An Uncertainty Estimation Model for Health Signal Prediction

Li Rong Wang[a], Thomas C. Henderson[b], Yew Soon Ong[a,c], Yih Yng Ng[d,e,f], Xiuyi Fan[a,f,*]

[a]*School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Ave, #32 Block N4 #02a, Singapore, 639798, Singapore, Singapore*
[b]*School of Computing, University of Utah, Merrill Engineering, 50 Central Campus Dr, Salt Lake City, 84112, Utah, United States*
[c]*Agency for Science, Technology and Research (A*STAR), 1 Fusionopolis Way, #20-10 Connexis, Singapore, 138632, Singapore, Singapore*
[d]*Tan Tock Seng Hospital, 11 Jln Tan Tock Seng, Singapore, 308433, Singapore, Singapore*
[e]*Ng Teng Fong Centre for Healthcare Innovation (CHI), 18 Jln Tan Tock Seng, Singapore, 308443, Singapore, Singapore*
[f]*Lee Kong Chian School of Medicine, 11 Mandalay Rd, #17-01, Singapore, 308232, Singapore, Singapore*

Abstract

Deep learning shows promise in leveraging health signal data for assisted living and preventive healthcare. However, deep learning models often exhibit overconfidence in predictions, necessitating the use of uncertainty estimation methods for accurate confidence assessment. Existing approaches for uncertainty estimation frequently suffer from poor usability or high computational complexity. In this study, we introduce the Reconstruction Uncertainty Estimate (RUE), a distributional uncertainty estimation method inspired by autoencoders, which addresses these limitations. Derived from any trained model through a two-step process – 1) training a separate multilayer perceptron as a decoder and 2) calculating uncertainty based on the reconstruction error of the decoder – RUE offers an easy-to-implement and computationally efficient solution to support point-of-care decision making. We also propose non-parametric and parametric methods to compute prediction intervals from RUE. Evaluation against Monte-Carlo dropout and the Gaussian Pro-

*Corresponding author. E-mail address: xyfan@ntu.edu.sg

cess Regressor on a health signal dataset from intensive care unit patients demonstrates RUE's superiority, showcasing high correlation with absolute error and the lowest area under the risk-coverage curve value. RUE-derived prediction intervals consistently outperform alternatives across key metrics. The robustness of RUE highlights its potential to enhance the reliability of deep learning applications in healthcare, paving the way for more informed decision-making.

Keywords: Uncertainty estimation, prediction interval, machine learning, neural networks

1. Introduction

Artificial intelligence holds significant potential in the field of healthcare, particularly with the exponential increase of health data. A substantial portion of this data comes in the form of health signals, such as vital sign readings. Therefore, to fully harness the benefits of this data, it is crucial to develop improved methods for extracting insights from health signals.

The integration of deep learning into health signal prediction has brought about notable advancements in healthcare, particularly in two key areas: 1) Living assistance, and 2) Early detection of health risks. Health signal prediction demonstrates promise in enhancing the quality of life for individuals with disabilities. Utilizing classical machine learning algorithms like Support Vector Machines (SVM) and random forest classifiers on signals from accelerometers and gyroscopes allows for the recognition of users' hand gestures, facilitating hands-free control of communication gadgets [1]. Another notable example would be the application of sparse Bayesian classifiers to analyze radar Doppler time-frequency signatures which enables the detection of falls, facilitating prompt assistance and minimizing complications from the fall, especially in the elderly [2]. Beyond fall detection, health signals have also shown promise in enhancing the quality of life for patients who are unable to sense bladder fullness, achieved through predicting bladder volume from afferent neural activity [2].

In addition to assisting in daily living activities, health signal prediction plays a pivotal role in preventive healthcare. For example, health signals have further been employed in predicting future epileptic seizures [2] and anticipating the onset of mental health crises [3]. Health signals are also employed in gait analysis [4]. Several studies have investigated the detection of

gait disorders and asymmetry using kinematics, kinetics, or neuromuscular signals, employing support vector machines (SVMs) with promising results. Early detection of gait disorders and asymmetry can enable medical professionals to intervene with protective measures, reducing the risk of fall-related injuries, especially in the elderly. These applications underscore the transformative potential of health signal prediction in improving patient outcomes and overall healthcare delivery.

Traditional methods for extracting insights from health signals involve time-series feature extraction, followed by the application of conventional machine learning techniques, such as SVM classifiers (as mentioned earlier), for downstream tasks. However, this approach is time-consuming and less suited for the healthcare domain, given the non-stationary nature of health signals [1]. Consequently, there has been a shift towards the development of deep learning methods that automate the feature extraction process to address the limitations of the classical approach. A notable example is the application of convolutional neural networks for the detection of myocardial infarctions from ECG signals [5].

However, the adoption of deep learning techniques has led to increasingly complex and unexplainable models. Moreover, recent advancements in neural network architectures have contributed to heightened model miscalibration [6], resulting in overconfident classification predictions. In this context, overconfidence refers to the tendency of the model to produce prediction probabilities that overstate its accuracy. For instance, predicting a patient has cancer with a 90% probability when the model's actual prediction accuracy for that patient is less than 90%. This is particularly concerning, as such overconfidence may lead to catastrophic consequences when models make predictions in scenarios for which they were not trained, resulting in erratic outcomes. In the medical context, where decisions are critical and often time-sensitive, some even carrying life-or-death implications, overconfident predictions can have profound consequences. Adding to the concerns, traditional neural networks applied to regression problems lack any indication of model confidence. To address the limitations of overconfident classification probability and provide an uncertainty score for neural network regressors, various uncertainty quantification methods have been developed. Well-known techniques for uncertainty estimation in neural networks include Bayesian Neural Networks (BNNs) [7], Monte Carlo dropout (MC Dropout) [8], and deep ensembles [9]. Beyond neural network techniques, the Gaussian Process Regressor (GPR) [10] stands out as a classical machine learning

model that not only provides reliable predictions but also generates a robust uncertainty estimate.

BNNs distinguish themselves from classical neural networks through the representation of their weights. In BNNs, weights are not treated as fixed values; instead, they are modeled as probability distributions known as posterior distributions. This modeling reflects the uncertainty associated with the weights given the dataset and the predicted input. During the inference phase, rather than producing a single deterministic output, a distribution of outputs is generated. Analyzing this output distribution provides an approximation of the uncertainty associated with the model's predictions.

MC Dropout is an uncertainty estimation technique applicable to any neural network model trained with dropout during training. It involves sampling predictions from a model when dropout is activated (random inactivation of nodes in a neural network) during inference. These samples approximate the output distribution from the model. The standard deviation of these prediction samples is proposed to act as an uncertainty estimate for the model, while its mean serves as the prediction.

Deep ensembles is a technique originally devised to enhance the performance of neural network models by independently training a group of models and aggregating their predictions, usually by taking the mean. Beyond its role as a performance enhancement method, deep ensembles also excel at generating reliable uncertainty estimates. The predictions from each model in the ensemble approximate the output distribution, like in MC dropout. Similar to MC dropout, the standard deviation of predictions serves as an uncertainty estimate for the deep ensemble. In literature, deep ensembles are often noted to produce more reliable uncertainty estimates compared to MC dropout [11], and this is hypothesized to be attributed to the high correlations between predictions in MC dropout.

GPR differs from a regular regressor by defining a distribution of functions instead of deriving a single function from the dataset. The model's prediction and uncertainty are determined by the mean and covariance functions.

Each of these techniques quantifies at least one of the following three types of prediction uncertainty commonly discussed in the literature: aleatoric, epistemic, and distributional uncertainty [12, 11]. In this study, our primary focus is on quantifying distributional uncertainty.

1. Aleatoric uncertainty arises from the inherent randomness in the data (e.g., noise).

2. Epistemic uncertainty is associated with the model's uncertainty and typically diminishes as the volume of training data increases.
3. Distributional uncertainty, at times categorized under epistemic uncertainty in the literature, pertains to the uncertainty linked to predictions on samples significantly different from the training set.

However, it is important to note that each of these methods has its limitations.

- BNNs have the capability to estimate both aleatoric and epistemic uncertainty, but they are challenging to train and incur a high computational cost, even when employing variational inference. Additionally, implementing BNNs often requires significant modifications to existing neural network architectures.

- In contrast, MC Dropout and deep ensembles do not necessitate changes to the model architecture, but they are limited to estimating epistemic and distributional uncertainty. These methods generate multiple predictions during inference for each input, with the mean and variance of these predictions serving as the final prediction and uncertainty measure for the input.

- MC Dropout achieves this by producing multiple sample predictions with slight variations through dropout activation during prediction. However, the sampling process in MC Dropout can lead to increased inference times, as the inference process has to be executed multiple times per output now instead of just once. This may affect its suitability for real-time applications.

- Deep ensembles, while not significantly affecting inference time, require longer training times due to the need to train multiple predictors.

- Vanilla GPR can estimate aleatoric, epistemic, and distributional uncertainty; however, it suffers from high space and time complexity.

In summary, the discussed methods suffer from one of the following three key limitations: 1) Poor usability due to required model modifications, 2) Higher prediction or training time, and 3) High computational cost. In this work, we propose a method that addresses all the above limitations. Our method focuses on estimating the distributional uncertainty of any given

trained neural network prediction model in a single pass without requiring modifications to its model architecture or training objective. The core concept draws inspiration from anomaly detection using autoencoders. We view any neural network model as consisting of two main components: 1) the feature extractor and 2) the prediction head. In our approach, we introduce another neural network, which we refer to as the decoder. The decoder is trained to reconstruct the prediction model's input based on the output of its feature extractor. Our hypothesis is that the reconstruction error, which quantifies the difference between the input and the reconstructed input, can serve as a reliable estimate of prediction error and hence, a reliable uncertainty score (Reconstruction uncertainty estimate; RUE). This hypothesis is grounded in two key assumptions:

1. The prediction model performs poorly on unfamiliar samples, resulting in high prediction error, and vice versa.
2. The decoder reconstructs unfamiliar samples poorly, leading to high reconstruction error, and vice versa.

Nevertheless, we have observed that, in many cases, it is valuable to provide not just a single estimation but also a prediction interval (PI). To address this, we have developed two methods — non-parametric and parametric — for estimating the prediction interval for each prediction based on the uncertainty score. Both methods aim to determine the PI (with lower and upper bounds L_i and U_i) that meets this condition:

$$P(L_i \leq y_i \leq U_i \mid \rho_i) = 0.95 \tag{1}$$

The non-parametric method calculates the margin of error of the PI as the 95th percentile of prediction errors (e) among all neighboring samples, where these samples are selected based on their similar reconstruction errors (ρ). In contrast, the parametric method assumes that uncertainty scores and prediction errors follow a multivariate Gaussian distribution. It derives a PI from the estimated conditional expectation ($E[e \mid \rho]$) and covariance ($Cov[e \mid \rho]$).

The contributions of this paper are threefold:

1. It proposes a distributional uncertainty estimate, RUE, that is computationally efficient and compatible with neural networks.
2. It introduces parametric and non-parametric methods for deriving prediction intervals from RUE.

3. It conducts a comprehensive analysis of uncertainty estimation performance on a publicly available health signal dataset from intensive care unit patients, comparing RUE and RUE-derived prediction intervals with MC Dropout and GPR.

The remainder of the paper is organized as follows: Section 2 provides details of the proposed uncertainty estimation and prediction interval generation methods, including a demonstration on a sine function approximation task. In Section 3, we outline the dataset, evaluation metrics, and baseline models used in our comparative study of uncertainty estimation methods. The findings from the experiment, covering model performance, uncertainty estimation, and prediction interval performance, are also presented in this section. Section 4 summarizes the key findings from our study, discusses the limitations of our proposed method, and outlines avenues for future work.

2. Methodology

In this section, we provide an overview of the proposed methods for uncertainty estimation and prediction interval generation.

2.1. Reconstruction Uncertainty Estimate

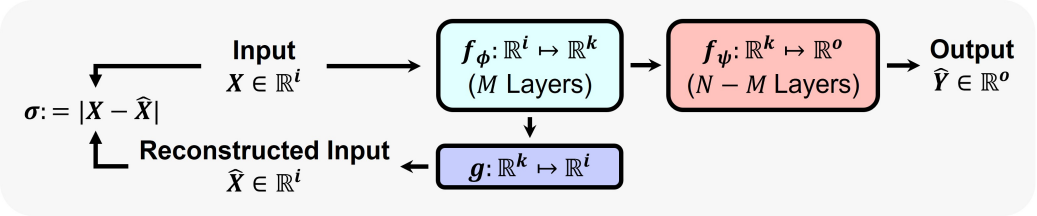


Figure 1: Model architecture for RUE.

We introduce a distributional uncertainty estimate, RUE (Reconstruction Uncertainty Estimate), inspired by anomaly detection with autoencoders. This uncertainty estimate is calculated in a single pass without any modifications to the model architecture or training objective.

With reference to Figure 1, consider a trained multi-layer perceptron (MLP) model $f_{\phi,\psi}$ with `N` layers of hidden nodes, trained with input $X \in \mathbb{R}^i$ and output $Y \in \mathbb{R}^o$. The model is parameterized by ϕ and ψ, which denote the first `M` and the last `N` $-$ `M` layers of nodes, respectively. We can interpret

f_ϕ as the feature extractor and f_ψ as the prediction head. In a regression problem, f_ϕ outputs a feature vector of size k, which is then used by f_ψ to compute continuous output values $\hat{Y}$, i.e., $f_{\phi,\psi} : \mathbb{R}^i \mapsto \mathbb{R}^o$. The MLP model $f_{\phi,\psi}$ is typically trained using gradient descent to minimize the discrepancy between Y and $\hat{Y} = (f_\psi \circ f_\phi)(X)$:

$$f_{\phi,\psi} = \arg\min_{\phi,\psi} ||Y - (f_\psi \circ f_\phi)(X)||^2. \tag{2}$$

We propose training another MLP, denoted as $g : \mathbb{R}^k \mapsto \mathbb{R}^i$ and referred to as the decoder. The decoder is trained to reconstruct the input X of the prediction model based on the output of its feature extractor, f_ϕ. g can also be trained using gradient descent to minimize the discrepancy between X and $\hat{X} = (g \circ f_\phi)(X)$:

$$g = \arg\min_{g} ||X - (g \circ f_\phi)(X)||^2. \tag{3}$$

We observe that f_ϕ and g form an autoencoder for X. Unlike a standard autoencoder model, where the encoder and the decoder are trained simultaneously, we adopt a two-step process. Initially, we train the encoder f_ϕ as part of the prediction model (Eqn. 2). Once f_ϕ is fully trained, we proceed to train g separately (Eqn. 3). This approach ensures that there is no compromise in prediction accuracy for $f_{\phi,\psi}$.

We hypothesize that for an instance $\mathbf{x} \in \mathbb{R}^i$ and its corresponding label $\mathbf{y} \in \mathbb{R}^o$, the reconstruction error of the decoder provides a reliable estimate of prediction error:

$$|\mathbf{y} - f_{\phi,\psi}(\mathbf{x})| \leftrightarrow |\mathbf{x} - (g \circ f_\phi)(\mathbf{x})|\,. \tag{4}$$

Thus, we can define

$$\sigma := |\mathbf{x} - (g \circ f_\phi)(\mathbf{x})|$$

as the *uncertainty estimate* for the prediction $f_{\phi,\psi}(\mathbf{x})$. We refer to σ as RUE (Reconstruction uncertainty estimate) in subsequent sections.

This hypothesis is grounded in two assumptions:

1. The prediction model $f_{\phi,\psi}$ performs well on inputs similar to the training set but poorly on unfamiliar inputs.
2. The decoder g, trained on the same training set, reconstructs inputs similar to the training set well, thereby achieving low reconstruction errors (small σ). For unfamiliar inputs, the decoder struggles to reconstruct the input, leading to high reconstruction errors (large σ).

The illustrated model architecture yields an uncertainty score represented by the reconstruction error σ between the actual and reconstructed inputs. σ is hypothesized to exhibit a positive correlation with the prediction error (Eqn. 4), serving as an indicator of the model's performance before the actual outcome is known. Through the use of RUE, we can identify inputs that fall outside the distribution by applying a RUE threshold. This threshold can either be user-defined or calculated as the maximum RUE in the validation set.

We demonstrate our proposed method using a toy dataset. We sample 20,000 x values from a uniform distribution, $X \sim U(-10, 30)$, covering the range from -10 to 30. Corresponding y values are generated with the equation $y = \sin(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 0.6)$. Data points where x is in the range $[-10, 0)$ and $(20, 30]$ are assigned to the test set (9921 data points), while those in the range $[0, 20]$ are randomly divided between the training and validation sets (5040 and 5039 data points, respectively). Figure 2 illustrates the model architecture used for predicting y given x on this toy dataset. Both the prediction model and the decoder were trained using the Adam optimizer with default Keras settings. Training is halted when the validation error begins to increase. The toy dataset and the model architecture used for prediction were adapted from [13].

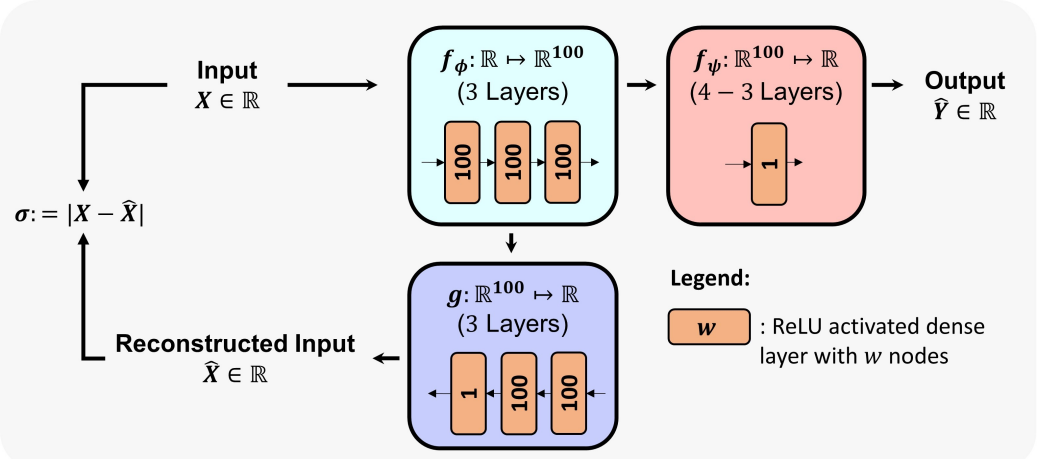


Figure 2: Model architecture for toy data set.

Figure 3 illustrates the behavior of RUE on points with varying distributional uncertainty. We note that RUE is high for samples in the test set (out-of-distribution samples) that exhibit high distributional uncertainty,

and low for samples in the training or validation set. Additionally, as x deviates further from the training and validation sets, RUE increases, aligning with the expected rise in distribution uncertainty as we move away from the training set. These observations highlight RUE's capability to quantify distributional uncertainty. Furthermore, we observe that RUE surpasses the levels seen in the training or validation sets when $x \leq 0$ or $x \geq 20$. This underscores RUE's ability to detect out-of-distribution examples, facilitated by a carefully selected threshold (such as the maximum RUE observed in the validation set).

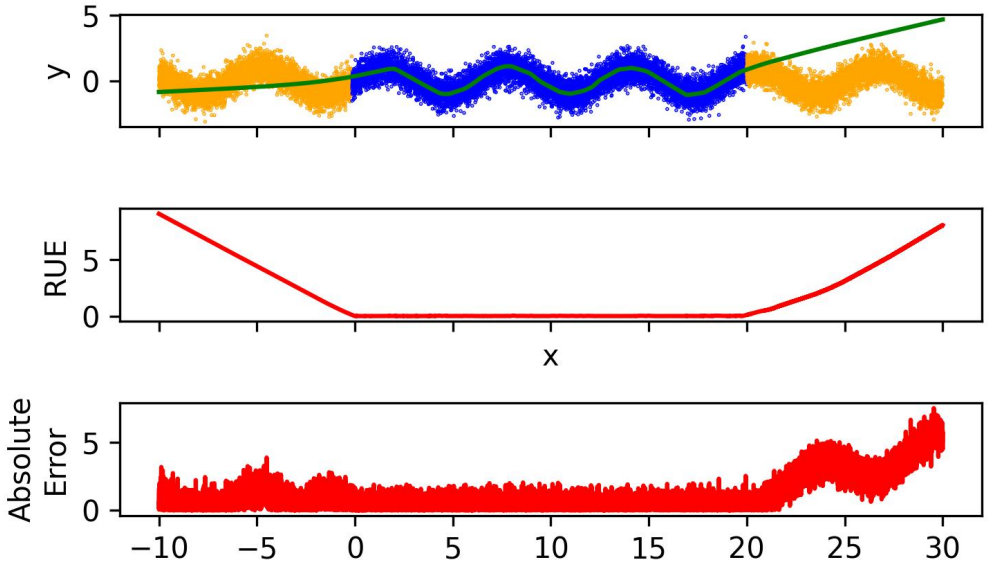


Figure 3: Model predictions, Reconstruction Uncertainty Estimation (RUE) and absolute prediction error. The green line graph represents model predictions. Orange and blue points form the test and training/validation set, respectively. Since the orange points are out of the range of the training and validation dataset, they are also considered out-of-distribution examples.

2.2. RUE-derived Prediction Intervals

In regression problems, it is valuable to offer not only a single uncertainty estimation but also a prediction interval (PI). To tackle this, we have devised two methods—non-parametric and parametric—for estimating the prediction interval for each prediction based on the reconstruction error. Both methods aim to establish a prediction interval that satisfies Equation 1.

2.2.1. Non-Parametric Prediction Intervals

The non-parametric method calculates the margin of error for a given point $x \in \mathbb{R}^i$ using the validation set of size n_v through the following procedure (also illustrated in Figure 4):

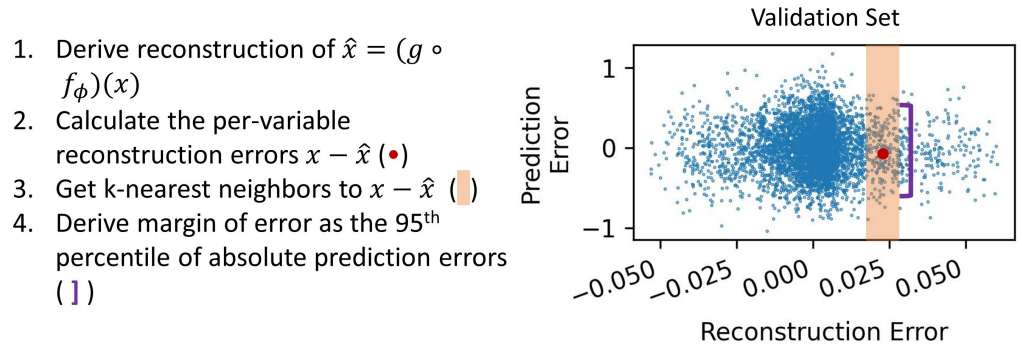


Figure 4: Non-parametric prediction interval.

1. Pass x through the feature extractor and the decoder to derive a reconstruction $\hat{x} = (g \circ f_\phi)(x)$.
2. Calculate the per-variable reconstruction error $x - \hat{x} \in \mathbb{R}^i$.
3. Find the nearest $k = \lfloor \sqrt{n_v} \rceil$ neighbors in the validation set with similar per-variable reconstruction errors.
4. Derive the absolute prediction errors of the k nearest neighbors.
5. The margin of error of x is the 95th percentile of the absolute prediction errors of the neighboring points.

In implementation, to expedite the search process in step 3, we employ a K-Dimensional Tree [14] fitted on the validation set reconstruction errors before inference. During inference, the tree is queried with x. The non-parametric method estimates the distribution of PIs conditioned on the reconstruction error through steps 3 and 4. Step 5 calculates the PI with 95% coverage from this estimated distribution. Using a percentile instead of a maximum reduces the impact of outliers in the validation set on the PI.

Figure 5 illustrates the prediction interval calculated using the non-parametric PI estimation method. We observe that the PI effectively covers the majority of the blue points (93.4% of the training set). However, it is important to note that since the PI is derived from the prediction errors of the validation

set, its coverage is limited to prediction values observed in that dataset. This limitation may result in constant PI values when the reconstruction error extends beyond the values present in the validation set, such as when $x \geq 20$ or $x \leq 0$.

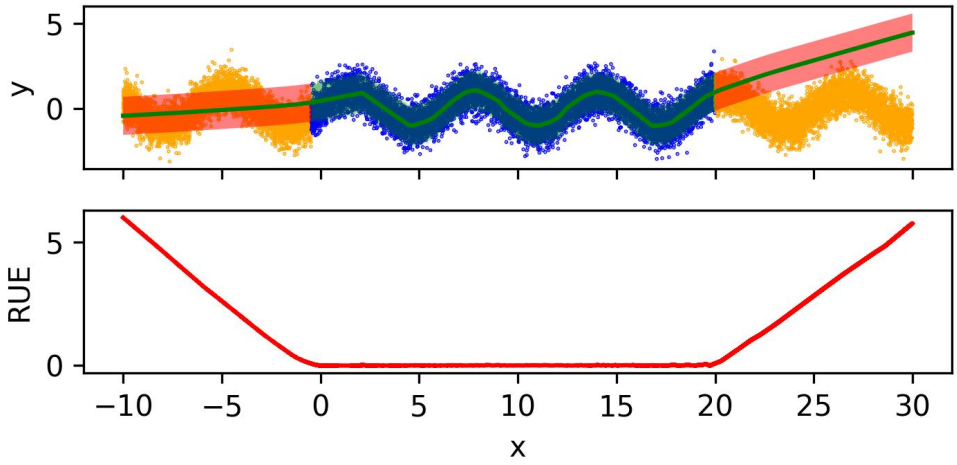


Figure 5: Non-parametric prediction interval on the toy dataset. The green line graph represents the model's predictions. Orange and blue points form the test and training/validation set, respectively. Since the orange points are out of the range of the training and validation dataset, they are also considered out-of-distribution examples. The shaded regions depict the prediction interval of the model; green regions are considered within the distribution (Reconstruction Uncertainty Estimate, RUE $\leq$ maximum RUE in the validation set), while red regions are considered out of distribution (indicating high distributional uncertainty).

2.2.2. Parametric Prediction Intervals

The parametric method assumes that both reconstruction errors ($\rho = x - \hat{x}$) and prediction errors ($e = y - \hat{y}$) follow a multivariate Gaussian distribution:

$$\begin{bmatrix} \rho \\ e \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_\rho \\ \mu_e \end{bmatrix}, \begin{bmatrix} \Sigma_\rho^2 & \Sigma_{\rho,e} \\ \Sigma_{e,\rho} & \Sigma_e^2 \end{bmatrix} \right) \tag{5}$$

Where μ_ρ and μ_e are the mean vectors of ρ and e, Σ_ρ^2 and Σ_e^2 are the covariance matrices of ρ and e, $\Sigma_{e,\rho}$ is the covariance matrix between e and ρ, and $\mathbf{\Sigma}_{\rho,e}$ is its transpose.

With this assumption, we can directly calculate the distribution of prediction errors conditioned on the reconstruction errors, $\mathcal{N}(\mu_{e|\rho}, \Sigma_{e|\rho})$. The mean $\mu_{e|\rho}$ and covariance matrix $\Sigma_{e|\rho}$ of this conditional distribution, given ρ, are calculated using the following two equations:

$$\boldsymbol{\mu}_{e|\rho} = \boldsymbol{\mu}_e + \boldsymbol{\Sigma}_{e,\rho}\boldsymbol{\Sigma}_{\rho}^{-1}(\rho - \boldsymbol{\mu}_\rho) \tag{6}$$

$$\boldsymbol{\Sigma}_{e|\rho} = \boldsymbol{\Sigma}_e - \boldsymbol{\Sigma}_{e,\rho}\boldsymbol{\Sigma}_{\rho}^{-1}\boldsymbol{\Sigma}_{\rho,e} \tag{7}$$

We derive a PI with a 95% coverage from the aforementioned conditional Gaussian distribution as follows:

$$F^{-1}(0.975|\mu_{e|\rho}, \Sigma_{e|\rho}) \tag{8}$$

Here, F^{-1} denotes the inverse cumulative distribution function operator of the conditional distribution. In practice, the mean vector and covariance matrix of the multivariate Gaussian are estimated using the sample mean and sample covariance of the reconstruction and prediction errors from the validation set:

$$\begin{bmatrix} \bar{\boldsymbol{\mu}_{\boldsymbol{\rho}}} \\ \boldsymbol{\mu_e} \end{bmatrix} = \frac{1}{n_v}\sum_{i=1}^{n_v}\begin{bmatrix} \rho_i \\ e_i \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} \Sigma_\rho^2 & \bar{\boldsymbol{\Sigma}}_{\rho,e} \\ \Sigma_{e,\rho} & \Sigma_e^2 \end{bmatrix} = \frac{1}{n_v - 1}\sum_{i=1}^{n_v}(\begin{bmatrix} \rho_i \\ e_i \end{bmatrix} - \begin{bmatrix} \mu_\rho \\ \mu_e \end{bmatrix})(\begin{bmatrix} \rho_i \\ e_i \end{bmatrix} - \begin{bmatrix} \mu_\rho \\ \mu_e \end{bmatrix})^T \tag{10}$$

Here, ρ_i and e_i represent vectors of per-input reconstruction error and per-output prediction error for validation sample i. This method assumes that the reconstruction and prediction errors follow a multivariate Gaussian distribution. While we acknowledge that this assumption may not always hold, we propose the mean log-likelihood (Eqn. 11) as a measure of prediction interval reliability.

$$\text{Average Log-Likelihood} = \frac{1}{N}\sum_{i=1}^{N}\log \hat{L}(\bar{\mu}, \bar{\Sigma}; x_i) \tag{11}$$

Here, $\hat{L}(\bar{\mu}, \bar{\Sigma}; x_i)$ represents the likelihood of x_i, derived from the probability density function of a multivariate Gaussian. The parameter N refers to the number of data points. A higher mean log-likelihood indicates that the

assumption is more likely to be true, and vice versa. This value can be compared with the mean log-likelihood of a dataset (of the same size) sampled from a multivariate Gaussian with identical mean and covariance matrix to assess the validity of the assumption.

Figure 6 illustrates the results of the parametric prediction interval method on the toy dataset. The parametric prediction interval covers 94.6% of data points in the training set. Expanding beyond the training set, we observe that the prediction interval increases to cover 98.9% of data points in the test set. In contrast to the non-parametric method, the parametric method does not have an upper bound for prediction interval width. The highly positive mean log-likelihood (5.593, compared with the simulated dataset) leads us to conclude that, in this toy dataset, the assumption that the reconstruction and prediction errors follow a multivariate Gaussian is accurate. This results in a reliable prediction interval that covers most of the out-of-distribution data points.

2.3. Summary of Proposed Methods

In summary, we introduce the Reconstruction Uncertainty Estimate (RUE) as a metric of distributional uncertainty. Leveraging a trained model, we employ a separate network, the decoder, to reconstruct the input based on the model's intermediate features. RUE is computed as the absolute difference between the reconstructed input and the actual input. We showcase RUE's effectiveness in identifying out-of-distribution data points with a carefully chosen threshold. Furthermore, we extend the utility of RUE by developing parametric and non-parametric methods for constructing prediction intervals. In the non-parametric approach, the prediction interval is estimated from the prediction errors of validation data points with similar reconstruction errors. Conversely, the parametric method involves fitting a multivariate Gaussian distribution to the reconstructed and prediction errors. The resulting distribution of prediction errors, conditioned on reconstructed errors, is then used to construct the prediction interval.

3. Experiment

In this section, we outline the experimental setup, including the dataset, evaluation metrics, and baseline models. We then present results comparing RUE and RUE-derived prediction intervals with existing methods.

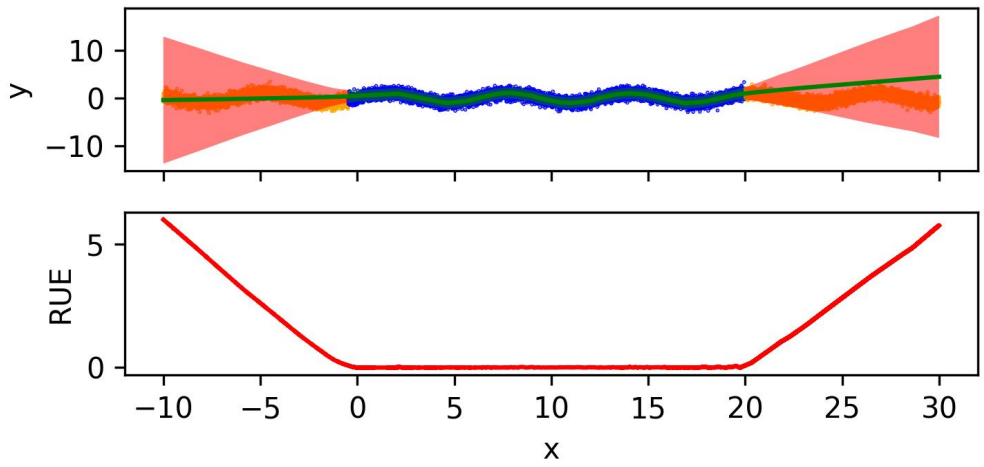


Figure 6: Parametric prediction interval on the toy dataset. The green line graph depicts the model's predictions. Orange and blue points form the test and training/validation set, respectively. Since the orange points are out of the range of the training and validation dataset, they are also considered out-of-distribution examples. The shaded regions represent the prediction interval of the model; regions in green are considered within distribution (Reconstruction uncertainty estimate, RUE $\leq$ maximum RUE in the validation set), while regions in red are considered out of distribution (high distributional uncertainty). The mean log-likelihood on the validation set is 5.593, and the mean log-likelihood of a simulated Gaussian dataset is 5.602.

3.1. MIMIC Dataset

To evaluate the effectiveness of RUE and both RUE-derived PI methods in the time series prediction of medical signals, we obtained medical signal data from the MIMIC database [15]. The dataset comprises signals from 121 intensive care unit (ICU) patients, each with a potentially different set of signals. The following preprocessing steps were applied to the dataset:

1. Select patients with all six of the following signals (patient states): ["ABPdias (mmHg)", "ABPmean (mmHg)", "ABPsys (mmHg)", "HR (bpm)", "RESP (bpm)", "SpO2 (%)"]. A total of 57 patients were selected.
2. Randomly assign 70%, 10%, and 20% of patients to the training, validation, and test sets, respectively. The prediction model will be trained on the training set, its parameters tuned on the validation set, and its performance evaluated on the test set. In the "Mini" dataset, the training set is further downsampled to 10 patients.
3. Remove all feature values less than zero or blood pressure values greater than 250.
4. Z-normalize the signals using the mean and standard deviation derived from the training set.
5. To reduce the signal frequency, downsample the signal to one sample per minute by calculating the mean and standard deviation of signals within each minute.
6. Drop rows with missing values.

"ABPdias (mmHg)", "ABPmean (mmHg)", "ABPsys (mmHg)" signals correspond to the diastolic, mean, and systolic arterial blood pressure of the patient, while "HR (bpm)", "RESP (bpm)", "SpO2 (%)" correspond to periodic measurements of the heart rate, respiration rate, and oxygen saturation of the patient.

With the preprocessed data, we trained the prediction model, denoted as $f_{\phi,\psi}(\mathbf{x})$, to forecast all six patient's states for one minute into the future (y_{t+1}), two minutes into the future (y_{t+2}), and three minutes into the future (y_{t+3}), utilizing the patient's signals from the past 5 minutes (x_{t-5} to x_t). Figure 7 illustrates the model architecture employed for prediction and reconstruction. We fine-tuned the width of each model ($w \in \{64, 128, 256, 512, 1024, 1536\}$) through grid-search, using validation loss as the performance metric. L2 regularization ($l2 = 0.01$) on hidden layers and early stopping ($patience = 20$) were implemented to mitigate overfitting.

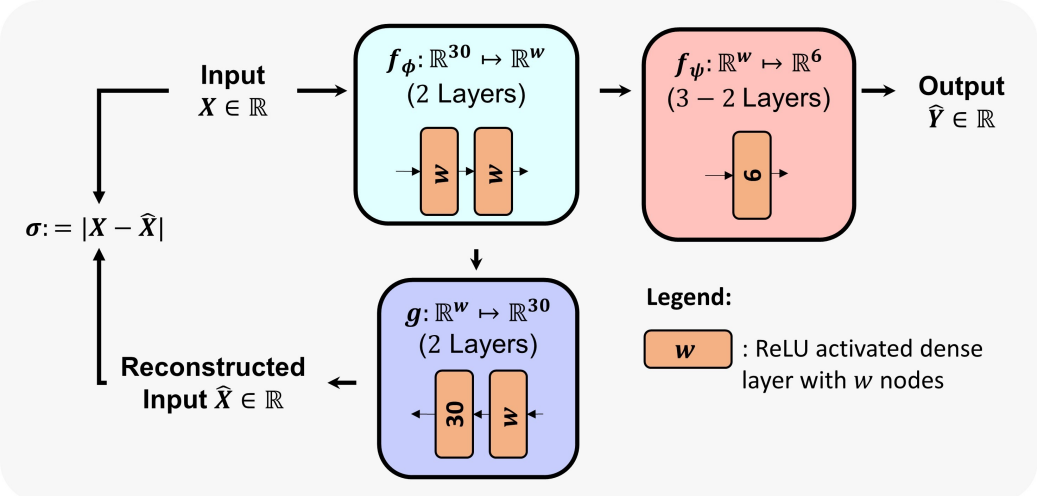


Figure 7: Model architecture for MIMIC data set.

3.2. Baseline Models

We compare RUE with uncertainty estimates and prediction intervals produced by two commonly used uncertainty estimation methods: Gaussian process regressor (GPR) [10] and Monte Carlo (MC) dropout [8].

The GPR models the underlying function of the data as a Gaussian process, representing a distribution of functions that could fit the data. It is characterized by its mean and standard deviation (Std), which serve as predictions and uncertainty estimates of the model. In our experiments, we used Scikit-learn's implementation of vanilla GPR with the default radial basis function kernel. Since vanilla GPR suffers from high time complexity ($\mathcal{O}(n^3)$) [16] and memory consumption ($\mathcal{O}(n^2)$) [17], we reduced the size of the training set to ten patients in the "Mini" data set.

Unlike GPR, MC dropout can be applied to neural networks trained with dropout without modifications to the model architecture. It proposes the activation of dropout during inference to generate multiple varied predictions for each input. The sample mean and variance of these predictions serve as the final prediction and uncertainty measure for the input. In implementing MC dropout for our experiments, we applied a dropout rate of 0.20 to the hidden layer of the prediction model. During inference, we generated 10 predictions for each input with dropout activated. The dropout rate and sample size were determined with reference to experiments conducted in [8] estimating uncertainty in regression problems.

For both MC dropout and GPR, we derived the prediction interval (PI)'s margin of error with 95% confidence using the standard deviation with Equation 12:

$$\text{Margin of Error} = 1.96 * z_{0.975} \tag{12}$$

3.3. Evaluation Metrics

For each of the three prediction tasks (predicting y_{t+1}, y_{t+2} and y_{t+3}), we assess the performance of all uncertainty estimates and prediction interval generation methods on the test set. We evaluate the uncertainty estimates using two metrics: the correlation between the uncertainty estimate and true prediction error, and the area under the risk curve (AURC) [18]. A positive correlation indicates a reliable uncertainty estimate. The AURC is computed by summing thresholded prediction errors, with 100 thresholds generated from equally spaced percentile values (0% to 100%). A lower AURC suggests a better uncertainty estimate. The AURC is known to be invariant to changes in the prediction model, allowing us to compare uncertainty estimates from models of varying predictive performance.

Prediction interval quality was assessed using three metrics derived from the prediction interval cost function introduced in [19]:

1. Prediction interval coverage probability (PICP): The proportion of points within the prediction interval (Eqn. 13); a higher percentage is better.

$$\text{PICP} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{y_i \in [L_i, U_i]} \tag{13}$$

   Here, n represents the total number of points, y_i denotes the ith ground-truth output, and L_i and U_i stand for the lower and upper bounds of the prediction interval. The PICP can be further converted into a penalty value known as the Coverage Penalty (CP), as defined in Equation 14:

$$\text{CP} = (0.951 - PICP)^2 \tag{14}$$

2. Prediction interval normalized average width (PINAW): The average normalized size of the prediction interval (Eqn. 15). When two intervals have the same PICP, the one with the smaller PINAW is considered better.

$$\text{PINAW} = \frac{1}{n \times R}\sum_{i=1}^{n}(U_i - L_i) \tag{15}$$

   Here, R is the range of the output variable.

3. Prediction interval normalized average failure distance (PINAFD): The average normalized distance of points outside the prediction interval (Eqn. 16). A smaller PINAFD is better.

$$\text{PINAFD} = \frac{\sum_{i=1}^{n} \mathbb{1}_{y_i \notin [L_i, U_i]} min(|y_i - U_i|, |L_i - y_i|)}{R \times \sum_{i=1}^{n} (\mathbb{1}_{y_i \notin [L_i, U_i]}) + \epsilon} \tag{16}$$

Here, ϵ is a small value introduced to prevent undefined values resulting from division by zero.

3.4. Experiment Results

In this section, we compare the prediction performance, reliability of uncertainty estimates, and the quality of prediction intervals among different uncertainty estimation methods. This comparison is based on the metrics discussed in Section 3.3 across all time horizons, as well as between models trained on the "Mini" and the complete training datasets.

3.4.1. Prediction Performance

Upon comparing the prediction performances of all three methods on the "Mini" dataset (Table 1), we observe that GPR exhibited the highest prediction errors among the three models. In contrast, MC dropout and direct prediction demonstrated very similar performance. It is noteworthy that, as the models forecast further into the future, prediction errors tend to increase. Additionally, when assessing the performance of models trained on the "Mini" dataset against those trained on the complete dataset, we observed that models trained with more data exhibited smaller prediction errors.

Table 1: Table showing test mean squared errors (MSE) for models trained on the "Mini" and whole dataset.

Data	Models	t+1	t+2	t+3
Mini	Direct Prediction	**0.099**	**0.134**	**0.148**
	MC Dropout	0.100	0.137	0.149
	GPR	0.491	0.535	0.557
Whole	Direct Prediction	0.089 (-10.4%)	0.121 (-9.9%)	0.131 (-11.1%)
	MC Dropout	0.09 (-10.2%)	0.121 (-11.2%)	0.132 (-11.3%)

3.4.2. Uncertainty Estimation Performance

Figure 8 illustrates the relationship between uncertainty estimates and absolute prediction errors. Table 2 compares the reliability of each uncertainty estimate. We observed that all three uncertainty estimates have a positive relationship with absolute prediction error. RUE consistently shows the strongest correlations with prediction error (0.487), followed by GPR and MC dropout standard deviations. We recognize that there is room for improvement in the correlation between RUE and absolute error. This is attributed to the fact that RUE solely focuses on distribution uncertainty, addressing only one aspect of predictive uncertainty. Subsequent studies could explore the integration of RUE with complementary methods to comprehensively capture both noise and model uncertainty. Generally, across all methods, as the time horizon increases, all uncertainty estimates become less correlated with absolute error. This observation may be explained by the increase in model uncertainty as the time horizon increases, which makes these uncertainty estimation methods less suited to quantify it. When comparing the reliability of uncertainty estimates with more training data available, we observed some drops in correlation for RUE and MC dropout standard deviation. Additionally, we noted that the change in correlation when models were trained on the whole dataset was less drastic for RUE compared to MC dropout. This suggests that RUE is more robust in the face of varying training dataset sizes compared to MC dropout.

Further comparing uncertainty estimates based on their risk-coverage curves (Figure 9) and the area under risk-coverage curves, we observed that the risk-coverage curves of MC dropout and GPR tend to follow a U-shape, while RUE exhibited a smoothly increasing logarithmic-like curve. The risk-coverage curves of RUE and GPR align with our expectations that as coverage increases, risk increases, unlike the relatively constant risk curve of MC dropout. We also noted that RUE is more reliable compared to the other two uncertainty estimates for all time horizons. Comparing their AURCs (Table 2), RUE's AURC was consistently the lowest among the three uncertainty estimates, followed by MC dropout and GPR standard deviation. The relative performances of MC dropout and GPR have switched in this metric, reflecting GPR's poorer predictive performance observed earlier in Table 1. When comparing the performance of models trained on the "Mini" and the whole dataset, we observed a reduction in AURC across both methods. We believe this reduction reflects the improved performance with more data, as

Table 2: Table showing the performance of each uncertainty estimate by comparing prediction error correlation and area under risk-coverage curve.

Data	UE	Correlation with AE			AURC		
		t+1	t+2	t+3	t+1	t+2	t+3
Mini	RUE	**0.487**	**0.439**	**0.362**	**13.624**	**16.025**	**17.384**
	MCD Std	0.117	0.168	0.149	18.271	20.377	21.342
	GPR Std	0.470	0.353	0.256	35.103	41.328	45.470
Whole	RUE	0.485 (-0.6%)	0.427 (-2.6%)	0.334 (-7.6%)	11.749 (-13.8%)	14.612 (-8.8%)	15.881 (-8.6%)
	MCD Std	0.163 (39.2%)	0.135 (-20.1%)	0.083 (-44.3%)	15.925 (-12.8%)	18.788 (-7.8%)	19.893 (-6.8%)

observed earlier (Table 1).

3.4.3. Prediction Interval Performance

Table 3 compares the prediction intervals derived from RUE, MC dropout, and GPR on three metrics: PICP, PINAW, and PINAFD. PICP measures the proportion of points within the prediction interval, PINAW measures the size of the prediction interval, and PINAFD measures the deviation of points outside the prediction interval (refer to Section 3.3 for more details). Ideally, a prediction interval should have a high PICP and low PINAW and PINAFD.

Comparing the four different prediction intervals, we observed that GPR generated prediction intervals with the highest PICP, and MC dropout produced the prediction interval with the smallest PINAW. Referring to Figure 10, we notice that GPR has the largest confidence interval; this allows GPR to achieve the higher coverage observed earlier (highest PICP). We also noticed that MC dropout has the narrowest prediction interval, consistent with the earlier observation of low PINAW.

Comparing the PINAFD, we observed that the non-parametric RUE prediction interval is comparable to MC dropout across all time horizons, with MC dropout showing greater performance on two of the three time horizons. Closely examining the equation for PINAFD (Eqn. 16) reveals that

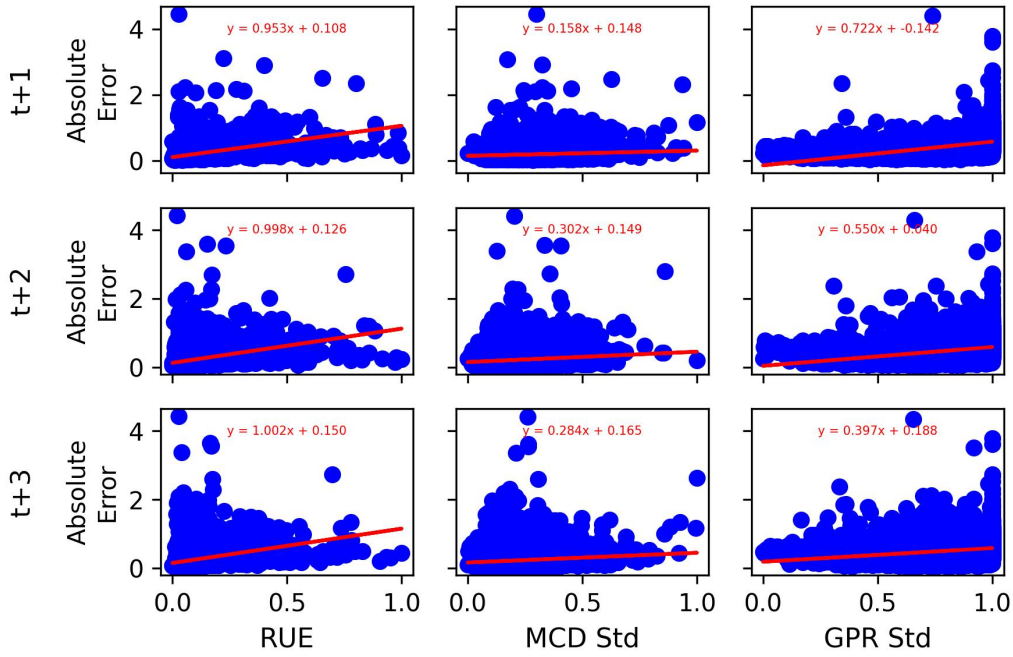


Figure 8: Relationship between uncertainty estimates and absolute prediction error for models trained on the "Mini" dataset. Uncertainty estimates have been min-max normalized for comparable visualization. The red line, accompanied by the red equation in each plot, represents the linear regression line between the uncertainty estimates and absolute error. Comparing the slopes of the scatter plots, we observe a more pronounced positive relationship with absolute error for RUE and GPR Std compared to MCD Std.

this may not indicate that MC dropout is the better prediction interval with respect to failure distance. PINAFD is derived as the normalized average distance of points outside the prediction interval. Since MC dropout has been observed to have poorer coverage (low PICP), the lower PINAFD value observed in MC dropout may instead reflect a prediction interval that fails to cover points that fall just outside the prediction boundaries. This leads to a smaller PINAFD since these points reduce the average failure distance.

The above observations highlight that these three measures of prediction interval quality need to be considered together; a good prediction interval should excel in all three aspects. Figure 11 compares all three aspects with star plots. The PICP was converted to a coverage penalty (CP) for easier visualization (refer to Section 3.3 for more details). An ideal prediction interval exhibits low CP, PINAW, and PINAFD. This is reflected on the star

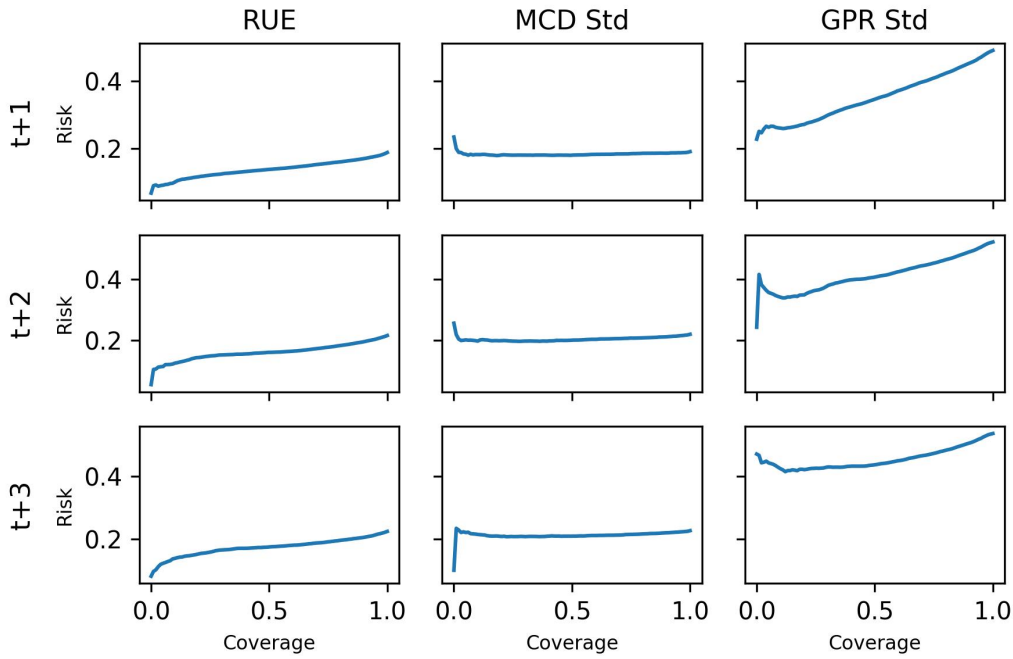


Figure 9: Risk-coverage curve for all uncertainty estimates for models trained on the "Mini" dataset.

plot as a prediction interval with the smallest triangle.

From the figure, we can clearly observe that the non-parametric RUE balances all three aspects the best, followed by the parametric RUE. MC dropout and GPR excel in different aspects; MC dropout has poor coverage but a small failure distance and a small prediction interval, whereas GPR exhibits the opposite pattern.

Upon comparing the prediction intervals across time horizons, we observed a clear increasing trend for RUE-derived prediction intervals in Table 3 and Figure 10. This observation aligns with the positive relationship between prediction error and the time predicted into the future (Table 1). However, this trend is not evident in GPR or MC dropout. This suggests that GPR and MC dropout prediction intervals may fail to capture the increasing uncertainty associated with predicting further into the future.

Upon comparing RUE and MC dropout prediction intervals trained with the "Mini" and the whole dataset, it is evident that training with more data generally improved the coverage and reduced the size of RUE-derived predic-

tion intervals. However, there appears to be an increase in failure distances. This could be a consequence of the improved coverage discussed earlier when comparing MC dropout and RUE non-parametric failure PINAFD. This suggests that RUE-derived prediction intervals can be enhanced as the prediction and decoder models are trained with more data. For MC dropout, a reduction in the prediction interval size was also observed. However, the effects of increased dataset size on its coverage or failure distances seem inconsistent. Overall, it appears that for both prediction interval generation methods, the prediction interval size reduced when the training dataset was increased. This demonstrates that both methods captured the reduction in model uncertainty when the model was trained with additional data.

4. Discussion and Conclusion

In this paper, we proposed a distributional uncertainty estimation method inspired by the autoencoder, called the Reconstruction Uncertainty Estimate (RUE). We suggest deriving RUE from any trained model with a two-step approach: 1) training a separate multilayer perceptron acting as a decoder, which reconstructs the input given the intermediate outputs of any trained model, and 2) calculating the uncertainty estimate as the reconstruction error of the decoder. We also propose non-parametric and parametric methods for deriving prediction intervals from RUE by estimating the distribution of prediction errors given reconstruction errors on the validation set.

We evaluated our uncertainty estimation methods against two baselines – Monte-Carlo dropout and the Gaussian Process Regressor – using a publicly available health signal dataset from intensive care unit patients. Our experiments demonstrated that the Gaussian Process Regressor has poorer predictive performance compared to artificial neural networks on this multivariate multi-output prediction problem. Additionally, we observed that RUE is the most reliable uncertainty estimate among the three methods, exhibiting the highest correlation with absolute error and the lowest area under the risk-coverage curve value.

RUE-derived prediction intervals also demonstrated the best performance when evaluating all three prediction interval metrics: coverage, interval size, and failure distance. Although RUE-derived prediction intervals have shown significant promise, it is important to note their dependency on the size of the validation set. Future work will explore alternative methods of deriving prediction intervals from RUE or investigate more direct approaches for

obtaining prediction intervals from a trained model.

In Section 3.4.2, we also observed that RUE does not encompass all facets of predictive uncertainty. RUE does not capture uncertainty from noise in the data or model uncertainty. Repeating our sine function simulation experiments with different noise levels (Figure 12), we observed that as noise increases, the concordance between prediction error and RUE is reduced. This suggests that to reliably quantify prediction uncertainty in all machine learning problems, RUE should be paired with aleatoric (noise) and epistemic (model) uncertainty estimation methods. Future studies could explore integrating RUE with complementary methods that capture both noise and model uncertainty.

Compared to existing uncertainty estimation methods, RUE offers significant operational advantages. Its computational efficiency during prediction, unlike methods such as Monte Carlo dropout and deep ensembles, enables real-time implementation in low-resource or edge computing environments. This facilitates integration into medical IoT devices, potentially allowing for smaller sensors. Reduced computational demands may also extend battery life and enhance overall cost-effectiveness with the use of low-powered chips, aligning with environmental sustainability goals. Beyond its operational efficiency, RUE stands out as the most reliable uncertainty estimate among the compared methods. However, certain considerations, like the need for a larger dataset for model validation, exist. Unlike Gaussian Process Regressor and Monte Carlo dropout, RUE also requires no changes to existing model architectures, allowing for seamless integration into existing neural network-based systems without costly retraining. Additionally, its faster training process, especially with extensive data, implies quicker deployment and the ability to swiftly push out new updates, marking a significant advantage in dynamic environments.

5. Declaration of Competing Interest

Declarations of interest: none

References

[1] M. Strzelecki, P. Badura, Machine Learning for Biomedical Application, Applied Sciences 12 (4) (2022) 2022. doi:10.3390/app12042022.

[2] G. Rong, A. Mendez, E. Bou Assi, B. Zhao, M. Sawan, Artificial Intelligence in Healthcare: Review and Prediction Case Studies, Engineering 6 (3) (2020) 291–301. doi:10.1016/j.eng.2019.08.015.

[3] S. Shinde, P. Rajeswari, Intelligent health risk prediction systems using machine learning: A review, International Journal of Engineering and Technology(UAE) 7 (2018) 1019–1023. doi:10.14419/ijet.v7i3.12654.

[4] P. Khera, N. Kumar, Role of machine learning in gait analysis: A review, Journal of Medical Engineering & Technology 44 (8) (2020) 441–467. doi:10.1080/03091902.2020.1822940.

[5] V. Jahmunah, E. Ng, R.-S. Tan, S. L. Oh, U. R. Acharya, Explainable detection of myocardial infarction using deep learning models with Grad-CAM technique on ECG signals, Computers in Biology and Medicine 146 (2022) 105550. doi:10.1016/j.compbiomed.2022.105550.

[6] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger, On Calibration of Modern Neural Networks, in: Proceedings of the 34th International Conference on Machine Learning, PMLR, 2017, pp. 1321–1330.

[7] A. A. Abdullah, M. M. Hassan, Y. T. Mustafa, A Review on Bayesian Deep Learning in Healthcare: Applications and Challenges, IEEE Access 10 (2022) 36538–36562. doi:10.1109/ACCESS.2022.3163384.

[8] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: international conference on machine learning, PMLR, 2016, pp. 1050–1059.

[9] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles, in: Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.

[10] C. K. Williams, C. E. Rasmussen, Gaussian processes for machine learning, Vol. 2, MIT press Cambridge, MA, 2006.

[11] N. Durasov, T. Bagautdinov, P. Baque, P. Fua, Masksembles for Uncertainty Estimation, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Nashville, TN, USA, 2021, pp. 13534–13543. doi:10.1109/CVPR46437.2021.01333.

[12] A. Malinin, M. Gales, Predictive Uncertainty Estimation via Prior Networks, in: Advances in Neural Information Processing Systems, Vol. 31, Curran Associates, Inc., 2018.

[13] J. Postels, F. Ferroni, H. Coskun, N. Navab, F. Tombari, Sampling-free epistemic uncertainty estimation using approximated variance propagation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 2931–2940.

[14] J. H. Friedman, J. L. Bentley, R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software (TOMS) 3 (3) (1977) 209–226.

[15] G. B. Moody, R. G. Mark, A database to support development and evaluation of intelligent intensive care monitoring, in: Computers in Cardiology 1996, IEEE, 1996, pp. 657–660.

[16] H. Liu, Y.-S. Ong, X. Shen, J. Cai, When gaussian process meets big data: A review of scalable gps, IEEE transactions on neural networks and learning systems 31 (11) (2020) 4405–4423.

[17] J. Wang, An intuitive tutorial to gaussian processes regression, arXiv preprint arXiv:2009.10862 (2020).

[18] Y. Ding, J. Liu, J. Xiong, Y. Shi, Revisiting the evaluation of uncertainty estimation and its application to explore model complexity-uncertainty trade-off, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 4–5.

[19] H. D. Kabir, A. Khosravi, S. Nahavandi, D. Srinivasan, Neural network training for uncertainty quantification over time-range, IEEE Transactions on Emerging Topics in Computational Intelligence 5 (5) (2020) 768–779.

Table 3: Comparison of prediction intervals from RUE (Non-Parametric and Parametric), MCD, and GPR, trained on the "Mini" and the whole dataset.

Data	Time Horizon	Method	PICP (↑)	PINAW (↓)	PINAFD (↓)
Mini	t+1	Non-Parametric	0.855	0.074	**0.0160**
		Parametric	0.912	0.111	0.0340
		MCD	0.637	**0.041**	0.0189
		GPR	**0.985**	0.380	0.0758
	t+2	Non-Parametric	0.878	0.093	0.0237
		Parametric	0.926	0.138	0.0421
		MCD	0.704	**0.055**	**0.0230**
		GPR	**0.981**	0.380	0.0699
	t+3	Non-Parametric	0.884	0.101	0.0299
		Parametric	0.947	0.156	0.0468
		MCD	0.580	**0.042**	**0.0233**
		GPR	**0.978**	0.389	0.0658
Whole	t+1	Non-Parametric	0.888 (3.9%)	0.071 (-4%)	0.019 (18.7%)
		Parametric	0.953 (4.5%)	0.117 (4.7%)	0.039 (15.8%)
		MCD	0.689 (8.1%)	0.04 (-0.9%)	0.019 (2.9%)
	t+2	Non-Parametric	0.899 (2.4%)	0.086 (-7.8%)	0.026 (10.7%)
		Parametric	0.946 (2.2%)	0.129 (-6.9%)	0.044 (5.4%)
		MCD	0.525 (-25.5%)	0.03 (-45.8%)	0.02 (-11.9%)
	t+3	Non-Parametric	0.887 (0.3%)	0.092 (-9.6%)	0.028 (-4.9%)
		Parametric	0.964 (1.8%)	0.15 (-3.8%)	0.054 (15.4%)
		MCD	0.539 (-6.9%)	0.032 (-25.3%)	0.022 (-5.2%)

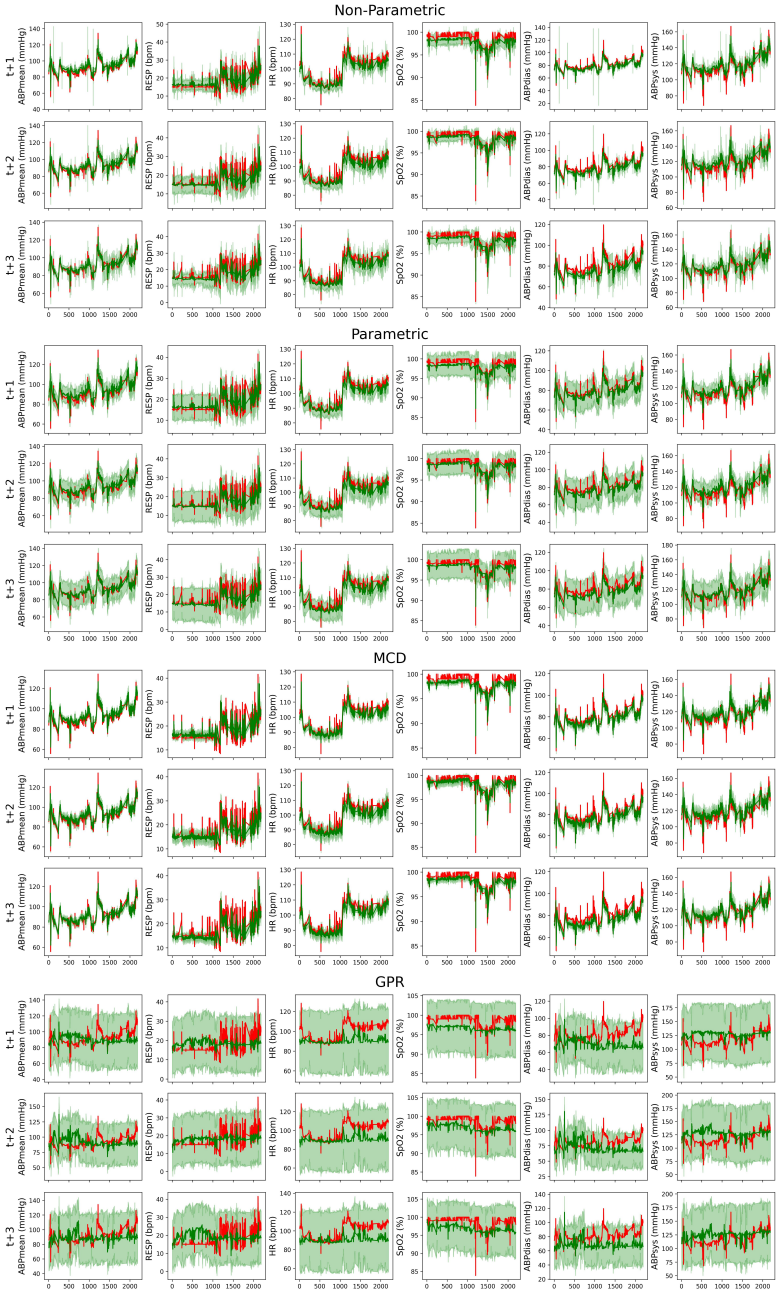


Figure 10: Comparison of prediction intervals produced by each method, for one patient (055n) on all six patient signals with different prediction horizons. The green line graph shows the model's predictions while the red shows the true signal values. The shaded lighter green regions define the prediction interval.

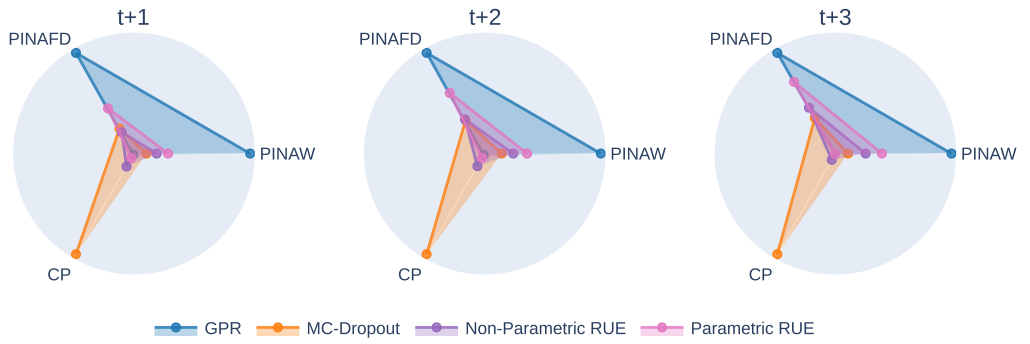
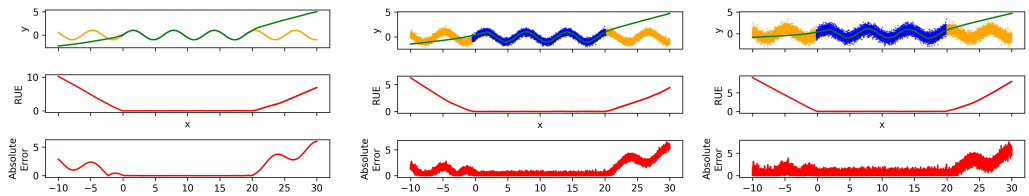


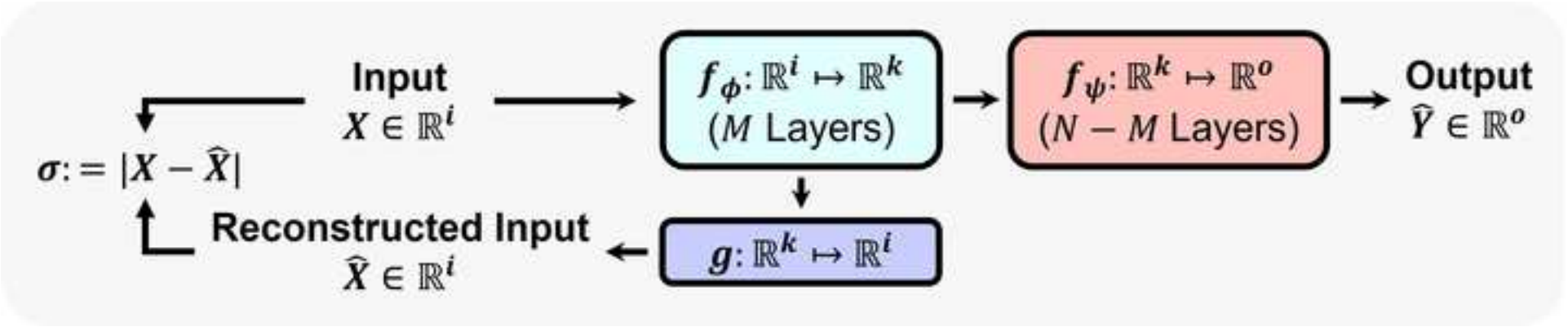
Figure 11: Starplot comparing the four prediction intervals across data sets of different time horizons. Models compared were all trained on the "Mini" dataset. CP stands for the coverage penalty calculated from the PICP.



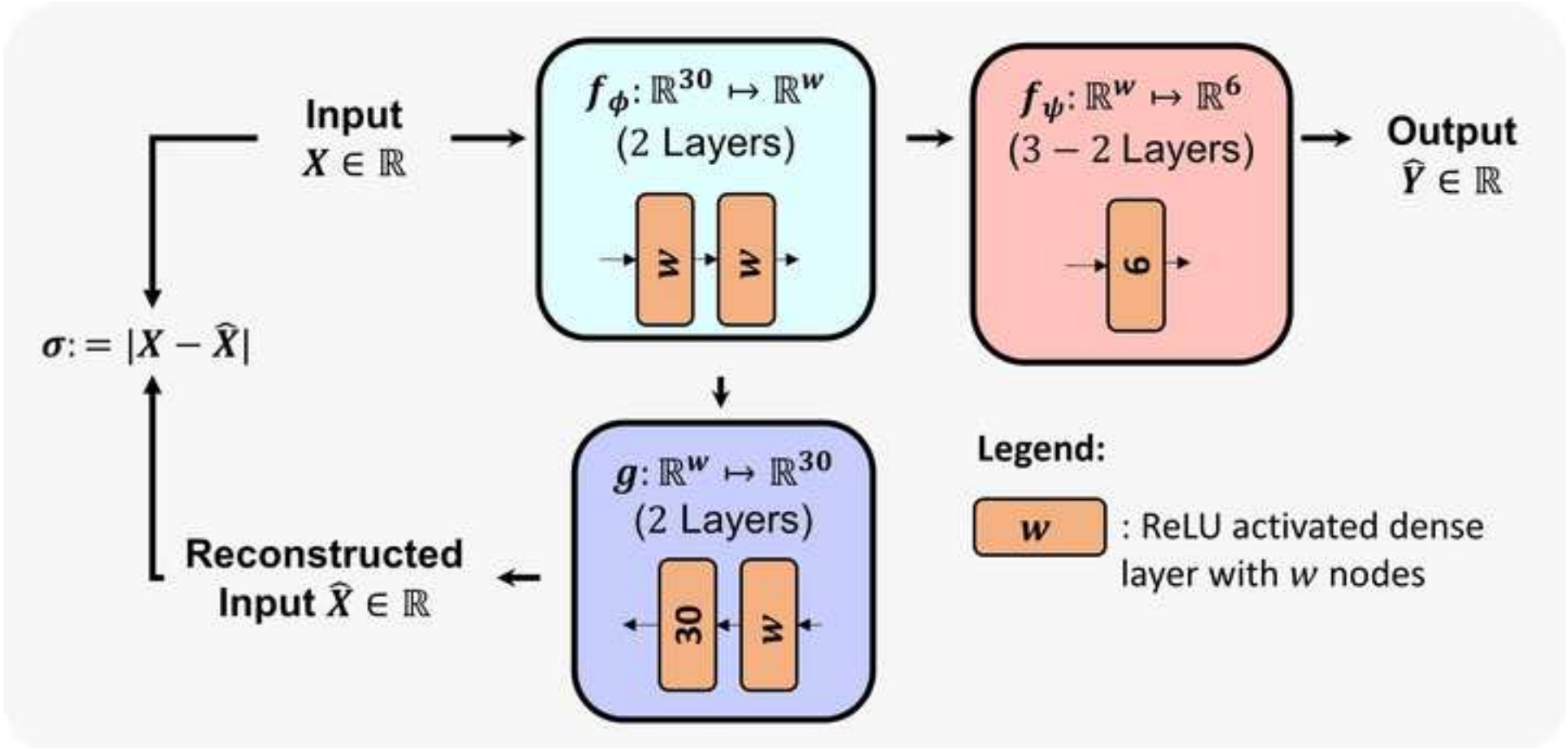
(a) Noise SD = 0 (b) Noise SD = 0.3 (c) Noise SD = 0.6

Figure 12: Relationship between prediction error and RUE on the sine simulation problem with different levels of noise (Different standard deviation; SD).

Figure

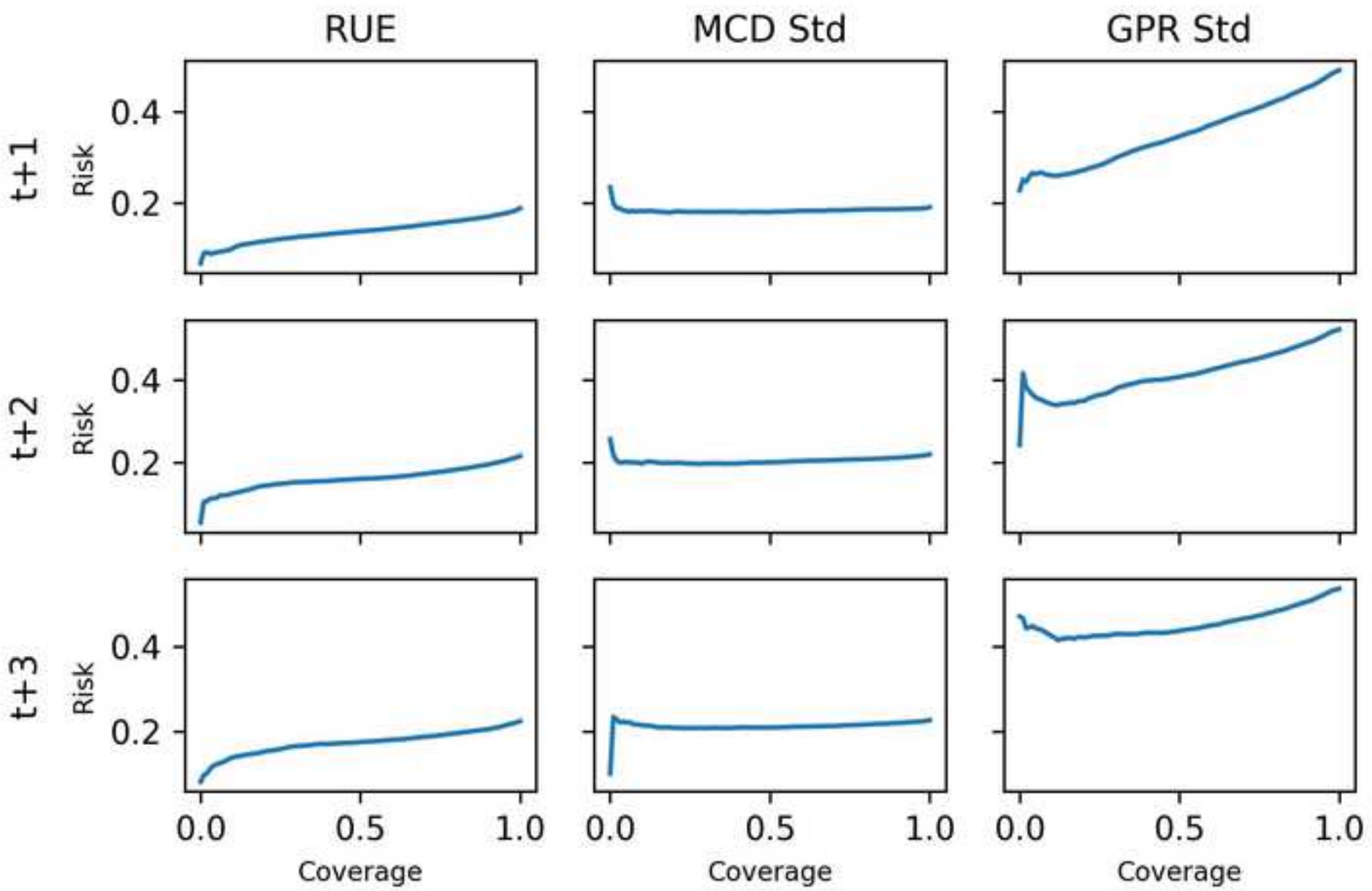


Figure



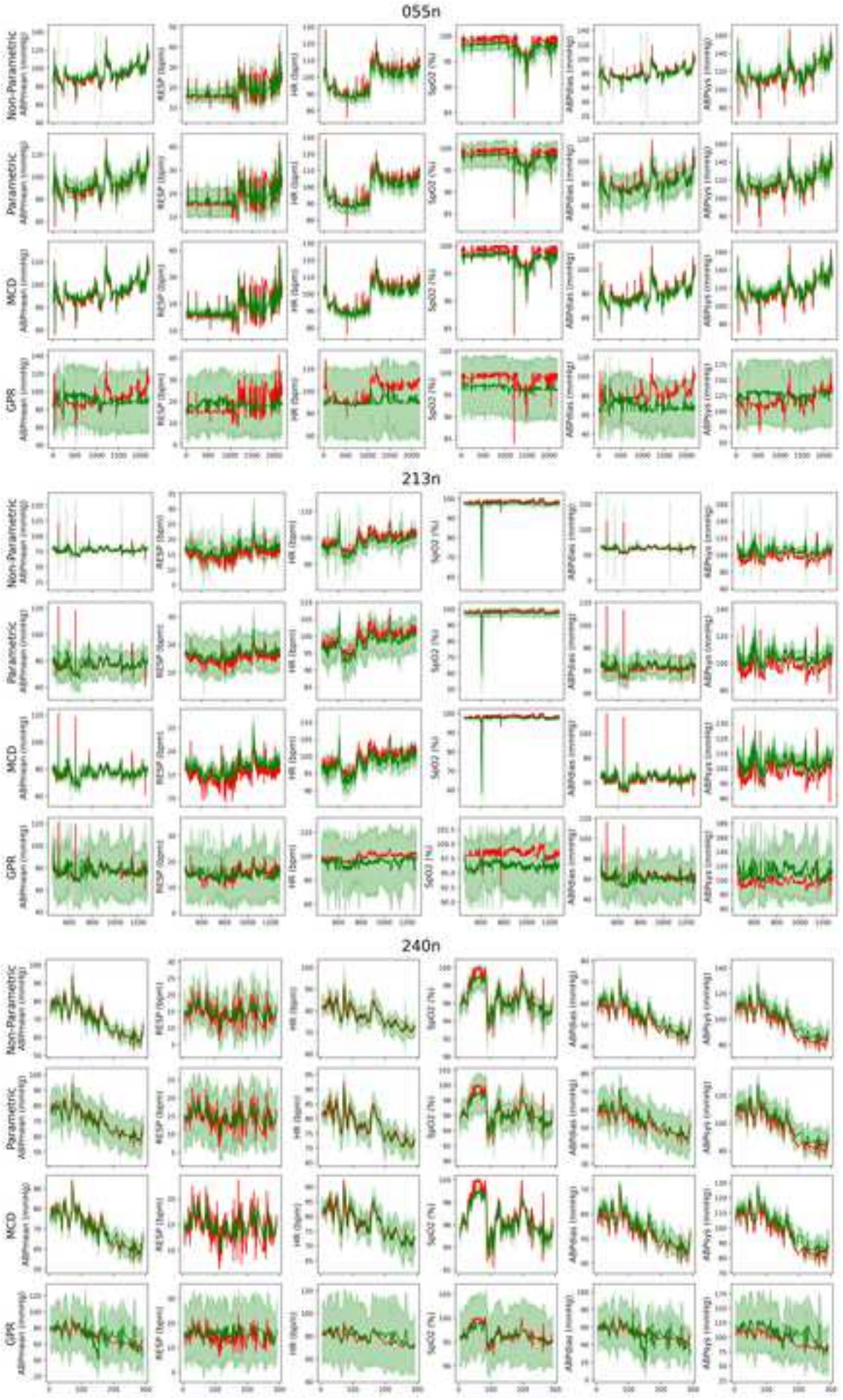
Figure

Click here to access/download;Figure;mimic_mini_aurc.jpg



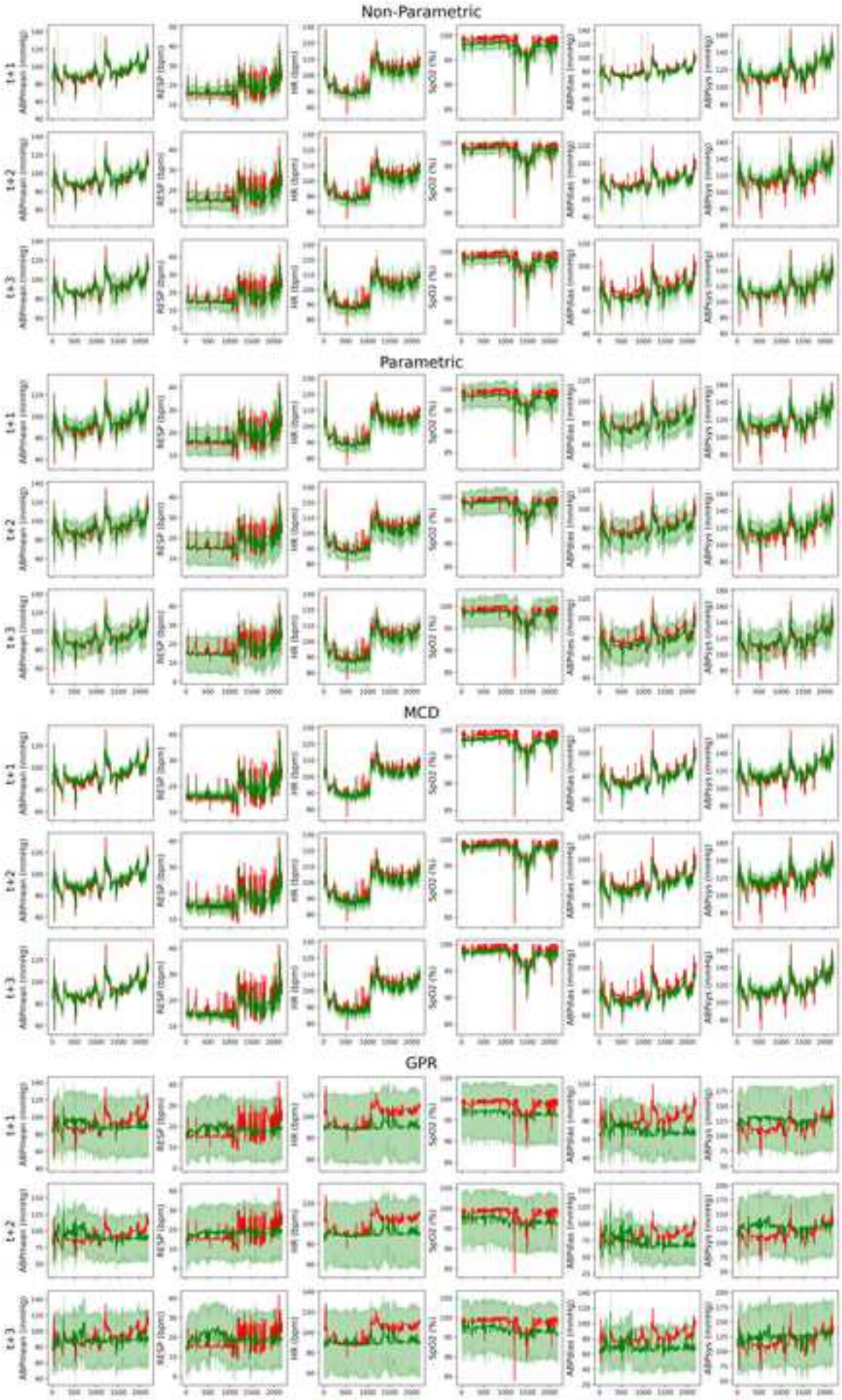
Figure

Click here to access/download;Figure;mimic_mini_compare_pi.jpg



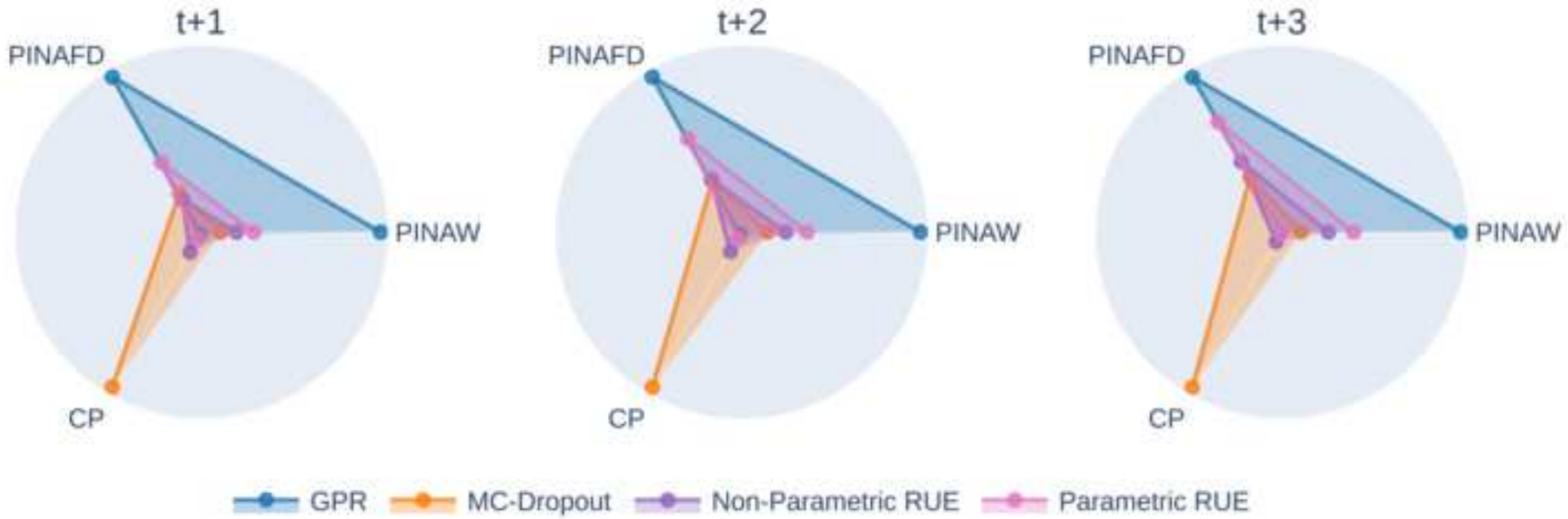
Figure

Click here to access/download;Figure;mimic_mini_compare_pi_across_time.jp



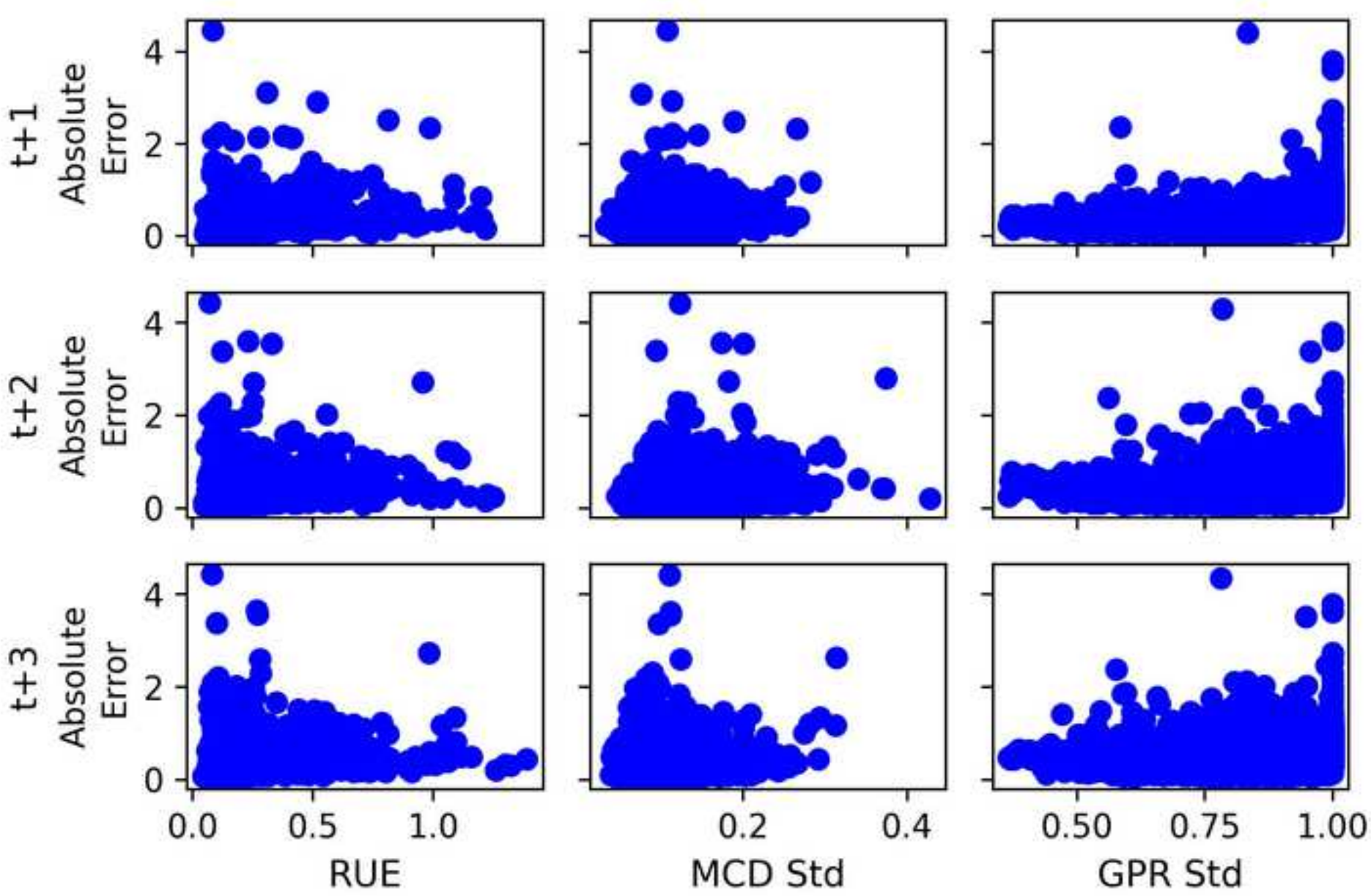
Figure

Click here to access/download;Figure;mimic_mini_compare_pi_starplot.png



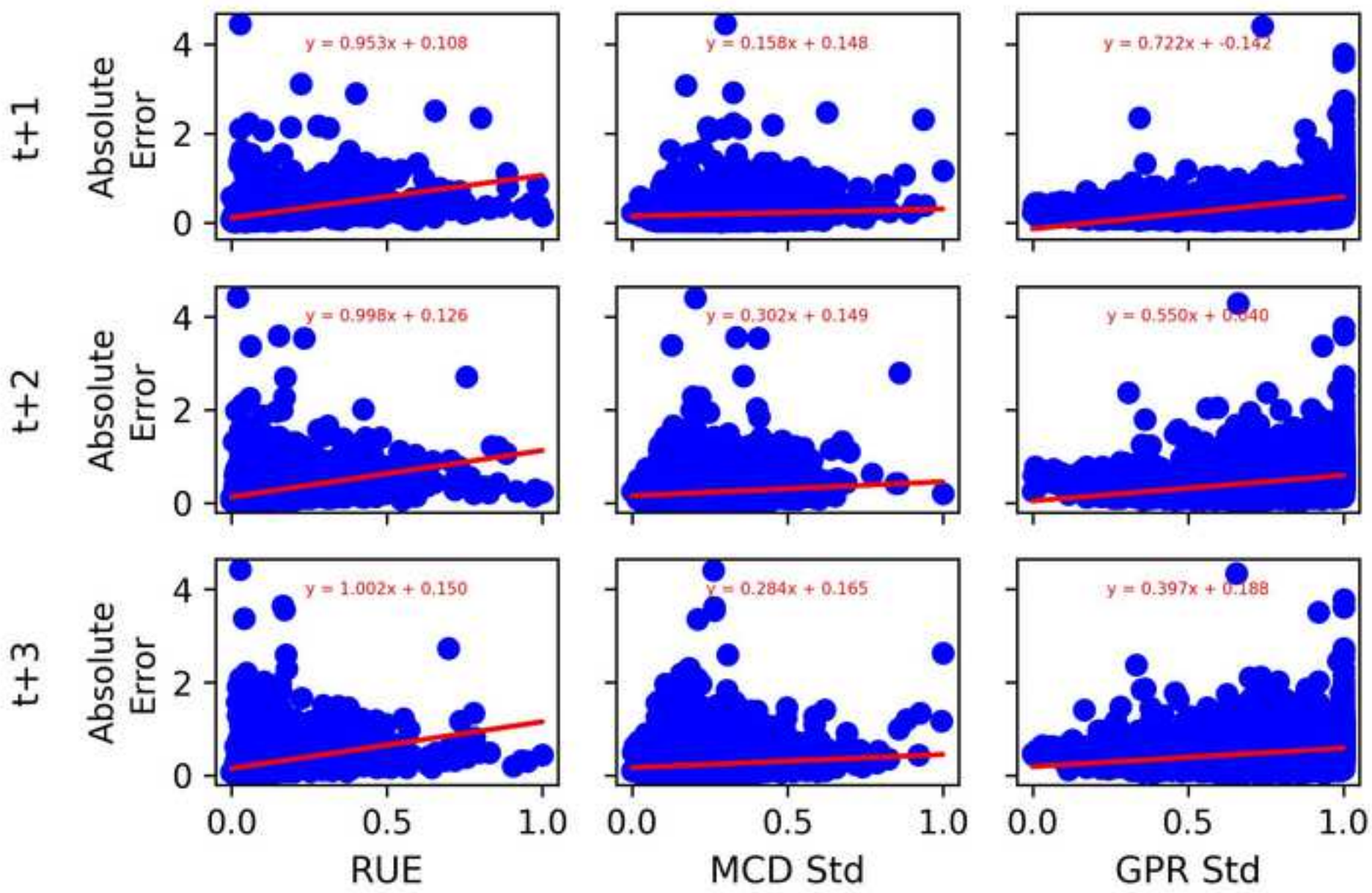
Figure

Click here to access/download;Figure;mimic_mini_ue_n_error.jpg



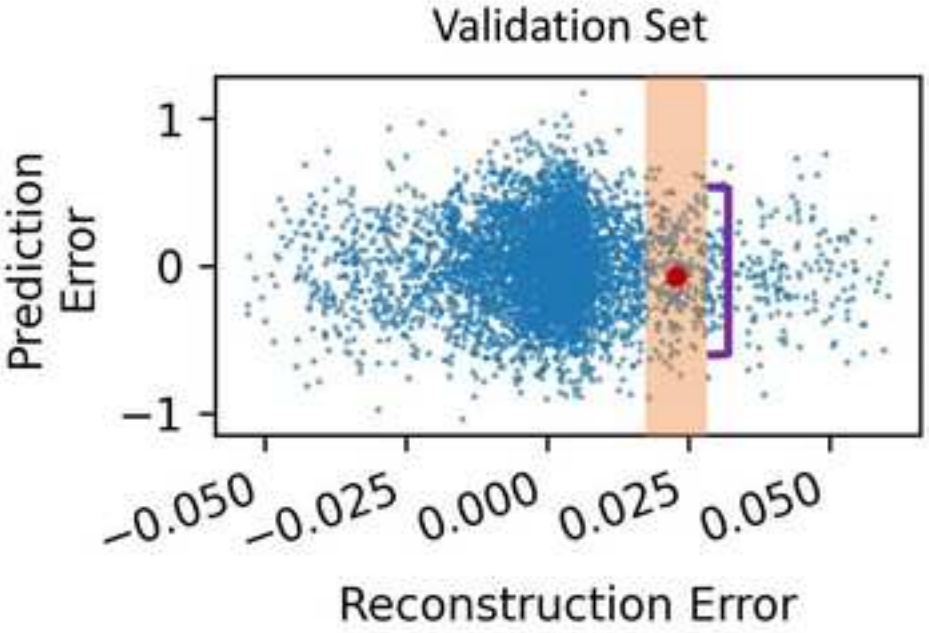
Figure

Click here to access/download;Figure;mimic_mini_ue_n_error_with_line.jpg

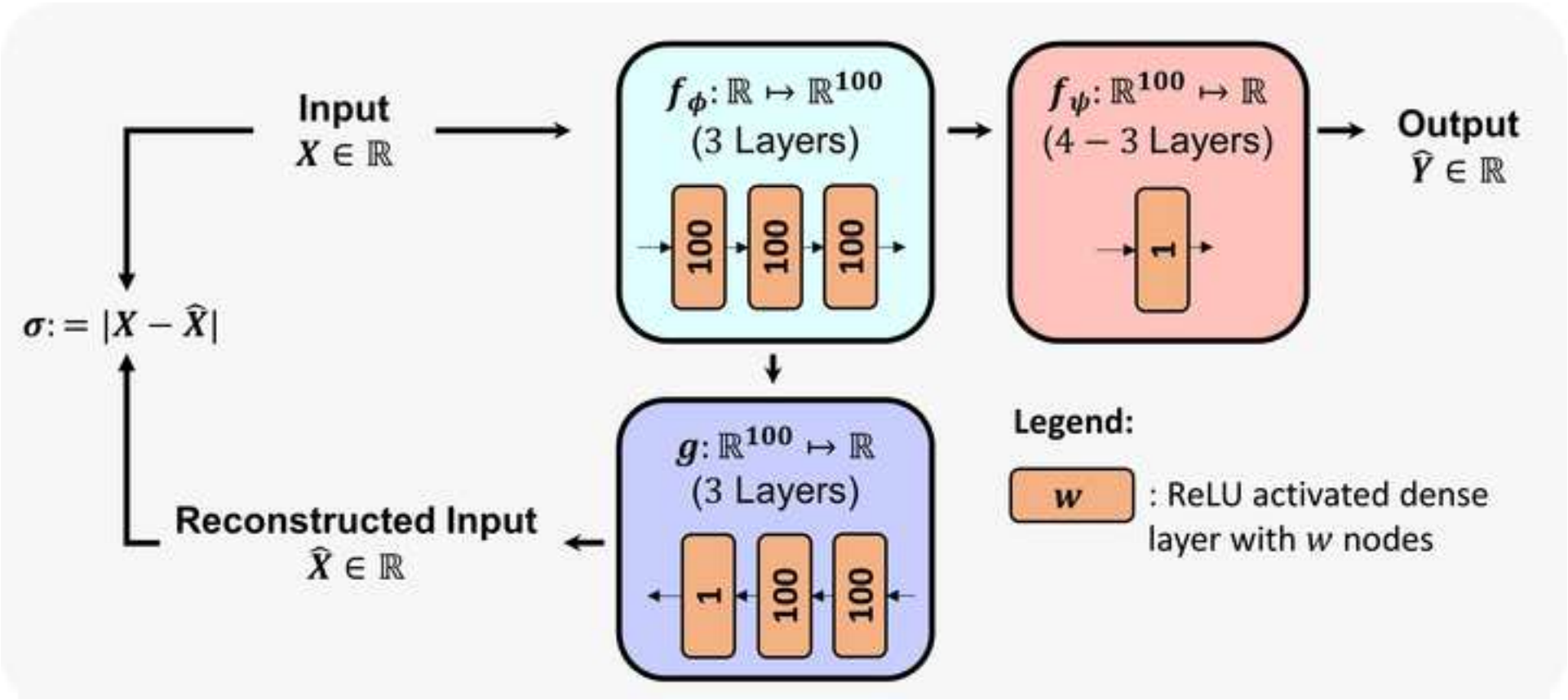


Figure

1. Derive reconstruction of $\hat{x} = (g \circ f_\phi)(x)$
2. Calculate the per-variable reconstruction errors $x - \hat{x}$ (•)
3. Get k-nearest neighbors to $x - \hat{x}$ ( )
4. Derive margin of error as the 95th percentile of absolute prediction errors ( ])

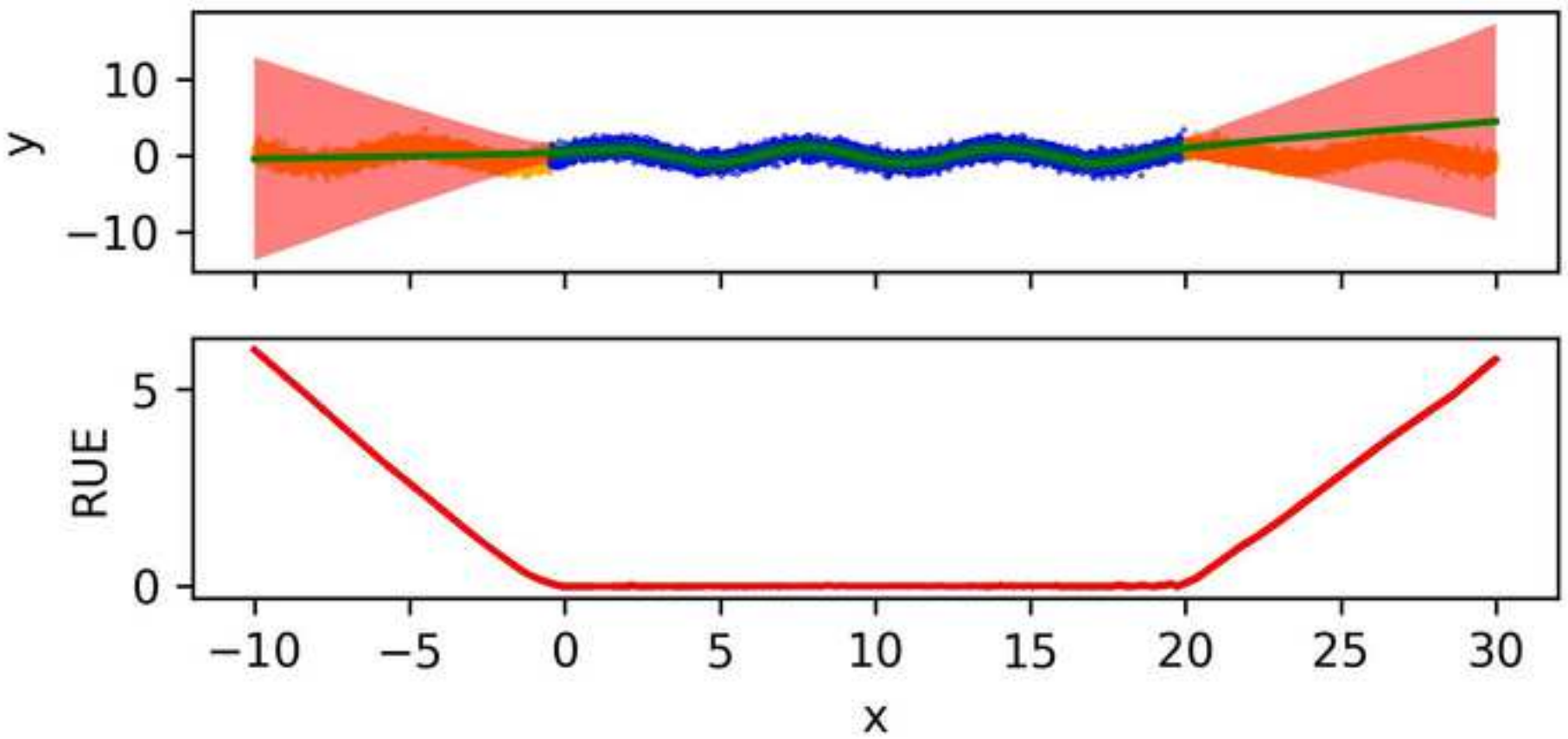


Figure



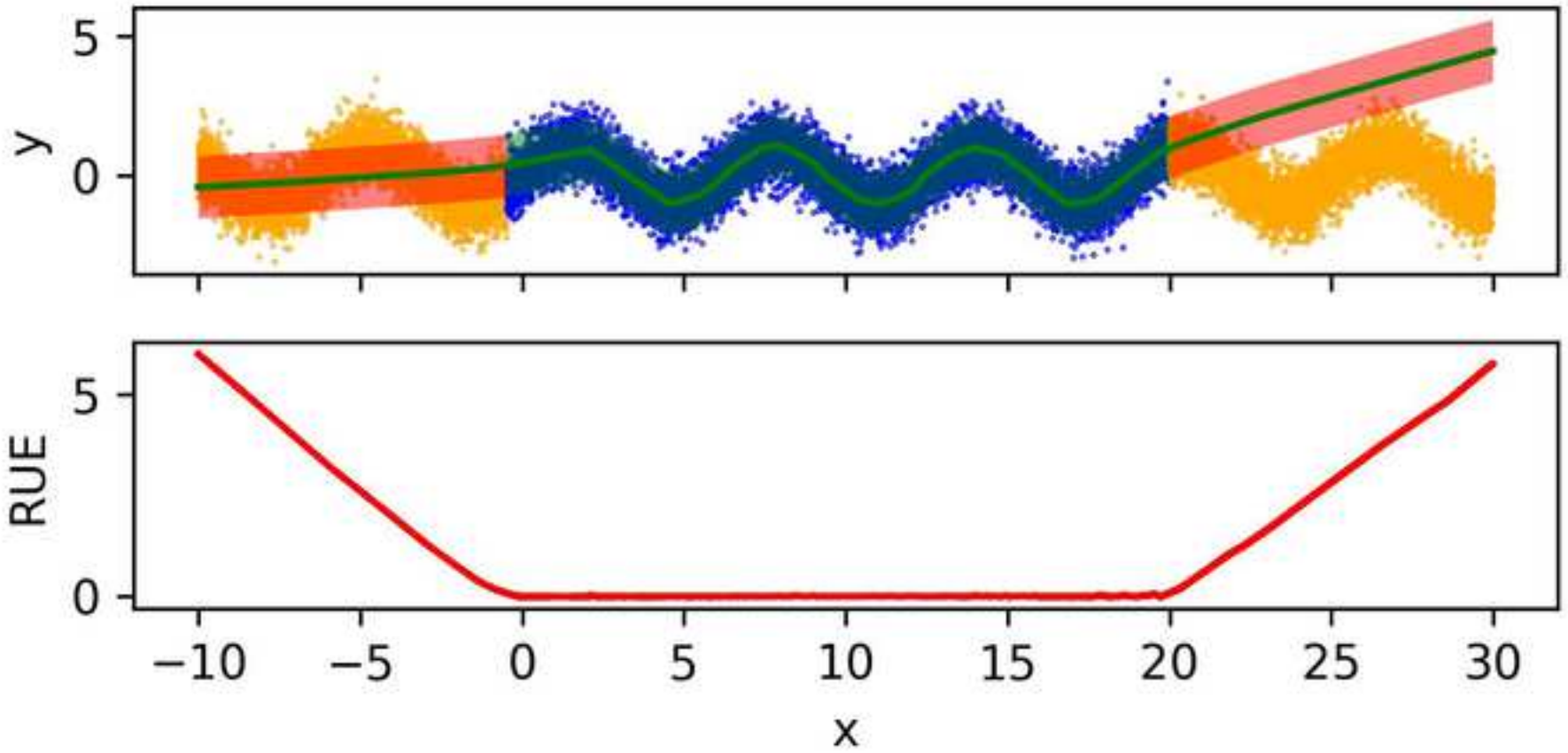
Figure

Click here to access/download;Figure;simulation_ci_cond.jpg



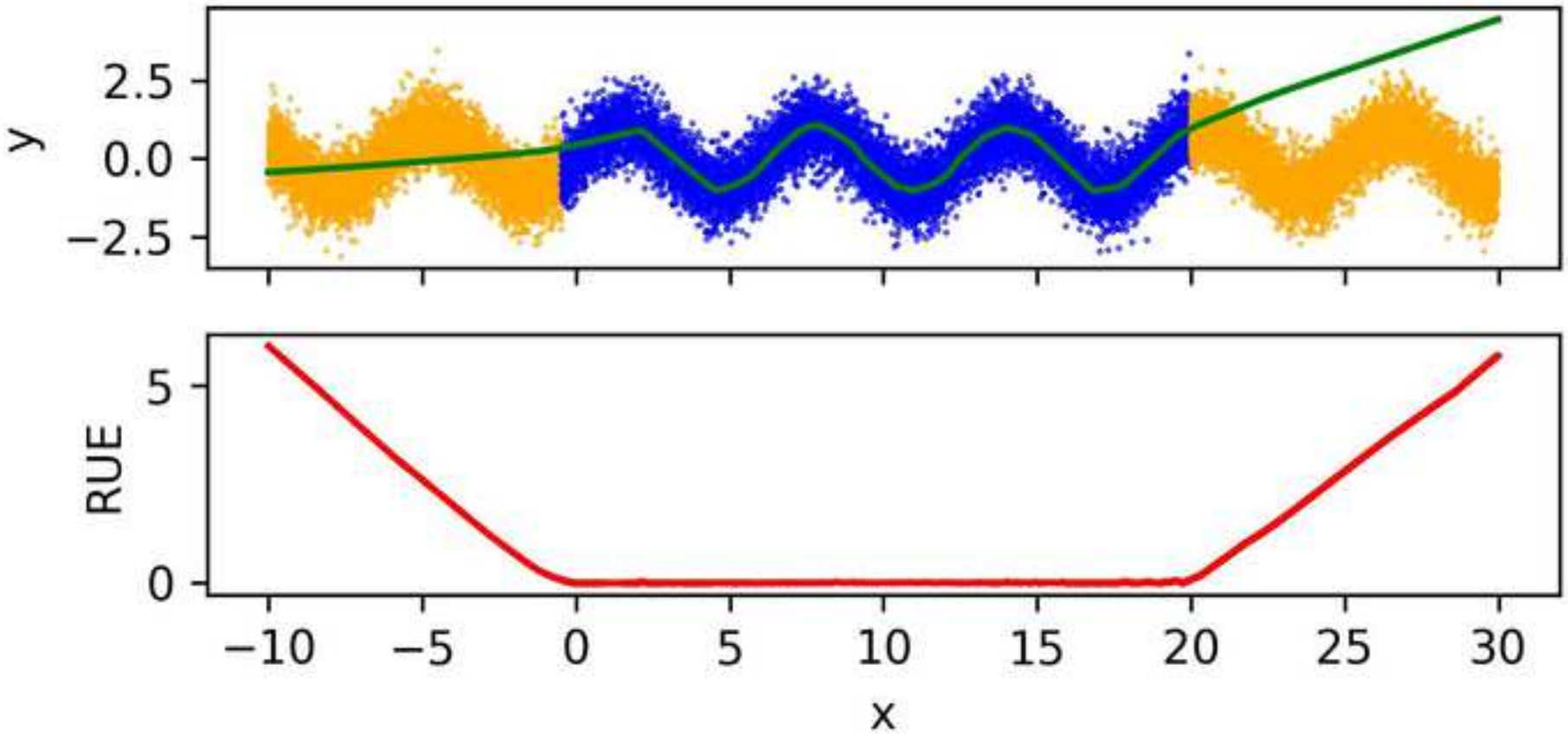
Figure

Click here to access/download;Figure;simulation_ci_knn.jpg



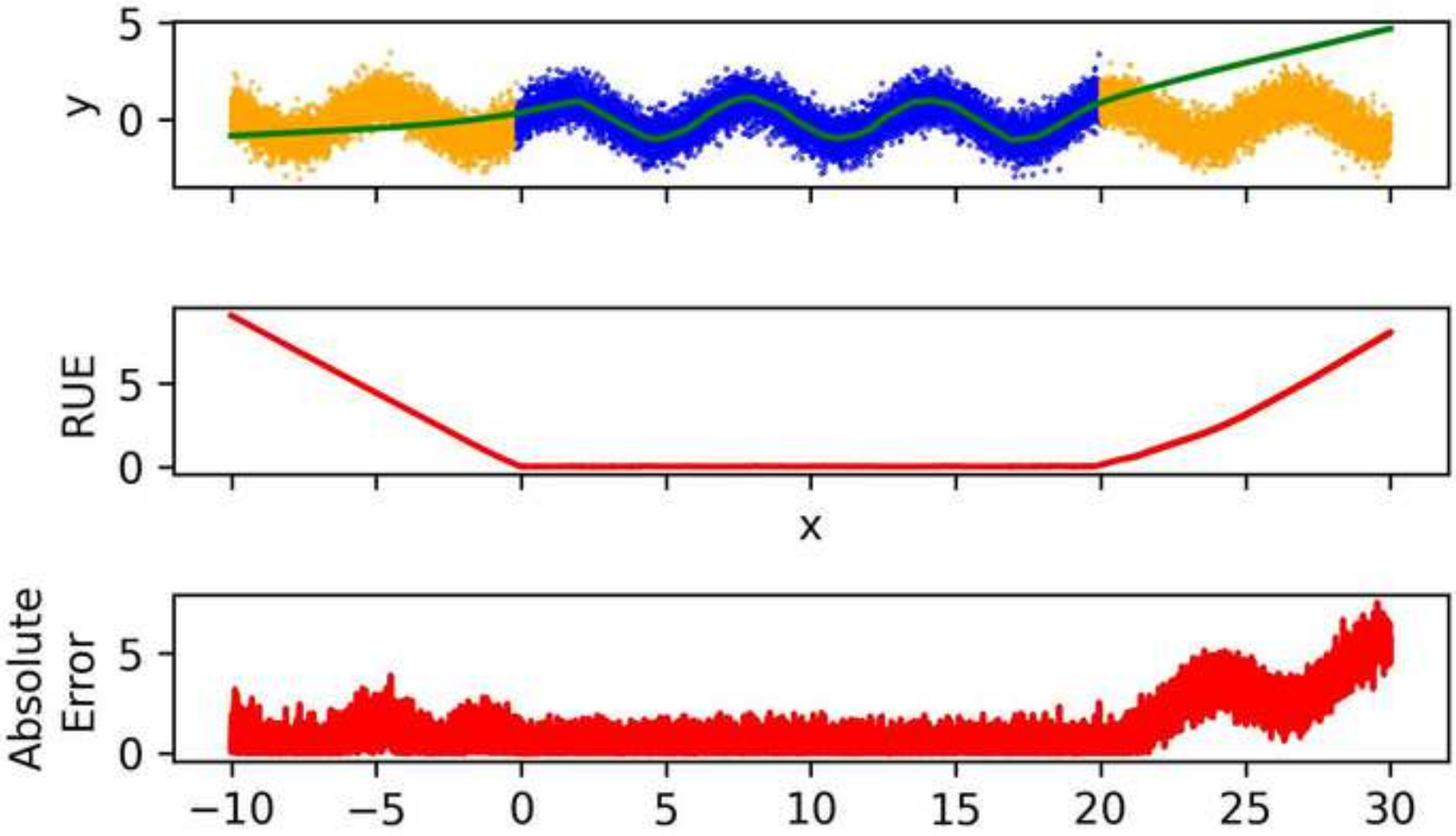
Figure

Click here to access/download;Figure;simulation_mae.jpg



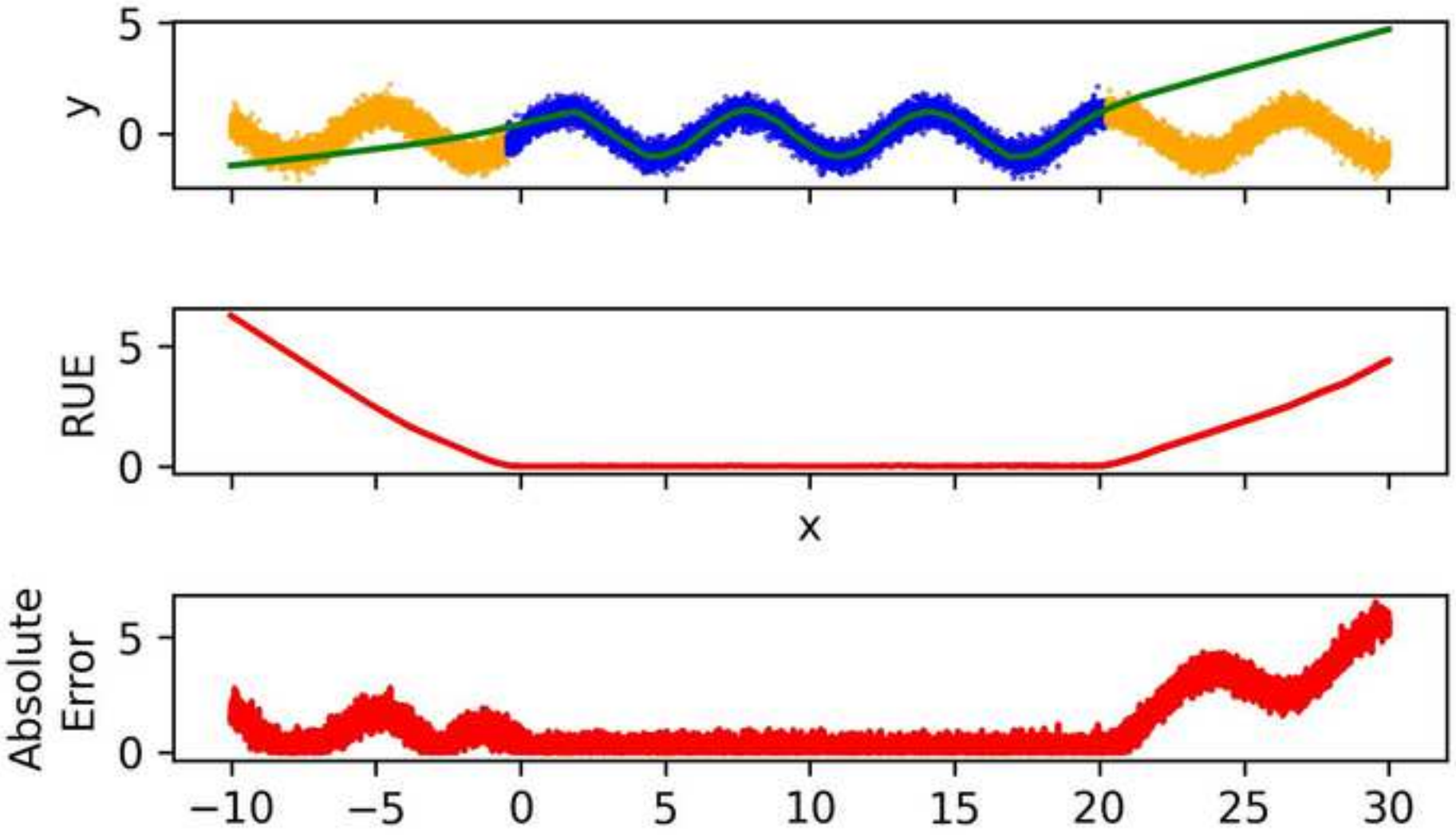
Figure

Click here to access/download;Figure;simulation_mae_with_ae.jpg



Figure

Click here to access/download;Figure;simulation_mae_with_ae_0.3noise.jpg



Figure

Click here to access/download;Figure;simulation_mae_with_ae_0noise.jpg