname{argmin}} \mathbb{E}L(Y, f(\mathbf{x})) \\
\hat{f}_{\mathcal{F}}^{n} &= \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i))
\end{aligned}$$

where $y_i = Z(\mathbf{x}_i)$ as above. Note that the risk of a function is $R(f) := \mathbb{E}L(Y, f(\mathbf{x}))$. When we cannot compute the true risk, we use the empirical risk. For the comparison of the model, the Access risk is defined as follows.

Definition 4.1.1 (Excess Risk)**.** *The* ***excess risk****, $\mathcal{E}(f)$, is defined by*

$$\mathcal{E}(f) := R(f) - R(\hat{f}_{Bayes}).$$

It compares the risk of f to the $\hat{f}_{Bayes}$.

We want this value to be reduced through sufficient observations. The access risk of $\hat{f}_{\mathcal{F}}^{n}$ can be decomposed as

$$\begin{aligned}
\mathcal{E}(\hat{f}_{\mathcal{F}}^{n}) &= R(\hat{f}_{\mathcal{F}}^{n}) - R(\hat{f}_{Bayes}) \\
&= \underbrace{R(\hat{f}_{\mathcal{F}}^{n}) - R(\hat{f}_{\mathcal{F}})}_{\text{estimation error}} + \underbrace{R(\hat{f}_{\mathcal{F}}) - R(\hat{f}_{Bayes})}_{\text{approximation error}}.
\end{aligned} \tag{4.1}$$

Note that $R(\hat{f}^n_{\mathcal{F}}) - R(\hat{f}_{\mathcal{F}})$ is the estimation error and $R(\hat{f}_{\mathcal{F}}) - R(\hat{f}_{Bayes})$ is the approximation error as described in (4.1). Here, $\mathcal{F}$ is the space to be used in the algorithm. Estimation error tends to be smaller as $\mathcal{F}$ is smaller and more training data is available. Approximation error tends to be smaller as $\mathcal{F}$ is bigger. The visualization of this decomposition is as shown in Figure 4.1. Here, we prefer to choose $\mathcal{F}$ that balances well between these two errors. If

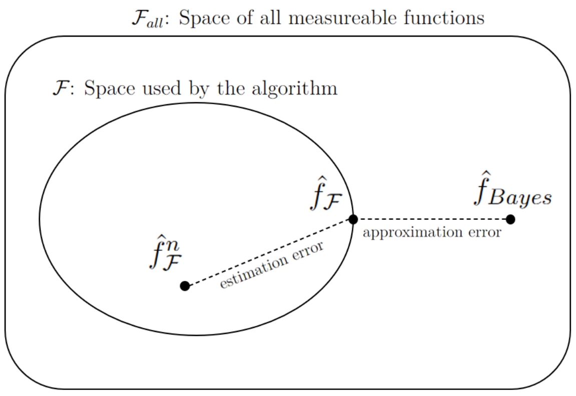
$\mathcal{F}_{all}$: Space of all measureable functions
$\mathcal{F}$: Space used by the algorithm
$\hat{f}_{\mathcal{F}}$
$\hat{f}_{Bayes}$
approximation error
$\hat{f}^n_{\mathcal{F}}$
estimation error

Figure 4.1: Visualization of access risk of $\hat{f}^n_{\mathcal{F}}$

you have a lot of training data, it would be good to choose a bigger $\mathcal{F}$. In deep learning training, which usually has a large number of data, a big $\mathcal{F}$, such as a deep neural network structure, is selected. It is similar when we think of data with a large number of time indexes. Thus, when we say our model space is $\mathcal{F}_{MLP} := \mathcal{NN}^{\rho}_{(p+K),1,(p+K)+3}$, we would like to argue that it will perform well because it has more capacity than $\mathcal{F}_{STUK}$. We show as follows that the function space of our model $\mathcal{F}_{MLP}$ has more capacity than $\mathcal{F}_{STUK}$ under some acceptable conditions.

Proposition 4.1.1. *Let $\mathcal{F}_{all} = C(V;\mathbb{R})$ for some compact set $V \subset \mathbb{R}^{p+K}$, $\mathcal{F}_{MLP} = \mathcal{NN}^{\rho}_{(p+K),1,(p+K)+3}$ for the ReLU activation function ρ, and $\mathcal{F}_{STUK} \subset \mathcal{C}(\mathbf{x}(\mathbf{s};t)) \cap \mathcal{F}_{all}$. L is the squared error loss function. Assume that $\hat{f}_{Bayes}$ =*

$\underset{f\in\mathcal{F}_{all}}{\operatorname{argmin}}\,\mathbb{E}L(Y,f(\mathbf{x}))$, *$\hat{f}_{Bayes} \notin \mathcal{F}_{STUK}$, and $\mathbb{E}[\{L(Y,f(\mathbf{x}))\}^2]$ is bounded for all $f \in \mathcal{F}_{all}$. Then $R(\hat{f}_{\mathcal{F}_{MLP}}) < R(\hat{f}_{\mathcal{F}_{STUK}})$.*

In above proposition, $\hat{f}_{Bayes} \notin \mathcal{F}_{STUK}$ means that $\hat{f}_{Bayes}$ is more complex to be in $\mathcal{F}_{STUK}$. Visualizing what Proposition 4.1.1. means is shown as in Figure 4.2. The lemmas required for proof are as follows (Durrett, 2019).

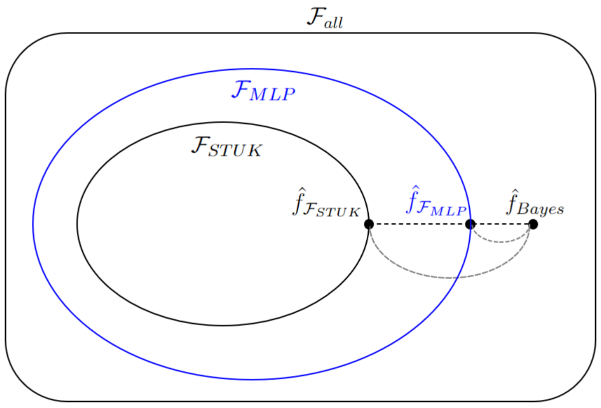

Figure 4.2: Visualization of bigger capacity

Lemma 4.1.1. *Let X and $X_1, X_1, \cdots$ be random variables. $X_n \to X$ a.s. if for all $\varepsilon > 0$, $\sum_{k=n}^{\infty} P(|X_k - X| > \varepsilon) \to 0$ as $n \to \infty$*

Lemma 4.1.2. *Let X and $X_1, X_1, \cdots$ be random variables. Assume that f is continuous and $X_n \to X$ a.s. then $f(X_n) \to f(X)$ a.s.*

Lemma 4.1.3. *Let X and $X_1, X_1, \cdots$ be random variables. Assume that $X_n \to X$ a.s. Let g, h be continuous functions with (i) $g \geq 0$ and $g(x) \to \infty$ as $|x| \to \infty$, (ii) $|h(x)|/g(x) \to 0$ as $|x| \to \infty$, and (iii) $\mathbb{E}g(X_n) \leq K < \infty$, $^{\forall}n$. Then $\mathbb{E}h(X_n) \to \mathbb{E}h(X)$ as $n \to \infty$.*

Proof of Proposition 4.1.1. By Theorem 2.3.1, $\mathcal{F}_{MLP} = \mathcal{NN}^{\rho}_{(p+K),1,(p+K)+3}$ is dense in $\mathcal{F}_{all} = C(V;\mathbb{R})$. Define the sequence $\{\varepsilon_n\}$ as $\varepsilon_n = 1/n$, for all

$n \in \mathbb{N}$. Then ${}^{\forall}n \in \mathbb{N}$, ${}^{\exists}f_n \in \mathcal{F}_{MLP}$ s.t. $\sup_{x \in V} |f_n(x) - \hat{f}_{Bayes}(x)| < \varepsilon_n$ since $\hat{f}_{Bayes}(x) \in \mathcal{F}_{all}$. Define the sequence $\{f_n\}$ as above. Note that

$$\begin{aligned} \varepsilon_n &> \sup_{x \in V} |f_n(x) - \hat{f}_{Bayes}(x)| \\ &\geq |f_n(t) - \hat{f}_{Bayes}(t)| \\ &\geq |f_{n+1}(t) - \hat{f}_{Bayes}(t)|, \ {}^{\forall}n \in \mathbb{N}, {}^{\forall}t \in V. \end{aligned}$$

For some $\varepsilon_0 > 0$, if $\varepsilon_0 > 1$, $|f_n(x) - \hat{f}_{Bayes}(x)| < \varepsilon_0, {}^{\forall}x \in V, {}^{\forall}n \in \mathbb{N}$. If $\varepsilon_0 \leq 1$, ${}^{\exists}n_0 \in \mathbb{N}$ s.t. $1/(n_0+1) < \varepsilon_0 \leq 1/n_0$. So $|f_n(x) - \hat{f}_{Bayes}(x)| < \varepsilon_0, {}^{\forall}x \in V, {}^{\forall}n (\in \mathbb{N}) \geq n_0$. Now, let $\varepsilon > 0$ be given. Then ${}^{\exists}M \in \mathbb{N}$ s.t. if $n \geq M$, $|f_n(x) - \hat{f}_{Bayes}(x)| < \varepsilon$, ${}^{\forall}x$. In other words, $f_n(x) \to \hat{f}_{Bayes}(x)$ as $n \to \infty$, ${}^{\forall}x \in V$.

Define $W_n = f_n(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})$. Then, for $n \geq M$, $|W_n| < \varepsilon \Rightarrow P(|W_n| > \varepsilon) = 0$. So, if $n \geq M$, then

$$\begin{aligned} &P(|W_n| > \varepsilon) + P(|W_{n+1}| > \varepsilon) + \cdots \\ &= P(|f_n(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})| > \varepsilon) + P(|f_{n+1}(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})| > \varepsilon) + \cdots \\ &= \sum_{k=n}^{\infty} P(|f_k(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})| > \varepsilon) \\ &= 0. \end{aligned}$$

That is, ${}^{\forall}\varepsilon > 0, {}^{\exists}M \in \mathbb{N}$ s.t. if $n \geq M$, $\left| \sum_{k=n}^{\infty} P(|f_k(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})|) \right| = 0$, which implies $\lim_{n \to \infty} \sum_{k=n}^{\infty} P(|f_k(\mathbf{x}) - \hat{f}_{Bayes}(\mathbf{x})| > \varepsilon) = 0$. So,

$$f_n(\mathbf{x}) \to \hat{f}_{Bayes}(\mathbf{x}) \ a.s.$$

by Lemma 4.1.1. Then, $(f_n(\mathbf{x}) - Y)^2 \to (\hat{f}_{Bayes}(\mathbf{x}) - Y)^2$ *a.s.* by Lemma 4.1.2. Since $\mathbb{E}[(f_n(\mathbf{x}) - Y)^4] < K$ for some $K \in \mathbb{R}$,

$$\lim_{n \to \infty} \mathbb{E}[(f_n(\mathbf{x}) - Y)^2] = \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2]$$

by Lemma 4.1.3.

Since $\arg\min_{f\in\mathcal{F}_{all}} \mathbb{E}(f(\mathbf{x}) - Y)^2 = \hat{f}_{Bayes} \notin \mathcal{F}_{STUK}$ and $\mathcal{F}_{STUK} \subset \mathcal{F}_{all}$, $\mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2] > \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2]$ (Equality holds when $\hat{f}_{Bayes} \in \mathcal{F}_{STUK}$). Let $\alpha = \mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2] - \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2] > 0$. Then, for $\alpha/2$, ${}^{\exists}f_{\mathcal{F}_{MLP}} \in \mathcal{F}_{MLP}$ s.t.

$$\mathbb{E}[(f_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] - \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2] < \alpha/2.$$

Then,

$$\mathbb{E}[(\hat{f}_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] - \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2] < \alpha/2$$

since $\mathbb{E}[(\hat{f}_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] \leq \mathbb{E}[(f_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2]$. So,

$$\mathbb{E}[(\hat{f}_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] < \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2] + \alpha/2.$$

Since $\mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2] = \alpha + \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2]$ for $\alpha > 0$,

$$\begin{aligned}\mathbb{E}[(\hat{f}_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] &< \mathbb{E}[(\hat{f}_{Bayes}(\mathbf{x}) - Y)^2] + \alpha/2 \\ &= \mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2] - \alpha/2 \\ &< \mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2].\end{aligned}$$

Thus,

$$[(\hat{f}_{\mathcal{F}_{MLP}}(\mathbf{x}) - Y)^2] < \mathbb{E}[(\hat{f}_{\mathcal{F}_{STUK}}(\mathbf{x}) - Y)^2]$$

and

$$\inf_{f\in\mathcal{F}_{MLP}} \mathbb{E}_{\mathbf{x},Y}[(f(\mathbf{x}) - Y)^2] < \inf_{f\in\mathcal{F}_{STUK}} \mathbb{E}_{\mathbf{x},Y}[(f(\mathbf{x}) - Y)^2].$$

This impiles $R(\hat{f}_{\mathcal{F}_{MLP}}) < R(\hat{f}_{\mathcal{F}_{STUK}})$ as desired. $\square$

The ensemble structure as (3.4) is not considered. This is for the convenience of the proof. This proposition simply shows the learning performance of DNNs on complex data.

4.2 Ensemble Method

In (3.4), we use the structure of an ensemble neuron network for deriving the final output. A similar structure is introduced by Gadgay et al., (2012). Although described later, this shows better results in the test step than in the absence of an ensemble. It divides the different features of the input. Then we get an aggregated predictor with the two version of predictors using different learning sets. It can be considered as Bagging predictor (Breiman., 1996).

Let $\mathcal{L} = \{(\mathbf{x}_i, y_i), j = 1, \cdots, n\}$ be a training set. Each $(\mathbf{x}, y) \in \mathcal{L}$ is obtained from the probability distribution $P(\mathbf{x}, y)$ independently. Let the predictor from $\mathcal{L}$ be $\hat{f}(\mathbf{x}, \mathcal{L})$. Suppose that we have training subsets $\mathcal{L}_k$ each including independent observed data from the distribution same as $\mathcal{L}$. Then, the average of $\hat{f}(\mathbf{x}, \mathcal{L})$ over $\mathcal{L}$ is the true Bagging predictor

$$f_A(\mathbf{x}) = \mathbb{E}_{\mathcal{L}}\hat{f}(\mathbf{x}, \mathcal{L}).$$

The average prediction error of the simgle set predictor $f(\mathbf{x}, \mathcal{L})$ is $e = \mathbb{E}_{\mathcal{L}}\mathbb{E}_{\mathbf{x},y}(y - \hat{f}(\mathbf{x}, \mathcal{L}))^2$. And the average prediction error in f_A is $e_A = \mathbb{E}_{\mathbf{x},y}(y - f_A(\mathbf{x}))^2$. Then, by the Jensen's inequality,

$$\begin{aligned}
e &= \mathbb{E}_{\mathbf{x},y}y^2 - 2\mathbb{E}_{\mathbf{x},y}\mathbb{E}_{\mathcal{L}}\hat{f}(\mathbf{x}, \mathcal{L}) + \mathbb{E}_{\mathbf{x},y}\mathbb{E}_{\mathcal{L}}(\hat{f}(\mathbf{x}, \mathcal{L}))^2 \\
&\geq \mathbb{E}_{\mathbf{x},y}y^2 - 2\mathbb{E}_{\mathbf{x},y}f_A(\mathbf{x}) + \mathbb{E}_{\mathbf{x},y}(\mathbb{E}_{\mathcal{L}}\hat{f}(\mathbf{x}, \mathcal{L}))^2 \\
&= \mathbb{E}_{\mathbf{x},y}y^2 - 2\mathbb{E}_{\mathbf{x},y}f_A(\mathbf{x}) + \mathbb{E}_{\mathbf{x},y}(f_A(\mathbf{x}))^2 \\
&= \mathbb{E}_{\mathbf{x},y}(y - f_A(\mathbf{x}))^2 \\
&= e_A
\end{aligned}$$

Therefore, f_A imporves f. The bagging estimate is defined as

$$\hat{f}_B(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}(\mathbf{x}, \mathcal{L}^{(b)}).$$

where $\mathcal{L}^{(b)},\ b = 1, \cdots, B$ are the training bootstrap samples. In our case, we have a covariate dataset $\mathcal{L}_1$ and a spatio-temporal information dataset $\mathcal{L}_2$ with the predictors $\hat{f}(\mathbf{x}, \mathcal{L}_1)$ and $\hat{f}(\mathbf{x}, \mathcal{L}_2)$. Our method sample two uncorrelated subsets without replacement, unlike traditional Bagging. And the two predictors are aggregated by giving weight instead of average. Overall, however, it is consistent with Bagging method.

To sum up, we consider the reason for using DNNs for learning complex data. And we consider the advantage of Bagging ensemble using divided training inputs.

Chapter 5

Application

We applied the prediction method using DNNs to real-world data to compare the test performance of any of our models with the existing spatio-temporal kriging methods. For the data, Korea's fine dust ($PM_{2.5}$ concentration) spatio-temporal data was used. Fine dust data is known to be non-gaussian and difficult to predict. So, we try to evaluate the performance on complex models by using our method to show better performance of this data. The application codes and data are in the Github repository: https://github.com/park4264/Spatio-Temporal-Prediction-Using-Deep-Neural-Networks

5.1 Detailed Structural Settings

Recall that we have two inputs, $\mathbf{x}_1(\mathbf{s};t) = \mathbf{x}(\mathbf{s};t) \in \mathbb{R}^p$ and $\mathbf{x}_2(\mathbf{s};t) = (\phi_1(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t))^\top \in \mathbb{R}^K$. Consider the second input $\mathbf{x}_2(\mathbf{s};t)$ and define it more specifically as

$$\begin{aligned}\mathbf{x}_2(\mathbf{s};t) &= (\phi_1(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t))^\top \\ &= (\phi_1(\mathbf{s};t), \cdots, \phi_{k_1}(\mathbf{s};t), \phi_{k_1+1}(\mathbf{s};t), \cdots, \phi_K(\mathbf{s};t))^\top,\end{aligned}$$

where the first k_1 basis include time information and the last $K - k_1 =: k_2$ basis include spatial information. Set $k_1 \approx 5T$ and $k_2 \approx 5m$, which seems to be able to heuristically contain spatio-temporal information well.

Next, the Wendland function was used as the basis function. Let θ_1 and θ_2 be scale parameters for the basis, respectively. Let $\{z_j\}, j = 1, \cdots, k_1$ be a one-dimensional gird, and $\{\mathbf{u}_j\}, j = 1, \cdots, k_{21}, \{\mathbf{v}_j\}, j = 1, \cdots, k_{22}$ be a rectangular gird. Then The basis function was set as

$$\phi_j(\mathbf{s}; t) = \begin{cases} \phi\Big(|t - z_j|/\theta_1\Big), & j \in \{1, \cdots, k_1\} \\ \phi\Big(||\mathbf{s} - \mathbf{u}_{j-k_1}||/\theta_2\Big), & j \in \{k_1 + 1, \cdots, k_1 + k_{21}\} \\ \phi\Big(||\mathbf{s} - \mathbf{v}_{j-k_1-k_{21}}||/\theta_1\Big), & j \in \{k_1 + k_{21} + 1, \cdots, K\} \end{cases}$$

where

$$\phi(x) = \begin{cases} \frac{1}{3}(1 - x)^6(35x^2 + 18x + 3), & x \in [0, 1] \\ 0, & \text{otherwise} \end{cases}.$$

Set the number of hidden layers to 3. The ReLU function, $\sigma(v) = \max\{v, 0\}$, is used as the activation function. Since it is a regression, the output function is set to the linear function, $g_k(v) = v$. Apply it to the data with the input set as above.

5.2 Data Description

As for the data, $PM_{2.5}$ concentration on Korea was obtained from AirKorea (https://www.airkorea.or.kr). Hourly PM2.5 concentration from 1:00 to 24:00 on January 9, 2022, February 1, 2022 and June 29, 2022 were considered. These three days are chosen as the representative days for high, normal and low PM2.5 concentrations. In this way, the performance of the model was evaluated for all cases by obtaining spatio-temporal data that include the time of three days of $PM_{2.5}$ concentration and (latitude, longitude) of the observation locations. Plotting the data of February 1, 2022 is shown in Figure 5.1.

Data on January 9, 2022, when $PM_{2.5}$ concentration is high, a total of 11544 data points over 24 hours were used at a total of 481 stations, with an average

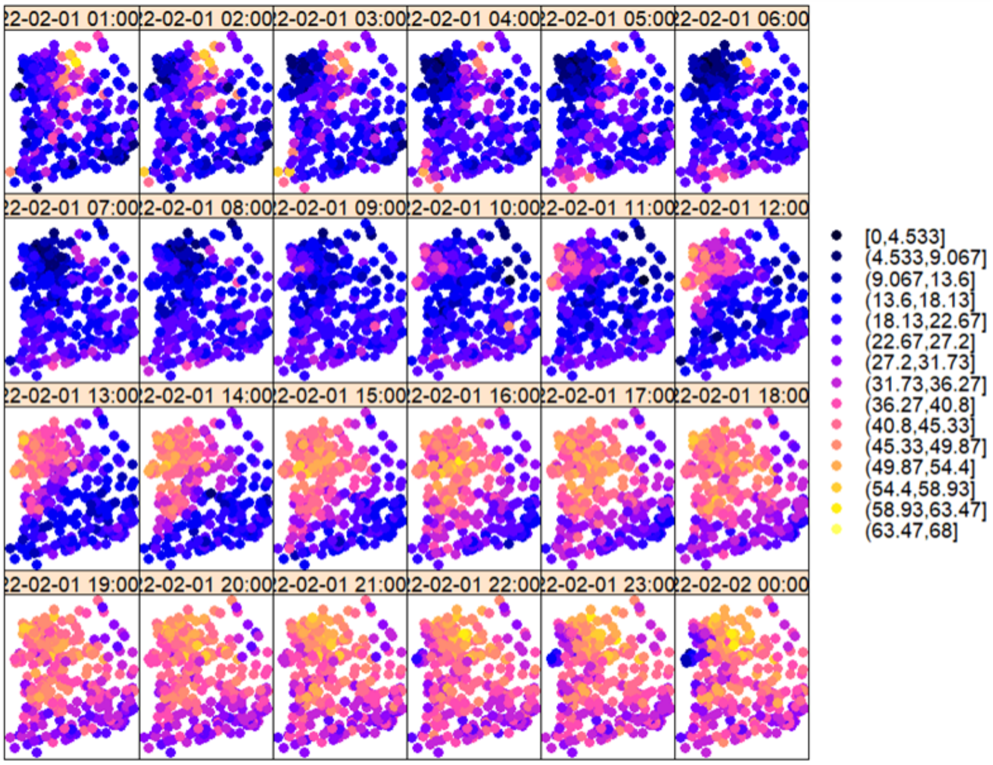

Figure 5.1: Data plot for the $PM_{2.5}$ concentration on February 1

value of $PM_{2.5}$ concentration being approximately $67.419\mu/m^3$. Next, data on February 1, 2022, when $PM_{2.5}$ concentration is normal, a total of 10704 data points over 24 hours were used at a total of 446 stations, with an average value of $PM_{2.5}$ of approximately $29.008\mu/m^3$. Lastly, data on June 29, 2022, when $PM_{2.5}$ concentration is low, a total of 10200 data points over 24 hours were used at a total of 425 stations, with an average value of $PM_{2.5}$ being approximately $5.153\mu/m^3$. For the observations, the histogram is shown in Figure 5.2. In addition, a summary being the observed data is shown in Table 5.1.

The source of the covariate data is the Korea Meteorological Administration (https://www.weather.go.kr). Six covariates were used: temperature, wind,

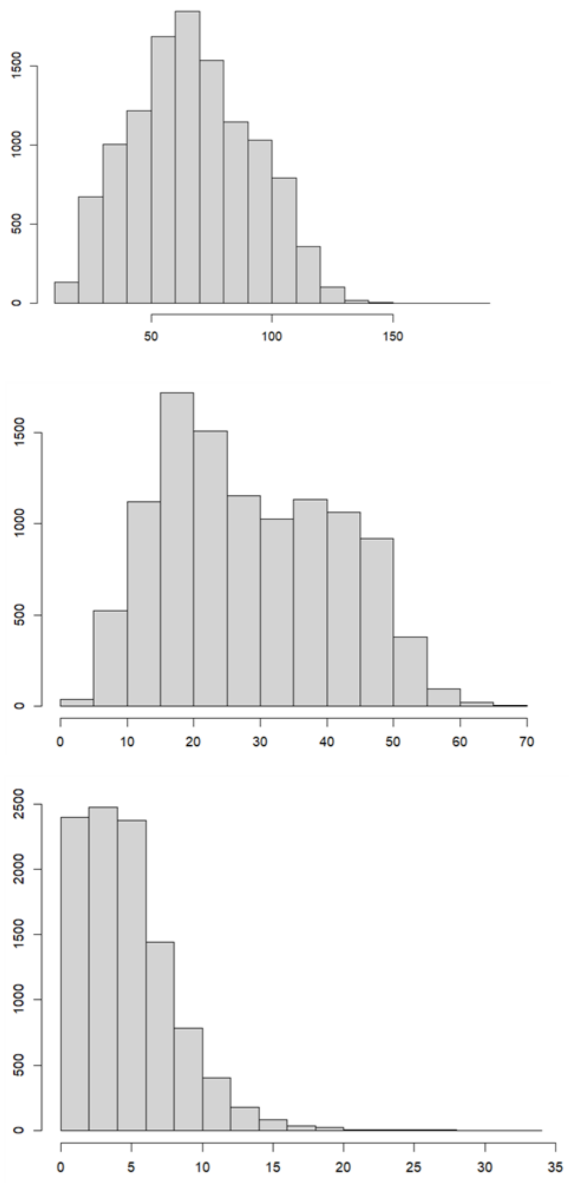

Figure 5.2: Data histogram for the $PM_{2.5}$ concentration on January 9 (Top), February 1 (Middel), and June 29 (Bottom).

	Location	Time	Total	$\text{PM}_{2.5}$ mean
January 9	481	24	11544	$67.419\mu/m^3$
February 1	446	24	10704	$29.008\mu/m^3$
June 29	425	24	10200	$5.153\mu/m^3$

Table 5.1: The number of locations and times observed, the total number of data, and $\text{PM}_{2.5}$ concentration mean by date

wind x-axis components, wind y-axis components, precipitation, and humidity. However, since the location of the observation station where the $\text{PM}_{2.5}$ concentration is observed and the observation station of the Korea Meteorological Administration is different, it is not completely consistent. So, we interpolated the covariate at the same latitude and longitude as the $\text{PM}_{2.5}$ concentration observed. As a method, we used simple spatio-temporal kriging.

5.3 Results

The K-fold cross-validation was used as a test method. In spatio-temporal data, there should be a difference from the normal K-fold cross-validation that randomly divides the training set and the verification set. Therefore, when the test was conducted for a certain spatio-temporal location, we should conduct the test except for all the corresponding locations and times. It is introduced by Meyer et al. (2018) under the name Leave-Location-and-Time-Out CV (LLTO Cross-Validation). The method is visualized in Figure 5.3.

For the test, 2 out of 24 hours were randomly extracted 5 times, and about 30 out of the number of stations were randomly extracted 5 times, and the test was conducted with a total of 25 folds, 60 validation points in a single fold. Note that there are 60 locations to test in a single fold, but as described above,

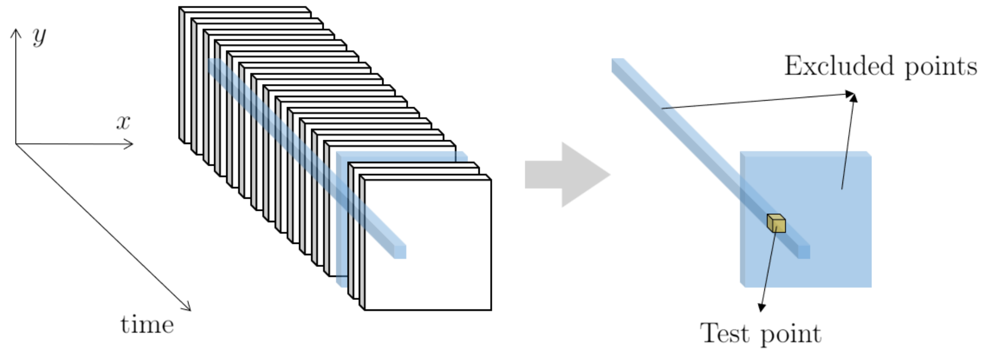

Figure 5.3: Visualization of the LLTO Cross-Validation

about 1000 out of about 10,000 training data are excluded.

Since it needs GPUs to perform 25 tests, the test was conducted under 1xNVIDIA-TESLA-V100 GPU environment of Google Cloud Platform (GCP, https://cloud.google.com). In practical application situations, we applied the Dropout method (Srivastava et al., 2014) which can reduce overfitting, and the Batch Normalization method (Ioffe et al., 2015) which can stabilize the model. In addition, considering that spatio-temporal kriging can be done comfortably in the package of the R environment, the traditional spatio-temporal kriging method is compared using the same dataset in the R environment.

There are two validation statistics for real-valued spatio-temporal processes: the Mean Squared Prediction Error (MSPE) and the Mean Absolute Prediction Error (MAPE)(Wikle et al., 2019). Note that for validation data $\{Z(\mathbf{s}_i;t_j) : i = 1,\cdots,m,\ j = 1,\cdots,T\}$ and predictions $\{\hat{Z}(\mathbf{s}_i;t_j) : i = 1,\cdots,m,\ j = 1,\cdots,T\}$ the MSPE is given by

$$MSPE = \frac{1}{mT}\sum_{j=1}^{T}\sum_{i=1}^{m}\left(Z(\mathbf{s}_i;t_j) - \hat{Z}(\mathbf{s}_i;t_j)\right)^2$$

and the MAPE is given by

$$MAPE = \frac{1}{mT}\sum_{j=1}^{T}\sum_{i=1}^{m}\big|Z(\mathbf{s}_i;t_j) - \hat{Z}(\mathbf{s}_i;t_j)\big|.$$

The performance of the method was evaluated using these two validation statistics.

We compute these statistics for the traditional kriging method (Kriging in Table 5.2) and the spatio-temporal prediction method using DNNs proposed in this paper (Ours in Table 5.2, Hereinafter referred to as Ours). In addition, statistics for six additional cases are also evaluated: Method 1, Method 2, Method 3, Method 4, Method 5, and Method 6 in Table 5.2. Define Method 1 as a way that has no ensemble structure. That is, for $p+K$ enlarged inputs $\mathbf{x}^*(\mathbf{s};t)$ in (3.2), the final output is computed by corresponding MLPs like (3.3). Method 2 proceeds the same process as Method 1 using only K basis without p covariates. Method 3 proceeds the same process as Method 1 using only p basis without K covariates. Recall that there are two steps for the methodology in this paper to define Method 4, Method 5, and Method 6. The first step is to obtain $\mathbf{f}(\mathbf{s};t) = (f_1(\mathbf{s};t), f_2(\mathbf{s};t))^\top \in \mathbb{R}^2$ by (3.3) , and the second step is to obtain final output $f(\mathbf{s};t)$ by (3.4). We can replace the first step with least squares regression for the multiple linear model. That is, for $i = 1, 2$,

$$f_i(\mathbf{s};t) = \mathbf{x}_i^1(\mathbf{s};t)^\top(\mathbf{X}_i^\top\mathbf{X}_i)^{-1}\mathbf{X}_i^\top\mathbf{Z}$$

where $\mathbf{x}_i^1(\mathbf{s};t) = (1, \mathbf{x}_i(\mathbf{s};t))^\top$ and $\mathbf{X}_i = \left[\mathbf{x}_i^1(\mathbf{s}_{ij};t_j)^\top : i = 1,\cdots,m; j = 1,\cdots,T\right]$ which is $mT\times(p+1)$ matrix. We can also replace the second step with a simple weighted average ensemble method. That is,

$$f(\mathbf{s};t) := w_1 f_1(\mathbf{s};t) + w_2 f_2(\mathbf{s};t)$$

where

$$w_1, w_2 = \underset{w_1, w_2}{\operatorname{argmin}} \frac{1}{N} \sum L(Z(\mathbf{s}_i; t_i), w_1 f_1(\mathbf{s}_i; t_i) + w_2 f_2(\mathbf{s}_i; t_i))$$

such that $w_1 + w_2 = 1,\ w_1, w_2 \geq 0$ with squared error loss function L. Method 4 is the same as replacing the second step of Ours with a weighted average ensemble. Method 5 is the same as replacing the first step of Ours with least squared regression. Method 6 is the same as replacing the first and second step of Ours with least squared regression and weighted average ensemble.

Table 5.2 provides the mean and standard deviation (SD in Table 5.2) results of the 25 validation MSPE and MAPE values for all of the above methods. First of all, comparing the results of Kriging, Ours, and Method 1, the performance of Ours is the best. Therefore, Ours is better than the traditional kriging method and method that do not include ensembles in terms of mean error performance. In addition, learning using basis decomposition shows good performance by considering the results of method 1 and method 2. Lastly, Considering the results of the rest of the methods (i.e., Method 4, Method 5, and Method 6) which have less complexity, Ours are almost always the best, but the rest also show comparable performance. In conclusion, Ours shows good performance for complex non-Gaussian spatio-temporal data by the results of the cross-validation test. The higher the average $\mathrm{PM}_{2.5}$ concentration, the greater the error value. Because of the difference in variance and complexity.

Considering the original purpose of interpolation, we can predict values at unobserved locations and times. Thus, we can obtain a super-resolution spatio-temporal map. Through the previous test results, it is expected that this will provide more accurate super-resolution map than those obtained by the traditional kriging method.

		January 9		February 1		June 29	
		MSPE	MAPE	MSPE	MAPE	MSPE	MAPE
Kriging	Mean	405.095	19.932	65.416	8.006	13.616	3.665
	SD	111.846	2.850	18.889	1.172	3.170	0.441
Ours	Mean	97.975	9.864	39.785	6.163	12.274	3.448
	SD	15.938	0.821	17.763	1.342	4.999	0.623
Method 1	Mean	142.664	11.763	48.008	6.790	13.372	3.612
	SD	53.983	2.075	19.718	1.381	4.552	0.570
Method 2	Mean	130.154	11.270	53.264	7.090	12.705	3.518
	SD	41.879	1.772	26.619	1.729	4.633	0.572
Method 3	Mean	271.870	16.286	77.764	8.695	13.176	3.591
	SD	87.994	2.577	28.103	1.472	4.179	0.533
Method 4	Mean	99.570	9.941	39.226	6.137	12.277	3.448
	SD	17.843	0.865	16.030	1.249	4.935	0.622
Method 5	Mean	101.024	9.995	40.679	6.233	12.556	3.493
	SD	21.448	1.061	17.773	1.353	4.792	0.595
Method 6	Mean	102.445	10.081	40.036	6.191	12.547	3.491
	SD	19.028	0.914	17.025	1.306	4.837	0.602

Table 5.2: Results that provides the mean and standard deviation of validation MSPE and MAPE values by date of $PM_{2.5}$ concentration observation.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This paper dealt with the spatio-temporal prediction method using DNN structure. For this, we studied the spatio-temporal random process, kriging method, and MLPs structure. Then, a novel methodology was specified using the properties studied. This method was expected to learn complex spatio-temporal data better using deep learning.

Also, we studied the theoretical properties of this method. By statistical learning theory, two learning models, our method and the traditional kriging method, were compared. As a result, we showed that the approximation error of our method is smaller than that of the traditional method. This means that the model has more capacity so the performance of our method is expected to be good for complex big data. We also considered the ensemble of predictors using different training sets.

In addition, We applied this method to real-world spatio-temporal data. As data, $PM_{2.5}$ concentration data in Korea was used. In this process, we specified a more detailed structure of our method. The K-fold cross-validation test was conducted and a test method for spatio-temporal data was introduced. These results provided that our method had a smaller prediction error than the traditional method.

6.2 Future Work

This paper proposed an interpolation method for the spatio-temporal data. Future study is to predict the future value of the spatio-temporal data. To do this, we seem to have a good idea through a time series of spatial processes (Wikle et at., 2019):

$$\{Y_t(\cdot) : t = 0, 1, \cdots\},$$

with the time space $D_t = \{0, 1, 2, \cdots\}$. In other words, deep learning is applied by treating our data as time series data. Neural network structures for predicting time series data include Long Short-Term Memory (LSTM, Hochreiter et al., 1997) and Gated Recurrent Unit (GRU, Chung et at., 2014). If there is enough time data, spatio-temporal data prediction using this idea is expected to perform well.

Bibliography

[1] Wikle CK, Zammit-Mangion A, Cressie N. (2019). Spatio-temporal statistics with R. Chapman and Hall/CRC Press

[2] Cressie N. (1993). Statistics for spatial data. Wiley

[3] Cressie N, Wikle CK. (2011). Statistics for spatio-temporal data. Wiley

[4] LeCun Y, Bengio Y, Hinton G. (2015). Deep learning. Nature

[5] Floriän Kastner, Benedikt Janßen, Frederik Kautz, Michael Hübner. (2018). Exploring deep neural networks for regression analysis.

[6] Wikle CK, Zammit-Mangion A. (2022). Statistical deep learning for spatial and spatio-temporal data. arXiv preprint arXiv:2206.02218

[7] Chen W, Li Y, Reich BJ, Sun Y. (2021). DeepKriging: Spatially dependent deep neural networks for spatial prediction. arXiv preprint arXiv:2007.11972

[8] José-María Montero, Gema Fernández-Avilés, Jorge Mateu. (2015). Spatial and spatio-temporal geostatistical modeling and kriging. Wiley

[9] Hastie T, Tibshirani R, Friedman J. (2009). The elements of statistical learning: Data mining, inference, and prediction, Second edition. Springer

[10] Ian Goodfellow, Yoshua Bengio, Aaron Courville. (2016). MIT Press

[11] Cybenko G. (1989). Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals, and Systems

[12] Kidger Patrick, Lyons Terry. (2020). Universal approximation with deep narrow networks. arXiv preprint arXiv:1905.08539

[13] Ulrike von Luxburg, Bernhard Schölkopf. (2011). Statistical Learning Theory: Models, Concepts, and Results. arXiv preprint arXiv:0810.4752

[14] Durrett R. (2019). Probability: Theory and examples, Fifth edition. Cambridge series in statistical and probabilistic mathematics

[15] Halsey Royden, Patrick Fitzpatrick. (2010). Real analysis, Fourth edition. Pearson

[16] Basawaraj Gadgay, Subhash Kulkarni, Chandrasekhar B. (2012). Novel ensemble neural network models for better prediction using variable input approach. International journal of computer applications

[17] Breiman L. (1996). Bagging predictors.

[18] Hanna Meyer, Christoph Reudenbach, Tomislav Hengl, Marwan Katurji, Thomas Nauss. (2018). Improving performance of spatio-temporal machine learning models using forward feature selection and target-oriented validation.

[19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of machine learning research

[20] Sergey Ioffe, Christian Szegedy. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167

[21] Sepp Hochreiter, Jürgen Schmidhube. (1997). Long short-term memory. Neural computation

[22] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint prearXiv:1412.3555

국문초록

크리깅은 관측한 시공간 데이터를 이용해 관측되지 않은 위치를 예측하는 통계적 기법이다. 데이터를 예측하여 보간하는데 시공간 데이터의 시공간 의존성을 이용한다. 그러나 복잡한 데이터의 경우 크리깅은 최적의 예측값이 되지 않을 수 있다. 최근 심층 신경망을 이용한 딥러닝은 많은 분야에서 활용되고 있다. 다층 퍼셉트론을 회귀에 활용할 수 있다는 점을 이용해 본 논문에서는 이 신경망 구조를 이용한 새로운 크리깅 방법을 제안한다. 이 방법은 더 복잡한 시공간 확률 과정을 학습할 수 있다. 그다음, 통계 학습 이론의 관점에서 기존의 크리깅 방법과 제안된 방법을 비교한다. 마지막으로, 실제 한국의 미세먼지 농도 데이터에 제안된 방법을 활용하여 이것의 성능을 평가한다. 여기서 교차 검증 방법을 사용한다.

주요어: 시공간 데이터, 크리깅, 딥러닝, 심층 신경망, 다층 퍼셉트론, 통계 학습 이론
학번: 2021-24984