# Automatic Variogram Inference Using Pre-Trained Convolutional Neural Networks

by

Abdelkerim Mokdad

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Mining Engineering

Department of Civil and Environmental Engineering
University of Alberta

© Abdelkerim Mokdad, 2023

# Abstract

Calculating and modeling a variogram from sparsely sampled data can be complex and time-consuming due to the need for expertise in selecting the appropriate parameters and fitting functions to the experimental variogram. To assist with variogram modeling, a novel approach based on Convolutional Neural Networks (CNNs) is implemented in this research to automatically calculate and model variograms from sparse data. CNNs are known for effectively processing grid-like data such as images and spatial maps making them suitable for geostatistical applications. To train the CNN model, a large set of Sequential Gaussian Simulation (SGS) realizations, each with different variogram model parameters, is resampled to generate the training data. Training a CNN directly on sampled data was not effective so each dataset is modeled using inverse distance; other interpolation techniques such as kriging and nearest neighbors were also tested, but inverse distance was found to be the most efficient for training data, the minimal number of input parameters for inverse distance is attractive.

The approach consists of the implementation of three CNNs: CNN-A predicts the major direction of continuity, CNN-1 estimates key variogram model parameters, and CNN-2 predicts the experimental variogram values at pre-set lag distances. Considering these CNNs, two workflows are implemented: (1) train CNN-1 to directly predict variogram model parameters (range, anisotropy ratio, nugget effect) followed by the inference of an automated model based on the predicted parameters (2) train a second CNN-2 to predict the experimental variogram values at specified lag distances which are easily autofit. Mean squared error (MSE) is used as the loss function in the training process. Once trained, the CNNs can predict the azimuth of the major direction, key variogram model parameters, and the experimental variogram of a given dataset. When applying CNN to datasets, data augmentation (rotation) is performed to obtain a set of variogram predictions that improves prediction robustness. Hyperparameter tuning and sensitivity analysis are performed to determine optimal parameters for the CNN by choosing configurations that minimize MSE and avoid overfitting.

The testing and validation of variogram prediction are conducted on various datasets, including the Walker Lake, and a data validation consisting of 110 geological images transformed to grayscale values. The results indicate that the prediction of variogram model parameters (CNN-1) is limited by a fixed number of variogram structures and the second workflow (CNN-2) involving predicting values at pre-set lag distances performs better without limiting the number of variogram structures. CNN-2 results in the prediction of an experimental variogram that can be easily fit with automated variogram programs achieving an R-squared value of 0.95 for the validation dataset. The workflows

are fully automated and applicable to sparse or dense data but are currently limited to 2D datasets and the output consists of normal score variograms.

# Dedication

I dedicate this work to my parents Ridha and Sousou who have always been a great source of motivation, for their endless sacrifices, and for teaching me that knowledge has no limits. To my siblings, Meriem for always being my role model, Med for constantly believing in me and Fatma for her unique spirit and care. To my nephews Dedo, Scan, Vonvon, Aydouda, and Zizou for their priceless presence. To my uncle Tio Mokh for boosting my creativity since my early age. To Marwa and Wael for being my family away from home. Last but certainly not least, to Rym for giving me lasting loves.

# Acknowledgments

I would like to express my sincere gratitude to my supervisor Dr. Jeff Boisvert for the great learning experience I gained during my MSc, through his expertise and enthusiasm for the research topic. His continuous guidance, scientific rigorousness, and encouragement would not only mark my MSc experience but also my future professional life.

I am extremely thankful to Professor Clayton Deutsch for his invaluable support and guidance throughout my MSc.

My deepest appreciation goes to my committee members for their honorable presence and their precious time in evaluating the proposed research.

I am grateful to the Centre for Computational Geostatistics and Mitacs for the financial support throughout the duration of my MSc.

I would like to deeply thank all CCG students for their positive attitude, work ethics, amazing friendliness, and for being my family away from home.

# Table of Contents

# List of Tables

# List of Figures

# List of Symbols

| Symbol | Description |
|---|---|
| $a$ | Variogram range |
| $b$ | Convolutional biases |
| $d$ | Scalar distance |
| $F$ | Feature map |
| $f_x$, $f_y$ | Feature map height and width |
| $w_x$, $w_y$ | Filter height and width |
| $h$ | Lag separation vector |
| $K$ | Number of categories to predict |
| $k$ | Category or point being predicted |
| $l$ | Network layer number |
| $M$ | Max pooling layer of the network |
| $m_x$, $m_y$ | Max pooling window height and width |
| $N(h)$ | Number of pairs of data separated by a lag vector $h$ |
| $n$ | Maximum number of neighboring data used for inverse distance estimation |
| $pad_x$, $pad_y$ | Spatial padding in the directions x and y |
| $\hat{p}(k)$ | Predicted probability |
| $p(k)$ | True probability |
| $P$ | Number of continuous points to predict |
| $st_x$, $st_y$ | Stride in the directions x and y |
| $\theta$ | Network inner parameters |
| $U$ | Input of the network |
| $u_x$, $u_y$, $u_d$ | Input height, width, and depth |
| $W$ | Convolutional filter or ensemble of weights of the filter |
| $W_{ij}$ | Value of the weight of the filter at the position $ij$ |
| $\hat{y}(k)$ | Predicted value of the output |
| $y(k)$ | True value |
| $Z$ | Random variable |
| $z(u)$ | Random variable outcome at location $u$ |

# List of Abbreviations

| Abbreviation | Description |
|---|---|
| 1-D | One-dimensional |
| 2-D | Two-dimensional |
| 3-D | Three-dimensional |
| ANNs | Artificial Neural Networks |
| ASMC | Automatic Semivariogram Modeling by Convolutional Neural Networks |
| CE | Cross Entropy |
| CNNs | Convolutional Neural Networks |
| GSLIB | Geostatistical software library |
| LMC | Linear Model of (Co)regionalization |
| ML | Machine Learning |
| MLE | Maximum Likelihood Estimator |
| MPS | Multiple-Point Statistics |
| MSE | Mean Squared Error |
| PDF | Probability Density Function |
| ppm | parts per million |
| RCNN | Recursive Convolutional Neural Networks |
| ReLU | Rectified Linear Unit |
| RGB | Red, Green, Blue |
| RMSE | Root Mean Square Error |
| SGS | Sequential Gaussian Simulation |
| TanH | Hyperbolic Tangent |

# Chapter 1

# Introduction

This Chapter explores the challenges involved in calculating and modeling experimental variograms. The importance of variogram modeling in various geostatistical workflows and the parameters involved in the process are discussed in Section 1.1. This Section also addresses how these challenges motivate the automation of variogram calculation and modeling, which is a primary objective of this research. The purpose of this thesis, which aims to automate the calculation and modeling of variograms, is outlined in Section 1.2 and the structure of this thesis is summarized in Section 1.3.

## 1.1 Problem Setting and Background

The process of calculating and modeling a variogram from sparse data can be a challenging task in geostatistics. A variogram is defined as the average square difference between two sampled data values, separated by the same lag vector (Journel and Huijbregts, 1978). It is an essential tool in geostatistical modeling that measures and describes the spatial variability of a random variable distributed in space (Rossi and Deutsch, 2014) and it is required to perform multiple geostatistical procedures such as estimation and simulation. The conventional workflow to infer and model a variogram from sparse data starts first by calculating the experimental variogram based on the location of the sampled data points and the values of the variable of interest. Once the experimental variogram is calculated, a series of known, analytical positive definite functions are fitted to the experimental variogram points (Goovaerts, 1997) resulting in a variogram model. The most commonly used positive definite functions are the spherical, Gaussian, and exponential models. These fitted models will be then used for estimation and simulation. The principal variogram model parameters that are obtained from the process are the sill, ranges, nugget effect, azimuth of the major and minor direction, and contribution of the variance (Chiles and Delfiner, 2012).

The conventional process of inferring and modeling a variogram from sparse data, as described above, can be a time-consuming and technically challenging task, requiring a significant amount of expertise in geostatistics. One of the major challenges is selecting the appropriate search and tolerance parameters for the calculation of the experimental variogram, such as the lag distance, the number of lags, the lag tolerance, and the azimuth tolerance. These parameters must be chosen carefully, as selecting inadequate values can result in experimental variograms that are noisy and difficult to model (David, 1988), especially in an early stage of geostatistical modeling, where there is limited data available. This makes the process highly dependent on the geostatistical knowledge and experience of the practitioner, which can be a limitation. Furthermore, the calculated experimental

variogram will be further subject to model fitting which is another source of complexity. The challenges associated with variogram modeling and calculation, as described above, have led to the need for automated methods. To address this problem, various robust and automatic variogram modeling techniques have been proposed and implemented in geostatistical workflows. These methods aim to minimize the number of parameters that need to be selected and to reduce the overall time required for modeling by the user. Some of these methods include the use of optimization algorithms and Machine Learning (ML) techniques.

Among the proposed methods modelling by iterative least squares (Desassis and Renard, 2013) is a method that uses an optimization algorithm to minimize the distance between the model and the experimental variogram. This method is based on the iterative least squares optimization algorithm, which is designed to find the best-fitting model parameters that minimize the objective function. The VARFIT program, proposed by Larrondo et al. (2003) is a semi-automatic variogram modeling approach, where an initial model is proposed, and then updates are made by iterating on the initial proposed model until a model that minimizes the objective function is obtained. Another semi-automatic method is Maximum Likelihood Estimator (MLE), developed by Todini and Pellegrini (1999) for semi-variogram parameters in ordinary kriging. This method is based on the assumption of a multi-normal distribution of the kriging cross-validation errors and by using MLE algorithm it can estimate the semi-variogram parameters. The problem of automatically calculating and modeling a variogram can be approached using ML techniques.

One of the most widely-used subsets of ML algorithms is Convolutional Neural Networks (CNNs). Originally proposed by LeCun et al. (1989) CNN is a supervised learning technique that has proven highly effective in a wide range of image processing and recognition tasks. These algorithms are able to learn to identify patterns and objects in images. Additionally, CNNs have been shown to excel in processing grid-like topology input data (Goodfellow et al., 2016) such as images and spatial maps. CNN algorithms have many advantageous properties that make them well-suited for geostatistical applications. With the ability to capture features from One-dimensional (1-D), Two-dimensional (2-D), and Three-dimensional (3-D) input grids, CNNs are able to preserve the spatial information of the data. They can predict both categorical (e.g., facies classification) and continuous variables (e.g., variogram range). Additionally, the final layer of the CNN can predict multiple output values, allowing for the prediction of an exhaustive subsurface model estimate, such as the values of a variable at unsampled locations or the value of the experimental variogram points at multiple lag distances. CNNs have been used in several research studies to tackle geostatistical problems. For example, Avalos and Ortiz (2020) proposed a new technique based on Recursive Convolutional Neural Networks (RCNN) for Multiple-Point Statistics (MPS) simulation, providing an alternative to traditional MPS methods by extracting features and relationships from spatial data. Similarly, Santos et al. (2021) employed a 3-D CNN algorithm to predict fluid flow through porous media by analyzing 3-D digital rock images, leveraging the spatial relationship between the porous medium

morphology and the fluid velocity field.

This thesis aims to investigate the advantages of using CNNs in geostatistics and the ease of applying them to gridded data. To address the problem of automatic variogram calculation and modeling, two methodologies are proposed. The first approach presents an automatic variogram modeling program that utilizes CNN algorithms to determine the key variogram model parameters, such as variogram ranges, the direction of maximum continuity, anisotropy ratio, and nugget effect. This method is noteworthy for its ability to generate a variogram model directly from the data without the need for intermediate steps, such as calculating the experimental variogram, in both major and minor directions of continuity. The second methodology is a two-step workflow, where a CNN is initially implemented to automate the calculation of experimental variogram values at different lag distances.

The primary goal of this research is to develop a workflow that automates the process of variogram modeling by minimizing the number of parameters that need to be selected by the user. This is achieved through the use of CNN algorithms for the automatic prediction of a stable experimental variogram, which is simpler to model and results in a more precise fit of the variogram. Furthermore, the proposed workflow significantly reduces the time required for user input, resulting in a more time-effective modeling process. These advantages provide a significant time reduction over traditional methods, which often require a large number of parameters and tolerances to be selected manually. In order to implement the proposed workflow, a CNN must be trained using appropriate training data. To generate this training data, multiple Sequential Gaussian Simulation (SGS) realizations are generated and resampled, with each realization being generated using different variogram models, characterized by various ranges, major directions of continuity, anisotropy ratios, and nugget effect values. The goal of using multiple combinations of variograms for building the training data is to ensure that the data is representative of a wide range of possible variograms and can provide reliable predictions for a variety of input data. This approach allows the CNN to be optimized to predict variogram model parameters for a wide range of input data, rather than being constrained to a single dataset. The optimization strategy applied in this work aims to generalize predictions to multiple input data, thus requiring the CNN model to be trained only once, rather than repeated for each case study. The resampled SGS realizations are then transformed into an exhaustive model estimation using inverse distance weighting. Other interpolation techniques such as kriging and nearest neighbors were also tested, but inverse distance weighting was found to be the most efficient for training data. The use of inverse distance estimates as input for training the CNN was chosen to help the network capture more spatial features, which improves the performance of the CNN model. Additionally, by using the inverse distance estimates, the network can learn from denser, more informative training data compared to using sparse data alone.

Historically, Artificial Neural Networks (ANNs) have been viewed by many practitioners as a ”black box” with limited control over the inner workings of the algorithm (Kanevski et al., 2004). However,

recent studies (Bochinski et al., 2017; Passos and Mishra, 2022) have shown that the performance of an ANN model can be significantly improved through the process of hyperparameter tuning. The spatial modeling process can be further enhanced by combining the ML with geostatistical tools (da Silva et al., 2022; Samson, 2019). To achieve the best possible results in variogram modeling using CNN, it is essential to find the optimal hyperparameters for the CNN architecture. Though an existing CNN model may already perform well, it can still be improved through the hyperparameter tuning process. However, the high number of hyperparameters in a CNN architecture makes it challenging to decide which to prioritize and what range of values to test when tuning a model. In this work, the most critical and relevant hyperparameters are first identified, then the values of these hyperparameters are adjusted and evaluated based on the resulting Mean Squared Error (MSE) of each of the tested models. The hyperparameters of the model that yields the lowest MSE are then used to implement the final CNN model.

## 1.2 Thesis Statement

The primary goal of this research is to present a CNN-based approach for automating the process of variogram calculation and modeling. The proposed approach aims to generate stable experimental variograms that are easier to model, predict the key variogram model parameters, minimize the number of parameters required to be selected by the user, reduce subjectivity in the modeling process, and save time for the user. By automating the calculation and modeling of variograms, this approach provides a less time-consuming alternative to traditional methods.

The thesis statement is: CNN-based approach for automating the process of variogram calculation and modeling which aims to generate stable experimental variograms, predict key variogram model parameters, minimize user interaction, and improve resource modeling efficiency and time-saving.

## 1.3 Thesis Outline

CHAPTER 1 of this thesis serves as the introduction and provides an overview of the problem setting and motivation for the research. The objectives and scope of the thesis are also outlined in this chapter.

CHAPTER 2 is a literature review that covers the key aspects of the proposed research, including the calculation and fitting of experimental variograms, the definition and applications of CNNs, and related research in the field of geostatistics.

CHAPTER 3 provides an overview of the proposed CNN algorithm and presents the general framework and methodology of the proposed automatic variogram inference workflow along with the significant findings obtained from its implementation.

CHAPTER 4 presents case studies that apply the proposed CNN approach to 2-D datasets.

CHAPTER 5 delves into the sensitivity analysis for the training data and CNN hyperparameters, and the importance of selecting the appropriate values for these parameters.
CHAPTER 6 concludes the thesis with a summary of the findings, recommendations for future work, and an analysis of the limitations of the proposed technique.

# Chapter 2

# Literature Review and Background

This Chapter provides a brief review of the fundamental concepts of spatial variability and variograms, including their calculation and modeling techniques. The literature review serves as the foundation and motivation for the implemented automatic variogram inference methodology. Section 2.1 is a literature review of the main aspects surrounding this research, i.e., calculating and fitting an experimental variogram, and summarizes alternatives to the variogram modeling procedure. Section 2.2 introduces the concept of ML and its applications in geostatistics. It provides a comprehensive overview of CNNs, including their definition, applications, and the main building blocks of their architecture. It also highlights the recent advancement and application of CNN in the domain of geostatistics.

## 2.1 Variogram Calculation and Modeling

The spatial variability of a random function $Z(\mathbf{u})$ can be characterized by the variogram function (Journel and Huijbregts, 1978), as defined by Equation 2.1. The variogram represents the average of the squared differences between two sample values, separated by a fixed lag distance (Goovaerts, 1997).

$$2\gamma(\mathbf{h}) = \frac{1}{N(\mathbf{h})} \sum_{i=1}^{N(\mathbf{h})} \left[ Z\left(\mathbf{u}_i\right) - Z\left(\mathbf{u}_i + \mathbf{h}\right) \right]^2 \tag{2.1}$$

Where $N$ is the number of paired points, $\mathbf{h}$ is the lag vector between two sampled points, and $Z(\mathbf{u}_i)$ represents the value of the variable $Z$ at the location $\mathbf{u}_i$.

Most geostatistical workflows involve the use of variograms, which is an essential tool in geostatistical modeling. Variograms measure and describe the spatial variability and continuity of a random variable distributed in space (Rossi and Deutsch, 2014). They are essential for performing geostatistical procedures such as estimation (e.g., kriging) and simulation (e.g., SGS). Estimation techniques such as kriging require solving kriging systems or normal equations, where the variogram model is used to account for the spatial variability between sampled and unsampled locations. Typically, quantifying spatial variability and variogram modeling are conducted through the following steps:

1. Calculate the experimental variogram.

2. Select and fit a suitable variogram model to the experimental variogram.

Step 1 starts by identifying the direction of major continuity in a spatial dataset by calculating the experimental directional variograms in multiple directions. This involves using different lag distances and directions, where there are sufficient numbers of paired data (Rossi and Deutsch, 2014). The variogram of the major direction of continuity is considered to be the one having the most continuity (Samal et al., 2011). In addition to calculating experimental variograms, other methods can also be used to detect directionality and patterns in a dataset. For example, data visualization can be useful in detecting patterns if enough samples are available. Another alternative visualization tool is the variogram map, which can be used to detect directions of anisotropy in the data. The variogram map is defined as the representation of variances of data points in different directions, where each grid cell represents the average of variances of a group of paired points aligned along the same direction (Chiles and Delfiner, 2012).

Step 2: Once the direction of major continuity detected, the next step is to model the calculated experimental variogram. The modeling process involves selecting the number and types of variogram structures and fitting a positive definite function to the experimental variogram (Goovaerts, 1997). The fitted analytical model may be composed of one or more structures, and the combination of structures is referred to as the Linear Model of (Co)regionalization (LMC). The LMC is a combination of positive definite structures, and among the most commonly used structures are the spherical, gaussian, and exponential types represented respectively by Equations 2.2, 2.3, and 2.4.

$$\text{Spherical } (d) = \begin{cases} 1.5(\text{ d/a}) - 0.5(\text{ d/a})^3, \text{ d} \leq \text{a} \\ 1, \text{ otherwise} \end{cases} \tag{2.2}$$

$$\text{Exponential(d)} \ = 1 - \exp(-3d/a) \tag{2.3}$$

$$\text{Gaussian(d)} \ = 1 - \exp\left(-3(d/a)^2\right) \tag{2.4}$$

Where $d$ is a scalar distance and $a$ is the range. Parameters are estimated by fitting the models to the experimental variograms. The spherical structure is often used as a first approximation, then more complex structures are tried in case of a poor fit. The modeling procedure outlined in steps 1 and 2, where the experimental variograms are calculated in multiple directions using the available sample data, and then modeled using positive definite functions, is commonly referred to as parametric modeling. This is because the variogram is represented by a set of parameters such as the range of the major and minor directions, the contribution of variance, the azimuth of the major direction, the anisotropy ratio, and the nugget effect. These parameters are used to represent the underlying spatial structure and patterns of the data.

Achieving satisfactory variogram modeling can be a challenging task, as it requires selecting a high number of parameters such as the lag distance, the number of lags, the lag tolerance, and the azimuth tolerance. These parameter selections have a direct impact on the final result and a poor selection can lead to noisy experimental variograms which are hard to model. There are various

automated and semi-automated alternatives available to assist with variogram modeling (Cressie, 1985; Desassis and Renard, 2013; Emery, 2010; Larrondo et al., 2003; Li et al., 2018; Mardia and Marshall, 1984). These alternatives can be useful for efficiently calculating experimental variograms and fitting functions.

The least-squares workflow is a commonly used optimization method in variogram modeling. It involves fitting a model to the experimental variogram by minimizing a cost function (Cressie, 1985). This cost function, also known as a misfit, measures the differences between the experimental variogram values and the fitted model at the same specified lag distances. Common measurements used to estimate the misfitting of the defined cost function include the MSE. Least squares have been widely used in automatic variogram modeling, with different optimization techniques being employed in the process. These techniques include random perturbation (Larrondo et al., 2003) simulated annealing (Emery, 2010; Wilde and Deutsch, 2012), the Gauss–Newton algorithm (Desassis and Renard, 2013), and the genetic algorithm (Li et al., 2018).

The maximum likelihood method is also used for variogram modeling, which involves directly estimating the variogram parameters, such as the range, sill, and nugget effect, by minimizing a negative log-likelihood function (Mardia and Marshall, 1984; Oliver and Webster, 2014). This method is considered to be attractive because it does not require an intermediate step, such as calculating the experimental variogram. However, one limitation of this method is that it is only effective if the variables under consideration are multivariate Gaussian distributed (Earle et al., 2007). The maximum likelihood method has been widely used in geostatistics (Dietrich and Osborne, 1991; Kitanidis, 1987; Kitanidis and Lane, 1985; Mardia and Marshall, 1984; Watkins and Mardia, 1992). Todini and Pellegrini (1999) developed a maximum likelihood estimator for variogram parameters in ordinary kriging, based on the assumption of a multivariate normal distribution of the kriging cross-validation errors. Overall, the maximum likelihood method is a useful approach to variogram modeling, as it provides a way to directly estimate variogram parameters without the need to calculate the experimental variogram. However, this method is only applicable in situations where the data is multivariate gaussian distributed.

In addition to the methods previously mentioned, Larrondo et al. (2003) proposed a program for semi-automatic variogram modeling referred to as VARFIT. The program starts by proposing an initial model with random parameters such as variance contributions, nugget effect, number of structures, and structure type, then iteratively updates the initial proposed values until a model that minimizes the objective function and reduces the error is obtained.

The problem of automatically calculating and modeling a variogram could also fall into the category of ML. Dimitrakopoulos (1993) proposed an ML algorithm that automatically calculates and evaluates an experimental variogram based on neural networks. This approach uses the capability of neural networks to learn patterns and relationships in data. Similarly, Manchuk and Deutsch (2019) proposed an ML algorithm that automatically fits an experimental variogram by implement-

ing a multi-layer feed-forward neural network that predicts the azimuth of the major direction, the anisotropy ratio, and the variance contributions. Also, Jo and Pyrcz (2022) proposed a 2-stage CNN approach to predict the variogram model parameters. This approach uses the capability of CNNs to learn spatial patterns in data. The results of those studies are promising and they show that the use of ML methods is a viable alternative to classical geostatistics methods in variogram modeling. The methodology implemented in this thesis is similar to the Automatic Semivariogram Modeling by Convolutional Neural Networks (ASMC) proposed by Jo and Pyrcz (2022) but it has some key differences. The three main differences are:

1. The primary advantage of the program implemented in this research is its generalizability, eliminating the need for retraining on each dataset. This is achieved through a pre-trained model that uses a larger training dataset (100 times larger; 100,000), which requires more implementation time but runs only once.

2. The present method does not use the sparsely sampled data as input but uses inverse distance estimates to train the CNN. This allows the network to learn from denser, more informative training data compared to using sparse data alone.

3. Another contribution of this work is the estimation of the experimental variogram and the nugget effect. In contrast, ASMC does not include the estimation of the nugget effect and experimental variogram.

## 2.2 Convolutional Neural Networks

CNNs (LeCun et al., 1989) are a type of ANNs that is widely used in ML for image recognition and feature extraction tasks. They are particularly effective at processing data with a grid-like topology (Goodfellow et al., 2016), such as images or spatial maps. CNNs consist of a feed-forward network that is generally composed of three main layers: a convolutional layer, a pooling layer, and a dense fully connected layer (Figure 2.1). The network is divided into two zones: a feature extraction zone and an inference zone (Avalos and Ortiz, 2019). In the feature extraction zone, the convolutional and pooling layers extract spatial features from the input data, while in the inference zone, the fully connected layer makes predictions based on those features. The weights $W$ and biases $b$ of the CNN, also known as its inner parameters $\theta = \{W, b\}$, play a key role in the training process. Depending on the nature of the problem, the prediction made by a CNN may be either a continuous or a categorical variable.

To extract spatial features from a gridded input, CNNs use convolutional filters, also known as kernels. These filters consist of multiple weights $W$ that slide over the input and perform element-wise multiplication between the kernel values and their corresponding values in the input (Haykin, 2009). The weights and biases are initially randomly assigned and are updated during each iteration of the

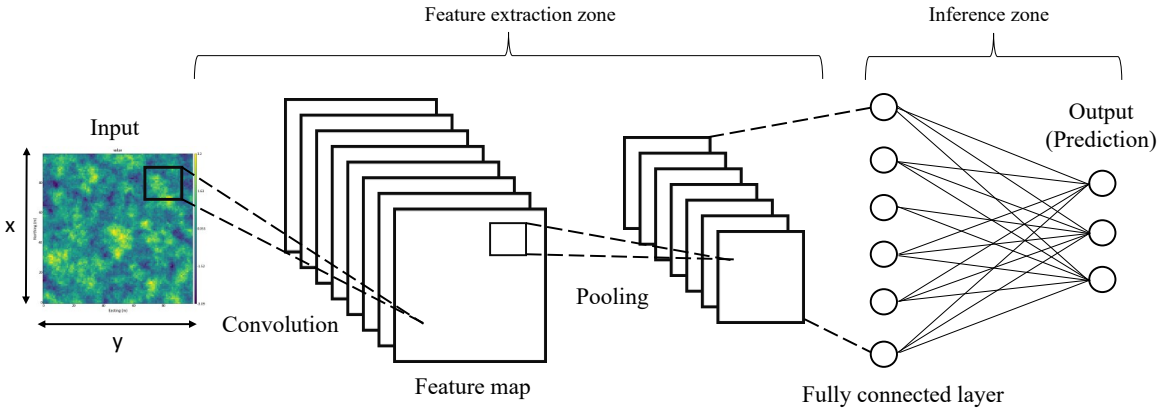


**Figure 2.1:** Main building blocks of a CNN : convolutional, pooling, and fully connected layers

training process until the error of the model, as measured by an objective function, is minimized. The training process involves the use of optimizers that employ gradient-based techniques (Kingma and Lei Ba, 2015) to minimize a loss function, such as MSE. In this way, CNNs can adaptively learn spatial features from the input data and use them to make predictions on unseen data.

### 2.2.1 CNN building blocks

This Section aims to provide a comprehensive overview of the key components and their functions in a 2-D CNN architecture, including input data $U$, convolutional filters composed by multiple weights $W$, bias vectors $b$, feature maps $F$, convolutional layers, activation functions, max-pooling layers $M$, fully connected layers, and various operations such as convolution, pooling, dropout, and batch normalization. Additionally, it also covers the process of training and optimizing these networks.

- Input data:

In a 2-D CNN, the input data is structured as a grid, with dimensions ($u_x$, $u_y$, $u_d$). The input height and width, $u_x$ and $u_y$, correspond to the number of rows and columns in the grid, respectively. The input depth, $u_d$, represents the number of channels in the input data $U$. For example, in image processing, a color image would have $u_d$ equal to 3, representing Red, Green, Blue (RGB) filters, while a grayscale image would have $u_d$ equal to 1. The grid-like structure of the input allows CNNs to efficiently process data with a spatial structure (Goodfellow et al., 2016).

- Convolutional layers:

These layers are the core of the CNN, they apply a convolution operation to the input data, which involves sliding a kernel across the data and computing an element-wise multiplication between the weights the filter $W_{(i,j)}$ and the values in the window of the input $U_{(i,j)}$ resulting in a feature map $F_l$. The obtained feature map is subsequently used as input for the next layer, which, in turn, generates a new feature map $F_{l+1}$ accordingly, this process is repeated for the number of layers

used, generating new feature maps for the remaining layers. The mathematical expression for the convolutional operation is given by Equation 2.5:

$$F_l(i,j) = \Big( \big( \mathrm{U} * W_l \big) (i,j) + b_l \Big) \tag{2.5}$$

Where $F_l$ is the output feature map, computed by applying the convolutional operation (*) between the input layer $U$ (which can be an image, spatial map or sampled data) and the filter $W$. The weights at position $(i, j)$ in the filter are denoted by $W_{(i,j)}$.

CNN uses various techniques to process and analyze data, including the use of convolutional filters, activation functions, and additional functions such as stride and spatial padding. These techniques work together to extract features and patterns from the input data (Guo et al., 2016). Stride is a parameter that determines the movement of the convolutional filters as they are applied to the input data. It controls the distance that the filters move in each step, and a larger stride results in a smaller output. Spatial padding is the number of pixels added to the edges of the input before the convolutional filters are applied (Yamashita et al., 2018). This padding is used to preserve the spatial dimensions of the input and prevent the output from being smaller than the input.

While these functions can be useful in certain situations, they are not always necessary (Avalos and Ortiz, 2019). The basic convolution operation and activation functions are essential components of a CNN, while stride and spatial padding may be used as additional techniques to optimize performance. Both the stride and spatial padding can be adjusted to trade off computation time and network performance.

$$f_x = \left[ \frac{u_x - w_x + 2pad_x}{st_x} \right] + 1 \tag{2.6}$$

$$f_y = \left[ \frac{u_y - w_y + 2pad_y}{st_y} \right] + 1 \tag{2.7}$$

Equations 2.6 and 2.7 represent the height $f_x$ and width $f_y$ of the output feature map. $u_x$ and $u_y$ are the height and width of the input, $pad_x$ and $pad_y$ are the values of padding applied in each direction to the input, $w_y$,$w_y$ are the sizes of the filters, and $st_x$, $st_y$ are the convolutional stride.

- Activation functions:

Activation functions, such as Rectified Linear Unit (ReLU), Hyperbolic Tangent (TanH), and Sigmoid, represented by Equations 2.8, 2.9 and 2.10 and illustrated in Figure 2.2, introduce non-linearity to the network, allowing it to learn complex and non-linear features in the data (Sharma et al., 2017).

$$\mathrm{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{2.8}$$

$$\mathrm{TanH}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.9}$$

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{2.10}$$

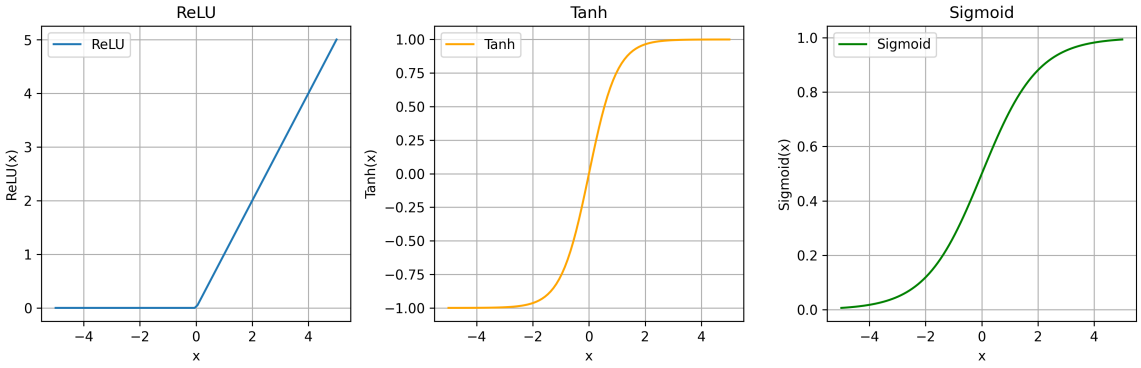


**Figure 2.2:** ReLU, Tanh, and Sigmoid activation functions

- Pooling layers:

Pooling layers are used to down-sample the input data by applying a pooling operation, such as max pooling, which selects the maximum value in each window (Zafar et al., 2022). These layers help to reduce the spatial dimensions of the data, as well as the number of learning parameters, improving statistical efficiency (Gholamalinezhad and Khosravi, 2020). Equations 2.11 and 2.12 are used to determine the dimensions of the feature map resulting from a pooling operation in a CNN. These equations have three key components: $f_x$ and $f_y$ are for the height and width of the input in the pooling operation. $M$ stands for max pooling, and $m_x$ and $m_y$ are the dimensions of the max pooling window used in the CNN.

$$fx = \left\lfloor \frac{u_x - m_x}{st_x} \right\rfloor + 1 \tag{2.11}$$

$$fy = \left\lfloor \frac{u_y - m_y}{st_y} \right\rfloor + 1 \tag{2.12}$$

- Fully connected layers:

These layers connect all the nodes in the previous layer to all the nodes in the current layer (Avalos and Ortiz, 2019), allowing the network to learn global patterns in the data. The output of the fully connected layers is passed through a final activation function to produce the final output of the CNN.

- Loss function:

A loss function is used to measure the prediction error of the network and the optimizer is used to adjust the weights and biases to minimize this error. Equation 2.13 is a common loss function for

CNNs called Cross Entropy (CE). It is used for categorical variables, and the goal is to minimize the difference between the predicted probability $\hat{p}(k)$ distribution and the true probability $p(k)$ distribution of the output. Equation 2.14 is another common loss function for CNNs is the MSE. It is used for continuous variables, and the goal is to minimize the difference between the predicted value $\hat{y}(k)$ and the true value of the output $y(k)$. In summary, the CE loss is used for classification problems and MSE is used for regression problems.

$$CE = -\sum_{k=1}^{K} (\log \hat{p}(k)) \cdot p(k) \tag{2.13}$$

$$\text{MSE} = \frac{1}{P} \sum_{k=1}^{P} \left( \widehat{\text{y}}_k - \text{y}_k \right)^2 \tag{2.14}$$

- Batch normalization:

Batch normalization (Ioffe and Szegedy, 2015) is a technique used to improve the stability and performance of CNN models. It works by normalizing the input data to have a mean of 0 and a standard deviation of 1 by subtracting the mean and dividing it by the standard deviation. This helps to improve the generalization of the model to new data, it also can improve the final performance of the model and speeds up the training process.

- Dropout:

Overfitting is a common issue in CNNs and other ML algorithms. It occurs when the model is too closely tailored to the specifics of the training data (Ying, 2019), resulting in poor performance on testing data or other datasets. Figure 2.3 (case 2) illustrates a clear case of overfitting, while Figure 2.3 (case 1) shows an ideal scenario with no overfitting. To prevent overfitting, there are several regularization techniques available, such as dropout. Dropout (Srivastava et al., 2014) is a popular method that randomly removes nodes from the network during training, which helps prevent the learned weights from adapting too much to the training data.

### 2.2.2 Training Process of a CNN:

The method of training a CNN includes adjusting the inner parameters of the network $\theta = \{W, b\}$, specifically the weights and biases, to minimize the difference between the predicted output and the target output on a labeled dataset. This process is done over several iterations, referred to as epochs, and the weights are adjusted based on the error calculated during each epoch. To train a CNN, the following steps are typically followed:

1. Prepare the dataset: The training data should be formatted and divided into training and validation sets. It is also common to apply data augmentation techniques, such as rotating or cropping the images, to increase the size of the dataset and improve the generalization of the model.

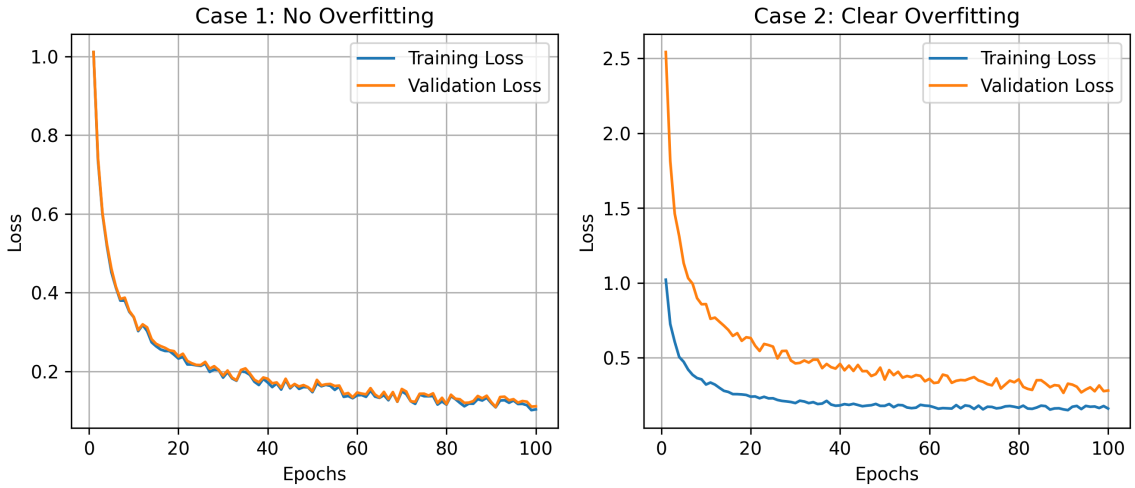


**Figure 2.3:** Loss curve over the training epochs of two CNN examples for both training and testing sets. No overfitting observed (case 1), and significant overfitting observed (case 2).

2. Define the model architecture: The CNN architecture should be defined, including the number and size of the convolutional and pooling layers, as well as the activation functions and regularization technique to be used.

3. Initialize the model parameters: The weights and biases of the filters should be initialized randomly.

4. Feedforward propagation of the network: The model makes predictions for a given input by performing a series of convolutions and pooling operations and applying the activation function.

5. Compute the loss: The difference between the predicted output and the true output is calculated using a loss function, such as the CE for categorical prediction and MSE for predicting continuous.

6. Backpropagation of the network: The gradients of the loss with respect to the model parameters are calculated using backpropagation.

7. Update the parameters: The model parameters are updated using an optimization algorithm, such as stochastic gradient descent, to reduce the loss.

8. The process is repeated for the specified number of epochs, using the updated parameters at each iteration.

9. Evaluate the model: The trained model is evaluated on the validation set to assess its performance. If the performance is satisfactory, the model can be used for prediction. If not, the model can be further tuned by adjusting the hyperparameters or the architecture through the hyperparameter tuning process.

## 2.3 Summary

This chapter provides a literature review and background on spatial variability and variograms. It starts by reviewing key concepts related to variogram modeling, including the calculation and fitting of experimental variograms. The chapter explores various automated alternatives for variogram modeling, such as the least-squares and maximum likelihood methods, which have been widely used in geostatistics. Additionally, the concept of ML and its applications in geostatistics is introduced, specifically focusing on CNNs, including their architecture, key components, and the process of training these networks. The literature review and background serve as the foundation for the proposed CNN-based approach in this thesis, which aims to automate variogram calculation and modeling.

## Chapter 3

# Automatic Variogram Inference with Convolutional Neural Networks

Chapter 3 presents the methodologies, results, and significant findings obtained from the two implemented workflows. The first methodology, referred to as workflow 1, consists of automatic parametric modeling and involves predicting key variogram model parameters directly from the data to infer the variogram model. The second methodology, workflow 2, calculates the normal score experimental variogram and fits the variogram model automatically.

The proposed methodologies are presented in Section 3.1 and involve several steps. First, a training dataset is created using SGS. Then, CNNs are trained for different tasks. The azimuth of the major direction is predicted using a first CNN denoted as CNN-A. Based on the estimated azimuth, another CNN, CNN-1, is trained to predict the key variogram model parameters, enabling the inference of the variogram model (workflow 1). Additionally, CNN-2, is trained to predict the experimental variogram values at pre-set lag distances and automatically fit the variogram model (workflow 2). The optimization of CNN hyperparameters and the application of data augmentation techniques, such as rotation, are performed to improve prediction robustness. The trained models are then tested using case studies.

Section 3.2 presents the results and findings of the implementation of these methodologies. Section 3.2.1 compares the predicted azimuth values to the true values using CNN-A. Section 3.2.2 evaluates the performance of CNN-1 in predicting the range, anisotropy ratio, and nugget effect. Section 3.2.3 presents the results of predicting the experimental variogram values using CNN-2. Section 3.2.4 provides a comparison between the results obtained from workflow 1 (using CNN-1) and workflow 2 (using CNN-2). Finally, Section 3.3 summarizes the key observations and discusses the results obtained from Sections 3.1 and 3.2. This Chapter aims to present an overview of the methodologies, their implementation, and the obtained results.

## 3.1 Methodology and Proposed Techniques

The proposed methodologies involve the following steps:

1. Creation of a training dataset using SGS.
2. Training Process:
    - Train a CNN that predicts the azimuth of the major direction.

- Based on the estimated azimuth, train a CNN to predict the key variogram model parameters and infer a model from the predicted parameters (workflow 1).
- Train a CNN to predict the experimental variogram values at different lag distances followed by an automatic fit of the variogram model (workflow 2).

3. Optimize CNN hyperparameters through hyperparameter tuning and conduct a sensitivity analysis to create the optimal training data for modeling a variogram.
4. Application of data augmentation (rotation) to improve the robustness of the prediction.
5. Model testing using case studies.

### 3.1.1 Training Data Creation

The process of generating training data to train the CNN is divided into several steps, as illustrated in Figure 3.1.

1. The training data generation process (Figure 3.1) involves resampling unconditional SGS realizations in a 100 x 100 grid using different variogram model parameters (Table 3.1). These SGS realizations are generated with different random seed numbers and parameters to improve CNN generalization and avoid the need to repeat training for each new dataset.
2. The relationship between the input and output is established by labeling each realization with its corresponding true variogram parameter value.
3. Transform the obtained sampled data from Step 1 into a subsurface model estimation using inverse distance. Inverse distance is used as input to enable the network to capture more spatial features, rather than feeding the network with only sparse sampled data.
4. The training data is randomly divided into two separate sets: the training set and the validation set, with the training set comprising 80% and the validation set 20% of the total dataset.

The training dataset comprises 100,000 realizations, with 80,000 used for training and 20,000 for validating. The large number of training samples is crucial for a CNN to effectively generalize predictions, learn, and capture features for various scenarios where the azimuth ranges from 0 to 180 degrees. The anisotropy ratio, varying from 1 to 3, the range is divided into 20 increments between 10 to 30. Additionally, the nugget effect is considered within the range of 0 to 0.3.

### 3.1.2 CNN Training

To predict the variogram model parameters and the experimental variogram, consider three different models CNN-A, CNN-1, and CNN-2 (Figure 3.2). CNN-A is used to learn the azimuth of the major direction. During training, CNN weights are updated and optimized by iteratively minimizing MSE

**Table 3.1:** Details of the variogram model parameters used to train the CNN models

| Variogram model parameter | Values |
|---|---|
| Major direction of continuity | Between 0 and 180 (degrees) |
| Range in the major direction | Between 10 and 30 (grid cells) |
| Anisotropy ratio | Between 1 and 3 |
| Nugget effect | Between 0 and 0.3 |
| Structure | Spherical |
| Number of nested structures | 1 |

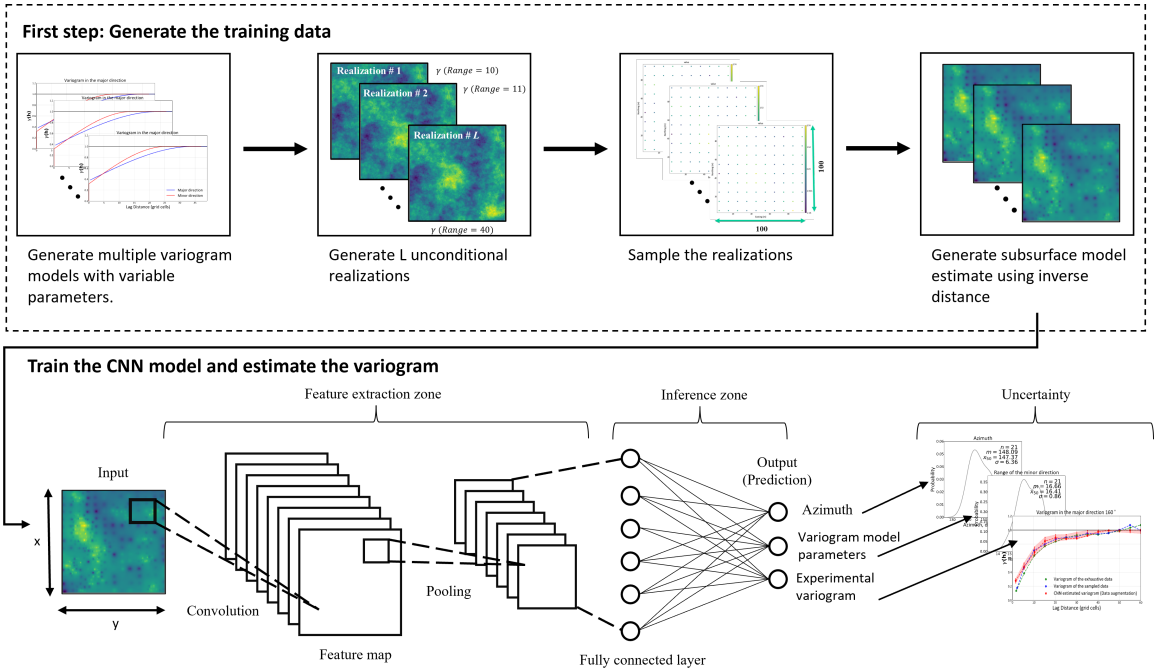


**Figure 3.1:** Workflow for the automatic calculation and modeling of the normal score variogram. The methodology includes the creation of training data, inverse distance estimation of the samples, and the training of three CNNs. The first CNN predicts the azimuth, while CNN-1 predicts key variogram model parameters.CNN-2 predicts experimental variogram values at preset lag distances.

and using the Adam optimizer until each input matches its target azimuth. Once the training is completed, CNN-A can predict the azimuth of the major direction of a given dataset.

For workflow 1, CNN-1 is trained on the output of CNN-A to predict the ranges in the major direction, anisotropy ratio, and the nugget effect. For workflow 2, CNN-2 is trained to predict the experimental variogram points at pre-set lag distances along the major/minor directions determined by CNN-A.

The azimuth is predicted first with CNN-A and 18 different models are implemented in CNN-1, where the azimuth values are divided into 18 increments separated by 10 degrees, ranging from 0 to 180 degrees. The resulting azimuth prediction from CNN-A is used to select an appropriate model for predicting the variogram range in the major direction, anisotropy ratio, and the nugget effect (Figure 3.2). In workflow 2, CNN-2 is trained to predict the experimental variogram values at pre-set lag distances (Figure 3.2).

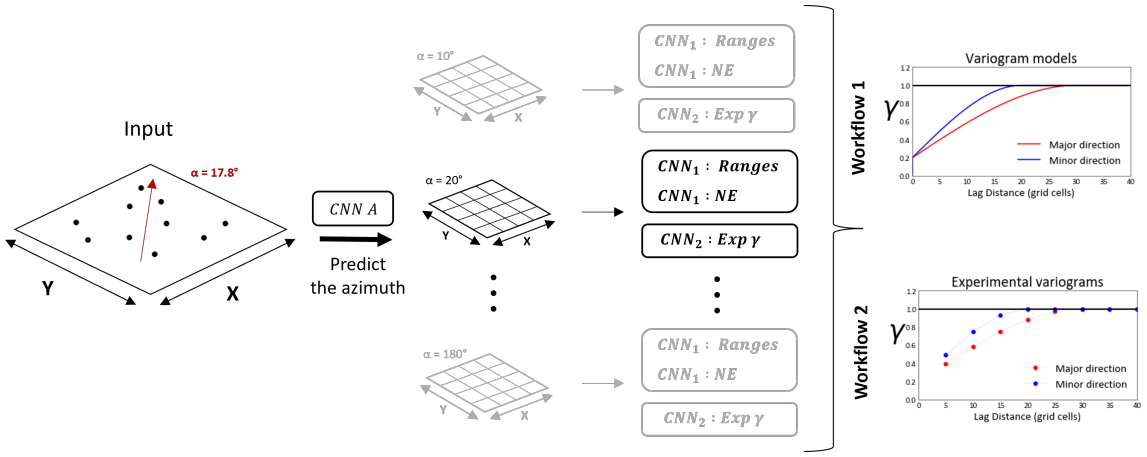


**Figure 3.2:** Schematic representation of the proposed methodology for predicting the variogram model parameters and the experimental variogram using three different CNN models. CNN-A predicts the azimuth, which is then used as input to CNN-1 for predicting the ranges in the major direction, anisotropy ratio, and the nugget effect in workflow 1. In workflow 2, CNN-2 predicts the experimental variogram values at preset distances.

The implemented CNNs use different architectures, which are obtained through hyperparameter tuning. The models with the lowest MSE and minimal overfitting are selected, with the key difference between workflows 1 and 2 being the number of convolutional layers and outputs of the final layer. Workflow 1 uses 4 convolutional layers and 3 outputs, representing the range in the major direction, anisotropy ratio, and nugget effect. In contrast, workflow 2 uses 6 convolutional layers and 13 outputs, representing variogram values at 13 lag distances, separated by 5-unit intervals. In this work, MSE is used as the loss function for CNN-A, CNN-1, and CNN-2. Figure 3.3 displays the MSE loss over 100 epochs for both the training and validation sets. All three CNNs exhibit a gradual decrease in loss over the 100 epochs, with the training and validation curves decreasing simultaneously. This indicates that there is no significant overfitting present, as evidenced by the minimal difference between the training and validation MSE values.

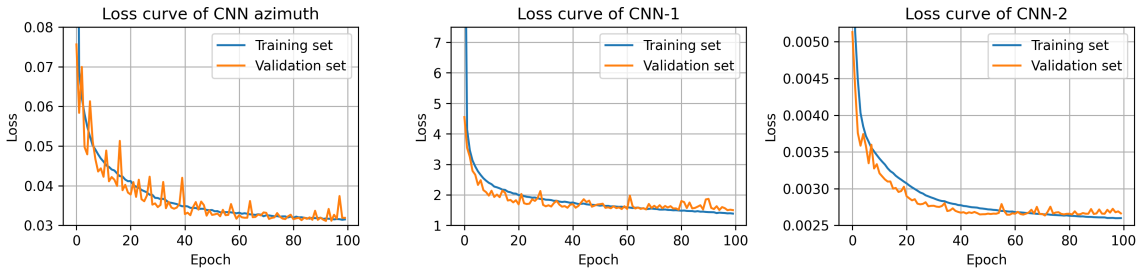


**Figure 3.3:** Training and validation loss for CNN-A, CNN-1, and CNN-2

The detailed structures of the CNN architectures used in this work are outlined in Tables 3.2, 3.3, and 3.4. The primary objective of the CNN is to extract spatial features from the training data and estimate experimental variograms and variogram model parameters, which is accomplished through a combination of convolution layers, activation functions, max-pooling layers, dropout layers, and

a dense fully connected layer with neurons representing the outputs. Each layer performs a unique function in feature extraction from the input data, as described below:

- The input is a 100x100x1 subsurface model obtained through inverse distance estimation of the sparsely sampled data. These input dimensions remain fixed for every trained CNN.
- The convolutional layers use filters that slide over the input data, performing convolutional operations. This process is repeated with multiple filters, resulting in multiple feature maps. As the architecture progresses, the number of filters increases, enabling the extraction of a larger number of features. In the first convolution, 8 filters were used, each measuring 3x3 in size, with the kernel size determined through hyperparameter tuning outlined in Chapter 5.
- The ReLU activation functions are applied to the output of the convolutional layers. The ReLU activation function is non-linear. This enables the convolutional layers to introduce non-linearity into the model and capture complex features.
- Pooling operation layer: The network then applies a maximum pooling operation to reduce the size of the feature map. This is done by taking the maximum value over a 2x2 window resulting in a feature map of size 50x50x8. The pooling operation is important as it reduces the number of learning parameters, improves statistical efficiency, and reduces computational time.
- A batch normalization layer is used to normalize the activations of a layer in a neural network. This normalization is done by subtracting the mean and dividing by the standard deviation of the activations for a given batch.
- The output goes through a dropout layer, which is a regularization technique that randomly drops neurons during the training, to prevent overfitting.
- Then the output goes through a flattened layer, which takes the output of the previous layers and flattens it into a 1-D array, to be passed to a fully connected dense layer.

These building blocks are repeated four times in CNN-A, four times in workflow 1 and six times in workflow 2, as determined through the hyperparameter tuning (Chapter 5). The selected architecture results in the highest accuracy with the lowest overfitting. Keras and TensorFlow version 2.4.0 (Chollet et al., 2015) are used to implement the CNN architecture described. The Adam optimizer, developed by Kingma and Lei Ba (2015), is used to optimize the weights of the convolutional filters during training.

During the training process, the CNN is presented with a set of input data and corresponding target outputs. The model then uses the convolutional filters with the initial randomly assigned weights and biases to process the input data. As the model processes, the weights and biases are

**Table 3.2:** The selected architecture of CNN-A used to predict 2 output values representing the sine and cosine of the azimuth of the major direction of continuity.

| Layer type | Output shape | Number of parameters involved |
|---|---|---|
| 2D convolution | (100, 100, 8) | 80 |
| Activation function (ReLU) | (100, 100, 8) | 0 |
| Maxpooling | (50, 50, 8) | 0 |
| 2D convolution | (50,50,16) | 1168 |
| Activation function (ReLU) | (50,50,16) | 0 |
| Maxpooling | (25,25,16) | 0 |
| 2D convolution | 25,25,32 | 4640 |
| Activation function (ReLU) | 25,25,32 | 0 |
| Maxpooling | 12,12,32 | 0 |
| 2D convolution | 12,12,8 | 2312 |
| Activation function (Tanh) | 12,12,8 | 0 |
| Dropout | 12,12,8 | 0 |
| Flatten | 1152 | 0 |
| Fully connected dense layer | 2 | 2306 |
| Output | 2 | |

**Table 3.3:** The selected architecture of CNN-1 used to predict 3 output values for workflow 1 representing the range in the major direction, anisotropy ratio, and nugget effect.

| Layer type | Output shape | Number of parameters involved |
|---|---|---|
| 2D convolution | (100, 100, 8) | 80 |
| Activation function (ReLU) | (100, 100, 8) | 0 |
| Maxpooling | (50, 50, 8) | 0 |
| 2D convolution | (50,50,16) | 1168 |
| Activation function (ReLU) | (50,50,16) | 0 |
| Maxpooling | (25,25,16) | 0 |
| 2D convolution | 25,25,32 | 4640 |
| Activation function (ReLU) | 25,25,32 | 0 |
| Maxpooling | 12,12,32 | 0 |
| 2D convolution | 12,12,8 | 2312 |
| Activation function (ReLU) | 12,12,8 | 0 |
| Dropout | 12,12,8 | 0 |
| Flatten | 1152 | 0 |
| Fully connected dense layer | 3 | 3459 |
| Output | 3 | |

adjusted to minimize the difference between the predicted output and the target output. This process of adjusting the weights and biases is known as backpropagation. To achieve the described backpropagation Adam optimizer is used. The Adam optimizer is an extension of the gradient descent algorithm that uses both the gradient of the loss function with respect to the parameters and the gradient information in order to update the parameters. In this research MSE is used as loss function in CNNs, which measures the difference between the predicted output and the target output. By minimizing the MSE, the model is able to learn the optimal weights and biases that will produce the most accurate predictions. In summary, the training process in a CNN is an iterative process that involves adjusting the weights and biases of the convolutional filters in order to minimize the difference between the predicted output and the target output. The Adam optimizer and the

**Table 3.4:** The selected architecture of CNN-2; used to predict 13 output values for workflow 2 representing 13 experimental variogram values at pre-set lag distances.

| Layer type | Output shape | Number of parameters involved |
|---|---|---|
| 2D convolution | (100, 100, 8) | 80 |
| Activation function (ReLU) | (100, 100, 8) | 0 |
| Maxpooling | (50, 50, 8) | 0 |
| 2D convolution | (50,50,16) | 1168 |
| Activation function (ReLU) | (50,50,16) | 0 |
| Maxpooling | (25,25,16) | 0 |
| 2D convolution | 25,25,32 | 4640 |
| Activation function (ReLU) | 25,25,32 | 0 |
| Maxpooling | 12,12,32 | 0 |
| 2D convolution | 12,12,64 | 18496 |
| Activation function (ReLU) | 12,12,64 | 0 |
| Maxpooling | 6,6,64 | 0 |
| 2D convolution | 6,6,128 | 73856 |
| Activation function (ReLU) | 6,6,128 | 0 |
| Maxpooling | 3,3,128 | 0 |
| 2D convolution | 3,3,8 | 9224 |
| Activation function (Sigmoid) | 3,3,8 | 0 |
| Dropout | 3,3,8 | 0 |
| Flatten | 72 | 0 |
| Fully connected dense layer | 13 | 949 |
| Output | 13 | |

MSE are used to optimize the training process and improve the accuracy of the model. Once the training process is complete, the model can be used to make predictions on unseen data.

### 3.1.3 CNN Hyperparameter Tuning and Sensitivity Analysis

- CNN hyperparameter tuning

Hyperparameter tuning of the CNN determines the optimal parameters for predicting the azimuth of the major direction, key variogram model parameters, and experimental variogram values. The parameters studied include the number of convolutional layers, kernel size, and dropout rate. The number of convolutional layers is varied from 2 to 8 in increments of 2, while the kernel size is tested at 2x2, 3x3, 4x4, and 5x5. The dropout rate is also examined at values of 0, 0.25, and 0.5. All possible combinations of these parameters are tried to determine the optimal set of hyperparameters for the CNN and summarized in Chapter 5. The parameters and their possible values selected for examination are listed in Table 3.5.

**Table 3.5:** Range of hyperparameter values tested

| Number of convolutional layers | Kernel size | Dropout |
|---|---|---|
| 2 layers | 2x2 | 0 |
| 4 layers | 3x3 | 0.25 |
| 6 layers | 4x4 | 0.5 |
| 8 layers | 5x5 | - |

The goal of the hyperparameter tuning analysis is to determine the optimal set of parameters for the network to improve its performance. By varying the number of convolutional layers, kernel size, and dropout rate, across the different configurations, the analysis aims to find the combination of these parameters that would result in the best performance.

- Sensitivity analysis to create the optimal training data:

The sensitivity analysis conducted in this work aims to create the optimal training data for a CNN to model a variogram by examining two main factors: (1) the sensitivity of the inverse distance parameters, and (2) the number of training data used. Regarding the sensitivity analysis of the inverse distance parameters, the parameters studied are the inverse distance power and the maximum data used. The values considered for these parameters are summarized in Table 3.6.

**Table 3.6:** Range of parameter values to test for inverse distance

| | **Considered Values** | | | | | | |
|---|---|---|---|---|---|---|---|
| **Inverse distance power** | 1 | 1.7 | 2 | 2.5 | 3 | 4 | 6 |
| **Maximum data used** | 12 | 24 | 48 | | | | |

The trained CNNs are then used to predict the experimental variogram for different case studies, and the average MSE is calculated for each model. The model with the lowest MSE value is selected as the most accurate model and its inverse distance parameters are considered optimal for the training process.

Regarding the sensitivity analysis of the number of training data, it is investigated by varying the number of training data from 1,000 to 100,000. This allows to understand the effect of the number of training data on model performance and determine the optimal number.

### 3.1.4 Application of Data Augmentation (Rotation)

Data augmentation is a commonly used ML technique that increases the number of training data by slightly modifying the original data (Maharana et al., 2022). This can be done through various processes such as flipping, rotation, adding noise, shifting, and cropping (Lashgari et al., 2020). In this work, data augmentation is performed by applying a 1-degree rotation of the dataset 20 times (Figure 3.4). This produces a set of variogram predictions where the results can improve prediction robustness and could be used to examine variogram uncertainty.

### 3.1.5 Model Testing Using a Case Study

The best-performing CNN models of CNN-A, CNN-1, and CNN-2 are selected from the training process through hyperparameter tuning based on their lowest MSE values. The proposed two methods: workflow 1 and workflow 2 are then used to predict the variogram model of a sparsely sampled input. The following are the steps used in the process :

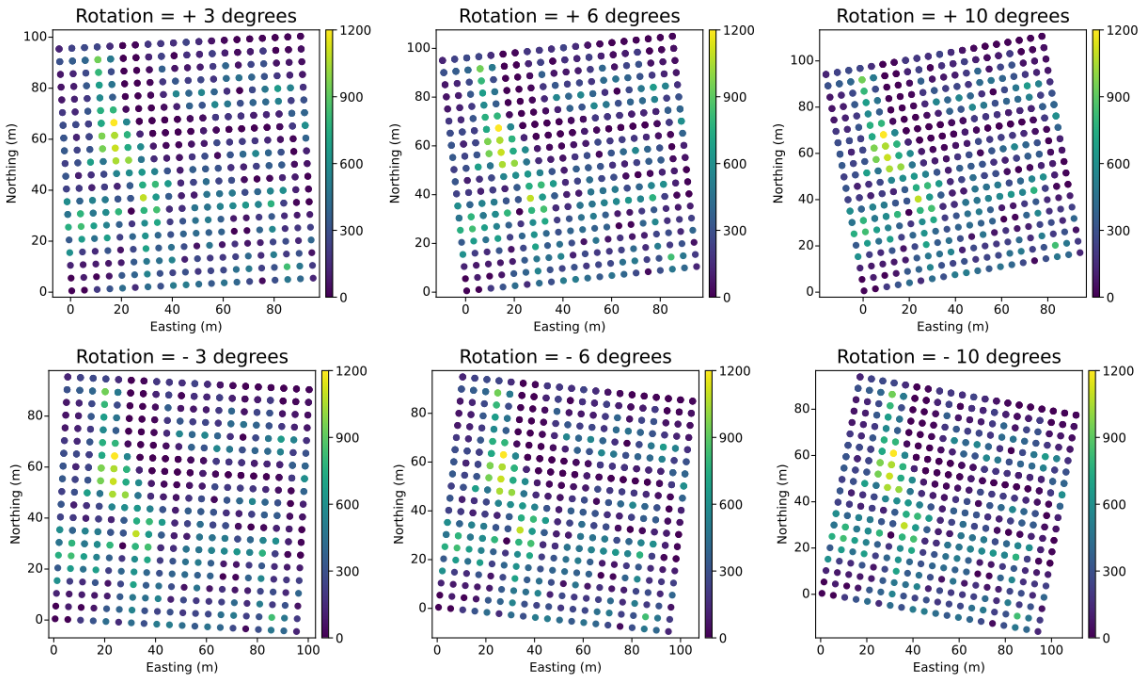


**Figure 3.4:** Data augmentation through rotations; 20 rotations applied to original datasets to increase the quantity of data and improve variogram prediction. The rotations consist of 10 clockwise and 10 counter-clockwise modifications by 1 degree each, leading to 20 generated datasets.

- Transform the original input sparse data to a 100 x 100 grid size to accommodate the input size of the trained CNNs.
- Transform the data to normal score values.
- Perform a subsurface model estimation of the transformed data using inverse distance.
- Pass the obtained subsurface model estimation through CNN-A to predict the azimuth of the major direction.
- Use the predicted azimuth in CNN-A to select one of 18 different models implemented in CNN-1, resulting in the prediction of key variogram parameters and inferring a variogram model.
- Predict the experimental variogram using CNN-2. The prediction consists of 13 experimental variogram points corresponding to 13 lag distances from 0 to 60 separated by 5 unit distance.

## 3.2 Results and Discussions

Results of the implementation of the workflows are presented. Section 3.2.1 compares the true azimuth values to the predicted values for both the training and validation sets using CNN-A. Section 3.2.2 evaluates the performance of CNN-1 in predicting the range, anisotropy ratio, and nugget effect, enabling the inference of the variogram model. Section 3.2.3 presents the results of

predicting the experimental variogram. Section 3.2.4 provides a comparison between the results obtained from workflow 1 (CNN-1) and workflow 2 (CNN-2).

### 3.2.1 Prediction of the Azimuth of the Major Direction (Performance of CNN-A)

The performance of CNN-A is evaluated in mapping sparsely sampled input data to the estimation of the azimuth of the major direction. Figure 3.5 compares the true azimuth of the training and validation sets to the predicted azimuth. 100,000 realizations are used to train CNN-A, with 80,000 realizations in the training set and 20,000 in the validation set; the model has an r-squared of 0.998 for the training set and 0.979 for the validation set. CNN-A performance is shown for four synthetically generated SGS realizations with azimuths of 30, 60, 90, and 120 degrees (Figure 3.6). The realizations are randomly sampled at 500 locations and used in CNN-A to predict the azimuth.

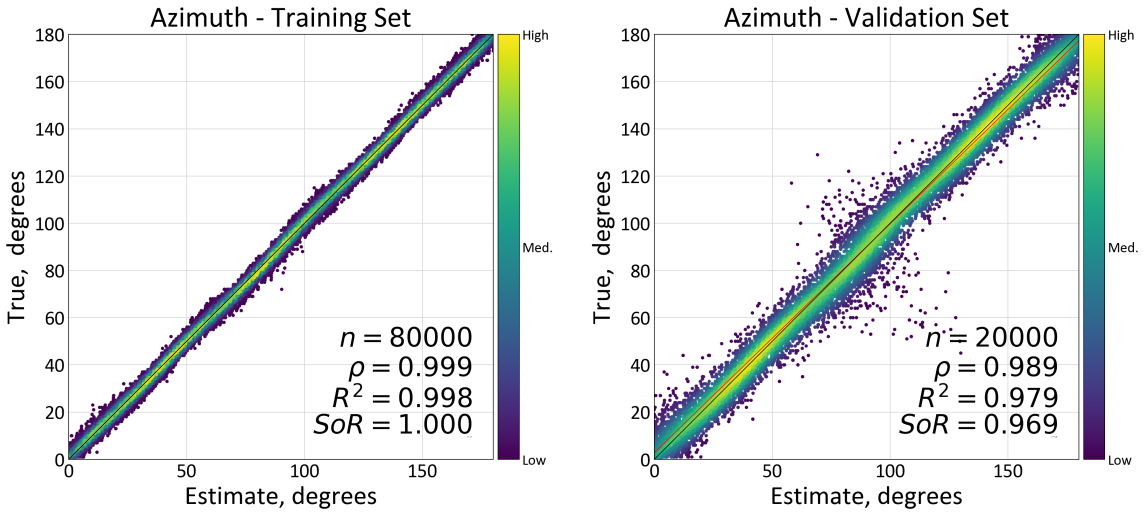


**Figure 3.5:** Scatter plots of the accuracy of the prediction of the azimuth of the major direction. The color bar is point density.

### 3.2.2 Parametric Modeling: Estimating Variogram Model Parameters (Performance of CNN-1)

Workflow 1 predicts variogram model parameters (range in the major direction, anisotropy ratio, and nugget effect), and is evaluated on both the training and validation sets. Figure 3.7 compares the true values to the estimated values of the key variogram model parameters. The r-squared of the range is 0.983 for the training set and 0.969 for validation. The r-squared of the anisotropy ratio is 0.953 for the training set and 0.945 for validation. The r-squared for the nugget effect is 0.933 for the training set and 0.867 for validation. Figure 3.8 compares the distribution of the target variogram parameters with those of the predicted values from both the training and validation data.

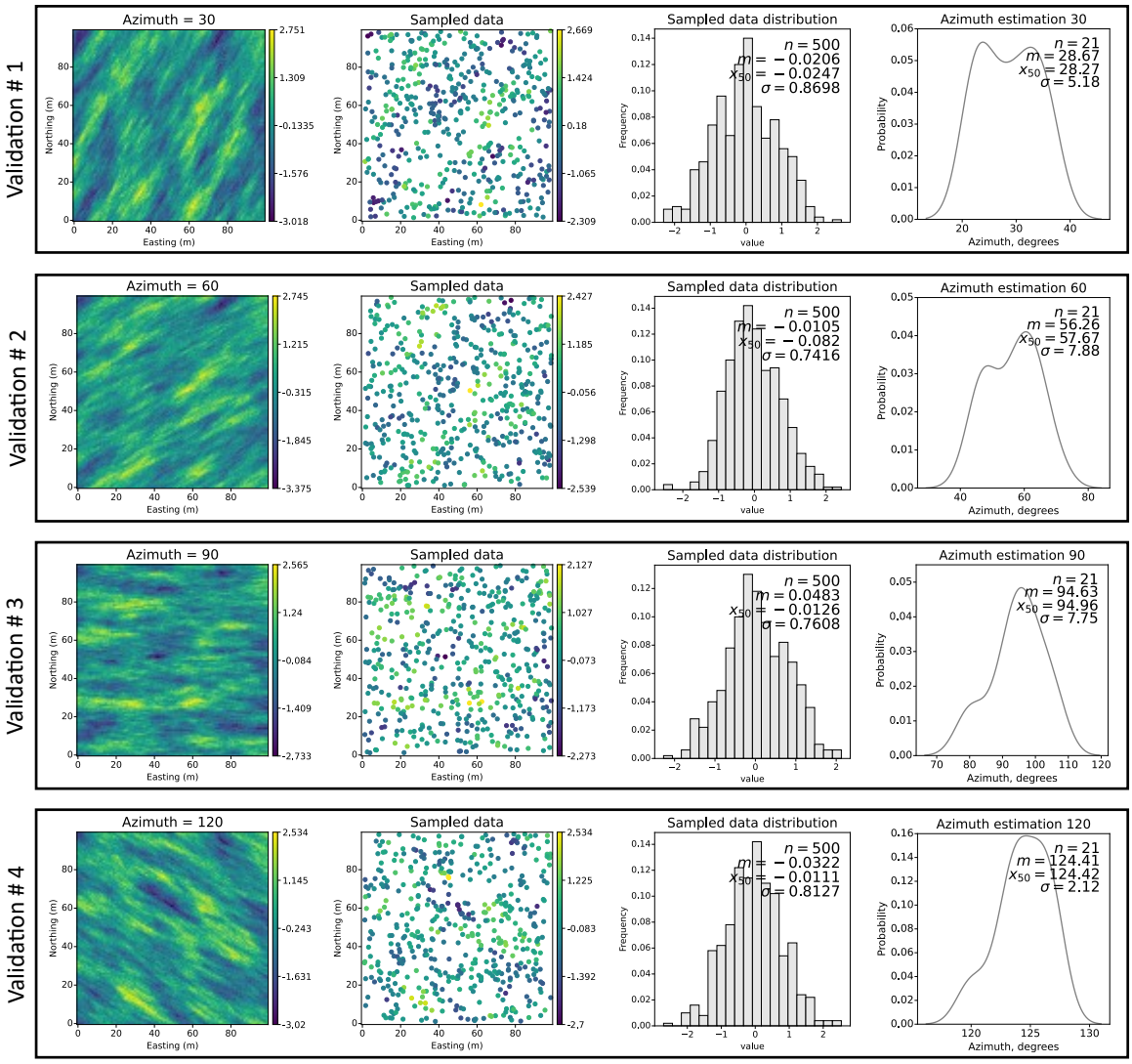


**Figure 3.6:** Visualization of CNN-A performance in predicting the azimuth of four validation datasets. The first column shows the SGS realizations and the second column displays the randomly sampled data locations. The third column visualizes the distribution of each sampled data, and the fourth and last column presents the CNN-A estimation of the azimuth along with the results of data augmentation.

The comparison reveals a strong alignment between the actual and predicted values. A variogram model is generated for the training and validation datasets based on the estimated parameters using CNN-A and CNN-1.

The performance of CNN-1, is also demonstrated on six synthetically generated SGS realizations. Six predictions are plotted in Figure 3.9. The results indicate that CNN-1 performs well for cases requiring one single spherical structure with a maximum nugget effect of 0.4 (validation 2 and 5 in Figure 3.9). However, it shows poor predictive performance for cases that require more than two structures (validation 1, 3, 4, and 6 in Figure 3.9). This limitation can be attributed to CNN-1 being trained to predict a single spherical structure with a maximum nugget effect of 0.3, leading to inaccurate variogram models when the experimental variogram requires two or more structures

or if the nugget effect exceeds 0.4 (e.g., validation 3 in Figure 3.9). These limitations motivated the development of CNN-2, which predicts the experimental variogram instead of the variogram model parameters.

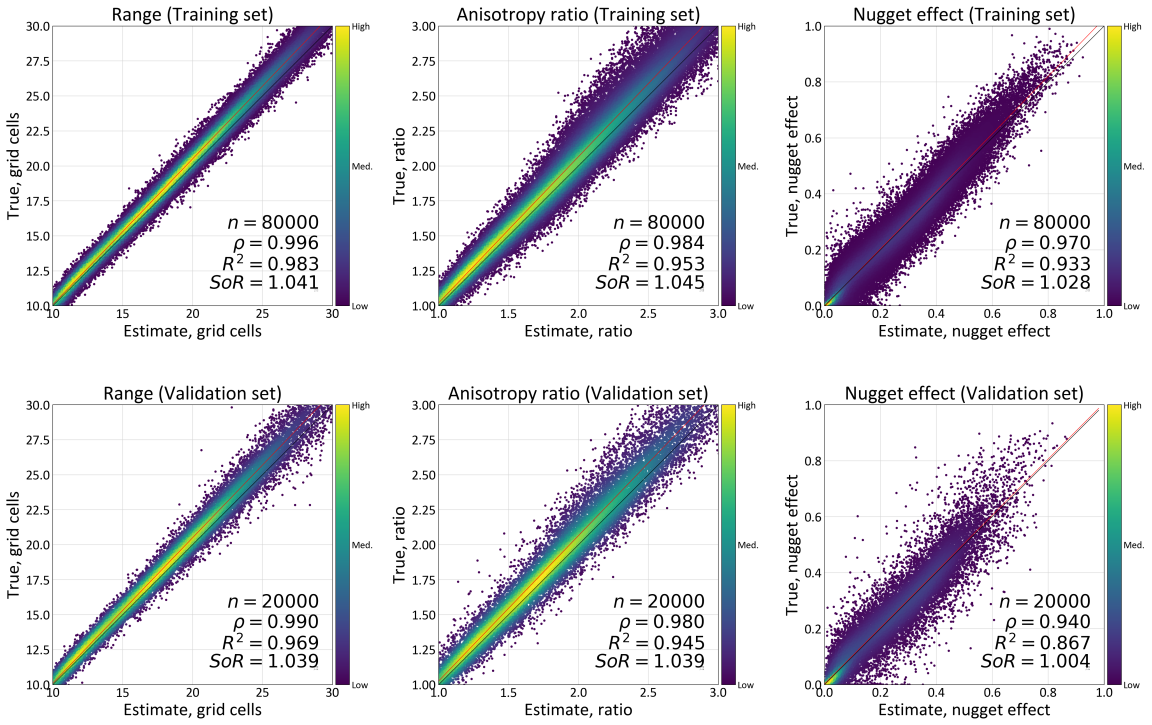


**Figure 3.7:** Comparison of estimated and true variogram model parameters. The scatter plots show the accuracy of the predictions of the model for the range in the major direction, anisotropy ratio, and nugget effect, with the training set above and validation set below.

### 3.2.3 Prediction of the Experimental Variogram (Performance of CNN-2)

Workflow 2 predicts the experimental variogram at pre-set lag distances without loss of automation as the prediction is always smooth and easily autofit using existing algorithms (Larrondo et al., 2003). Note that this is different than a typical variogram autofitting workflow as the experimental variogram is automated and does not require the user to set lag tolerances, distances, or directions. In Figure 3.10, the performance of the model is evaluated on 100,000 input data, with 80,000 data sets for training and 20,000 for validation. The results show an average r-squared value of 0.95 for the validation when comparing the predicted experimental variogram with the true experimental variogram. The performance of CNN-2 is also demonstrated on six synthetic examples (Figure 3.11). The comparison demonstrates a strong agreement between the true and predicted variograms, with an r-squared value exceeding 0.92 and a significant improvement over workflow 1 (Figure 3.9). The red-shaded area in Figure 3.11 represents the uncertainty introduced using data augmentation.

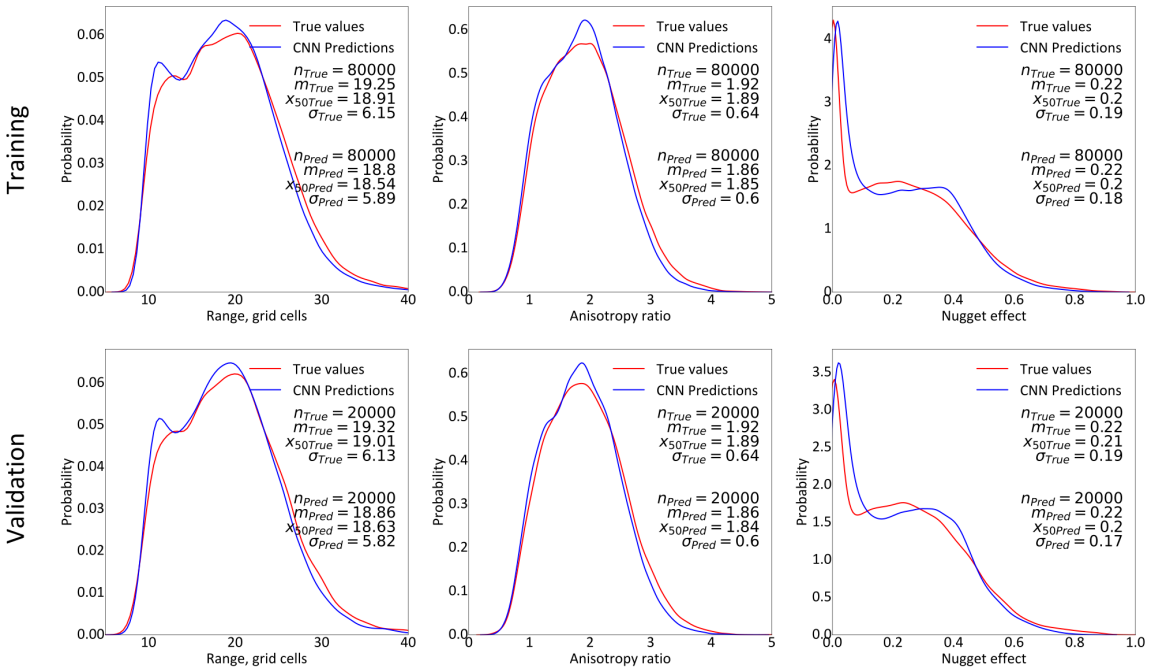


**Figure 3.8:** Probability Density Function (PDF) of actual and predicted variogram model parameters.

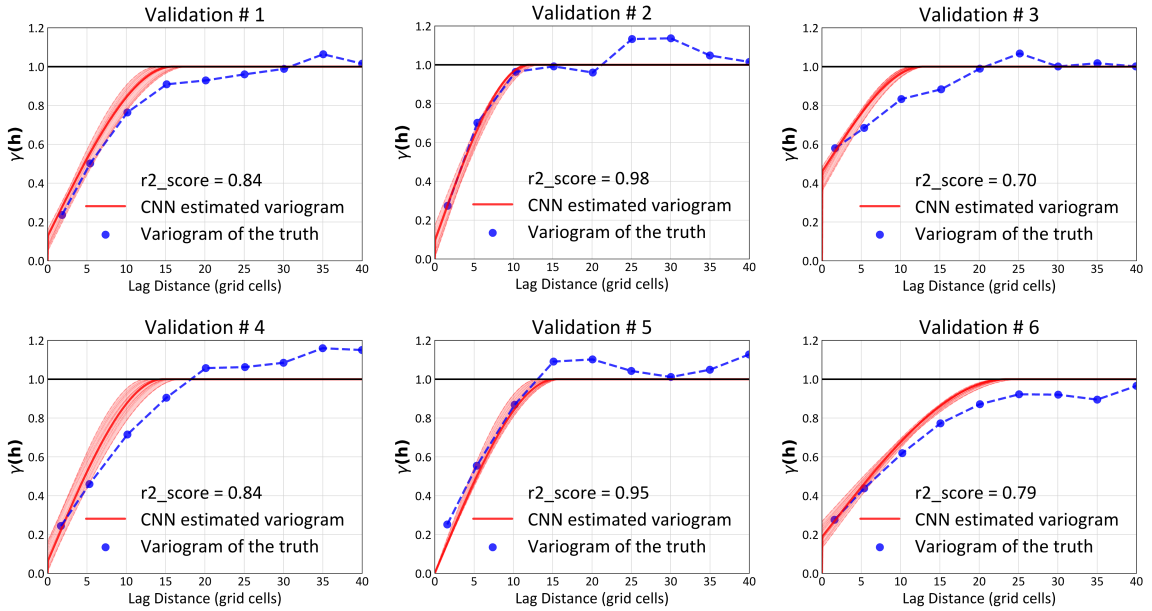


**Figure 3.9:** Demonstration of CNN-1 on six datasets. True variogram of the realizations (blue) and estimated variogram (red). Data augmentation is shown as the red-shaded zone.

### 3.2.4 Comparison Between the Results of Workflow 1 and Workflow 2

This subsection presents a comparison between workflow 1 and workflow 2. Figure 3.12 illustrates the performance of CNN-1 and CNN-2 for the six selected realizations. In four out of six cases, CNN-2 demonstrates superior performance, achieving a higher r-squared value compared to CNN-1. However, for realizations 2 and 5, both workflows perform similarly, because these variograms are

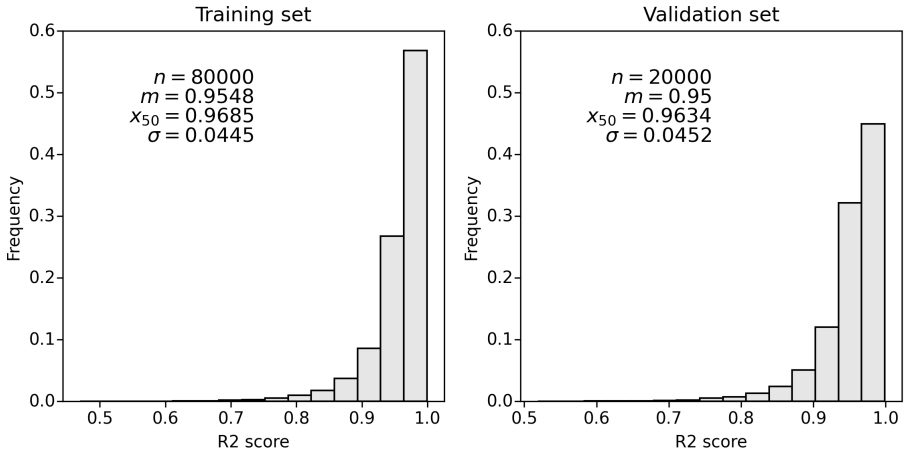


**Figure 3.10:** Histograms of the r-squared of CNN variogram using workflow 2 for training and validation datasets. The average r-squared value is 0.95.

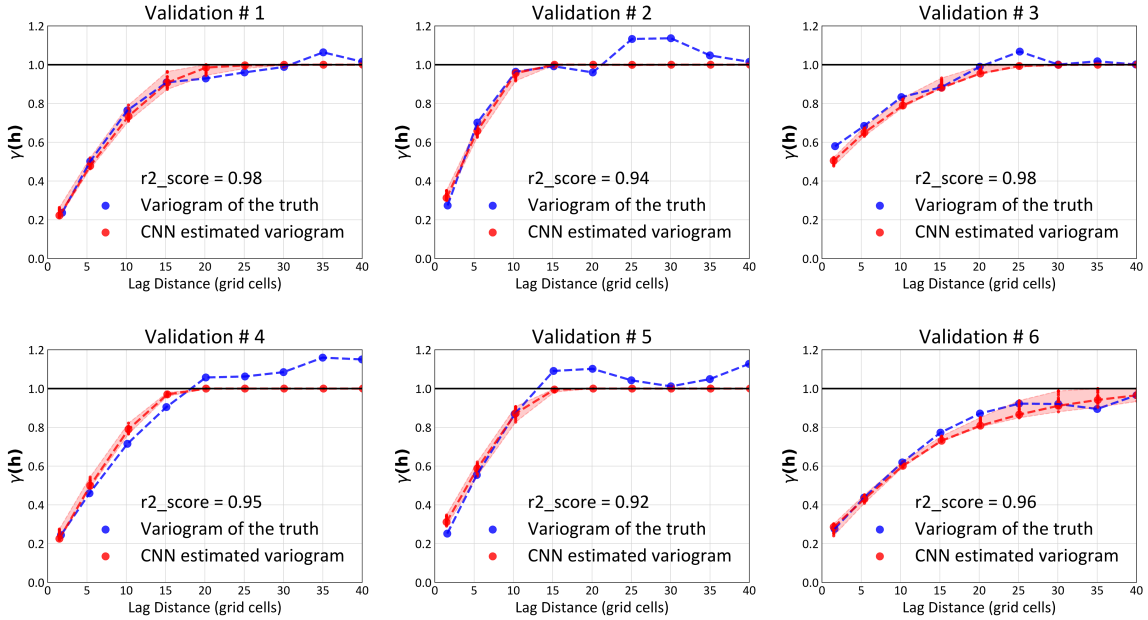


**Figure 3.11:** Demonstration of CNN-2 on six synthetic examples. True variogram of the realizations (blue) and estimated variogram (red). Data augmentation is shown as the red-shaded zone.

well-fitted by a single variogram structure (Figures 3.9 and 3.11); the improvement of CNN-2 is due to the limitations of workflow 1 for variograms that require two or more structures (Figure 3.9). Workflow 1 could be modified to consider additional variogram structures but the number of required parameters to predict would quickly grow and would require very large training sets and training multiple CNNs. Workflow 2 is preferred because it is not limited by the number of structures, performs better, and has more potential to be extended to 3D.

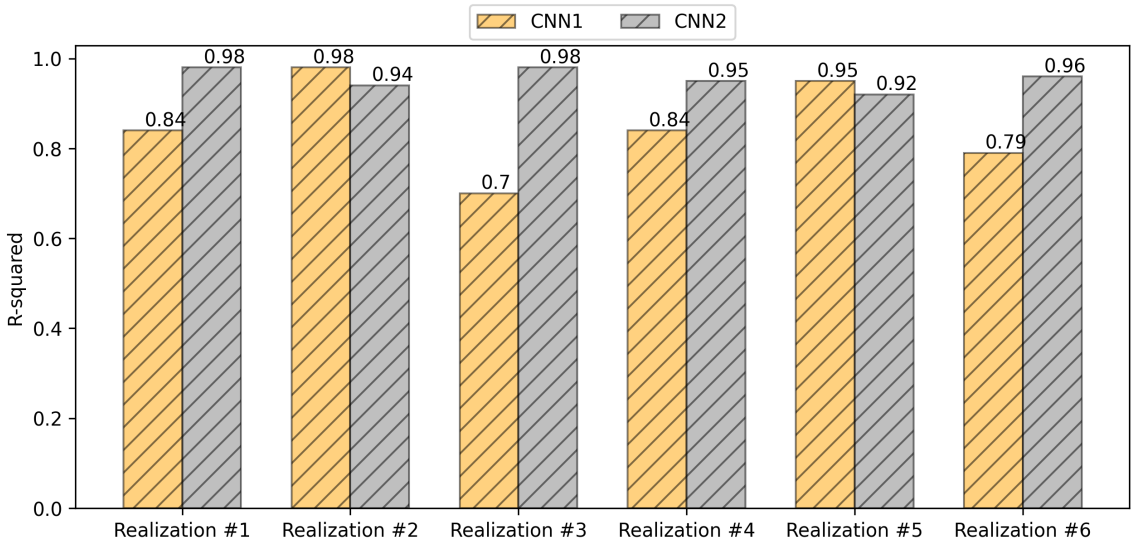


**Figure 3.12:** R-squared for CNN-1 and CNN-2 on six realizations

## 3.3 Summary

This Chapter proposes two methodologies that involve the use of CNNs to automatically calculate and model normal score variograms in the major and minor direction of continuity of a given 2-D dataset. The first methodology, workflow 1, directly predicts the key variogram model parameters, allowing for the inference of the variogram model. The second methodology, workflow 2, calculates the normal score experimental variogram at set lag distances. The proposed methodologies involve creating a training dataset through unconditional SGS realizations, training a CNN, tuning hyperparameters, conducting sensitivity analysis, and applying data augmentation that could potentially be considered as uncertainty. The performance of CNN-A in predicting the azimuth is evaluated. The results show that CNN-A can predict the azimuth of the major direction with an r-squared of 0.998 for the training set and 0.979 for the validation set. The model is further tested on four synthetic SGS realizations with varying azimuths, and the results demonstrate the ability of the model to estimate the major direction of continuity. Followed by the evaluation of the performance of CNN-1 in predicting key variogram model parameters, including range in the major and minor directions, anisotropy ratio, and nugget effect. The results show an r-squared of 0.983 for training in predicting the range, 0.953 for the anisotropy ratio, and 0.933 for the nugget effect. The model is then inferred from the predicted variogram model parameters. However, the predictive performance varies across datasets, indicating the potential limitations of the model in predicting complex variogram structures. Specifically, when dealing with variograms that require more than one spherical structure or have a high nugget effect, the accuracy of the model may be compromised. To overcome the limitations of CNN-1, CNN-2 is implemented, which predicts the experimental variogram from the data and automatically models it, allowing the possibility of fitting the experimental variogram with

more than one structure. The results show an average r-squared value of 0.95 when comparing the predicted experimental variogram with the true experimental variogram of the validation dataset. The performance of the model is also visualized by comparing the predicted experimental variogram with the true experimental variogram for six datasets, and the results show a close match between the two. The results of this chapter demonstrate the potential of CNNs in predicting azimuth, experimental variograms, and variogram models. However, the limitations of the model in predicting complex variogram structures are evident. For instance, workflow 1 only predicts a single spherical structure in 2-D. On the other hand, workflow 2 predicts smooth, robust, and easily modeled experimental variograms but is limited to cases where the nugget effect is lower than 0.3. To overcome these limitations, a more sophisticated model should be developed that includes additional training data to cover a wider range of inputs. However, this approach can be computationally expensive. In Chapter 4 the workflows are tested on real case studies to better understand the performance and the limitations, allowing to address the challenges posed by complex variogram structures and computational complexity.

# Chapter 4

# Evaluating CNN-Based Workflows for Variogram Inference: Case Studies and Performance Analysis

This Chapter presents a series of case studies that evaluate the performance of the two proposed workflows for variogram inference using CNNs. Workflow 1 predicts the key variogram model parameters which allows the inference of the variogram model and workflow 2 predicts the experimental variogram. To enhance the robustness of the predictions, data augmentation techniques are employed and could be used in uncertainty modeling. Through these case studies, the predictive accuracy and limitations of the two workflows are evaluated, considering different datasets and addressing the challenges involved in variogram inference using CNNs.

- The first case study is based on the Walker Lake dataset (Isaaks and Srivastava, 1989). Three sets of data are taken from this dataset with different sampling configurations: regular sampling, random sampling, and the originally published sampling (Isaaks and Srivastava, 1989).
- The second case study introduces the data validation data set, where the CNN automatic variogram inference algorithm is applied to measure prediction accuracy and validate the performance of the proposed workflows. This project involves 110 geological images, and the results are categorized into subsets based on their respective MSE values.

## 4.1 The Walker Lake Dataset

### 4.1.1 Data Description

The first case study uses the Walker Lake dataset (Isaaks and Srivastava, 1989) which is characterized by two variables (U and V) measured in parts per million (ppm). The dataset is exhaustively sampled at a spacing of 1m, with 78,000 points distributed over a 260 x 300 rectangular grid (Figure 4.1). This domain is sampled at 470 locations. Three sets of data are taken from the Walker Lake dataset with different sampling configurations : the first set is regularly sampled at about 5 unit spacing, the second is randomly sampled data, and the third is the originally published sampled data (Isaaks and Srivastava, 1989).

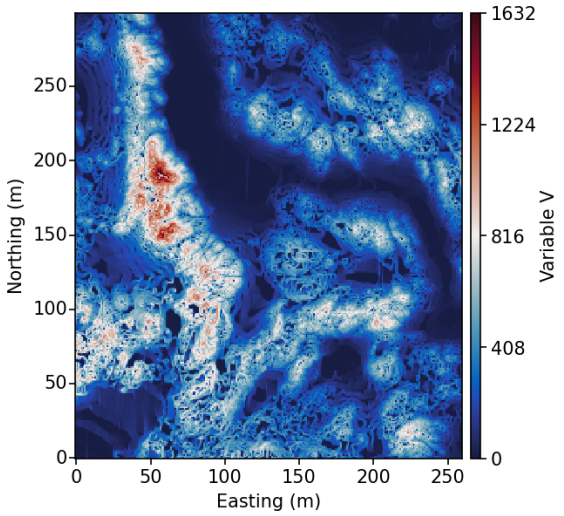


**Figure 4.1:** Location map of the exhaustive variable V

### 4.1.2 Evaluation of CNN-A on the Walker Lake Dataset

In this Section, the performance of CNN-A in predicting the azimuth of the major direction of continuity is evaluated using three different sampling configurations of the Walker Lake dataset. Figure 4.4 displays the location map and histogram of these datasets. To assess the accuracy of CNN-A, the variogram of the Walker Lake data set is calculated first using the Geostatistical software library (GSLIB) program Varcalc (Deutsch and Journel, 1998) in eight different directions, separated by 22.5 degrees (0, 22.5, 45, 67.5, 90, 112.5, 135, and 157.5 degrees), to identify the direction of major continuity. Figure 4.2 highlights that the direction of 157.5 degrees exhibits the longest continuity as indicated by its highest range. CNN-A is then implemented on the three sampling configurations, and data augmentation is applied. The results are summarized in Figure 4.3. In the case of the regularly sampled data, the CNN predicts a mean estimated azimuth of 160.35 degrees. For the randomly sampled data, CNN-A predicts 153.8 degrees, and for the original sampled data, it predicts 148.09 degrees, with the original sampling configuration exhibiting the highest uncertainty. CNN-A performs best with the regularly sampled data, as it closely aligns with the longest range calculated from the directional variograms.

### 4.1.3 Evaluation of Workflow 1 on the Walker Lake Dataset

The performance of workflow 1 is evaluated on the Walker Lake dataset (Figure 4.4). Key variogram model parameters are predicted, and a model is inferred. All predictions are carried out with normal score units. In the case of the originally sampled data, the CNN estimates a range of 25.88 units in the major direction and 16.66m in the minor direction for the variable V. The estimated variogram models for variable V in the major and minor directions are plotted and compared to the automatically fitted variogram model. Experimental variogram points are included for visual

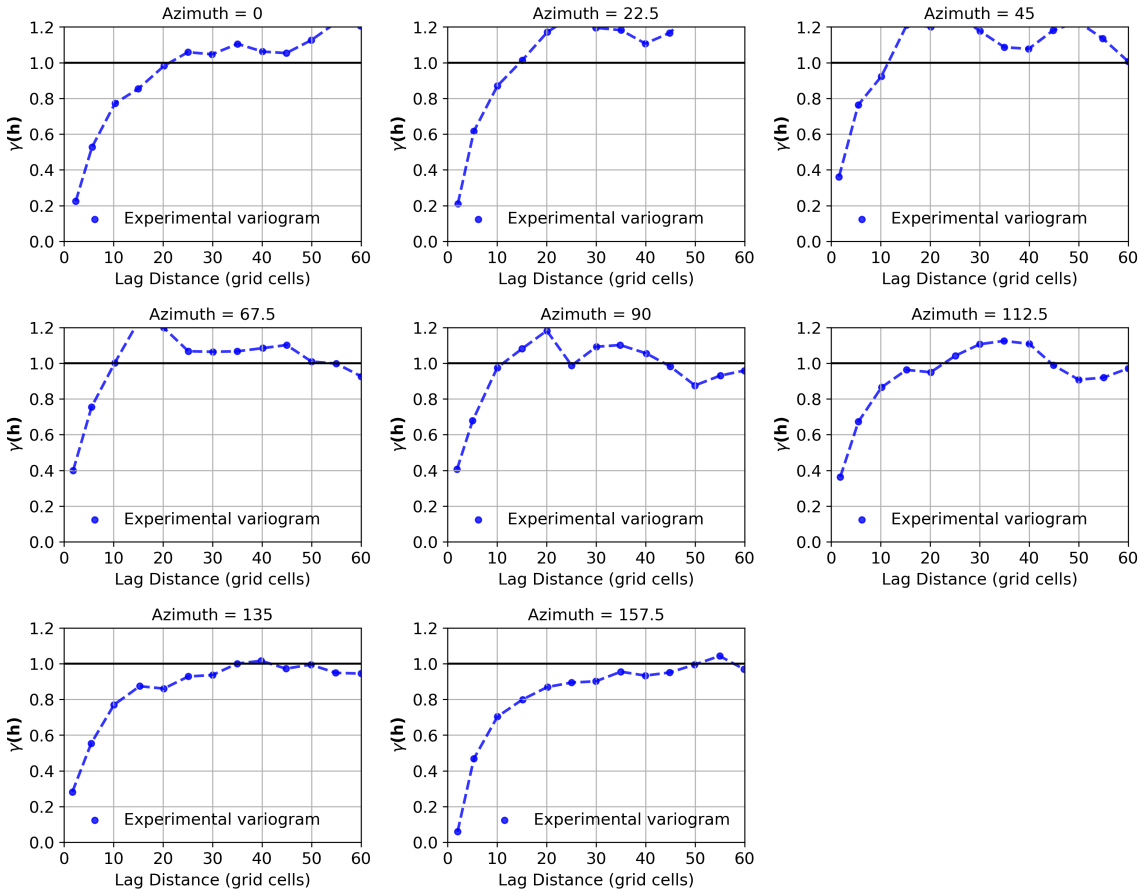


**Figure 4.2:** Experimental variogram calculation of the Walker Lake dataset. Identifying the longest direction of continuity; 157.5 degrees exhibits the longest continuity as indicated by its highest range.

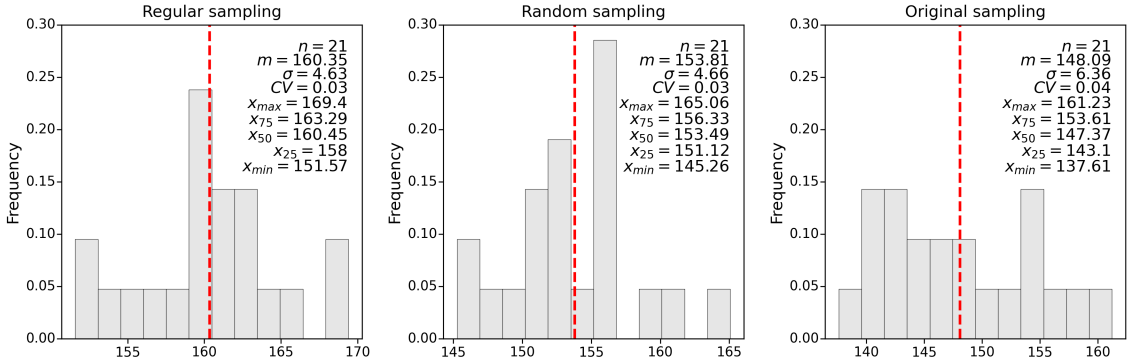


**Figure 4.3:** Comparison of CNN-A predictions of the azimuth on different sampling configurations of the Walker Lake with data augmentation. The red dashed lines show the mean predictions for each dataset configuration.

comparison. For the three sets of samples, the MSE ranges between 0.005 and 0.018. Poor predictive performance is observed in some variograms (Figure 4.4) because the training of workflow 1 is limited to one spherical structure and the variable V requires at least two structures. Rotation-based data augmentation is applied to obtain a set of variogram models (red zone in Figure 4.4). This could be

used for the examination of variogram uncertainty, but the reasonableness of this uncertainty was not assessed.

### 4.1.4 Evaluation of Workflow 2 on the Walker Lake Dataset

Workflow 2 predicts experimental variogram values at different lag distances in the major and minor directions of continuity determined by CNN-A, summarized in Figure 4.5. MSE using workflow 2 ranges between 0.0017 and 0.0045, a significant improvement over workflow 1 with MSE values between 0.005 and 0.018. The shaded red area represents prediction using data augmentation. In comparison to workflow 1, workflow 2 demonstrates a clear improvement in predicting the experimental variograms and is not limited by the number of structures.

## 4.2 Data Validation Dataset

### 4.2.1 Data Description

The dataset includes 110 2D geological images, each serving as a diverse case study (Mokdad et al., 2022). These datasets are derived from RGB geological images, converted to grayscale values (Figure 4.6), and sampled in various configurations to replicate real case studies. The images can be used as inputs for various geostatistical techniques, allowing researchers and users to benchmark their newly implemented methods since exhaustive data is provided for each dataset, which is not possible in real case studies. In this work, data validation images are transformed to normal score units in a 100 by 100 domain size to match the input size of the trained CNN. These images are then used to predict the experimental variogram and variogram model parameters through workflow 1 and workflow 2.

### 4.2.2 Steps

The following Section outlines the steps taken to predict the experimental variogram and variogram model of the 110 2D images from the data validation dataset:

- Rescale the data to a 100 by 100 domain size to match the input size of the trained CNN.
- Randomly sample the data at 400 locations, generating a set of 110 sampled datasets.
- Generate a subsurface model estimation of the sampled data using inverse distance.
- Apply a normal score transformation of the obtained subsurface model estimation to ensure compatibility with the trained CNN, as it is trained on normal score data.
- Predict the variogram model by estimating the variogram model parameters for all the transformed data obtained in step 4, using workflow 1 and based on the continuity determined by CNN-A.

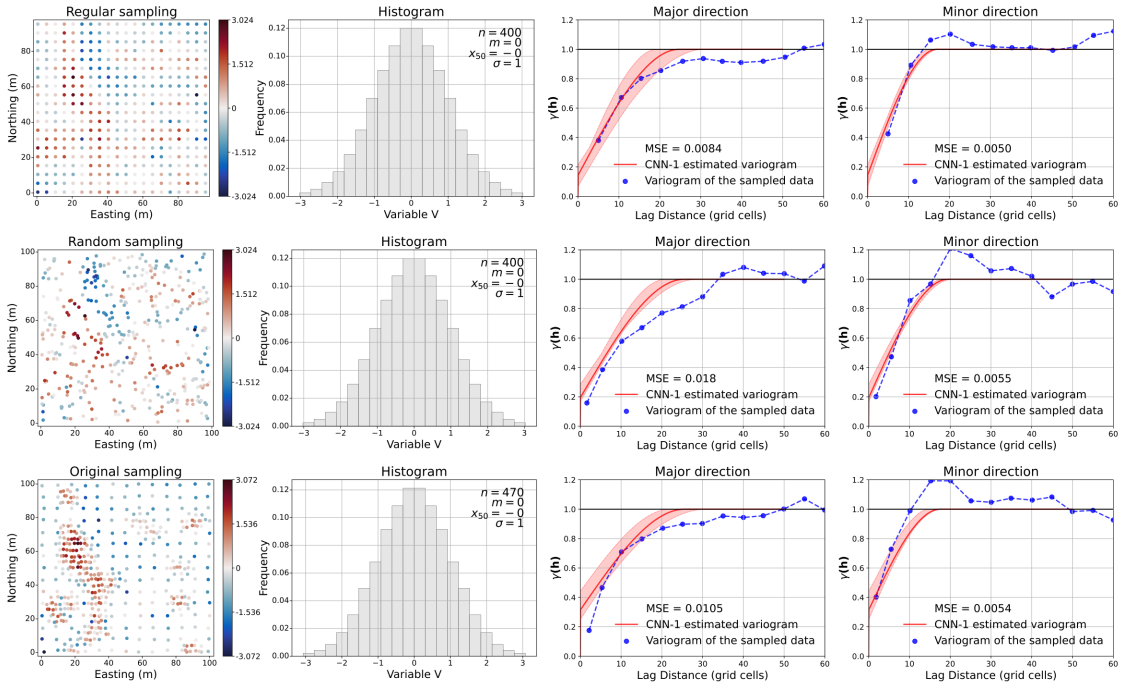


**Figure 4.4:** True and estimated variogram models (workflow 1) for variable V in the major and minor directions.

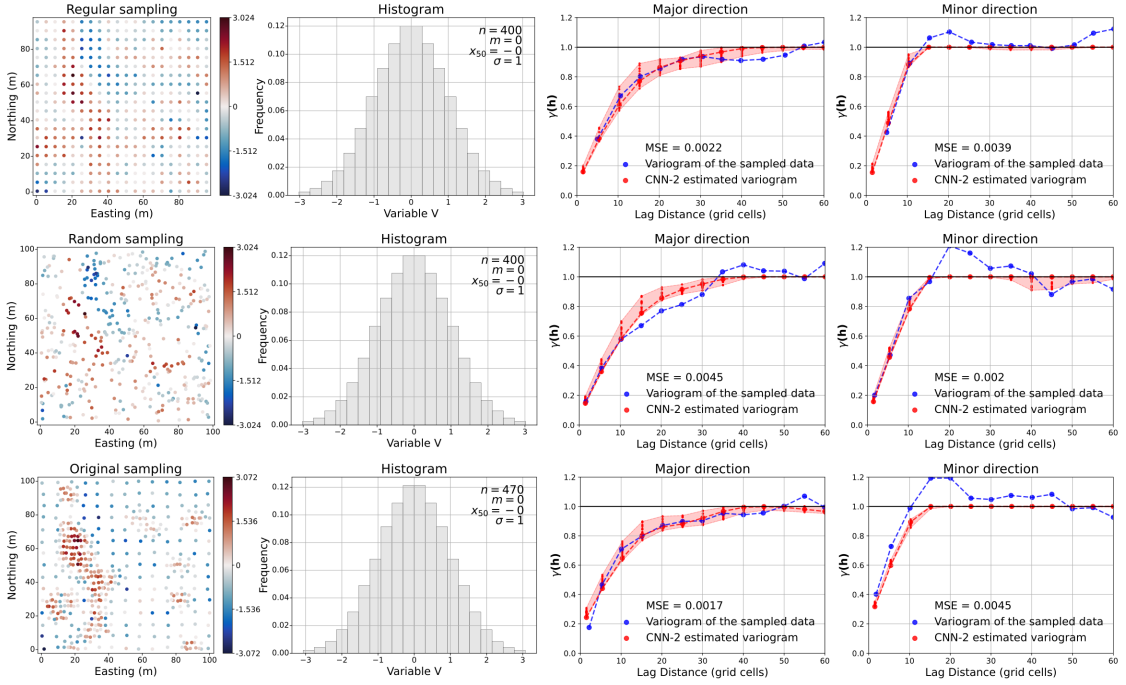


**Figure 4.5:** Predicted experimental variogram values (workflow 2) of the three selected datasets of variable V. Red points are estimated with workflow 2.

- Predict the experimental variogram for all the transformed data obtained in step 4, using workflow 2 and based on the continuity determined by CNN-A.
- Perform a comparison between the results obtained from workflow 1 and workflow 2.

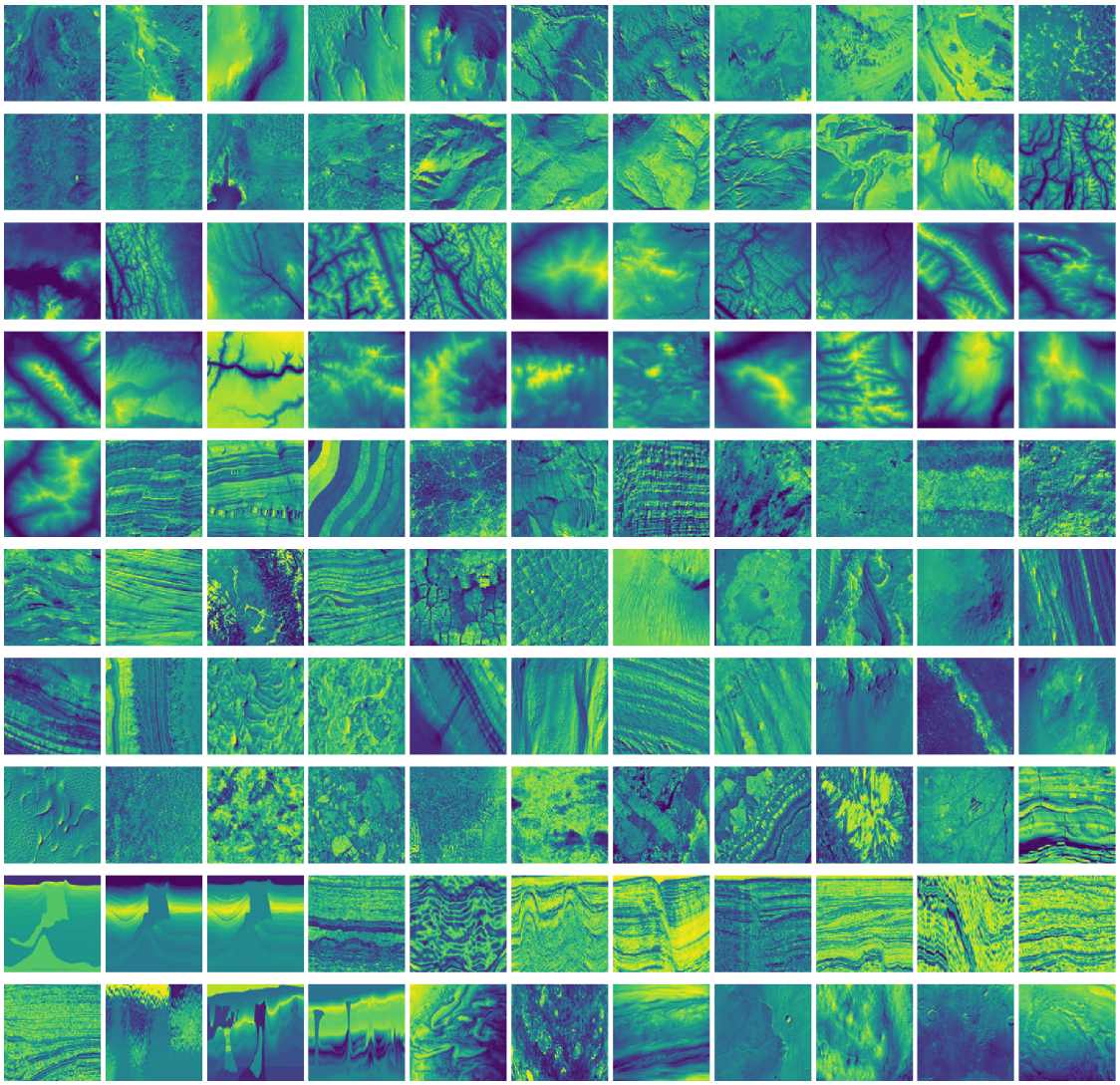

**Figure 4.6:** 2D Visualization of the images of the Data Validation Project (consisting of 110 2D geological Images)

In summary, the approach adopted in this work to predict the experimental variogram and variogram model of the 110 2D images using CNN, starts by first rescaling and sampling the data images and generating subsurface model estimations using inverse distance. These estimations are then used in two separate workflows: workflow 1 predicts the variogram model parameters, while workflow 2 predicts the experimental variogram. Finally, a comparison is performed between the results obtained from the two workflows.

### 4.2.3 Evaluation of Workflow 1 on the Data Validation Project Dataset

In this Section, workflow 1, is demonstrated on the data validation project dataset. The goal is to predict key variogram model parameters and generate an inferred variogram model based on

these predictions. To assess the accuracy of the predictions, the MSE value is calculated, which quantifies the closeness between the predicted model and the experimental true variogram model at the corresponding lag distances. Figure 4.7 illustrates the results of the proposed workflow 1 when tested on the data validation project, along with a histogram distribution of the MSE values. In Figure 4.7, the average MSE value is 0.035. The results are categorized into three subsets based on their respective MSE values: images with high accuracy (plotted in green) with an MSE lower than the 25th percentile (P25), images with medium accuracy (plotted in yellow) with an MSE between the 25th percentile (P25) and the 75th percentile (P75), and images with low accuracy (plotted in red) with an MSE higher than the 75th percentile (P75). Figure 4.8 showcases an example of variogram model prediction using workflow 1 for each category, accompanied by the corresponding MSE value. It is worth noting that cases with a high nugget effect, cases requiring more than one structure, and cases exhibiting spatial continuity for more than a third of the domain tend to result in poor predictive performance when using workflow 1.

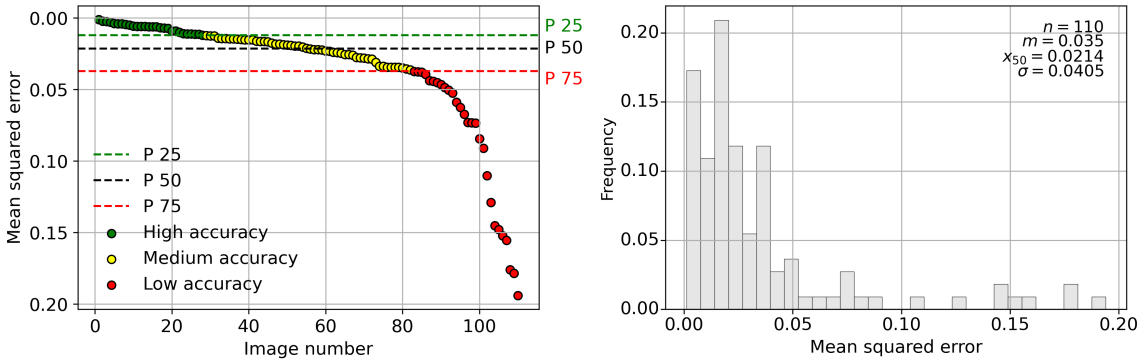


**Figure 4.7:** Results of the proposed workflow 1 tested on the data validation project which consists of 110 images collected from geological images along with a histogram distribution of the MSE values. Results are categorized into high accuracy (green), medium accuracy (yellow), and low accuracy (red) based on the MSE values.

### 4.2.4 Evaluation of Workflow 2 on the Data Validation Project Dataset

In this Section, the results obtained from applying workflow 2 to the data validation dataset in order to predict the experimental variogram are presented. Once the predictions are made, the MSE values are calculated to evaluate the accuracy of the predicted variogram compared to the true experimental variogram at the respective lag distances. Figure 4.9 provides an overview of the results of workflow 2 on the data validation project, along with the histogram distribution representing the MSE values obtained. Analyzing Figure 4.9, the average MSE value for workflow 2 is 0.0148. Similar to workflow 1, the results are classified into three subsets based on their corresponding MSE values. Some images exhibit poor performance, plotted in red. Figure 4.10 shows a representative example of the experimental variogram prediction using workflow 2 for each category, along with the associated

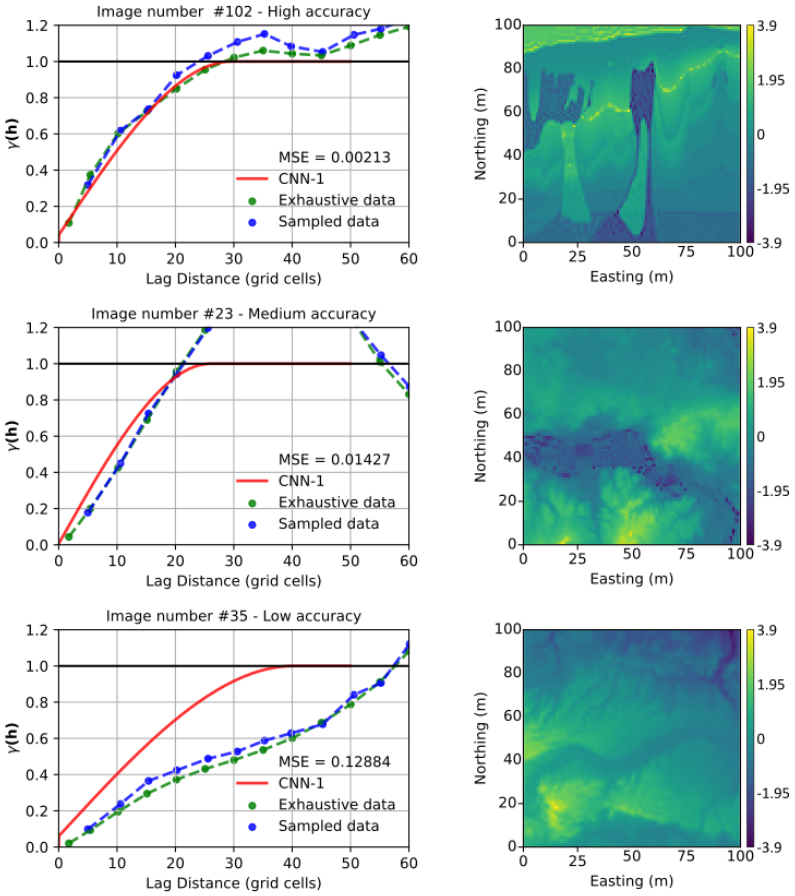


**Figure 4.8:** Example of variogram model prediction using workflow 1 for each category along with its respective MSE value.

MSE value. It is worth noting that, In the case of workflow 2, high nugget effects above 0.3 contribute to poor predictive performance.

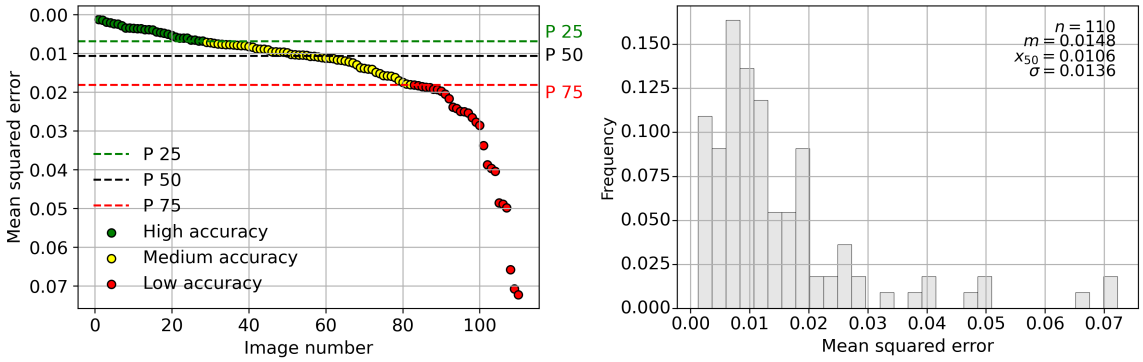


**Figure 4.9:** Results of the proposed workflow 2 tested on the data validation project which consists of 110 images collected from geological images along with a histogram distribution of the MSE values. Results are categorized into high accuracy (green), medium accuracy (yellow), and low accuracy (red) based on the MSE values.

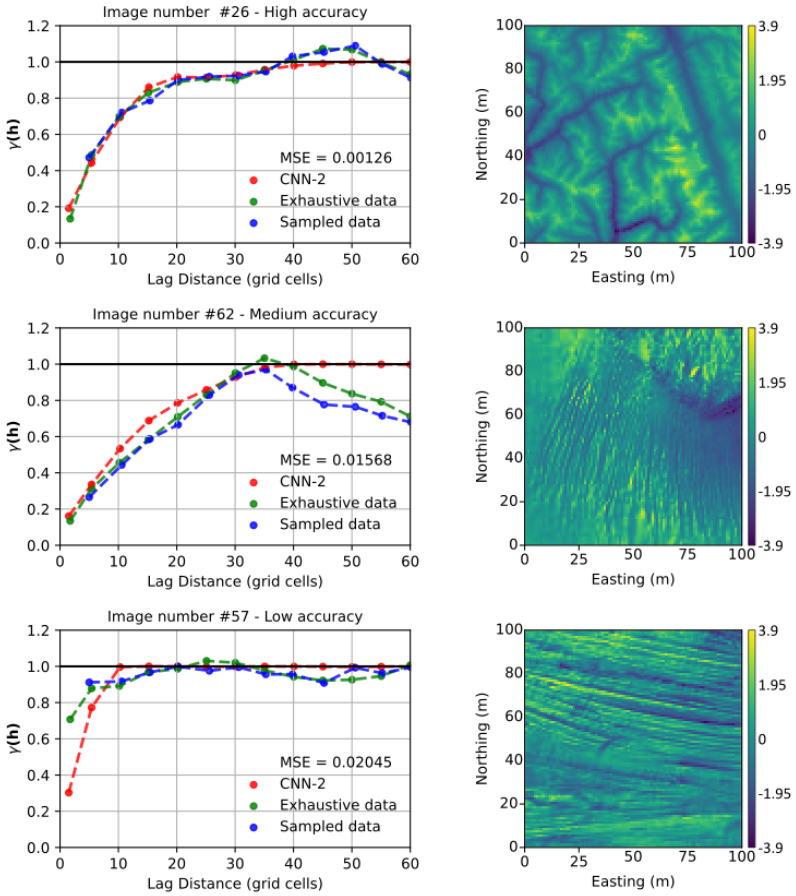


**Figure 4.10:** Example of experimental variogram prediction using workflow 2 for each category along with its respective MSE value.

### 4.2.5 Comparative Analysis of the Two Proposed Workflows

Figures 4.11 and 4.12 compare the MSE results of the two workflows, showing their respective performance in predicting the experimental variogram and the variogram model. Comparing workflow 1 and workflow 2, it is evident that workflow 2 has lower MSE values and performs better overall. While workflow 1 is limited to predicting a single spherical structure, workflow 2 is not restricted to a specific number of structures. This flexibility allows workflow 2 to achieve better predictive accuracy. In the case of workflow 1, the results obtained from the data validation project dataset show an average MSE of 0.035. On the other hand, workflow 2 demonstrates superior performance on the same data validation dataset. The average MSE value for workflow 2 is 0.015. It should be noted that both workflows exhibit poor predictive performance when dealing with case studies that have a nugget effect higher than 0.3. This suggests that high nugget effects pose challenges to accurate predictions regardless of the CNN workflow used, due to the training data not being trained to predict such cases. The models are trained on a limited maximum nugget effect of 0.3, which may not adequately capture the complexity of higher nugget effect cases. As a result, the ability of the model to generalize predictions on cases with high nugget effects becomes compromised.

In summary, workflow 2 outperforms workflow 1 with lower MSE values and greater flexibility in handling multiple structures.

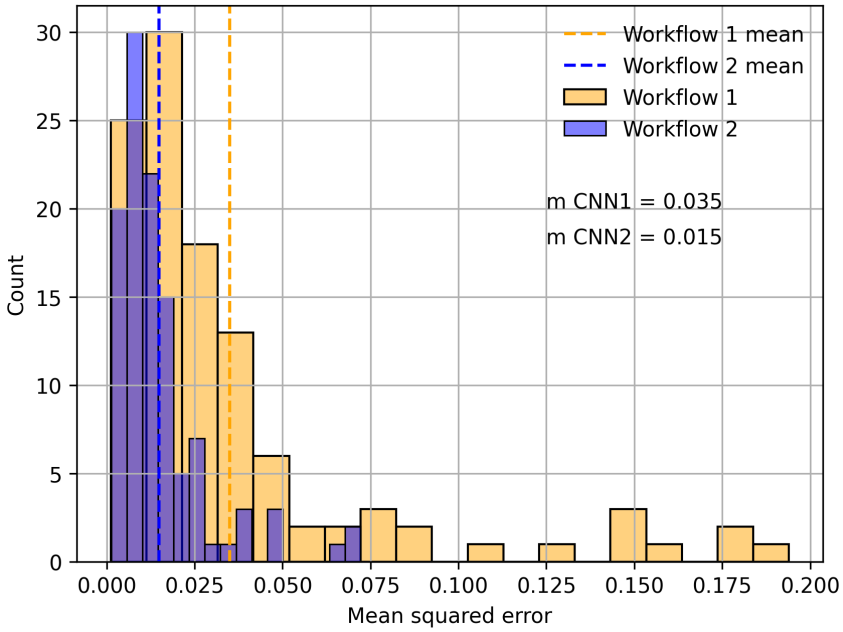


**Figure 4.11:** Comparison of MSE results for workflow 1 and workflow 2.

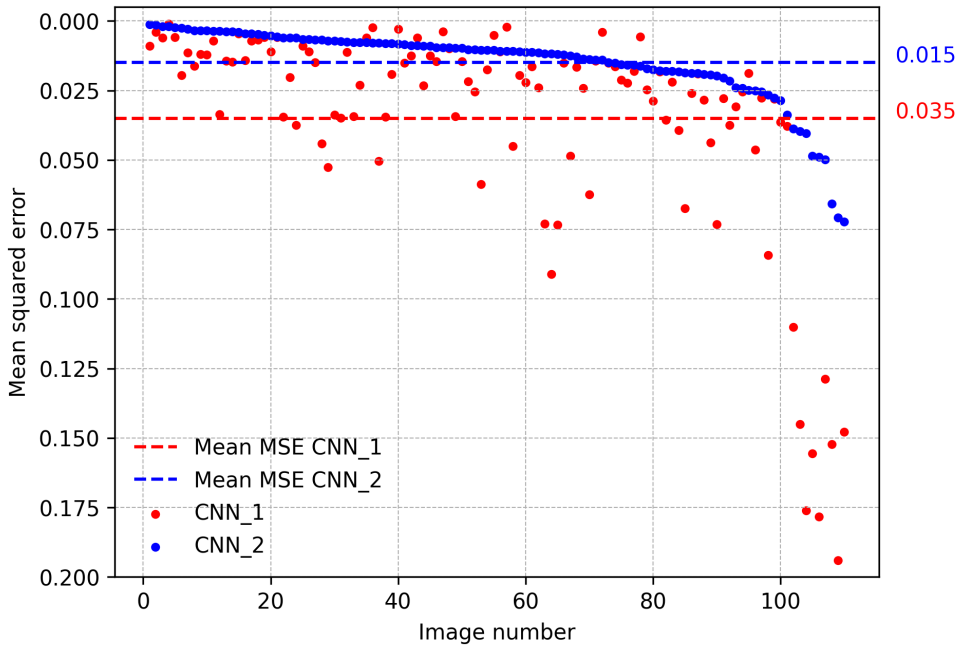


**Figure 4.12:** MSE results comparison between workflow 1 and workflow 2 for each image. Overall, workflow 2 outperforms workflow 1.

Despite CNN-2 having a lower average MSE, CNN-1 outperforms CNN-2, 10 times out of the 110 examples of the data validation project. Figure 4.12 illustrates cases where CNN-1 demonstrates

superior performance over CNN-2. Six of these cases are plotted in Figure 4.13, providing a comprehensive study of when CNN-1 performs better. Figure 4.13 displays the exhaustive variogram, the variogram predicted with CNN-1, and the variogram predicted by CNN-2, revealing that CNN-1 tends to perform similarly or better when the experimental variograms require only one spherical structure. For instance, in examples (37, 67, and 102). It is also noted that CNN-1 performs better in case studies with a nugget effect between 0.3 and 0.4, such as examples (12, 67, 95). This is because CNN-1 is based on parametric estimation and directly predicts the value of the key variogram parameters rather than experimental variogram points, which includes predicting the nugget effect. However, this improvement is not observed above the 0.4 threshold, where both CNN-1 and CNN-2 result in poor predictive performance such as in example 79. Similarly, cases, where CNN-2 performs better, are plotted in Figure 4.14. For example, numbers 4 and 26 demonstrate that when the variogram requires two or more structures, CNN-2 performs better. Another example, 101, shows that even though the MSE is slightly lower for CNN-2, CNN-1 also performs well when the spatial continuity is less than a third of the domain, has a nugget effect of less than 0.3, and when the experimental variogram requires only one structure. In such cases, both CNN-1 and CNN-2 are reliable. Additionally, examples 43 and 44 reveal that when spatial continuity is higher than a third of the domain, CNN-2 tends to perform better. Furthermore, CNN-2 is more suitable for non-spherical structures than CNN-1, as seen in examples 41 and 43.

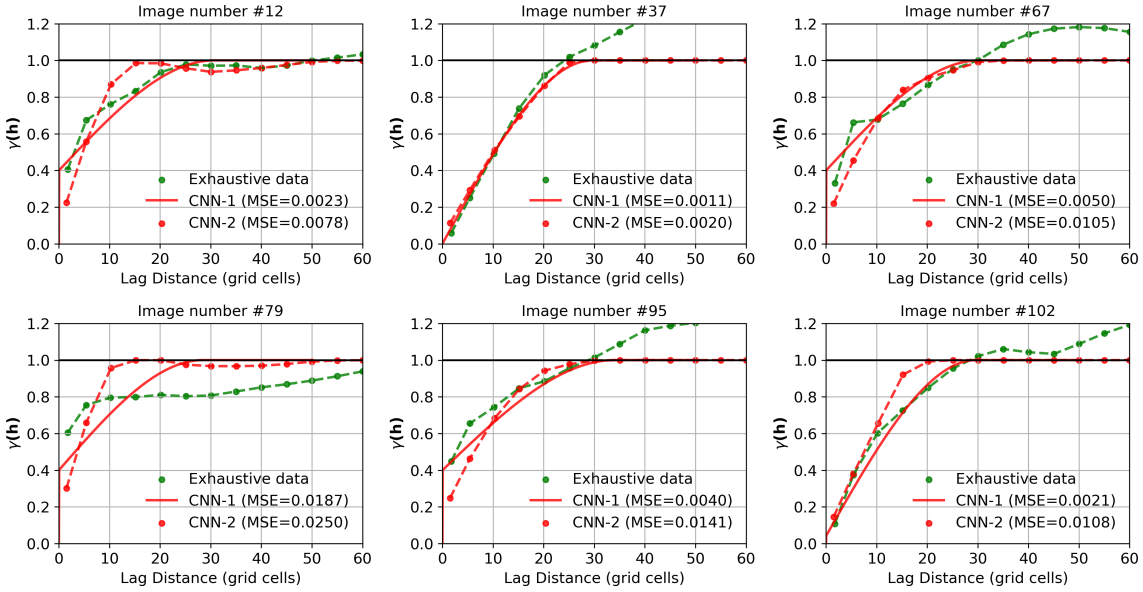


**Figure 4.13:** Comparative analysis of cases where CNN-1 outperforms CNN-2 in variogram prediction.

## 4.3 Limitations of the Proposed Workflows

This Section discusses the limitations of the proposed workflows for automatic variogram prediction using CNN. The first limitation studied is the impact of the nugget effect on the predictive perfor-

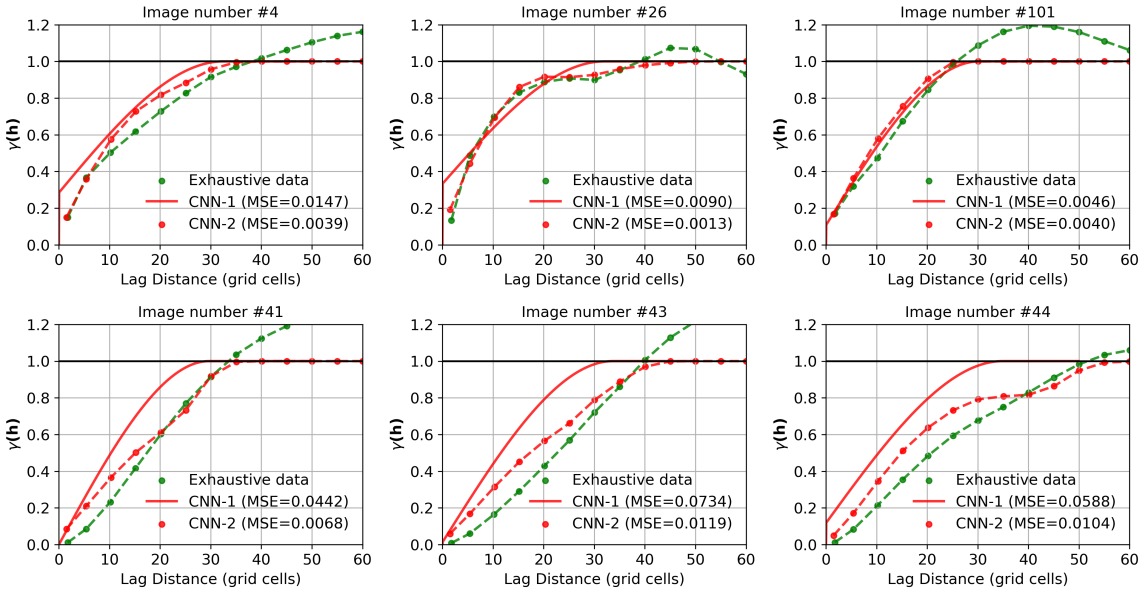


**Figure 4.14:** Comparative analysis of cases where CNN-2 outperforms CNN-1 in variogram prediction.

mance of the model. The second limitation is training the CNN on normal score data instead of using the original values.

### 4.3.1 Impact of the Nugget Effect on MSE

In Figure 4.15, the relationship between the nugget effect and MSE values is explored to better understand its impact on automatic variogram prediction using CNN-2, which is considered the preferred workflow in this research due to its lower mean MSE than CNN-1. The plot consists of green dots representing high accuracy and red dots indicating poor predictive performance. The results are categorized into four groups based on their respective MSE values. In Group A, referred to as "High accuracy" the MSE is less than the 25th percentile (P25), indicating low MSE compared to the other groups, is characterized by a low nugget effect, and serves as the benchmark in this analysis. Group B, labeled as "High nugget effect / low accuracy" shows a higher MSE above the 75th percentile (P75) and a high nugget effect, resulting in poor predictive performance. Group C, referred to as "Low nugget and low accuracy" demonstrates low accuracy despite having a low nugget effect. Lastly, Group D, labeled as "Low accuracy," has the highest MSE among the four groups.

Group A, represented by the green dots in Figure 4.15, stands as the benchmark with the highest accuracy among the four groups. It exhibits an average MSE of 0.0039, which is notably lower than the mean MSE of all data from CNN-2, which is 0.0148. Additionally, Group A is characterized by a nugget effect below 0.3. In Figure 4.16, six variogram predictions using CNN-2 are shown, demonstrating low MSE. The CNN model effectively handles case studies with nugget effects below 0.3. There are two key points to highlight, explaining why the model performs poorly when dealing

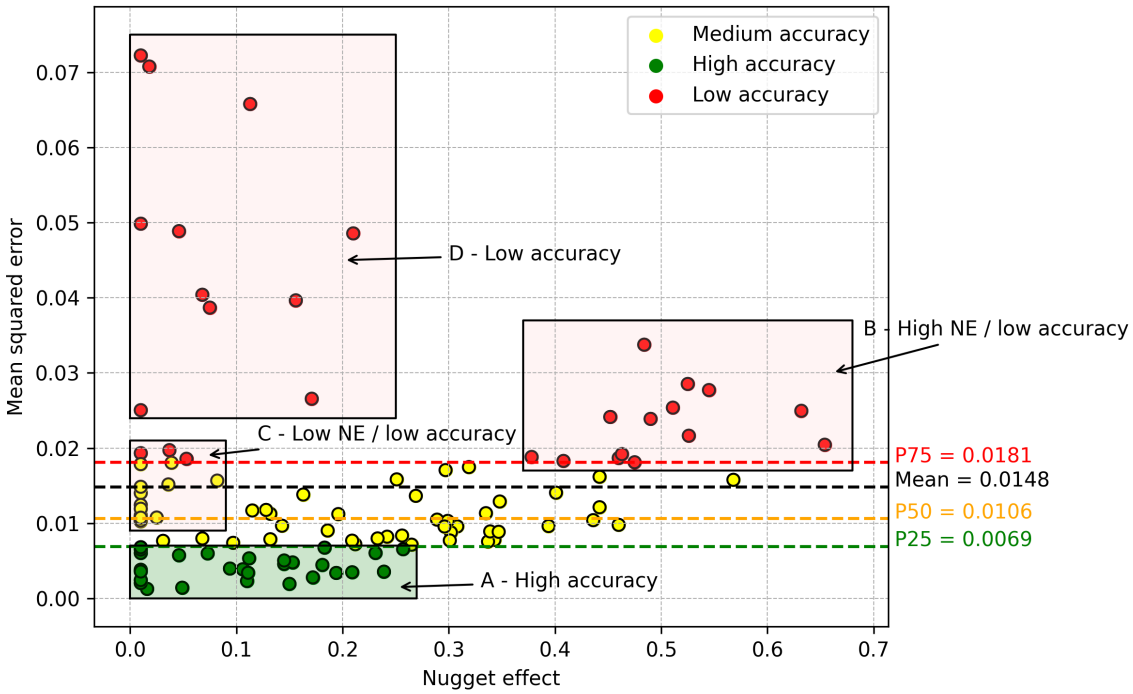


**Figure 4.15:** Exploring the impact of the nugget effect on CNN-2. The green dots represent data points with high accuracy, while the red dots indicate poor predictive performance. The results are categorized into four groups based on their respective MSE values. Group A, "High accuracy" exhibits low MSE (below P25). Group B, "High nugget effect / low accuracy" shows higher MSE and a high nugget effect. Group C, "Low nugget and low accuracy" demonstrates low accuracy despite having a low nugget effect. Lastly, Group D, "Low accuracy" displays the highest MSE among the four groups.

with examples with a high nugget effect. First, the model is trained to predict variogram ranges with nugget effects below 0.3. Secondly, the decision to select a nugget effect higher than 0.3 should be made by the geo-modeler, rather than being assigned automatically. Although occurrences are rare, it is possible to encounter deposits with a nugget effect exceeding 0.3 (Pyrcz and Deutsch, 2014). Figure 4.17, shows six selected variograms from Group B, characterized by high nugget effects above 0.3 and low accuracy, clearly illustrating that high nugget effect cases lead to poor predictive performance.

In Figure 4.18, six variograms are selected from Group C, where despite having a low nugget effect, it resulted in low accuracy. In Figure 4.18, it is apparent that despite the low nugget effect, the variograms of Group C rise steeply. The ratio between the nugget effect and the third lag distance of the variograms of the exhaustive data of Group C, compared to the same ratio for the benchmark Group A (shown in Figure 4.19). The mean of the ratio for the variograms of Group C, which rise more steeply, is three times higher than the ratio for the benchmark Group A. This suggests that overall, CNN-2 results in poor predictive performance when dealing with variograms that rise steeply, despite two examples that are well predicted in Group A with a ratio higher than 40. Lastly, Group D, referred to as "Low accuracy" shows six variograms in Figure 4.20 that exhibit poor performance. In these cases, the variograms demonstrate zonal anisotropy where they do not reach the sill (Rossi

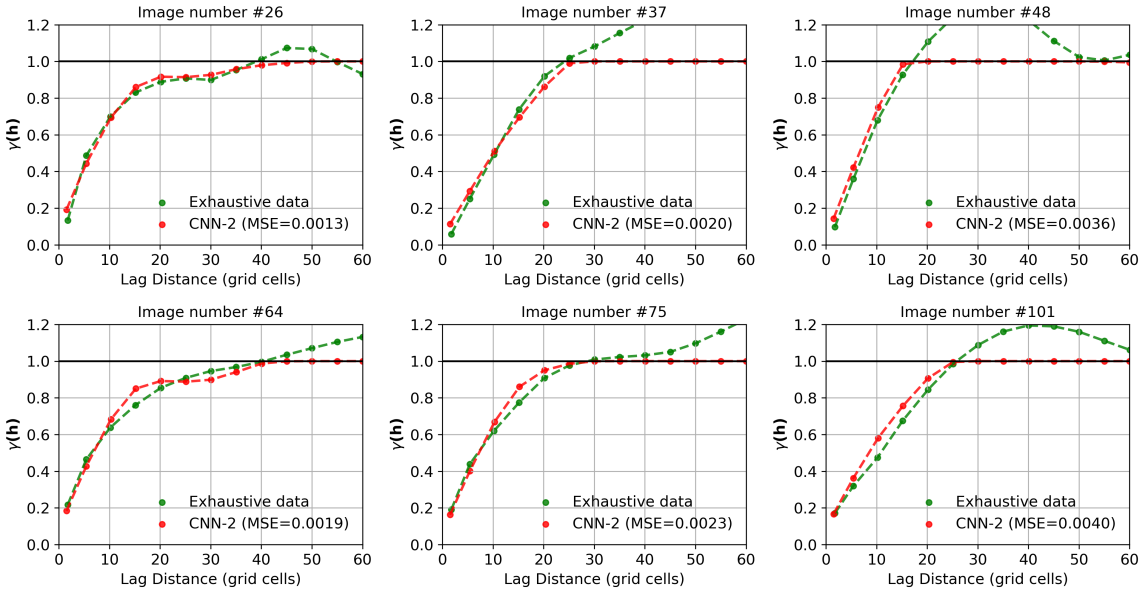


**Figure 4.16:** Variogram predictions for Group A "High accuracy" - Demonstrating high accuracy and low MSE compared to the other groups.

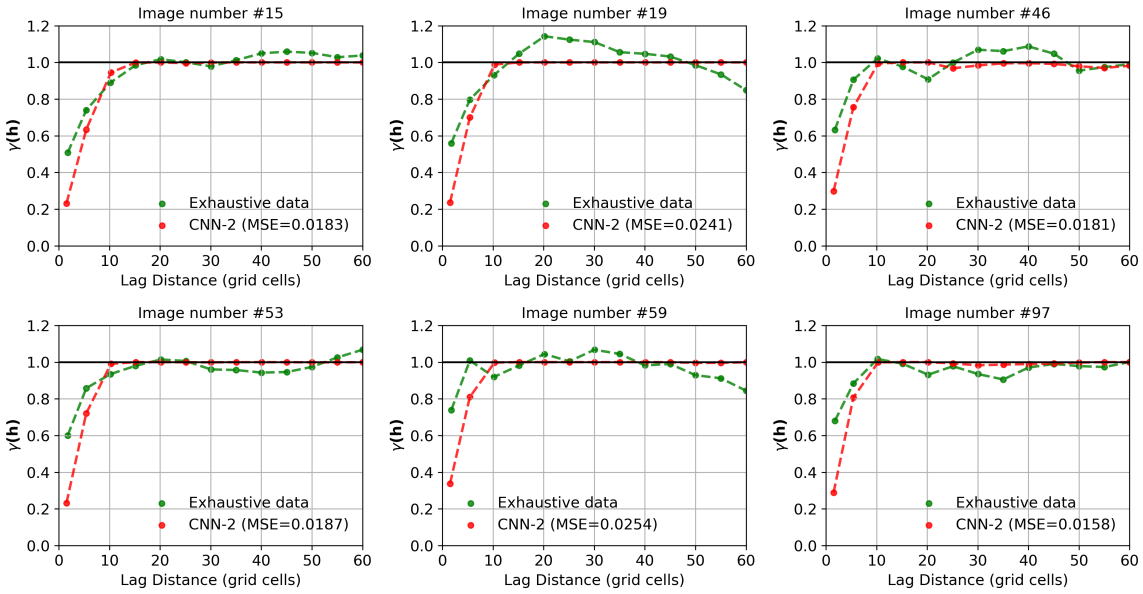


**Figure 4.17:** Variogram predictions for Group B, "High nugget effect / low accuracy"; six selected variograms from Group B, where the nugget effects exceed 0.3.

and Deutsch, 2014), as CNN-2 is not trained for cases with zonal anisotropy, leading to data shift and decreased predictive accuracy.

### 4.3.2 Prediction of Normal Score Variograms

The output of the two proposed workflows results in a normal score variogram as the training data are in normal score units. Training a CNN on normal score data, as opposed to using the original values, offers several advantages. Here are some key reasons why normal score transformation is

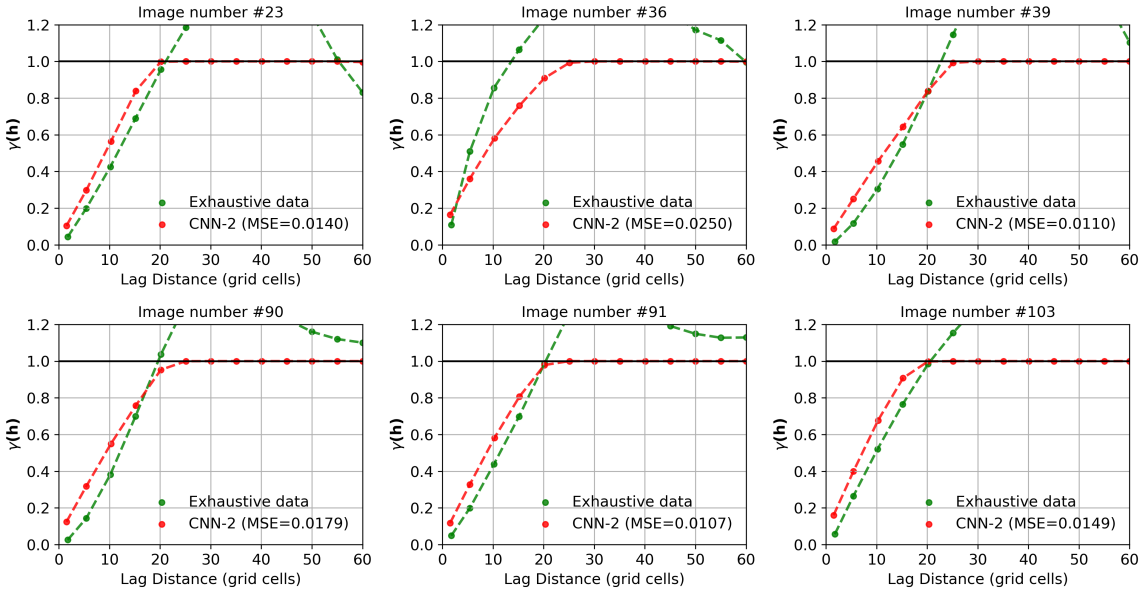


**Figure 4.18:** Variogram predictions for Group C " Low nugget and low accuracy" – Reveal a steep rise in the variogram.

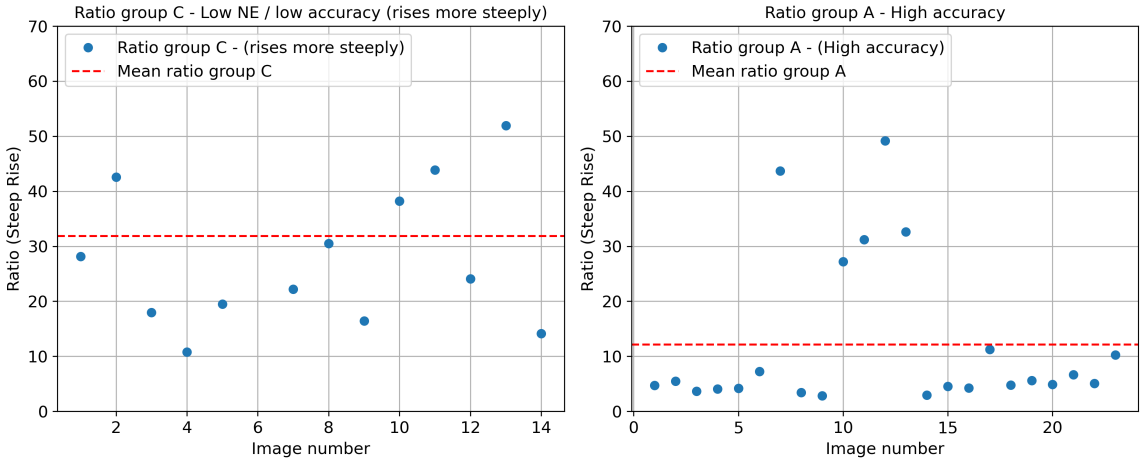


**Figure 4.19:** Ratio between the nugget effect and the third lag distance of Group C: variograms with steep rise compared to variograms from the benchmark Group A.

employed in the CNN training of this research:

- **Generalization:** By bringing the values to a similar range, the optimization algorithm can more efficiently update the network weights to handle different ranges of case studies. Neural networks are sensitive to the scale of input data (Nejedly et al., 2019). By normal score transforming the data, the network is less sensitive to variations. This makes the network more robust and generalizable.

- **Avoiding dominance of large values:** If the original values have a wide range, the larger values can dominate the learning process compared to the smaller values. Normal score transformations ensure that all values are scaled appropriately, preventing any particular value from

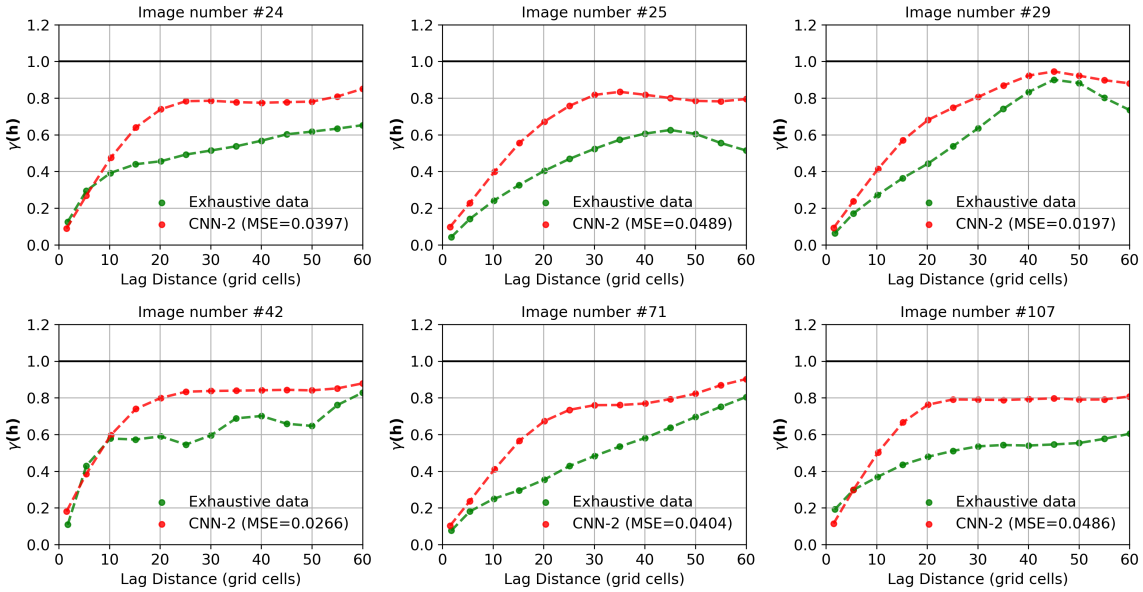


**Figure 4.20:** Variogram predictions for Group D " Low accuracy". These variograms demonstrate zonal anisotropy leading to poor predictive performance.

dominating others (Kothari and Oh, 1993).

- **Compatibility with activation functions:** Some activation functions, such as sigmoid or tanh, are sensitive to the range of input values. Normal score transformations ensure that the input to these activation functions falls within a reasonable range, allowing them to operate more effectively (Hao et al., 2020).

It is also worth noting that mining applications are increasingly adopting the normal score variogram (Pyrcz et al., 2018) for its practical use in modern geostatistical algorithms and softwares.

## 4.4 Summary

This Chapter presents an evaluation of two CNN-based workflows for variogram inference. Workflow 1 predicts key variogram model parameters, while Workflow 2 predicts experimental variograms. The case studies involve different datasets, including the Walker Lake dataset and the data validation project with 110 geological images. The case studies provide insights into the predictive accuracy and limitations of the workflows. Workflow 1 is limited to a single spherical structure, leading to some poor predictions. Workflow 2 performs better overall, with higher predictive accuracy since it is not limited by a fixed number of structures achieving lower MSE values and greater flexibility in handling variograms that requires multiple structures. CNN-1 performs well when the variogram requires only one spherical structure, while CNN-2 is more suitable for variograms with multiple structures. Both workflows are affected by variograms exhibiting zonal anisotropy and case studies characterized by a high nugget effect.

The poor predictive performance observed in the limitation section for both workflows is primarily due to datashift. These workflows demonstrate suboptimal results when tested on variograms that fall outside the training data, particularly when encountering conditions such as high nugget effect, range exceeding a third of the domain, zonal anisotropy, and non-spherical structures.

The use of normal score data in CNN training provides advantages such as better generalization, avoiding dominance of large values, and compatibility with activation functions. Overall, the case studies provide valuable insights into the performance and limitations of the proposed workflows, contributing to the understanding of variogram inference using CNNs in geostatistics.

## Chapter 5

# Hyperparameter Tuning: Optimizing CNN Performance for Variogram Inference

Chapter 5 focuses on hyperparameter tuning in CNNs and the selection of optimal models for predicting azimuth, key variogram model parameters, and experimental variogram values. The Chapter begins by providing background information on hyperparameters in CNNs and their impact on network performance. The hyperparameters discussed include learning rate, number of layers, kernel size, pooling size, activation functions, and dropout rate. The Chapter then describes the sensitivity analysis performed on CNN-A (predicting azimuth) and the selection of the optimal model based on the MSE and the presence of overfitting. Similarly, the process is repeated for CNN-1 (predicting key variogram model parameters) and CNN-2 (predicting experimental variogram values). Furthermore, the Chapter explores the selection of appropriate training and validation datasets for CNNs. The Chapter also investigates the influence of inverse distance parameters on CNN predictions, as the inverse distance technique is selected to be used as training input data. The impact of the maximum number of neighboring data to be used and the inverse distance exponent (power value) on variogram prediction accuracy using CNN are analyzed. Optimal values for these parameters are determined. Lastly, the sensitivity of CNN performance based on the size of the training and validation datasets is examined. Different sizes of training and validation data are tested.

## 5.1 Hyperparameter Tuning

In CNNs, hyperparameters are settings that are set before the training process begins and affect the performance of the network (Jin, 2022). Parameters, on the other hand, are learned during the training process (Avalos and Ortiz, 2019) and represent the weights and biases $\theta = \{W, b\}$ of the network, with $\theta$ representing the parameters, $W$ representing the weights, and $b$ representing the biases. Hyperparameters in a CNN include:

- Learning rate: Controls the step size at each iteration during the training process (Kingma and Lei Ba, 2015).
- Number of layers: Determines the depth of the network and its complexity.
- Kernel size: Specifies the dimensions of the filters used for convolution operations.

- Pooling size: Sets the dimensions of the pooling operation used for downsampling.
- Activation functions: Apply non-linear functions to the output of each neuron to help capture complex features.
- Dropout rate: Refers to the regularization parameter used in dropout, which is employed to prevent overfitting.

CNN sensitivity analysis consists of hyperparameter tuning. It involves analyzing how changes in the values of hyperparameters affect the performance of the CNN (Cardoza et al., 2022). By adjusting and evaluating different hyperparameter configurations, it is possible to select the settings that result in the best performance for a given CNN task. In this Chapter, out of the six mentioned hyperparameters, three are selected for tuning: the number of layers, the kernel size, and the regularization parameters. These parameters do not have default values, and their optimal values depend on each specific implementation case. However, the learning rate is kept fixed at 0.0001, which is the recommended default value for all available optimizers in TensorFlow (Wu and Liu, 2023). Similarly, sensitivity analysis is not performed for the pooling size, instead, the default value provided by TensorFlow is also used. The default pooling size in TensorFlow is set to 2x2 (Graham, 2014). The choice of a 2x2 pooling size also aligns with common practice in many CNN architectures (Abdel-Hamid et al., 2014; Avalos and Ortiz, 2020; Jo et al., 2021; Jo and Pyrcz, 2022; Santos et al., 2021) because it provides a balanced approach between reducing spatial dimensions and preserving important spatial feature information. While alternative pooling sizes like 3x3 or 4x4 can be used, they would capture fewer spatial features since pooling layers are used to downsample input feature maps, thereby reducing their size for better computational efficiency. A larger pooling size would result in a smaller output.

Regarding the selection of activation functions, the specific choices made in this research are based on the characteristics of the target output values of each of the three proposed CNNs. For CNN-A, which predicts the azimuth of the major direction, TanH function is selected. This choice is motivated by the fact that, during the azimuth prediction process, the CNN model needs to estimate the cosine and sine of the azimuth angle. The TanH activation function outputs values between -1 and 1, which is suitable for this purpose. In the case of CNN-1, which predicts the key variogram model parameters, ReLU activation function is selected. The variogram range in the generalized domain of 100 by 100 falls within the range of 10 to 30. As ReLU activation has no upper limit, it is appropriate for this task. For CNN-2, which predicts the experimental variogram values at different lag distances, the sigmoid activation function is selected. This choice is made because the sigmoid function outputs values between 0 and 1, which aligns well with the standardized experimental variogram values.

### 5.1.1 Hyperparameter Tuning and Model Selection for CNN-A: Predicting the Azimuth

This Section aims to assess the performance of the proposed CNN model in predicting the azimuth of the major direction, referred to as CNN-A, by varying the values of the hyperparameters. The hyperparameters selected for investigation in this study include kernel size, number of convolutional layers, and dropout rate. As there are no default values for these parameters and the selection of these values is not fixed and can vary depending on the specific case or application under investigation, the range of values is determined based on common practices reported in the literature and diverse previous implementations of CNNs in relevant studies (Avalos and Ortiz, 2020; Cardoza et al., 2022; Jo et al., 2021; Jo and Pyrcz, 2022; Santos et al., 2021). The specific ranges of values chosen for each parameter are summarized in Table 5.1.

**Table 5.1:** Range of hyperparameter values investigated for CNN-A

| Hyperparameters | | |
|---|---|---|
| Number of convolutional layers | Kernel size | Dropout rate |
| 2 layers | 2x2 | 0 |
| 4 layers | 3x3 | 0.25 |
| 6 layers | 4x4 | 0.5 |
| - | 5x5 | - |

By selecting hyperparameters within their respective ranges, 36 different configurations of the model are generated. These combinations are tested on 100,000 datasets, with 80,000 used for training and 20,000 for validation. The number of epochs is set to 100. The criteria for selecting the best-performing model in predicting azimuth using CNN-A consists of evaluating the MSE on both the training and validation data, as well as assessing overfitting, which measures the difference between the errors from the training and validation datasets.

The 36 proposed CNN models are implemented, and the results are presented in Figure 5.1. This Figure is divided into three main Sections, each corresponding to 12 models with a specific number of convolutional layers. Models 1 to 12 represent the results of models implemented with 2 convolutional layers, models 13 to 24 with 4 convolutional layers, and models 25 to 36 with 6 convolutional layers. Each Section further divides the results into four Subsections based on the kernel size used: models 1 to 3 with a 2x2 kernel, models 4 to 6 with a 3x3 kernel, models 7 to 9 with a 4x4 kernel, and models 10 to 12 with a 5x5 kernel. Additionally, within each kernel size, the three models differ in their dropout values. The first model has a dropout value of 0, the second model has a dropout of 0.25, and the third model has a dropout of 0.5.

In Figure 5.1, it is observed that the model with the lowest MSE for the validation data (plotted in blue) is model 36, with an MSE of 0.029. Similarly, model 36 also achieves the lowest MSE for the training data, with an MSE of 0.004. However, this model exhibits significant overfitting and cannot be used reliably. Therefore, the selected optimized model is model 14, which has 4 convolutional

layers, a 2x2 kernel size, and a dropout value of 0.25. This model demonstrates the lowest MSE among all the models with minimum overfitting, with MSE values of 0.0318 for validation and 0.0314 for training. It is important to note that a CNN model with the lowest MSE achieved through hyperparameter tuning may result in overfitting, leading to poor generalization on unseen data. In contrast, a model with higher MSE but less or no overfitting exhibits better generalization capabilities, making it more reliable for predictions on new data.

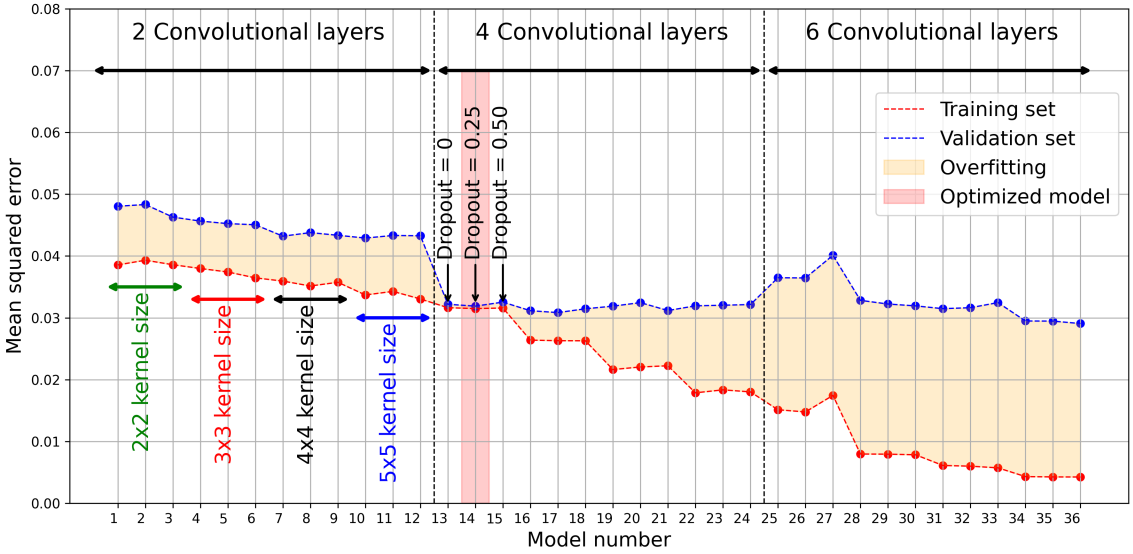


**Figure 5.1:** Performance of CNN-A with varying hyperparameters for predicting azimuth. The Figure is divided into three Sections based on the number of convolutional layers, each containing four Subsections representing different kernel sizes. The optimal model (model 14) offers a balance between accuracy and minimum overfitting.

In the conducted sensitivity analysis of hyperparameters for the three CNN architectures, the impact of varying the number of convolutional layers on model performance is studied. Specifically, 2, 4, and 6 convolutional layers are conducted, and in the case of CNN-1 and CNN-2, the implementation of 8 convolutional layers is added. During the experimentation with CNN-A, it became evident that increasing the number of convolutional layers from 4 to 6 led to a significant increase in overfitting. This is determined by evaluating the difference in the MSE between the training and validation datasets. To visualize this relationship, Figure 5.2 illustrates the variation in MSE difference (overfitting) across different layer configurations. Figure 5.2 demonstrates that augmenting the number of layers beyond 4 in CNN-A resulted in a notable rise in overfitting. As a result, given the increase observed between 4 and 6 layers, it is deemed unnecessary to extend the analysis to include 8 layers for CNN-A. Thus, the hyperparameter tuning process for CNN-A concluded at the maximum of 6-layer configuration. However, for CNN-1 and CNN-2, an extended investigation involving 8 convolutional layers is conducted, as there is no significant increase in overfitting observed between 4 is 6 layers.

Among the 36 trained models with different parameter combinations, Figure 5.3 shows the three

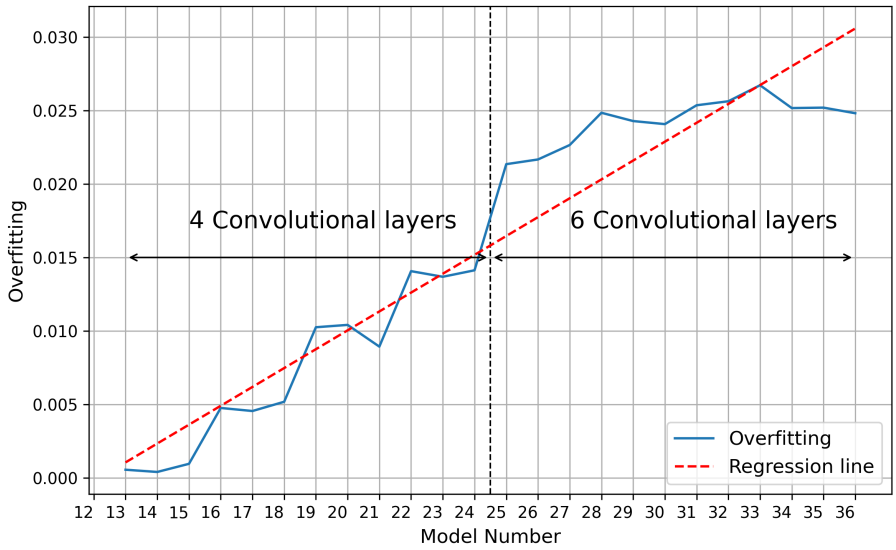


**Figure 5.2:** Impact of the number of convolutional layers of CNN-A on overfitting; assessing overfitting calculated based on the difference between the MSE of the training and the validation.

best-performing models that result in minimal overfitting: model 13 (4 convolutional layers, a 2x2 kernel size, and a dropout value of 0), model 14 (4 convolutional layers, a 2x2 kernel size, and a dropout value of 0.25), and model 15 (4 convolutional layers, a 2x2 kernel size, and a dropout value of 0.5). Out of the three models, model 14 is considered the optimal model. It achieves the lowest MSE values for both the training and validation datasets, with an MSE of 0.0314 for training and 0.0318 for validation. The key criteria for selecting the optimized model is the absence of overfitting, as this ensures the ability of the model to generalize effectively.

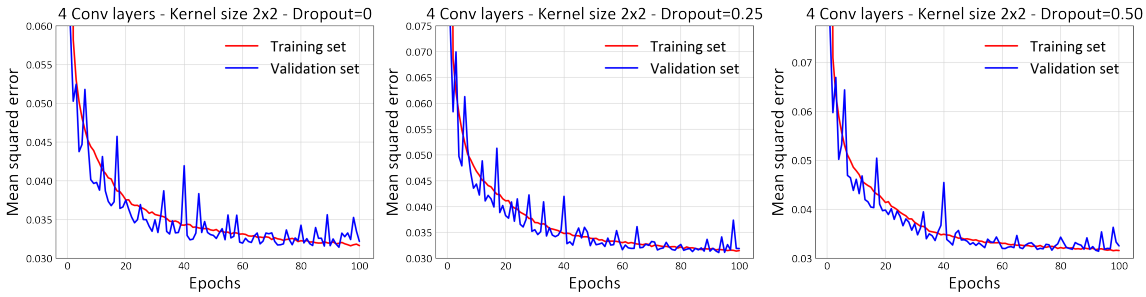


**Figure 5.3:** Comparison of best-performing models of CNN-A with minimal overfitting.

Figure 5.4 demonstrates the three worst-performing models in terms of overfitting: model 34 (6 convolutional layers, a 5x5 kernel size, and a dropout value of 0), model 35 (6 convolutional layers, a 5x5 kernel size, and a dropout value of 0.25), and model 36 (6 convolutional layers, a 5x5 kernel size, and a dropout value of 0.5). These models exhibit evident overfitting. Although their training MSE are lower than that of the chosen optimized model (model 14), it is crucial to prioritize generalization capabilities and avoid overfitting issues. Overfitting occurs when a model excessively fits the training

data, leading to poor performance on unseen validation data. Thus, despite their lower training MSE values, these models are not selected as the optimal choice due to their pronounced overfitting behavior.

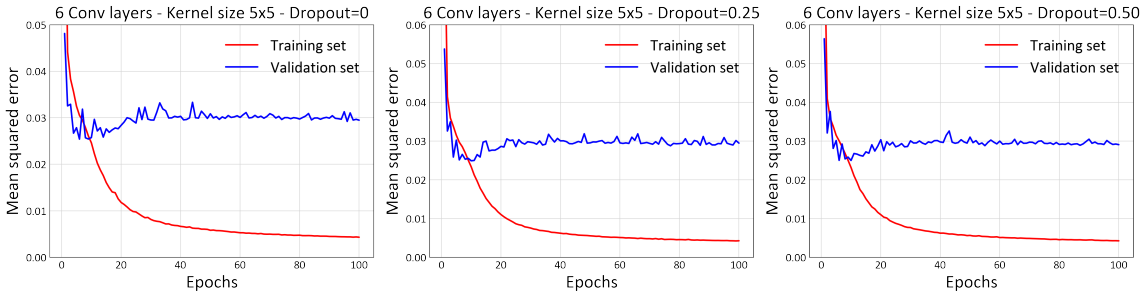


**Figure 5.4:** Comparison of worst-performing models of CNN-A with overfitting.

### 5.1.2 Hyperparameter Tuning and Model Selection for CNN-1: Predicting Key Variogram Model Parameters

Similar to CNN-A, the impact of different hyperparameters is investigated on the model performance of CNN-1 and CNN-2. The only difference is that CNN-1 and CNN-2 are also tested with 8 convolutional layers, as there is no apparent overfitting between using 4 and 6 layers. Consequently, this led to the generation of 48 models instead of 36 for CNN-A, as the addition of another value in the number of layers increased the number of combinations. The specific ranges of values chosen for each parameter are summarized in Table 5.2.

**Table 5.2:** Range of hyperparameter values investigated for CNN-1.

| Hyperparameters | | |
|---|---|---|
| Number of convolutional layers | Kernel size | Dropout rate |
| 2 layers | 2x2 | 0 |
| 4 layers | 3x3 | 0.25 |
| 6 layers | 4x4 | 0.5 |
| 8 layers | 5x5 | - |

For CNN-1, 48 different configurations are generated. The model is trained for 100 epochs, and the MSE is used to evaluate performance. Figure 5.5 shows the MSE on both the training and validation datasets, as well as overfitting. The model selected for CNN-1 is model 22, which has 4 convolutional layers, a 5x5 kernel size, and a dropout value of 0. This model demonstrates the lowest MSE among all the models with minimum overfitting, with MSE values of 1.49 for validation and 1.38 for training.

Figures 5.6 and 5.7 show, respectively, the three best-performing models that result in minimal overfitting: models 22,23 and 24, and the three worst-performing: models 46, 47, and 48. Out of the three best-performing models, model 22 achieved the lowest MSE values for both the training

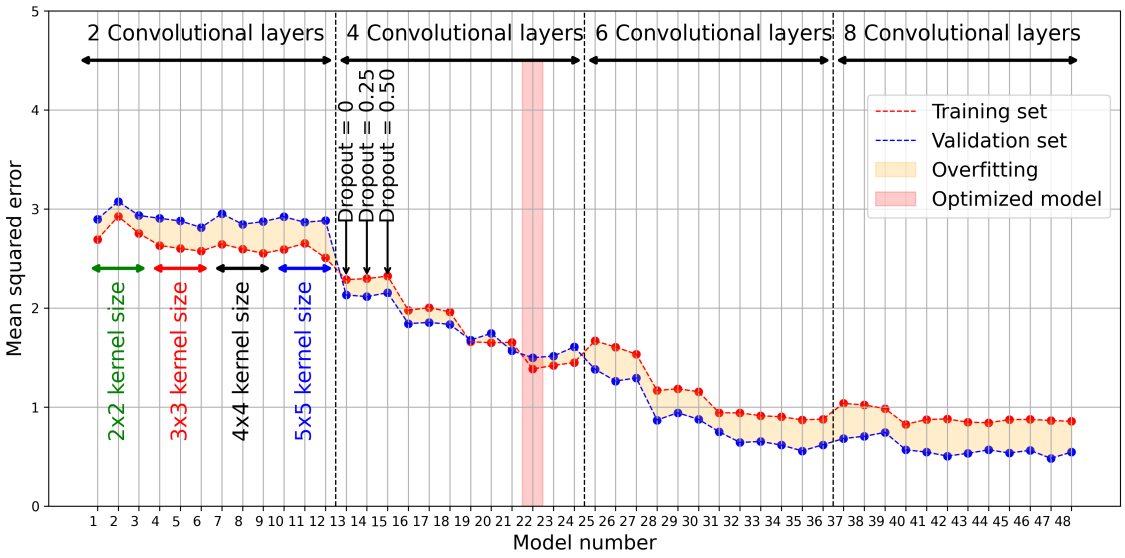


**Figure 5.5:** Performance of CNN-1 with varying hyperparameters for predicting the key variogram model parameters. The Figure is divided into four Sections based on the number of convolutional layers, each containing four Subsections representing different kernel sizes. The optimal model (model 22) strikes a balance between accuracy and minimum overfitting.

and validation datasets, with an MSE of 1.38 for training and 1.49 for validation. Considering the absence of overfitting, model 22 was chosen as the optimal model for CNN-1.

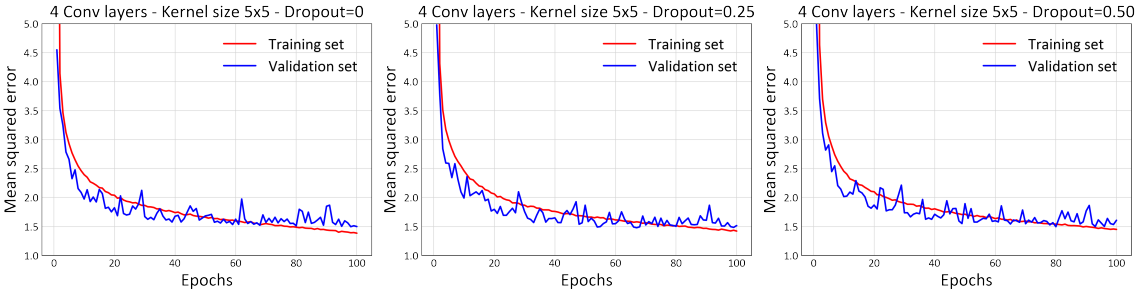


**Figure 5.6:** Comparison of best-performing models of CNN-1 with minimal overfitting.

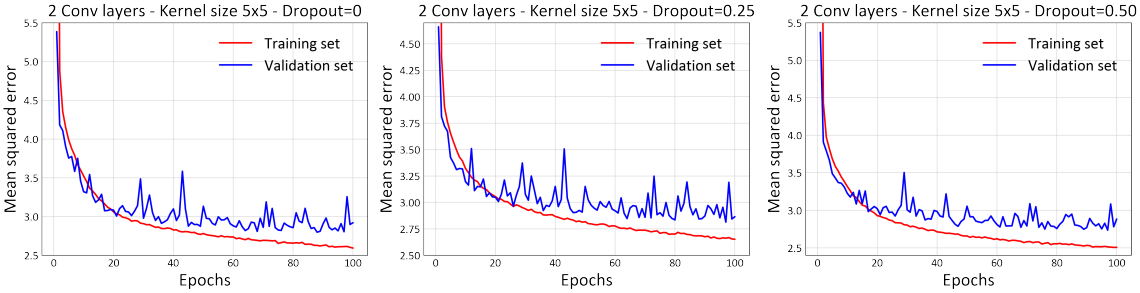


**Figure 5.7:** Comparison of worst-performing models of CNN-1 with overfitting.

### 5.1.3 Hyperparameter Tuning and Model Selection for CNN-2: Predicting the Experimental Variogram

48 different configurations are also generated for CNN-2. The specific ranges of values chosen for each parameter are summarized in Table 5.3. After training the models for 100 epochs, the optimized model is selected based on the MSE. Among the 48 trained models with different hyperparameter combinations, Figure 5.8 displays the evaluation of the performance of CNN2 based on the MSE; the optimized model selected is model 29.

**Table 5.3:** Range of hyperparameter values investigated for CNN-2.

| Hyperparameters | | |
|---|---|---|
| Number of convolutional layers | Kernel size | Dropout rate |
| 2 layers | 2x2 | 0 |
| 4 layers | 3x3 | 0.25 |
| 6 layers | 4x4 | 0.5 |
| 8 layers | 5x5 | - |

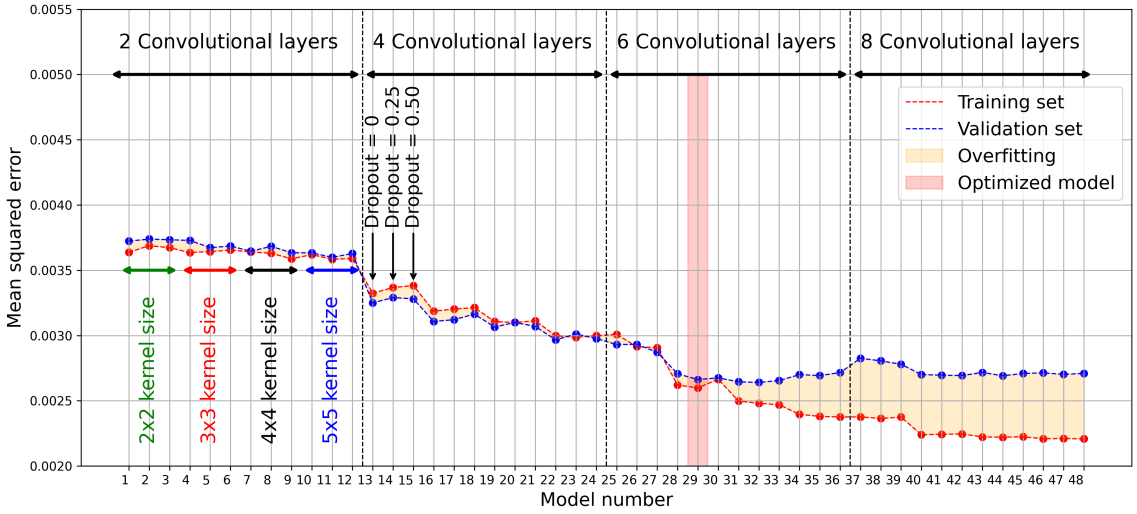


**Figure 5.8:** Performance of CNN-2 with varying hyperparameters for predicting the experimental variogram. The figure is divided into four Sections based on the number of convolutional layers, each containing four Subsections representing different kernel sizes. The optimal model (model 29) strikes a balance between accuracy and minimum overfitting.

Figures 5.9 and 5.10 show, respectively, the three best-performing models with minimal overfitting: models 28, 29, and 30, and the three worst-performing models with overfitting: models 46, 47, and 48. Among the three best-performing models, model 29 exhibited the lowest MSE values for both the training and validation datasets, with an MSE of 0.00259 for training and 0.00266 for validation. Consequently, model 29 is selected as the optimal model for CNN-2.

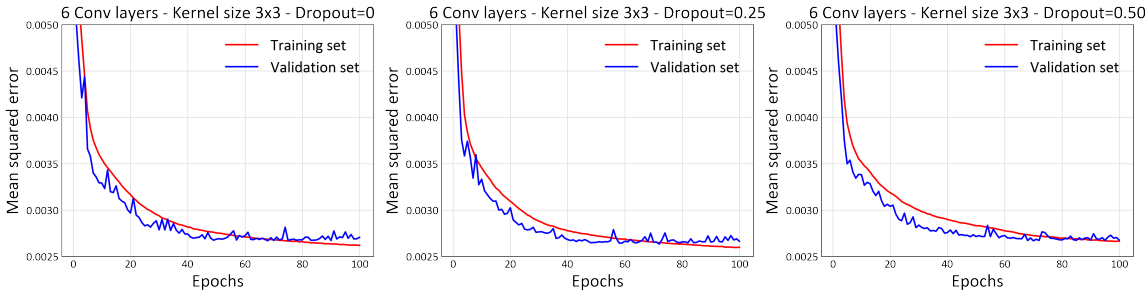


**Figure 5.9:** Comparison of best-performing models of CNN-2 with minimal overfitting.

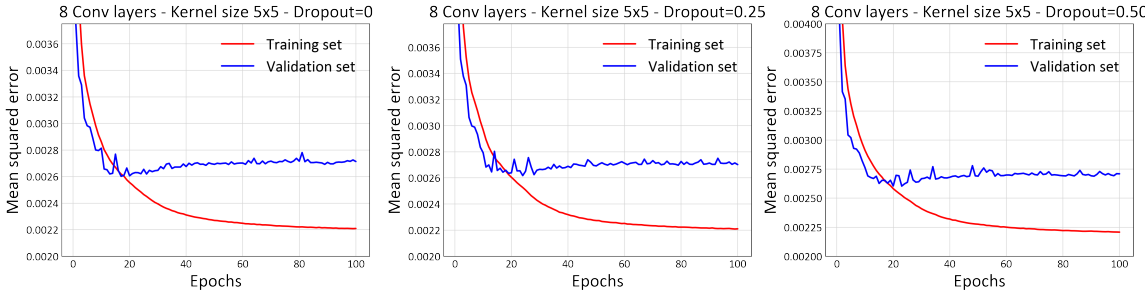


**Figure 5.10:** Comparison of worst-performing models of CNN-2 with overfitting.

## 5.2 Dataset Selection for Training and Validating CNN-based Variogram Prediction

The selection of an appropriate dataset for training and validating a CNN is crucial in constructing an efficient CNN. As discussed in Chapter 3, the workflow for generating the training data involves generating multiple unconditional SGS realizations with varying key variogram model parameters and directions of major continuity. These simulated data are then resampled to reproduce real case studies. The sampled data, along with their corresponding variogram model parameters, azimuth of the major direction, and experimental variogram, are used to train the CNNs.

The objective of the proposed CNNs is to predict the key variogram parameters, experimental variogram, and azimuth from the labeled data of each dataset. By mapping the sampled data obtained from SGS realizations to their labeled data, CNN learns to predict outputs for unseen sampled data. However, relying only on sampled data leads to poor predictive performance. To enhance the ability of the CNN to capture more spatial features, the workflow is based on a two-stage approach. In the suggested two-stage workflow, subsurface model estimations are generated and used to train the CNN instead of feeding the network with sampled data alone. This modification proved to be more effective in improving predictive performance. Several techniques are investigated for generating subsurface model estimations, including ordinary kriging, inverse distance, k-nearest neighbor, two-stage CNN, sampled data without subsurface model estimation, and SGS realizations with labeled variograms used to generate the sampled data.

However, it is important to address a particular constraint related to ordinary kriging. Ordinary kriging requires the use of a variogram, but there is no direct access to the variogram when making predictions for unseen data in real case studies since the purpose of this work is to predict the variogram itself. Manual calculation of the variogram to generate a kriging estimate that will be then used to predict the variogram is not feasible, as the goal of this work is to automate the variogram calculation. To overcome this constraint, ordinary kriging is implemented using an arbitrary variogram and applied to all the sampled training data. The arbitrary variogram has a range of 20, a nugget effect of 0.1, and an anisotropy ratio of 2. Choosing a range of 20 grid units is motivated by the fixed domain size of 100 x 100, used to generalize predictions. The SGS realizations are generated using a variogram range ranging from 1/10 to approximately 1/3 of the domain size, equivalent to a range between 10 and 30. The range of 20 is selected since it falls in the middle.

An additional subsurface estimation technique requiring clarification is the two-stage CNN, which involves two steps and is a modified version of the method proposed by Jo and Pyrcz (2022). First, a subsurface model estimation is generated using a first CNN that maps the sampled data to an exhaustive estimation. This CNN is trained using sampled data as the input and SGS realizations as the target output. Once trained, the two-stage CNN generates a subsurface model estimation based on the input sampled data. The subsurface estimations generated by the CNN are then used as input training data to predict the variogram values at pre-set lag distances. By incorporating a two-stage prediction workflow instead of training a CNN based on only sampled data, the aim is to enhance the ability of the CNN to learn spatial features and improve its performance. Subsurface model estimations are generated using the proposed techniques and implemented to predict experimental variogram values at pre-set lag distances using CNN-2 to select the most efficient technique to generate the training and validation data.

To evaluate the performance of each estimation technique, MSE of the CNNs are plotted based on the training and validation data, as shown in Figure 5.11. It is noted that feeding the CNN with only sampled data resulted in the highest MSE, indicating that this approach is not sufficient to accurately map the input to the desired output. The best performance is achieved using SGS realizations as training data. However, despite its superior performance, using SGS realizations as training data is not feasible due to several reasons. First, these realizations are considered truth and have the variogram that the main objective is to calculate embedded. Hence, comparing the model trained on SGS realizations with the other techniques is unfair. Secondly, when testing the model with unseen data, using SGS realizations is not feasible. This is because the variogram is required to generate SGS realizations, making it impossible to automatically generate variogram estimation without prior calculation of the variogram to be able to generate realizations as it is explained with ordinary kriging. This method is included to generate a baseline for comparing errors rather than as a feasible method for implementation. Among the estimation techniques considered, the inverse distance method performes the best, with an MSE of 0.0052 for training data and 0.0056

for validation. This method is attractive as it does not rely on a variogram and aligns with the goal of predicting the variogram. Consequently, it is selected as the most suitable for generating training data in this specific scenario of automatic variogram prediction. However, it is important to note that this conclusion does not imply that inverse distance is superior to ordinary kriging as an estimation method. The implementation of kriging in this study is not fair due to the use of an arbitrary variogram, potentially impacting its efficiency. Therefore, a fair comparison cannot be made. The decision to use inverse distance for generating training data is specific to the CNN variogram estimation technique being studied in this research and does not indicate that inverse distance outperforms ordinary kriging or machine learning estimation using CNNs.

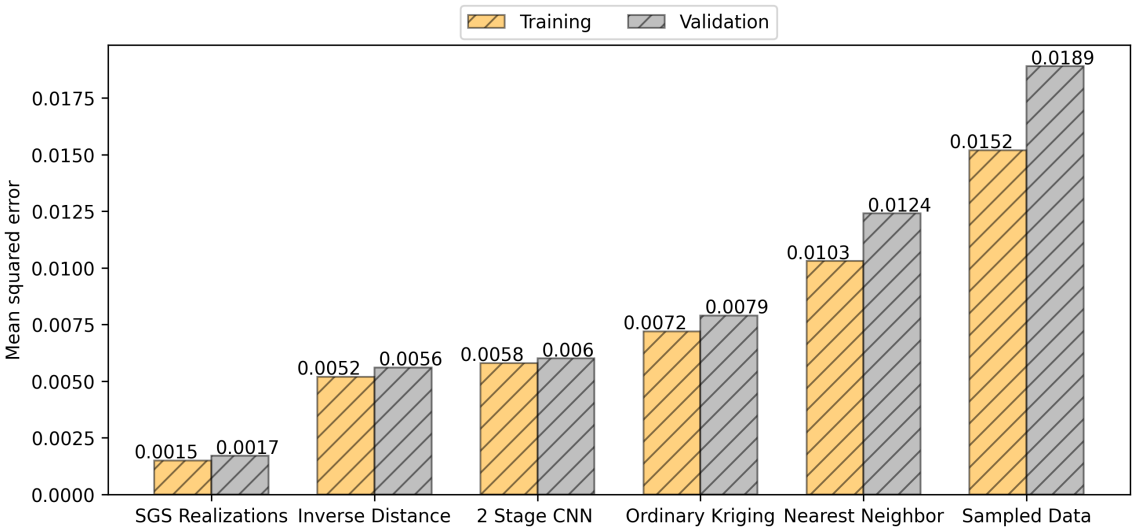


**Figure 5.11:** Evaluating the performance of estimation and simulation techniques for generating training and validation data to train the proposed CNN.

## 5.3 Exploring the Influence of Inverse Distance Parameters on CNN Predictions

As demonstrated in the preceding Section, the analysis revealed that the implementation of the inverse distance method is the most suitable proposed technique for generating training data. The next step is to investigate the sensitivity of the inverse distance method to its key parameters and its influence on CNN estimations.

One of the key parameters is the inverse distance exponent, also referred to as the power value which determines the rate at which the influence of the data on the inverse distance weights decreases with distance (Isaaks and Srivastava, 1989). By adjusting the power value, it is possible to control the influence of distant data points on the predicted values. A higher power value results in a lower influence of the distant points, giving more importance to the closest points. A lower power value

allows more influence of distant points and results in a prediction closer to the mean (Babak and Deutsch, 2009).

In addition to the sensitivity analysis on the power value, the impact of the number of neighboring data to be used for the estimation of the accuracy of the CNN predictions is explored. By varying the number of the maximum data used, it is possible to assess the optimal parameters to automatically predict a variogram. This analysis provides a relationship between the inverse distance parameters and the CNN predictive performance.

### 5.3.1 Impact of Maximum Neighboring Data on Variogram Prediction Using CNN

To investigate the impact of the maximum number of neighboring data used, multiple training data sets are generated with different maximum number of data used (n=12, 24, and 48). These datasets are then implemented in CNN-2 as training and validation data to predict the experimental variogram for the 110 case studies in the validation dataset. Through a comparative analysis of the resulting predictions, the aim is to identify the maximum number of data that leads to the lowest MSE. Figure 5.12 presents the obtained results. The model trained with a maximum number of data n=48 results in the best performance with an average MSE of 0.0348 across the 110 case studies, followed by the model trained with n=24 resulting in an average error of 0.0619. The model trained with n=12 yielded the worst average MSE value of 0.067.

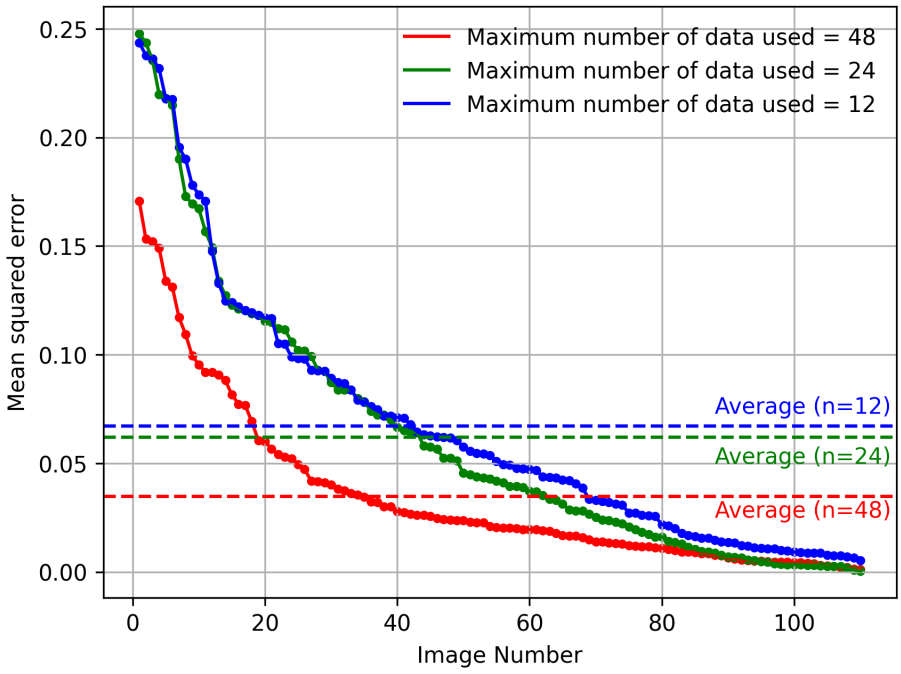


**Figure 5.12:** Impact of maximum neighboring data used on variogram prediction using CNN-2 applied on the 110 case studies of the validation dataset: Comparative analysis and optimal training configuration.

In addition to the 110 case studies, the generated CNN models using various maximum numbers of data are implemented on the Walker Lake dataset to predict the experimental variogram values using CNN-2. Figure 5.13 presents the predicted variograms compared to the true variogram, accompanied by a plot illustrating the relationship between the maximum number of neighboring data used and the MSE. The results align with the implementation of the 110 case studies from the data validation project, where the model run with n=48 results in the lowest MSE of 0.0017. Based on these findings, the model trained with a maximum number of data used, n=48, is selected as the optimal choice.

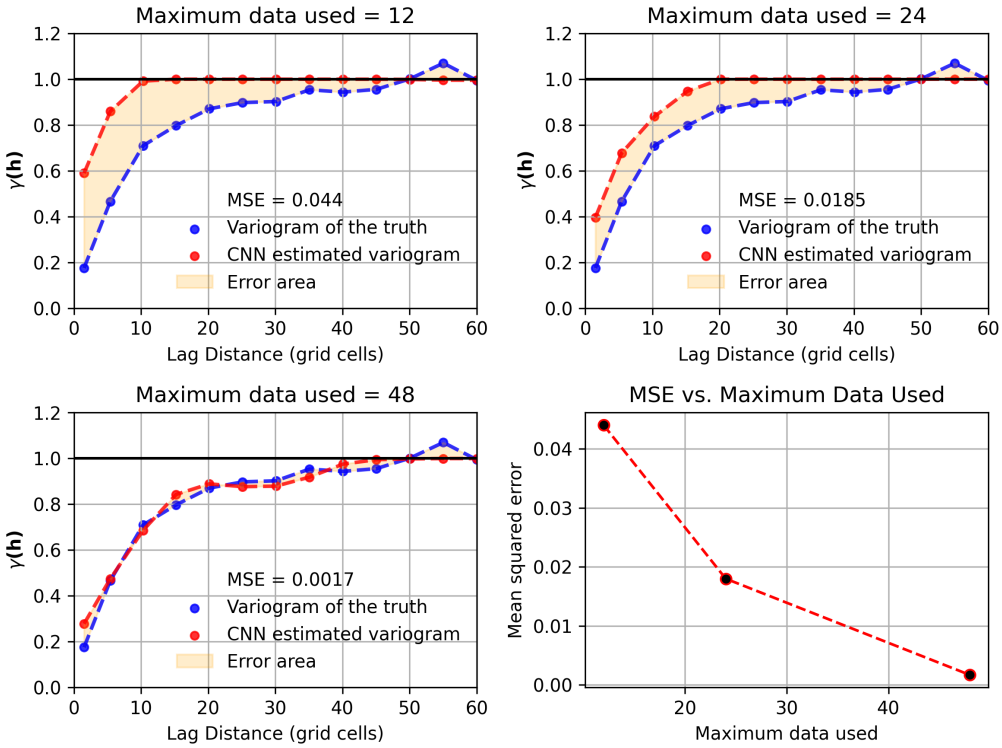


**Figure 5.13:** Impact of maximum neighboring data used on variogram prediction using CNN-2 applied on Walker Lake dataset: Comparative analysis and optimal training configuration.

### 5.3.2 Impact of the Inverse Distance Exponent on Variogram Prediction Using CNN

This Section investigates the influence of the power value on the accuracy of CNN predictions. To explore the impact of different power values, the values: 1, 1.7, 2, 2.5, 3, 4, and 6 are considered. While the most commonly used value is 2, it is important to note that this choice is not obligatory, and other values can be selected based on the specific requirements of the problem (Babak and Deutsch, 2009).

Different sets of training data based on the various selected power values are generated. These sets of training data are then used in CNN-2 to predict the experimental variogram values at different

lag distances. The training dataset resulting in the lowest MSE is selected. Figure 5.14 illustrates the results obtained from implementing CNN-2 with different power values. The model trained with a power value of 2 demonstrated the best performance, exhibiting an average MSE of 0.0348. This result aligns with the commonly used power value of 2 in the inverse distance.

However, the model trained with a power value of 1.7 also achieved a competitive performance, with an average MSE of 0.03684. This indicates that a slightly lower power value can still provide accurate predictions. On the other hand, when higher power values are used, the accuracy of the predictions decreases. The models trained with power values of 2.5, 3, 4, and 6 resulted in average MSE values of 0.0414, 0.0409, 0.0447, and 0.0462, respectively.

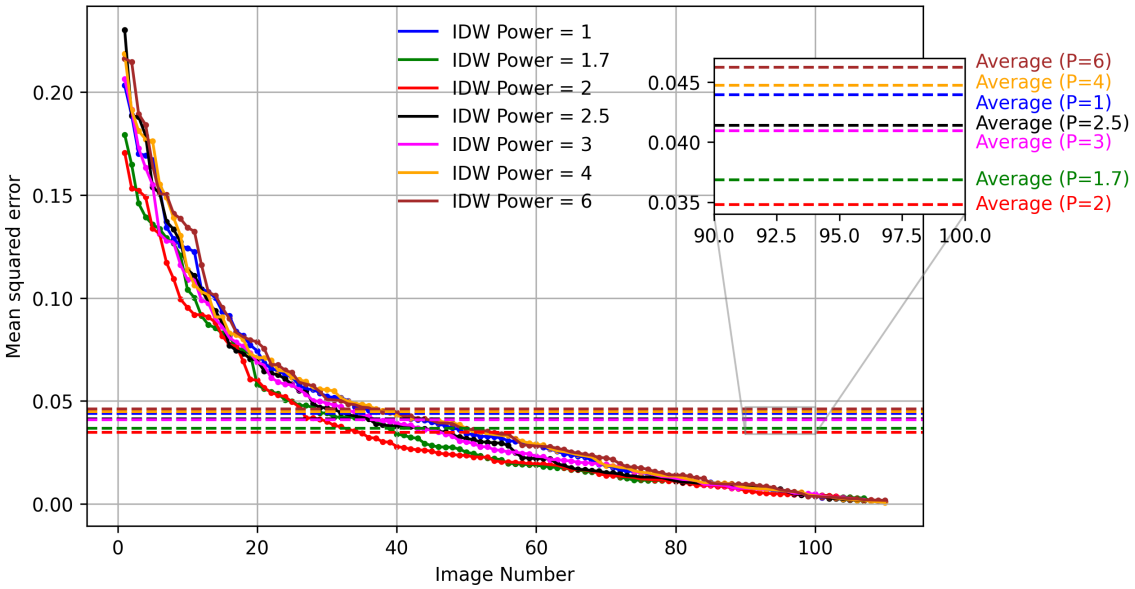


**Figure 5.14:** Impact of the inverse distance exponent on variogram prediction using CNN-2 applied on the 110 case studies of the validation dataset: Comparative analysis and optimal training configuration.

In addition to the analysis conducted on the 110 case studies from the validation dataset, the impact of the power value on the accuracy of CNN predictions is also evaluated using the Walker Lake dataset. The same set of power values (1, 1.7, 2, 2.5, 3, 4, and 6) are considered for this analysis. Figure 5.15 presents the results obtained when implementing CNN-2 with the different proposed power values on the Walker Lake dataset. Similar to the findings from the validation dataset, the model trained with a power value of 2 exhibited the best performance with an MSE of 0.0017. Based on these results, it can be concluded that the power value of 2 results in optimal performance in terms of predicting the experimental variogram values using CNN-2.

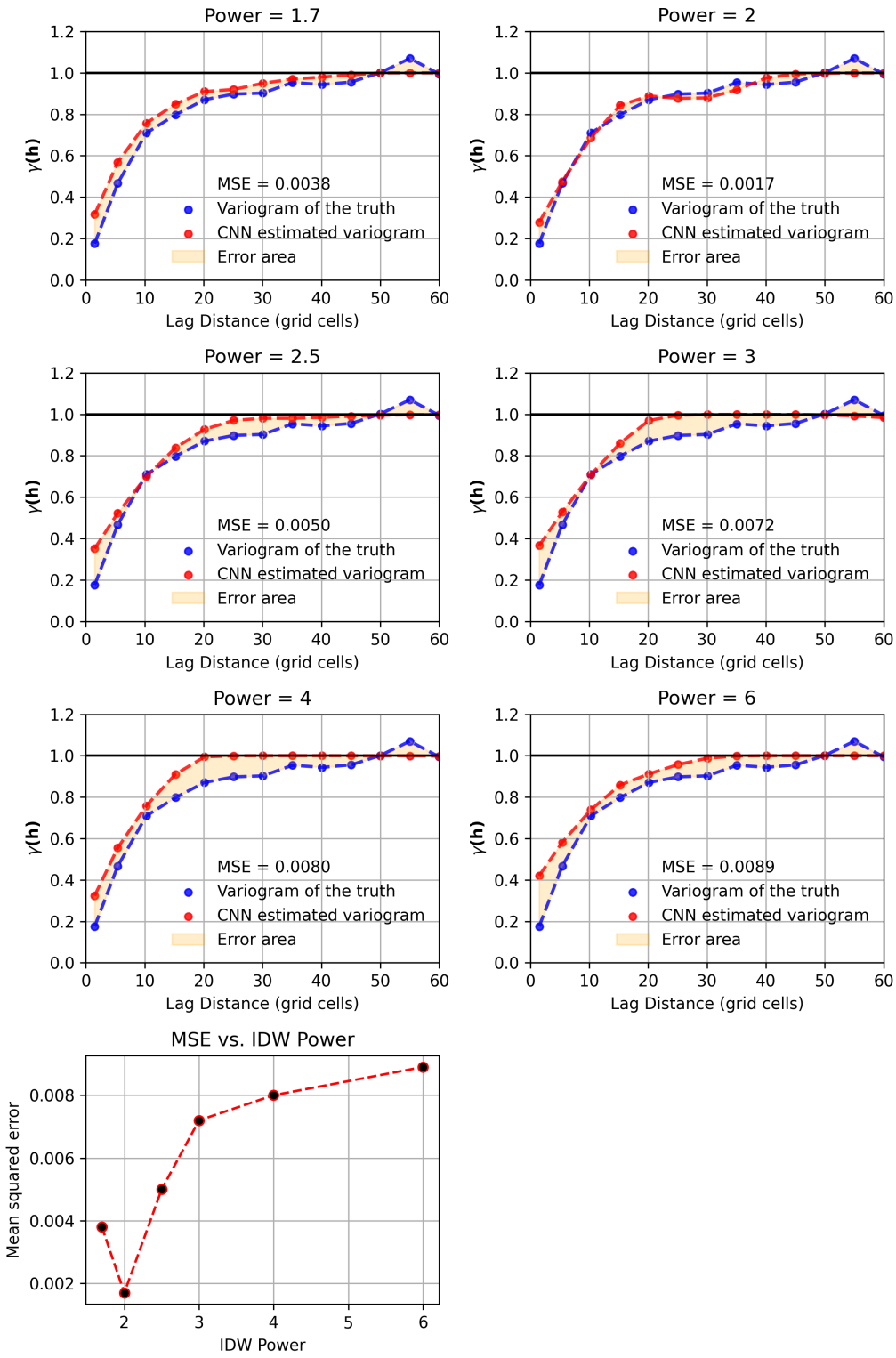


**Figure 5.15:** Impact of the inverse distance exponent on variogram prediction using CNN-2 applied on Walker Lake dataset: Comparative analysis and optimal training configuration.

### 5.3.3 Sensitivity Analysis of CNN-2 Performance Based on Training/Validation Data Size

In this Section, a sensitivity analysis is conducted for CNN-2 based on the number of training/validation data used. The selected numbers of data are 1000, 2000, 5000, 10000, 25000, 50000, 75000, and 100000. Due to computational constraints, the analysis is stopped at 100,000 training data, as using a larger number is computationally expensive when running CNN. The CNN model used in the analysis is CNN-2, which predicts experimental variogram values. The training data constitutes 80% of the dataset, while the remaining 20% is used for validation. Figure 5.16 displays the curves of the training and validation results. It can be observed that the MSE decreases as the number of samples increases for both curves. However, the training data curve consistently exhibits lower MSE compared to the validation data curve. Among the different configurations tested, the CNN model trained using 100,000 data points yielded the best results, with an MSE of 0.0025 for training and 0.0027 for validation.

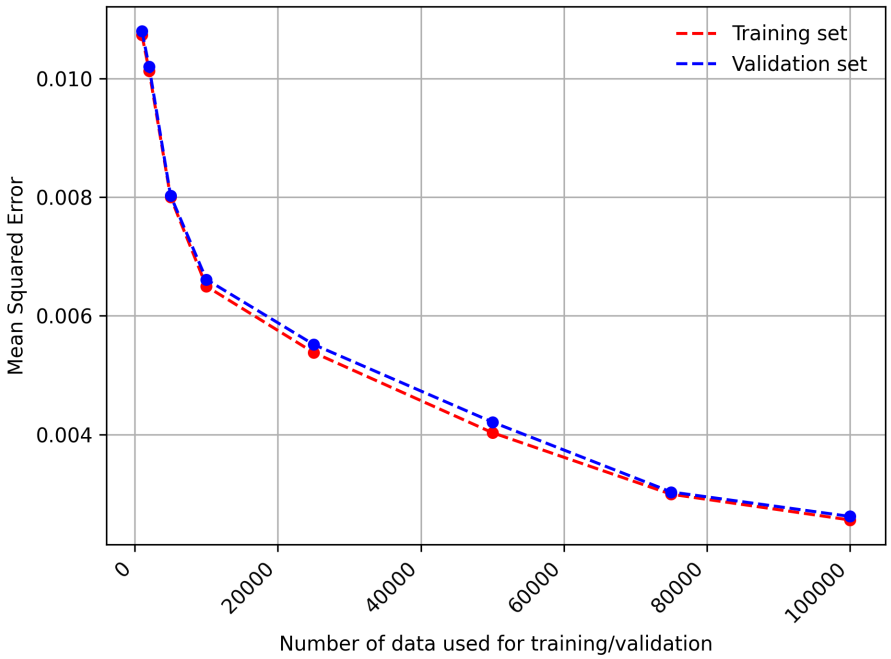


**Figure 5.16:** MSE results of CNN-2 with varying the number of training and validation data.

## 5.4 Summary

In this Chapter, hyperparameter tuning is performed to optimize the performance of the proposed CNN models for variogram prediction. Sensitivity analysis is conducted on the number of convolutional layers, kernel size, and regularization parameters. The optimal hyperparameter configurations are selected based on the MSE and the presence of overfitting.

For CNN-A, the model with 4 convolutional layers, a 2x2 kernel size, and a dropout value of 0.25 is selected as the optimal model. It demonstrated the lowest MSE values for both the training and validation datasets, with minimal overfitting. Similarly, for CNN-1, the model with 4 convolutional layers, a 5x5 kernel size, and a dropout value of 0 is selected as the optimal model. For CNN-2, the model with 6 convolutional layers, a 3x3 kernel size, and a dropout value of 0.25 is selected as the optimal model.

The selection of the training and validation dataset is crucial for the performance of the CNN models. The inverse distance method is found to be the most suitable technique for generating training data. The impact of inverse distance parameters, such as the power value and the maximum number of neighboring data, are also investigated. The optimal values were determined to be a power value of 2 and a maximum number of neighboring data of 48.

Furthermore, sensitivity analysis is conducted on the training and validation data size. The analysis revealed that increasing the training data size led to improved performance with lower MSE values. However, computational constraints limited the analysis to 100,000 training data.

In conclusion, the hyperparameter tuning process and dataset selection significantly influenced the performance of the CNN models for variogram prediction. The optimized models demonstrated improved accuracy and generalization capabilities, making them reliable for predicting experimental variogram values, key variogram model parameters, and azimuth of the major direction. The findings from this Chapter provide valuable insights for selecting optimal hyperparameter configurations and dataset generation techniques for CNN-based variogram prediction tasks.

# Chapter 6

# Conclusions

This Chapter outlines the main conclusions drawn from the implementation and application of the CNN-based methodologies, demonstrating their potential and limitations. The primary objective of this research is to develop a CNN-based method for automating variogram calculation and modeling, with the aims of predicting the major direction of continuity, generating stable experimental variograms, predicting key variogram model parameters, minimizing user interaction, and reducing user modeling time. To achieve these goals, two workflows are implemented. The first workflow involves predicting key variogram model parameters, while the second workflow directly generates a normal score stable experimental variogram. The computational time when applying the workflows is minimal because the CNN is pre-trained. The effectiveness and limitations of the methodologies are discussed along with recommendations and major contributions made in this research.

## 6.1 Contributions

The workflows offer a CNN-based alternative for automatically calculating the azimuth of the major continuity and two alternatives for automating variogram inference using CNNs. The first workflow involves predicting key variogram model parameters, such as the range of the major direction, anisotropy ratio, and nugget effect. These predicted parameters are then used to infer the variogram model. The second workflow directly generates a robust and stable normal score experimental variogram. This variogram is stable and easy to model. This workflow avoids the traditional experimental variogram calculation process and eliminates the need for selecting search and tolerance parameters. Both workflows simplify the modeling process.

The primary advantage of the program implemented in this research is its generalizability, eliminating the need for retraining on each data set. This is achieved through a pre-trained model that uses a large training dataset (100,000). Hyperparameter tuning, crucial to the performance of CNN models, is performed, optimizing the number of layers, kernel size, and dropout rate. The selected CNN model configurations are based on minimizing MSE and avoiding overfitting.

Several case studies are conducted to assess the performance and predictive accuracy of the implemented workflows. The Walker Lake dataset, presented with different sampling configurations, and a data validation project involving 110 geological images are used to provide insights into the strengths and limitations of the methodologies. Workflow 2 is considered the most efficient methodology, effectively handling variograms requiring multiple structures.

The data validation images used as case studies are derived from grayscale values of different geo-

logical images. These images can be used as inputs for various geostatistical techniques, enabling researchers and users to benchmark their newly implemented methods. Unlike real case studies, these images provide exhaustive data for each dataset to facilitate validation.
The main contribution of this research is the incorporation of CNNs into geostatistical workflows and exploring the impact of hyperparameter tunning in CNNs on variogram inference. The implemented workflows extend beyond traditional variogram calculations.

## 6.2 Limitations

There are limitations to the implemented approaches. The CNN models can only predict normal score variograms in 2-D. The current CNN-based approach is designed and implemented for 2-D datasets. While this is suitable for many geostatistical applications, it may not be directly applicable to 3-D cases. Extending the approach to 3-D datasets may require significant additional complexities such as additional training data and computational effort.
CNN-1, designed for predicting key variogram model parameters, is only capable of predicting a single spherical structure with a maximum range of 1/3 of the domain size and a maximum nugget effect of 0.3. This restricts its applicability to datasets where the spatial continuity can be accurately modeled with a single structure, limiting the generalizability of the model. Datasets with complex spatial structures requiring multiple nested structures cannot be effectively modeled using the current version of CNN-1. Additionally, CNN-2, which predicts normal score experimental variograms, results in poor predictive performance with variograms with a high nugget effect and zonal anisotropy.
The performance of the CNN models is sensitive to the selection of hyperparameters. While the study in Chapter 5 performed hyperparameter tuning, there is still a possibility that some specific configurations may not have been explored exhaustively, leading to suboptimal model performance. Further research and exploration of hyperparameters could improve model accuracy and robustness.
The performance of the CNN-based approach is highly dependent on the variability of the training dataset. If the training data is limited or does not sufficiently represent the target dataset, the predictions may exhibit bias. This is particularly evident when implementing CNN-2 in cases with high nugget effects, zonal anisotropy, variograms showing a steeper rise, and variograms requiring more than one structure for CNN-1. Both models are not trained for such cases, resulting in poor predictive performance.

## 6.3 Future Work

The CNN-based approach could be extended to 3-D, which would increase the applicability of the model to a wider range of geostatistical case studies. This would, however, require substantial computational power and a higher number of training data to ensure robust and accurate predic-

tions. Although an extension of CNN-2 applied to 3-D datasets is presented in Appendix 2. This extension involves predicting the experimental variograms of 3-D datasets in the horizontal direction (horizontal minor/horizontal major) for each stratigraphic layer.

Future work should focus on the enhancement of the training dataset. To improve model flexibility, modifications to CNN-1 should be made to allow it to predict multiple nested structures, accommodating datasets with complex spatial continuity. The complexity of variograms could be further addressed by developing and training a more advanced CNN-1 model that can handle multiple nested structures, instead of a single structure. This would enhance the capability of the model to predict complex spatial continuity and improve its overall predictive performance. To improve the performance of CNN-2, the training data should include cases with high nugget effects, zonal anisotropy, and steep variograms. This approach will help the model better capture spatial patterns, contributing to its improved performance. Employing a wider variety of datasets would improve the overall predictive performance of both models. Although computationally expensive, adding additional training data, in this case, would address the limitations. Future research could also aim to explore more hyperparameter configurations, optimizing model performance by conducting more exhaustive hyperparameter tuning.

## 6.4 Final thoughts

The CNN workflows explored have the potential to be efficient for real data if enough samples are available and if the variables under analysis are within a stationary domain. The first explored workflow, CNN-1, can be particularly useful for predicting key variogram model parameters and inferring the normal score variogram model for cases of spatial continuity that require a single spherical structure with a maximum range of 1/3 of the domain and a nugget effect of less than 0.3. For cases requiring one or more nested structures with a nugget effect of less than 0.3 and no zonal anisotropy, the second workflow, CNN-2, is recommended. CNN-2 is the preferred workflow due to its lower error and its applicability in cases requiring one or more nested structures. The workflows reduce user modeling time and can be used as an initial model or scripted into a fully automated simulation/estimation workflow. However, to effectively apply this technique and carefully predict the variogram, it is crucial to consider and understand the geological factors that play a major role in the variogram modeling process.

# References


Abdel-Hamid, O., Mohamed, A. R., Jiang, H., Deng, L., Penn, G., and Yu, D. (2014). Convolutional neural networks for speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 22(10):1533–1545.

Avalos, S. and Ortiz, J. M. (2019). Convolutional neural networks architecture A tutorial. *Predictive Geometallurgy and Geostatistic Lab*, pages 159–168.

Avalos, S. and Ortiz, J. M. (2020). Recursive convolutional neural networks in a multiple-point statistics framework. *Computers & geosciences*, 141:104522.

Babak, O. and Deutsch, C. V. (2009). Statistical approach to inverse distance interpolation. *Stochastic Environmental Research and Risk Assessment*, 23(5):543–553.

Bochinski, E., Senst, T., and Sikora, T. (2017). Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. pages 3924–3928.

Cardoza, I., García-Vázquez, J. P., Díaz-Ramírez, A., and Quintero-Rosas, V. (2022). Convolutional Neural Networks Hyperparameter Tunning for Classifying Firearms on Images. *Applied Artificial Intelligence*, 36(1).

Chiles, J.-P. and Delfiner, P. (2012). *Geostatistics: modeling spatial uncertainty*, volume 713. John Wiley & Sons.

Chollet, F. et al. (2015). keras.

Cressie, N. (1985). Fitting variogram models by weighted least squares. *Journal of the International Association for Mathematical Geology*, 17(5):563–586.

da Silva, C. Z., Nisenson, J., and Boisvert, J. (2022). Grade Control with Ensembled Machine Learning: A Comparative Case Study at the Carmen de Andacollo Copper Mine. *Natural Resources Research*, 31(2):785–800.

David, M. (1988). Handbook of applied advanced geostatistical ore reserve estimation.

Desassis, N. and Renard, D. (2013). Automatic variogram modeling by iterative least squares: univariate and multivariate cases. *Mathematical geosciences*, 45:453–470.

Deutsch, C. V. and Journel, A. G. (1998). GSLIB: geostatistical software library and user's guide. Second edition. *GSLIB: geostatistical software library and user's guide. Second edition.*

Dietrich, C. and Osborne, M. (1991). Estimation of covariance parameters in kriging via restricted maximum likelihood. *Mathematical Geology*, 23:119–135.

Dimitrakopoulos, R. (1993). Artificially intelligent geostatistics: A framework accommodating qualitative knowledge-information. *Mathematical Geology*, 25(3):261–279.

Earle, C. D., King, E. M., Tsay, A., Pittman, K., Saric, B., Vailes, L., Godbout, R., Oliver, K. G., and Chapman, M. D. (2007). High-throughput fluorescent multiplex array for indoor allergen exposure assessment. *Journal of Allergy and Clinical Immunology*, 119(2):428–433.

Emery, X. (2010). Iterative algorithms for fitting a linear model of coregionalization. *Computers and Geosciences*, 36(9):1150–1160.

Gholamalinezhad, H. and Khosravi, H. (2020). Pooling methods in deep neural networks, a review. *arXiv preprint arXiv:2009.07485*.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning: Machine Learning Book. page 785.

Goovaerts, P. (1997). *Geostatistics for natural resources evaluation*. Applied Geostatistics.

Graham, B. (2014). Fractional Max-Pooling. pages 1–10.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., and Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48.

Hao, W., Yizhou, W., Yaqin, L., and Zhili, S. (2020). The Role of Activation Function in CNN. *Proceedings - 2020 2nd International Conference on Information Technology and Computer Application, ITCA 2020*, pages 429–432.

Haykin, S. (2009). *Neural networks and learning machines, 3/E*. Pearson Education India.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Isaaks and Srivastava, R. M. (1989). An introduction to applied geostatistics.

Jin, H. (2022). Hyperparameter Importance for Machine Learning Algorithms. pages 1–8.

Jo, H., Pan, W., Santos, J. E., Jung, H., and Pyrcz, M. J. (2021). Machine learning assisted history matching for a deepwater lobe system. *Journal of Petroleum Science and Engineering*, 207(February):109086.

Jo, H. and Pyrcz, M. J. (2022). Automatic semivariogram modeling by convolutional neural network. *Mathematical Geosciences*, 54(1):177–205.

Journel, A. G. and Huijbregts, C. J. (1978). *Mining Geo-statistics*.

Kanevski, M., Parkin, R., Pozdnukhov, A., Timonin, V., Maignan, M., Demyanov, V., and Canu, S. (2004). Environmental data mining and modeling based on machine learning algorithms and geostatistics. *Environmental Modelling and Software*, 19(9):845–855.

Kingma, D. P. and Lei Ba, J. (2015). 15iclr-ADAM. *Iclr*, pages 1–15.

Kitanidis, P. K. (1987). Parametric estimation of covariances of regionalized variables 1. *JAWRA Journal of the American Water Resources Association*, 23(4):557–567.

Kitanidis, P. K. and Lane, R. W. (1985). Maximum likelihood parameter estimation of hydrologic spatial processes by the gauss-newton method. *Journal of Hydrology*, 79(1-2):53–71.

Kothari, S. C. and Oh, H. (1993). Neural Networks for Pattern Recognition. *Advances in Computers*, 37(C):119–166.

Larrondo, P. F., Neufeld, C. T., and Deutsch, C. V. (2003). Varfit: A program for semiautomatic variogram modeling. *Fifth Annual Report of the Centre for Computational Geostatistics, Uni-*

*versity of Alberta, Edmonton*, page 24.

Lashgari, E., Liang, D., and Maoz, U. (2020). Data augmentation for deep-learning-based electroencephalography. *Journal of Neuroscience Methods*, 346(July):108885.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Li, Z., Zhang, X., Clarke, K. C., Liu, G., and Zhu, R. (2018). An automatic variogram modeling method with high reliability fitness and estimates. *Computers and Geosciences*, 120:48–59.

Maharana, K., Mondal, S., and Nemade, B. (2022). A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, 3(1):91–99.

Manchuk, J. G. and Deutsch, C. V. (2019). Covariance Fitting with Tensorflow. 2019:1–7.

Mardia, K. V. and Marshall, R. J. (1984). Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika*, 71(1):135–146.

Mokdad, A., Binakaj, D., and Boisvert, J. B. (2022). Data Validation Project : Validation of 114 Spatial 2D Datasets ( Non synthetic Data ). *Twenty-fourth Annual Report of the Centre for Computational Geostatistics, University of Alberta, Edmonton*, pages 1–8.

Nejedly, P., Cimbalnik, J., Klimes, P., Plesinger, F., Halamek, J., Kremen, V., Viscor, I., Brinkmann, B. H., Pail, M., Brazdil, M., Worrell, G., and Jurak, P. (2019). Intracerebral EEG Artifact Identification Using Convolutional Neural Networks. *Neuroinformatics*, 17(2):225–234.

Oliver, M. A. and Webster, R. (2014). A tutorial guide to geostatistics: Computing and modelling variograms and kriging. *Catena*, 113:56–69.

Passos, D. and Mishra, P. (2022). A tutorial on automatic hyperparameter tuning of deep spectral modelling for regression and classification tasks. *Chemometrics and Intelligent Laboratory Systems*, 223(February).

Pyrcz, M., Deutsch, C., and Deutsch, J. (2018). Transforming data to a gaussian distribution. *Geostatistics Lessons*.

Pyrcz, M. J. and Deutsch, C. V. (2014). *Geostatistical reservoir modeling*. Oxford university press.

Rossi, M. E. and Deutsch, C. V. (2014). *Mineral resource estimation*.

Samal, A., Sengupta, R., and Fifarek, R. (2011). Modelling spatial anisotropy of gold concentration data using GIS-based interpolated maps and variogram analysis: Implications for structural control of mineralization. *Journal of Earth System Science*, 120(4):583–593.

Samson, M. J. (2019). Mineral Resource Estimates with Machine Learning and Geostatistics. *A thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Mining Engineering*, (December):1–112.

Santos, J. E., Yin, Y., Jo, H., Pan, W., Kang, Q., Viswanathan, H. S., Prodanović, M., Pyrcz, M. J., and Lubbers, N. (2021). *Computationally Efficient Multiscale Neural Networks Applied to Fluid Flow in Complex 3D Porous Media*, volume 140. Springer Netherlands.

Sharma, S., Sharma, S., and Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Todini, E. and Pellegrini, F. (1999). A maximum likelihood estimator for semi-variogram parameters in kriging. In *geoENV II Geostatistics for Environmental Applications: Proceedings of the Second European Conference on Geostatistics for Environmental Applications held in Valencia, Spain, November 18–20, 1998*, pages 187–198. Springer.

Watkins, A. and Mardia, K. (1992). Maximum likelihood estimation and prediction mean square error in the spatial linear model. *Journal of Applied Statistics*, 19(1):49–59.

Wilde, B. J. and Deutsch, C. V. (2012). Automatic Variogram Modeling from Censored Variogram Volumes. 2012:1–4.

Wu, Y. and Liu, L. (2023). Selecting and Composing Learning Rate Policies for Deep Neural Networks. *ACM Transactions on Intelligent Systems and Technology*, 14(2).

Yamashita, R., Nishio, M., Do, R. K. G., and Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9:611–629.

Ying, X. (2019). An overview of overfitting and its solutions. In *Journal of physics: Conference series*, volume 1168, page 022022. IOP Publishing.

Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A. K., and Almotairi, S. (2022). A comparison of pooling methods for convolutional neural networks. *Applied Sciences*, 12(17):8643.

## Appendix A

# Appendix A: Subsurface Model Estimation Using CNN

Appendix A presents methodology and results for automating subsurface model estimation using CNNs from sampled data. The objective of this technique is to automate the estimation process, minimize parameter selection, and reduce modeling time. The methodology consists of two main steps:

- Design and generation of the training data
- Training a CNN

To demonstrate the performance of the proposed methodology, it is applied to the Walker Lake dataset. The model generated by the CNN is compared to the exhaustive data, and the difference between the estimated values and the truth is evaluated.

## A.1 Methodology

The methodology implemented in this study is similar to the method proposed by Jo and Pyrcz (2022) but several changes have been made. These changes include incorporating the influence of the nugget effect during training data generation and modifying the CNN architecture by adjusting the number of layers, kernel size, and the type of activation function. The proposed CNN method in this workflow automates the process and eliminates the need for variogram calculations, reducing the number of parameters required for estimation. The workflow is divided into two steps: the design and generation of the training data, followed by the second step, which involves training a CNN.

### A.1.1 Step 1: Design and Generation of the Training Data

To generate the training and validation data, 1000 SGS realizations using the input data are generated. These realizations are created using different combinations of variogram parameters (Table A.1). For each combination, random values are chosen for each parameter.
For the azimuth, 1000 values are randomly generated and then one value from this set is selected for each combination. The chosen azimuth values fall within the range of 0 to 180 degrees. Similarly, the range values are determined to be approximately within 1/10 to 1/3 of the domain size, which is 128 by 128, resulting in range values between 10 and 40. Additionally, 1000 anisotropy ratio values

between 1 and 3 are selected, and the nugget effect values are chosen to be between 0 and 0.3. It is worth noting that all of the variograms used in the realizations have a single spherical structure (Figure A.1). The SGS realizations are then sampled. The CNN is trained using sampled data as the input and SGS realizations as the target output.

**Table A.1:** Details of the variogram model parameters used to train the CNN models.

| Variogram model parameter | Values |
|---|---|
| Major direction of continuity | Between 0 and 180 (degrees) |
| Range in the major direction | Between 10 and 40 (grid cells) |
| Anisotropy ratio | Between 1 and 3 |
| Nugget effect | Between 0 and 0.3 |
| Structure | Spherical |
| Number of nested structures | 1 |

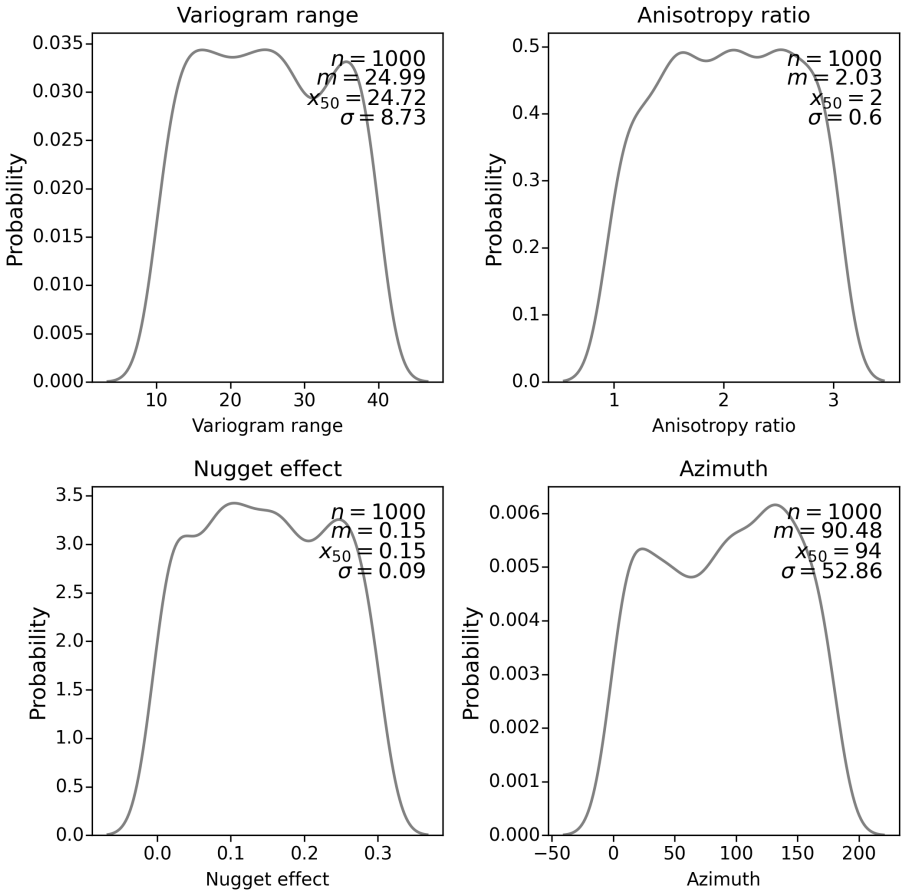


**Figure A.1:** Variogram parameters for generating realizations used as training data.

### A.1.2 Step 2: Training a CNN

To map sparsely sampled data to subsurface model estimation, the CNN is trained using the generated training data. The dataset of 1000 realizations is divided into 800 samples for training and

200 for validation. The generated data is then fed into the CNN network, enabling the training of a model for generating subsurface model estimations. The CNN architecture (Figure A.2 and Table A.2) contains 5 convolutional layers, one dense fully connected layer, and 4 upscaling layers. A kernel size of 2x2 and an upscaling factor of 2x2 are used, along with the sigmoid activation function, selected through hyperparameter tuning to obtain the best model. Once trained, this model can predict subsurface model estimation using input sparsely sampled data, eliminating the need for prior calculation of a variogram.

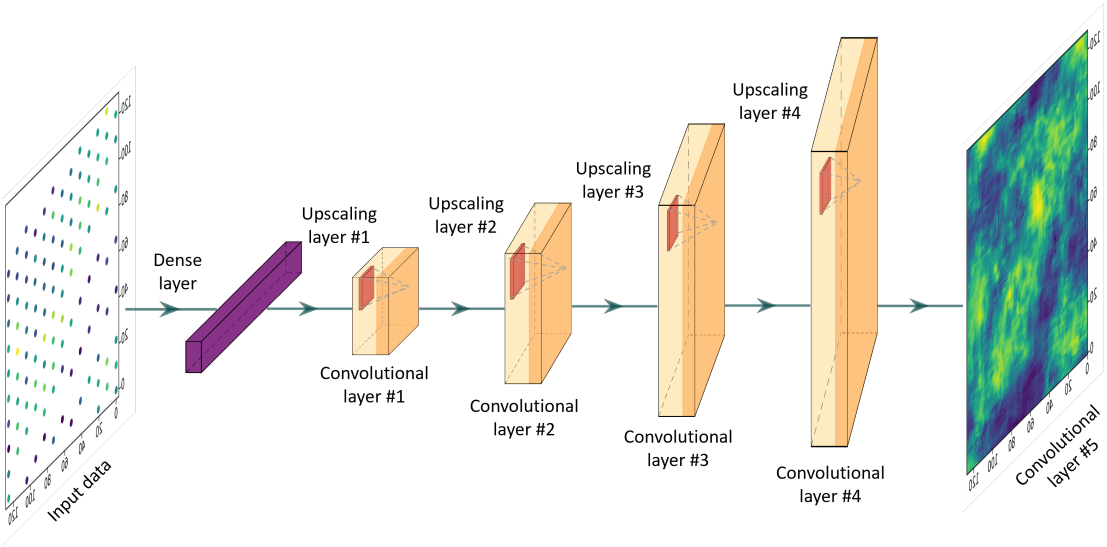


**Figure A.2:** Architecture of the proposed CNN used to predict a subsurface model estimation.

## A.2 Application of the Proposed CNN Methodology

The proposed methodology is applied to the Walker Lake to demonstrate its feasibility and assess its accuracy. Figure A.2 illustrates the sampled data from the Walker Lake dataset. Figure A.4 displays the subsurface model estimation generated by the proposed CNN model, the exhaustive data and the difference map that indicates the error between the target truth and the estimation. Figure A.5 showcases data reproduction. It is important to note that the proposed CNN technique does not reproduce the data at their locations but demonstrates a correlation of 0.871. While the mean of the sampled data is fairly reproduced by the CNN technique, it tends to overestimate. The mean of the sampled data is 0.28, whereas the mean of the data at the sampled data location predicted by the CNN is 0.3. As the Walker Lake dataset also includes exhaustive data, a comparison between the truth and the predicted value by the CNN is conducted (Figure A.6), resulting in a correlation of 0.746 and an Root Mean Square Error (RMSE) of 0.115.

**Table A.2:** The proposed architecture of CNN used to predict a subsurface model estimation.

| Layer type | Output shape | Number of parameters involved |
|---|---|---|
| Fully connected dense layer | (1024) | 482304 |
| Reshape | (8,8,16) | 0 |
| 2D upsampling | (16,16,16) | 0 |
| 2D convolution | (16,16,16) | 1040 |
| Activation function (ReLU) | (16,16,16) | 0 |
| 2D upsampling | (32,32,16) | 0 |
| 2D convolution | (32,32,16) | 1040 |
| Activation function (ReLU) | (32,32,16) | 0 |
| 2D upsampling | (64,64,16) | 0 |
| 2D convolution | (64,64,8) | 520 |
| Activation function (ReLU) | (64,64,8) | 0 |
| 2D upsampling | (128,128,8) | 0 |
| 2D convolution | (128,128,4) | 132 |
| Activation function (Sigmoid) | (128,128,4) | 0 |
| 2D convolution | (128,128,1) | 17 |
| Output | (128,128,1) | |

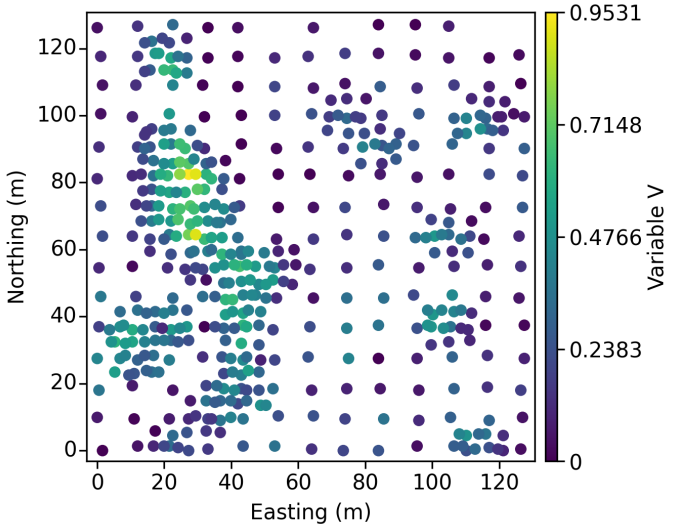


**Figure A.3:** Walker Lake dataset.

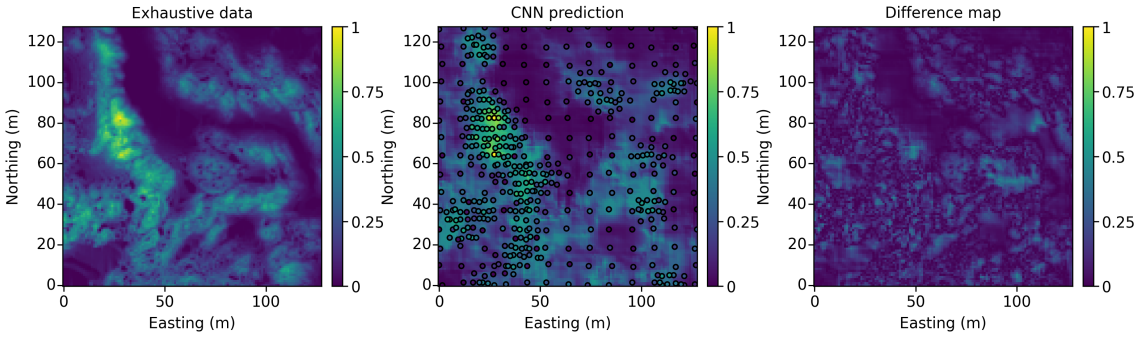


**Figure A.4:** Subsurface model estimation using CNN compared to the truth and error map.

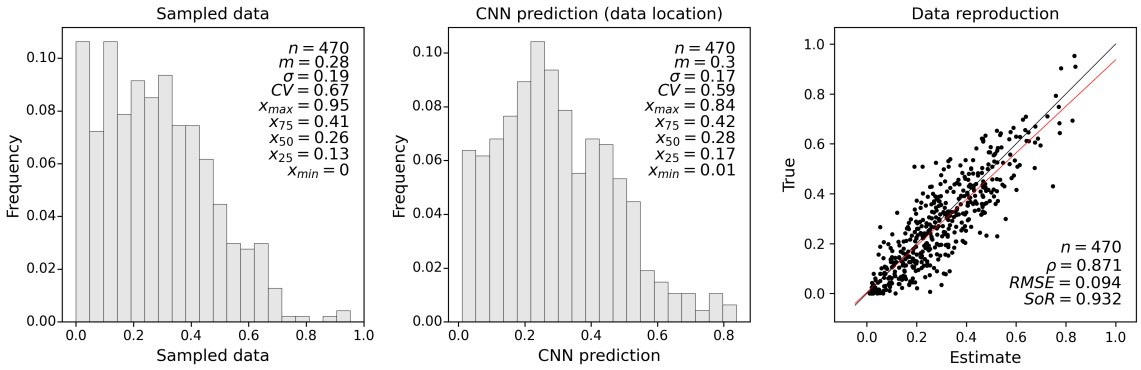


**Figure A.5:** Accessing data reproduction using the CNN estimate.

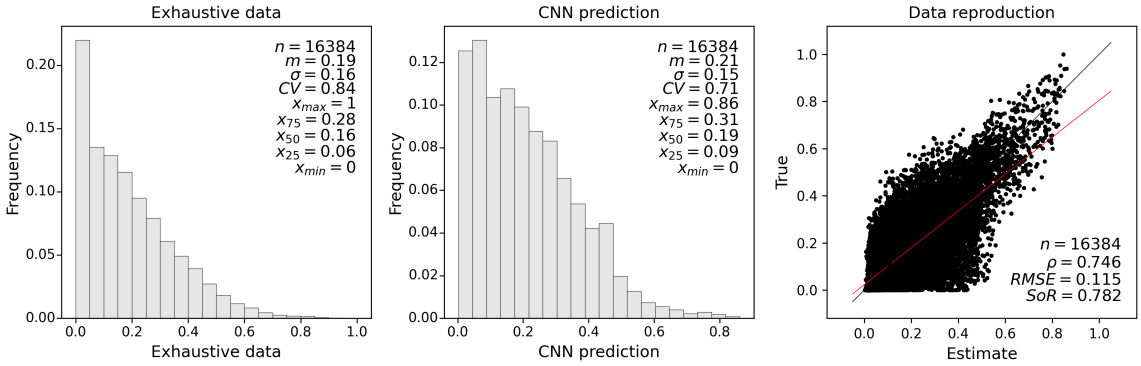


**Figure A.6:** Comparison of truth and CNN predictions.

## A.3 Summary

A methodology for automating subsurface model estimation using CNNs is presented. The objective of this technique is to reduce parameter selection and eliminate the need for variogram calculations, thereby minimizing modeling time. The methodology consists of two main steps: generating training data and training a CNN model. To demonstrate the practicality and accuracy of the proposed methodology, it is applied to the Walker Lake dataset. The CNN-generated subsurface model estimation is compared to the exhaustive data, and the difference between the estimated values and the truth is evaluated. The results showed a correlation of 0.746 and an RMSE of 0.115. However, the main limitation of this workflow is that it does not reproduce the data.

## Appendix B

# Appendix B: Application of the CNN-Based Approach on a 3D Synthetic Dataset

Appendix B presents a CNN-based method for automatically predicting the azimuth of the major direction of contunity and computing the experimental variogram of a 3-D dataset. A synthetic dataset using SGS is used to demonstrate the workflow. The 3-D model is sliced into 2-D layers along the vertical direction (stratigraphic layers), and the pre-trained 2-D CNN model predicts the azimuth and the variograms for each sliced layer. The average variogram is then selected as the most representative. The vertical variogram is calculated using a semi-automatic program. This approach uses the pre-trained CNNs implemented in this research, making it applicable to 3-D datasets without the requirement of training new CNN models or adding additional training data. The accuracy of the approach is evaluated by comparing the calculated variograms with the true variograms.

## B.1 Methodology

The steps of the proposed method are as follows:

- Data set creation: A three-dimensional dataset is simulated using SGS with an extension of 100 x 100 x 100 nodes. The dataset is sampled at 5-unit spacing in the X and Y directions and 1-unit spacing in the vertical direction. The simulation is based on a one-structure spherical variogram with a major direction of continuity of 160 degrees, a range of 30 in the major direction, a range of 20 in the minor direction, and a nugget effect of 0.01. Figure B.1 represents the generated unconditional 3-D model.
- Slice the 3-D model: The sampled 3-D model is initially sliced along the vertical direction into 100 horizontal 2-D models. Each of these slices represents a distinct 2-D layer of the data.
- Azimuth and variogram prediction for horizontal directions: For each sampled slice, inverse distance is applied. The generated inverse distance estimates for each layer are then used to predict the azimuth and experimental variogram in both the major and minor horizontal directions using respectively CNN-A and CNN-2.

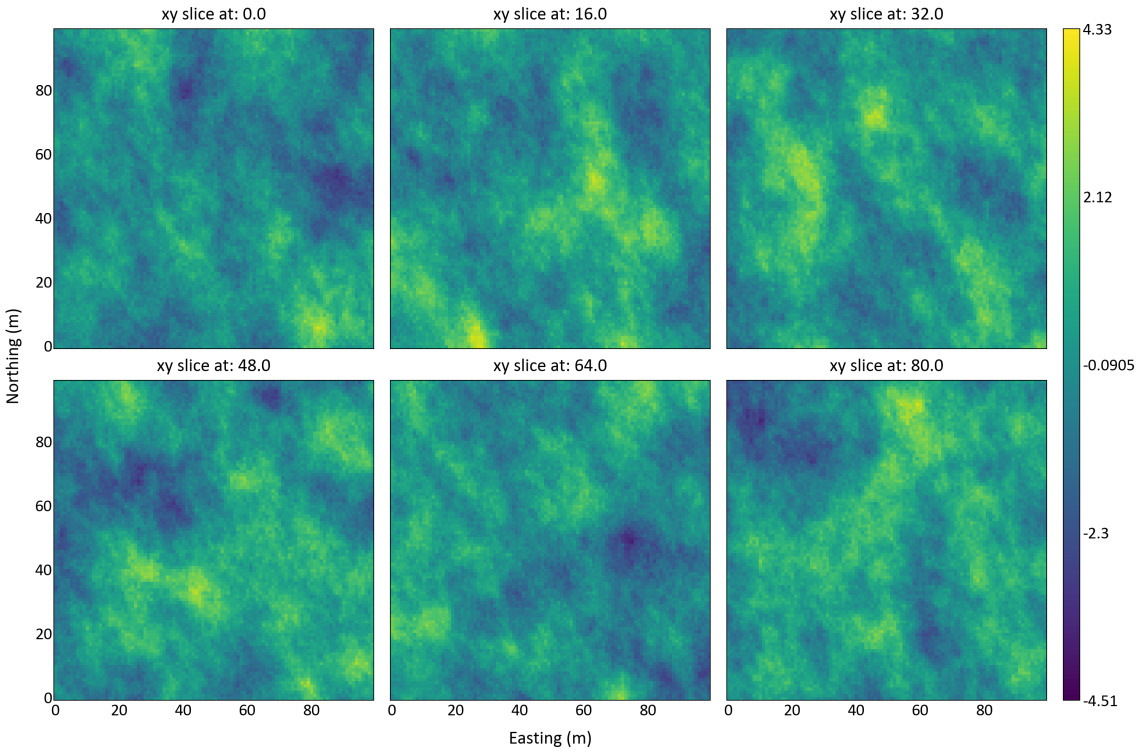


**Figure B.1:** Unconditional 3-D model generated using SGS. The model represents a synthetic dataset with a size of 100 x 100 x 100 nodes, based on a one-structure spherical variogram, characterized by a major direction of continuity of 160 degrees, a range of 30 units in the major direction, a range of 20 units in the minor direction, and a nugget effect of 0.01.

- Obtaining the representative horizontal variogram: To obtain the most representative horizontal variogram for both major and minor directions, the average of the predicted variograms for all the layers is computed, resulting in the most representative variogram.

- Vertical variogram calculation: The vertical variogram can be easily calculated using Varcalc (Deutsch and Journel, 1998) since the data is sampled at one unit spacings in the vertical direction. The densely sampled data can obtain a stable variogram in the vertical direction.

- Comparison with the true variograms: The horizontal variograms in both major and minor directions are automatically calculated using CNN-2. These calculated variograms are then compared to the true variograms obtained from the exhaustive data. Since the data in this case is synthetically generated, there is access to the truth. Additionally, the calculated variograms are also compared to the sampled variogram.

Using the pre-trained models from this research to predict variograms in 2-D, this approach is extended to 3-D datasets without training a 3-D CNN, thus eliminating the need to augment the number of training data with new components such as the vertical variogram.

## B.2 Results

The azimuth of the major direction of each layer is predicted in 2-D using the pre-trained CNN-A model proposed in this thesis. In Figure B.2, the histogram displays the distribution of the 100 predicted azimuths, which correspond to the predicted azimuths of each layer extracted from the 3-D model. The average of all the predicted azimuths is computed, resulting in a value of 159.97. The 3-D synthetic model is generated with a major direction of continuity set at 160 degrees. The range of predictions varies from a minimum of 137.84 to a maximum of 178.05.

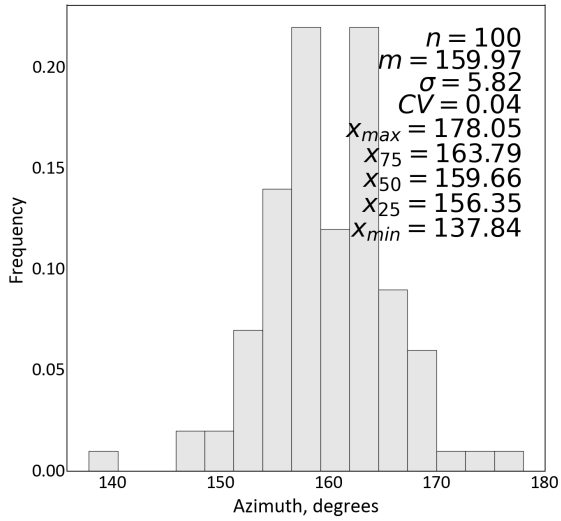


**Figure B.2:** Distribution of the predicted azimuth for 100 layers Using CNN-A. The histogram displays the values of the predicted azimuths, with an average of 159.97. The 3D synthetic model is generated with a major direction of continuity set at 160 degrees.

The variograms of each layer are predicted in 2-D, considering both major and minor horizontal directions of continuity, using the pre-trained CNN-2 model implemented in this thesis. Figures B.3 and B.4 illustrate the results of 20 out of the 100 predicted experimental variograms in the major and minor directions, respectively. These 100 variograms correspond to the predicted variograms of each layer. To obtain the most representative variogram, the average of all the predicted variograms is computed. Figures B.5 and B.6 present respectively the results of the average generated variogram from the CNN prediction, compared to the variogram of the truth, both in the major and minor directions of continuity. Additionally, the vertical experimental variogram is calculated (Figure B.7).

## B.3 Summary

Appendix B introduces a CNN-based method for predicting the azimuth of the major direction and computing the experimental variogram of a 3-D dataset. It involves generating a synthetic dataset using SGS, slicing it into 2-D layers, and using a pre-trained CNN model to predict variograms for

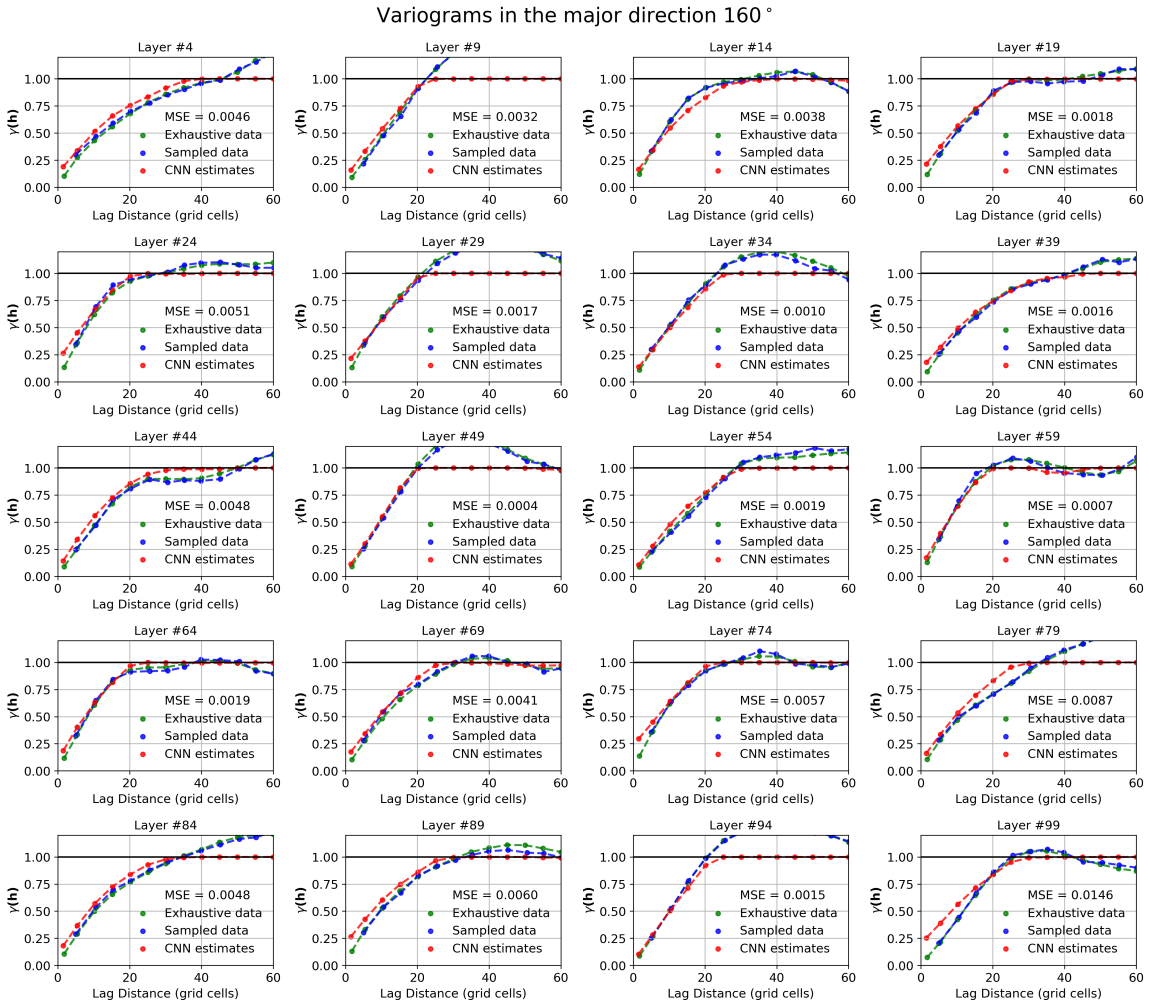


**Figure B.3:** Results of the CNN predicted variograms for 20 layers in the major direction. For display purposes, only 20 variograms are plotted, although 100 variograms are estimated. The plotted variograms are selected by taking one variogram for every 5 layers, starting from layer 4 and ending with layer 99.

each layer. The azimuth and the average variogram are then selected as the most representative ones. The approach offers an automatic way to calculate variograms in horizontal directions without training a new 3-D CNN.

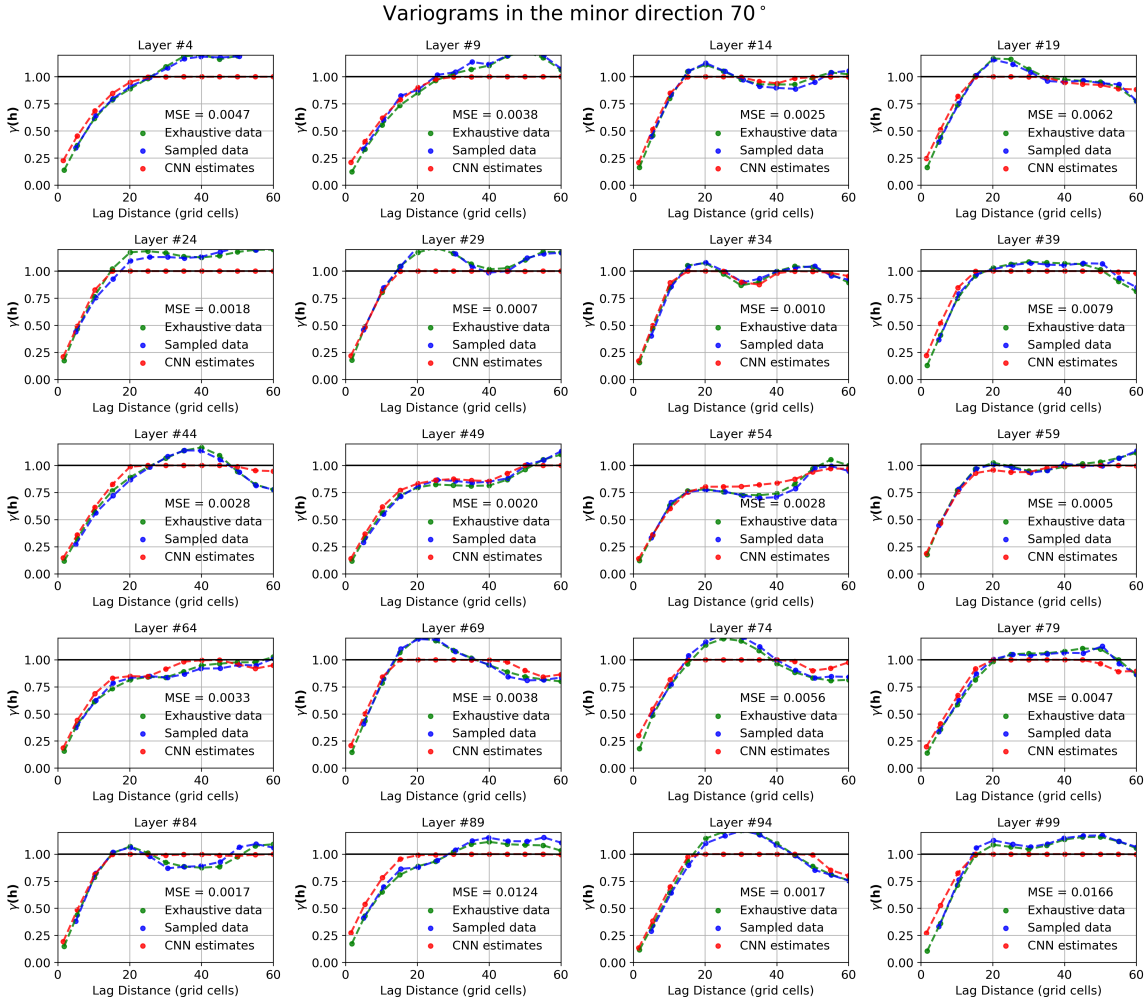


**Figure B.4:** Results of the CNN predicted variograms for 20 layers in the minor direction.

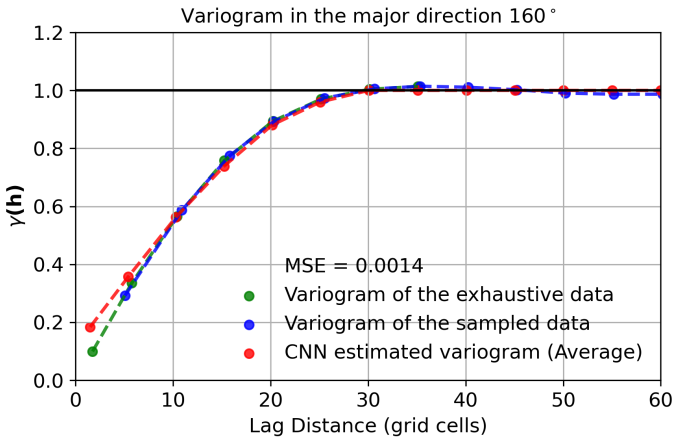


**Figure B.5:** Comparison of the average CNN predicted variogram and true variogram in the major direction.

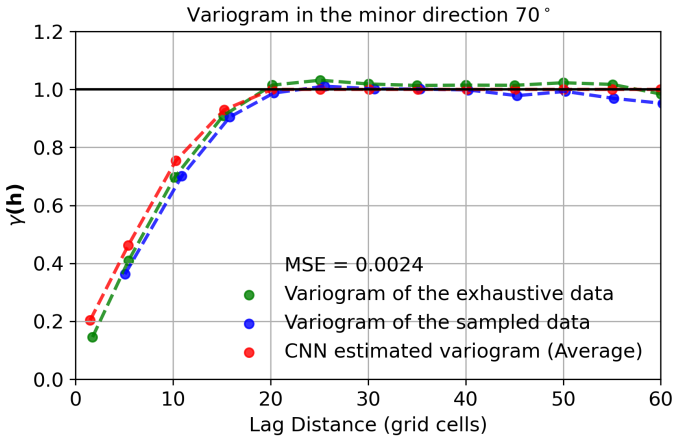


**Figure B.6:** Comparison of the average CNN predicted variogram and true variogram in the minor direction.

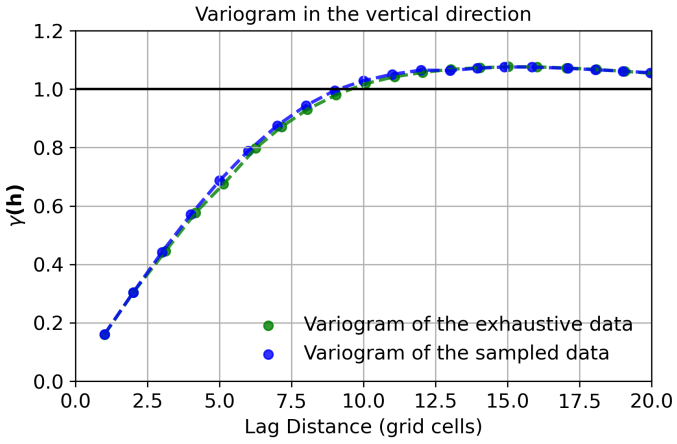


**Figure B.7:** Vertical experimental variogram.

## Appendix C

# Appendix C: Code Availability

The notebooks used to generate the results are available in the GitHub repository:

Generating training data and training a CNN:

https://github.com/Karim-Mokdad/Automatic-Variogram-Inference-Using-Pre-Trained-Convolutional-Neural-Networks

Testing CNN with the Walker Lake dataset:

https://github.com/Karim-Mokdad/Testing-CNN-with-the-Walker-Lake-dataset