Tube Loss: A Novel Loss Function for Prediction Interval Estimation

Anonymous authors
Paper under double-blind review

Abstract

This paper proposes a novel loss function called Tube Loss, developed for the simultaneous estimation of the lower and upper bounds of a Prediction Interval (PI) in regression problems, including probabilistic forecasting in autoregressive frameworks. The PIs obtained through empirical risk minimization using Tube Loss exhibit superior performance compared to those derived from existing approaches. A theoretical analysis confirms that the estimated PIs asymptotically attain a user-specified confidence level $1 - \alpha$.

A distinctive feature of Tube Loss is its ability to shift the PI along the support of the response distribution through a tunable parameter, allowing the intervals to better align with high-density regions of the distribution. This is especially valuable for generating tighter intervals when the response distribution is skewed. Moreover, the method allows further narrowing of PIs through recalibration.

Unlike several prior techniques, the empirical risk associated with Tube Loss can be efficiently optimized via gradient descent. Extensive experiments demonstrate the robustness and accuracy of the proposed method in delivering high-quality PIs across a range of models, including kernel machines, neural networks, and probabilistic forecasting frameworks.

1 Introduction

In a regression framework, machine learning (ML) models aim to predict the value of a real-valued random variable y, commonly referred to as the dependent variable, given the value of a vector of input variables $\boldsymbol{x}$, known as the independent variable. However, providing only a point prediction of y without an uncertainty quantification may not be useful for applications. **Uncertainty quantification (UQ)** becomes particularly critical when the stakes of prediction errors are high. Consider, for instance, the scheduling of a replacement for a vital component in a nuclear reactor, where failure could be catastrophic. Merely reporting that the expected remaining lifespan is 2.5 years offers limited utility for planning preventive maintenance. On the other hand, stating that the component's lifespan is expected to lie between 2 and 3 years with 99% confidence conveys significantly more actionable information. Such ranges for the predicted values of y are termed *prediction intervals (PIs)*, and are associated with a specified confidence level (e.g., 99% in this case).

Suppose we are given n training samples $\{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n\}$, and we must now predict the unknown value of y_{n+1} at a test point $\mathbf{x} = \mathbf{x}_{n+1}$. We assume $\{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n+1\}$ are i.i.d. from an arbitrary joint distribution $p(\boldsymbol{x}, y)$ of the random vector $(\boldsymbol{x}, y)$, with feature vector $\boldsymbol{x} \in \mathcal{R}^p$

typically being a vector-valued input, and response variable $y \in \mathcal{R}$. For the sake of notational convenience, we use $(\boldsymbol{x}, y)$ to denote both the random vector as well as their specific values, whenever the distinction is evident from the context.

Given an algorithm $\mathcal{A}$ and a training set $\{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n\}$, a regression model is fitted to the training set to generate $\hat{\mu}(\mathbf{x})$, the predicted value of y at $\mathbf{x}$ obtained by minimizing the empirical risk

$$arg \min_{\mu} \sum_{i=1}^{n} \rho(y_i, \mu(\mathbf{x}_i)), \tag{1}$$

where, $\rho(y, \mu(\mathbf{x}))$ is a suitably chosen loss function representing deviation of $\mu(\mathbf{x})$ from y.

In conventional regression set up, typically $\hat{\mu}(\mathbf{x})$ is an estimate of the conditional expectation $\mathbb{E}(y \mid \mathbf{x})$ and $\rho(y, \mu(\mathbf{x})) = (y - \mu(\mathbf{x}))^2$. Often, a quantile of the conditional distribution $y|\mathbf{x}$, specifically the median instead of the mean, may be preferred as the predictor of y, especially when the conditional distribution is heavy-tailed.

In general, for quantifying the uncertainty associated with a predictor $\hat{\mu}(\boldsymbol{x})$ of y_{n+1}, a widely used measure is the *prediction interval* $[\hat{\mu}_1(\boldsymbol{x}_{n+1}), \hat{\mu}_2(\boldsymbol{x}_{n+1})]$ with a specified confidence level $1 - \alpha$, which is defined as

$$P\big[\hat{\mu}_1(\mathbf{x}_{n+1}) \leq y_{n+1} \leq \hat{\mu}_2(\mathbf{x}_{n+1})\big] \geq 1 - \alpha. \tag{2}$$

Equation (3) is formulated for any joint distribution $p(\boldsymbol{x}, y)$ and for any sample size n. The probability involved in this expression is *marginal*, taken over all samples $\{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n + 1\}$. A procedure aimed at constructing high-quality (**HQ**) prediction intervals must satisfy two essential criteria.

First, it should achieve the desired confidence level $1 - \alpha$ without imposing strong distributional assumptions. Second, the resulting intervals should be as narrow as possible across the input space, thereby ensuring that the predictions remain maximally informative.

The quality of a prediction interval is typically evaluated using two metrics: the **Prediction Interval Coverage Probability (PICP)** and the **Mean Prediction Interval Width (MPIW)**, both computed on the test set $\{(\boldsymbol{x}_i, y_i) : i = n + 1, n + 2, \ldots, n + m\}$. These measures are formally defined as

$$PICP = m^{-1} \sum_{i=n+1,2,\ldots,n+m} k_i, \tag{3}$$

where,

$$k_i = \begin{cases} 1, & \text{if } \; y_i \in [\hat{\mu}_1(x), \hat{\mu}_2(x)], \\ 0, & \text{Otherwise}, \end{cases} \tag{4}$$

and

$$MPIW = m^{-1} \sum_{i=n+1,n+2,\ldots,n+m} [\hat{\mu}_2(x_i) - \hat{\mu}_1(x_i)]], \tag{5}$$

respectively. A **High Quality (HQ)** prediction interval is expected to minimize the **MPIW** while ensuring that the **PICP** remains greater than or equal to $1 - \alpha$, the desired confidence level.

To address the problem of uncertainty quantification in regression tasks, various methods for constructing prediction intervals (PIs) have been proposed in the literature. Traditional approaches such as the *delta method* (Hwang & Ding (1997); Chryssolouris et al. (1996)), the *Mean-Variance Estimation (MVE)* technique (Nix & Weigend (1994)), and *Bayesian inference*-based models (MacKay (1992)) typically assume that the observational noise follows an independent Gaussian distribution. However, the performance of these approaches tends to deteriorate significantly when the underlying data distribution is highly skewed or exhibits heavy tails. To overcome these limitations, several alternative strategies have been proposed (e.g., Chung et al. (2021), Cui et al. (2020)), which involve first estimating the conditional mean $\mathbb{E}(y \mid \boldsymbol{x})$, and subsequently modeling the distribution of the residuals or their moments.

A more direct and model-agnostic strategy for constructing a **PI** with confidence level of $1-\alpha$ is to estimate $[F_q(\boldsymbol{x}), F_{q+1-\alpha}(\boldsymbol{x})]$ where, $F_q(\mathbf{x})$ is the $q \in (0,1)$-th quantile of the conditional distribution of y given $\mathbf{x}$ defined by $P(y \leq F_q(\mathbf{x})|\mathbf{x}) = q$. In quantile regression, an estimate of $F_q(\mathbf{x})$, say, $\hat{F}_q(\mathbf{x})$, is obtained by replacing $\rho(y, \mu(\mathbf{x}))$ in (1) with the pinball loss function (Koenker & Bassett Jr (1978), Koenker & Hallock (2001)), given by

$$\rho_q(y, \mu(\mathbf{x})) = \begin{cases} q(y - \mu(\mathbf{x})), & \text{if} \quad y \geq \mu(\mathbf{x}), \\ (q-1)(y - \mu(\mathbf{x})), & \text{otherwise.} \end{cases} \tag{6}$$

It is noteworthy that obtaining the lower and upper bounds of this interval involves solving two distinct optimization problems to estimate $\hat{F}_q(\boldsymbol{x})$ and $\hat{F}_{q+1-\alpha}(\boldsymbol{x})$. Furthermore, to ensure that the resulting interval qualifies as a **HQ PI**, one must select an optimal value of q (within the feasible range satisfying $q + 1 - \alpha \in (0,1)$) that minimizes the **MPIW** in the test set:

$$\text{MPIW} = \frac{1}{m} \sum_{i=n+1,n+2,\ldots,n+m} \left[\hat{F}_{q+1-\alpha}(\mathbf{x}_i) - \hat{F}_q(\mathbf{x}_i) \right].$$

Determining the optimal value of q necessitates repeatedly solving two separate optimization problems over a grid of candidate q values, which renders the overall procedure for estimating the prediction interval computationally demanding.

A few questions arise naturally at this point: can the lower and upper bounds of a **PI** be estimated *simultaneously* through a single optimization problem? In other words, is it possible to obtain **PI** as a direct output of a single optimization process? Furthermore, if such a formulation is feasible, the quality of the resulting interval would evidently depend on the choice of the underlying loss function. Hence, the central question becomes: what kind of loss function can yield **good-quality PIs**?

To address these issues, this paper introduces a novel loss function that produces **good-quality PIs**, referred to as the *Tube Loss* function. It may be viewed as a two-dimensional generalization of the *pinball loss* function, endowed with distinctive properties that enable the simultaneous estimation of both the lower and upper bounds of a **PI**. The intervals derived using the Tube Loss are shown, both theoretically and empirically, to asymptotically achieve the nominal coverage level. A formal proof of this result is provided.
Moreover, by tuning a hyperparameter of the loss function, the resulting interval can be shifted along the support of the response distribution y, thereby allowing it to better capture regions of high data density. As demonstrated in the experimental analysis, this feature proves particularly effective in reducing the **MPIW**, especially when the response distribution is asymmetric. Additionally, further reduction in **MPIW** on the test set can be achieved by incorporating a penalty term in

the loss function with a user-defined parameter, particularly when the **PICP** on the validation set substantially exceeds the nominal coverage level $1 - \alpha$.
In the experimental results presented in Section 4, the proposed Tube Loss is employed to generate **PIs** across various regression frameworks, including *Kernel Machines*, *Neural Networks*, and *Probabilistic Forecasting* using deep neural networks. The **PIs** derived from the Tube Loss are observed to outperform those obtained using existing loss functions.

The rest of the paper is structured as follows. Section 2 provides a review of existing approaches for **PI** estimation in regression frameworks. Section 3 presents the proposed Tube Loss function, describing its fundamental properties, its implementation within kernel machines and neural networks for **PI** estimation, and its extension to probabilistic forecasting. Section 4 discusses the results of extensive numerical evaluations. Finally, Section 5 concludes the paper by summarizing the main findings and outlining possible directions for future research.

2 Related Work

In this section, we present a brief review of existing methodologies that generate prediction intervals directly as the outcome of a single optimization problem, bypassing the intermediate step of quantile estimation. This line of research has evolved independently from the traditional quantile regression framework. However, most existing approaches primarily focus on **uncertainty quantification (UQ)** of **neural network (NN)** outputs. In contrast, the Tube Loss-based methodology proposed in this work extends the applicability of such direct **PI** estimation to *Kernel Machines* and *Probabilistic Forecasting*, in addition to **NNs**.

2.1 Interval Regression (IR)

In a distribution-free framework, Khosravi et al. (2011) were the first to propose estimating prediction intervals directly by minimizing the empirical risk associated with the following loss function:

$$\mathcal{L}_{LUBE} = \mathbf{NMPIW}\Big(1 + \gamma(\mathbf{PICP})e^{-\eta(\mathbf{PICP}-(1-\alpha))}\Big), \tag{7}$$

where $\mathbf{NMPIW} = \frac{\mathbf{MPIW}}{R}$, and $R = \max(y_i) - \min(y_i)$ denotes the range of the response values y_i. Note that $in(7), \mathbf{NMPIW}$ and **PICP** are computed based on the training set. The parameter η is a user-defined constant that amplifies small deviations between **PICP** and $1 - \alpha$. The indicator function $\gamma(\mathbf{PICP})$ is defined as 0 if $\mathbf{PICP} \geq (1 - \alpha)$, and 1 otherwise. Here, **LUBE** stands for "Lower and Upper Bound Estimation."

LUBE has been extensively used by the researchers in various **NN**-based application settings, such as energy load predictions (Pinson and Kariniotakis, 2013; Quan et al., 2014), wind speed forecasting (Wang et al., 2017; Ak et al.,2013), prediction of landslide displacement (Lian et al., 2016), gas flow (Sun et al., 2017), and solar energy (Galvan et al., 2017).

A major drawback of $\mathcal{L}_{LUBE}$ is that it's a function of **PICP**, a step function, and its gradient is zero almost everywhere (with respect to Lebesgue measure). Thus, the gradient descent (**GD**) method could not be used for the minimization of empirical risk. This is inconvenient considering that **GD** is now the standard method for training NNs. Instead, they propose simulated annealing (SA), a non-gradient-based method for optimization. Later, various other non-gradient-based methods are used, including Genetic Algorithms (AK et al., 2013), Gravitational Search Algorithms (Lian et al., 2016), PSO (Galvan et al., 2017), and Artificial Bee Colony Algorithms (Shen et al, 2018).

Pearce et al. (2018), building on the work of Khosravi et al. (2011), propose the Quality Driven (**QD**) loss function, given by

$$\mathcal{L}_{QD} = MPIW_{capt.} + \lambda \frac{n}{\alpha(1-\alpha)} \mathcal{L}_{PICP} \tag{8}$$

where $\mathbf{MPIW}_{capt.} = \frac{\sum_{i=1,2,\ldots,n}(\hat{\mu}_2(x_i)-\hat{\mu}_1(x_i)).k_i}{\sum_{i=1,2,\ldots,n} k_i}$, k_i is as defined in (4), λ is the user-defined trade-off parameter and $\mathcal{L}_{PICP} = max\big(0, (1-\alpha-PICP)\big)^2$. Notice that in contrast to **MPIW**, $\mathbf{MPIW}_{capt.}$ captures the average width for those points lying inside the **PI**. As discussed in Pearce et al. (2018) (Section 3.3.1), like **LUBE**, $\mathcal{L}_{QD}$ involves **PICP**, and thus, **GD** method cannot be used. For enabling the use of **GD** method, they propose approximating **PICP** by the product of two sigmoid functions, a technique used earlier by Yan et al. (2004). They also propose using an ensemble of **NN** models instead of a single **NN**. It is found to yield **PIs** with shorter average widths.

Salem et al. (2020) modified $\mathcal{L}_{QD}$ by adding two more penalty terms, one ($\mathcal{L}_{MSE}$) for generating a good point prediction and the other ($\mathcal{L}_P$) for enforcing the integrity of the results, i.e., avoiding crossing of **PI** bounds and point prediction out of **PI** bounds.

They named it $\mathcal{L}_{QD^+}$. It is defined as

$$\mathcal{L}_{QD+} = (1-\lambda_1)(1-\lambda_2)\mathbf{MPIW}_{capt.} + \lambda_1(1-\lambda_2)\mathcal{L}_{\mathbf{PICP}} + \lambda_2\mathcal{L}_{\mathbf{MSE}} + \xi\mathcal{L}_P, \tag{9}$$

where,

$$\mathcal{L}_{\mathbf{MSE}} = n^{-1} \sum_{i=1,2,\ldots,n} (\hat{\mu}(x_i) - y_i)^2, \tag{10}$$

and

$$\mathcal{L}_P = n^{-1} \sum_{i=1,2,\ldots,n} [max(0, \hat{\mu}_1(x_i) - \hat{\mu}(x_i)) + max(0, \hat{\mu}(x_i) - \hat{\mu}_2(x_i))]. \tag{11}$$

Finally, they proposed a novel method for aggregating the outputs received from the ensemble using a loss function, assuming that the posterior predictive distribution from a single model is split normal.

2.2 Simultaneous Quantile Regression (SQR)

Tagasovska and Lopez-Paz (2019) propose simultaneous estimation of a large number of quantiles by minimizing the empirical risk (cf. (1)) of standard quantile regression with a pinball loss function (cf. (6)). The minimization is carried out by using a stochastic gradient descent algorithm, sampling fresh random quantile levels $q \sim \mathcal{U}(0,1)$ for each training sample and mini-batch during training. Thus, **SQR** provides an estimate of the entire conditional distribution of y given x and thus, it enables them to "model non-gaussian, skewed, asymmetric, multimodal, and heteroscedastic" noise.

Given all possible quantiles, to generate a **PI**, any pair that satisfies the coverage constraint may be chosen. Because it estimates a large number of quantiles rather than the two bounds of **PI**, it faces significantly more challenges in learning during training. In spite of its added flexibility, its performance in terms of attaining nominal coverage is found to be inferior to that of the standard interval regression methods, such as those based on $\mathcal{L}_{QD}$ (cf. Pouplin et al. (2024)).

2.3 Relaxed Quantile Regression (RQR)

As an alternative to **SQR**, Pouplin et al. (2024) propose a novel loss function that directly outputs the bounds of **PI** with a target coverage $1-\alpha$ without requiring the intermediate step of estimating

quantiles. This approach they called Relaxed Quantile Regression (RQR). The proposed loss function is a kind of extension of the Pinball Loss function and is given by,

$$\mathcal{L}_{1-\alpha}^{RQR} = \begin{cases} (1-\alpha)\kappa, & \text{if } \kappa \geq 0, \\ -\alpha\kappa, & \text{if } \kappa < 0, \end{cases} \tag{12}$$

where, $\kappa = (y - \mu_1)(y - \mu_2)$. Notice that, for $\kappa < 0$, y will be inside **PI** and for $\kappa \geq 0$, it is on or outside **PI**, but $\mathcal{L}_{1-\alpha}^{RQR}$ remains positive in both cases. However, for smaller values of α (such as 0.01, 0.05, 0.1), the value of $\mathcal{L}_{1-\alpha}^{RQR}$ is much smaller for $\kappa < 0$ than for $\kappa \geq 0$. The multipliers of κ are so chosen as to satisfy the coverage constraint.

PI generated by solving the optimization problem $\arg\min_{\mu_1,\mu_2}\{E_Y(\mathcal{L}_{1-\alpha}^{RQR}(\mu_1(x), \mu_2(x), Y))\}$ possess some good theoretical properties, such as, it attains coverage, and it is unbiased and consistent.

Notice that the loss function (12) is the product of two errors, the deviations of the true value from the bounds of the **PI**, and thus, it provides asymmetric penalization. However, being a quadratic function of errors, **PI** bounds obtained by this method may be sensitive to the presence of outliers, similar to the case of expectiles. Furthermore, the presence of outliers is more likely to trigger the crossing of bounds, and thus, it may affect the conditional coverage of the **PI** near the crossings. Our experimental results in Section 4 support this claim.

3 Interval Estimation Using Tube Loss Function

In this section, we present the **Tube Loss** function, which offers several advantages over the loss functions discussed earlier. First, unlike equation (12) used in **RQR**, the Tube Loss is a *linear* function of the errors, making the estimated bounds less sensitive to outliers. Second, in contrast to the intervals obtained through **RQR**, the bounds of the Tube Loss-based interval never intersect. Third, unlike the loss functions employed in the **QD** and **QD+** approaches, the Tube Loss allows direct optimization of the empirical risk using the **GD** method. In the latter methods, $\mathcal{L}_{QD}$ and $\mathcal{L}_{QD+}$ must be approximated by smooth functions to enable gradient-based optimization, and the quality of the resulting intervals depends heavily on the quality of these approximations. The experimental results presented in Section 4 vindicate this claim.

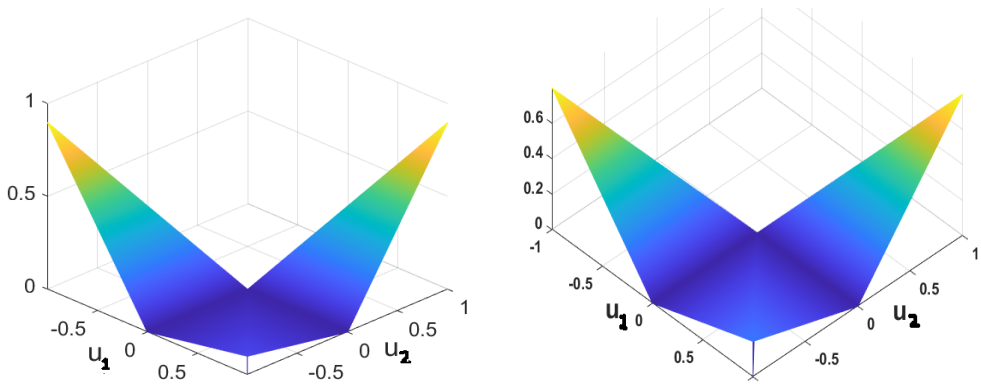
Fourth, a distinctive feature of the Tube Loss is that its output intervals can be shifted along the support of y by tuning a hyperparameter r. This flexibility allows the interval to better capture dense regions of the response distribution, thereby helping produce **tight** intervals, particularly when the conditional distribution of y given x is asymmetric. Fifth, by introducing a user-defined regularization parameter δ into the minimization problem, the Tube Loss enables an effective trade-off between **PICP** and **MPIW**, leading to narrower intervals in practice—especially when the **PICP** obtained on the validation set significantly exceeds the nominal coverage level $1 - \alpha$.

Finally, we demonstrate that the intervals derived from the Tube Loss function asymptotically attain the nominal coverage. In other words, as the training sample size $n \to \infty$, the empirical coverage converges to the nominal confidence level. A theoretical proof of this result is provided.

3.1 Tube Loss Function

Let us define the prediction errors $u_1(= y - \mu_1)$ and $u_2(= y - \mu_2)$ (cf. (12)). For a given nominal coverage $(1-\alpha)$ and $u_2 \leq u_1$ $(\mu_2 \geq \mu_1)$, we define the Tube loss function as

$$\mathcal{L}^r_{(1-\alpha)}(u_1, u_2) = \begin{cases} (1-\alpha)u_2, & \text{if} \quad u_2 > 0, \\ -\alpha u_2, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \; ru_2 + (1-r)u_1 \geq 0, \\ \alpha u_1, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \; ru_2 + (1-r)u_1 < 0, \\ -(1-\alpha)u_1, & \text{if } \; u_1 < 0, \end{cases} \tag{13}$$



(a) t=0.9 (b) t=0.8

Figure 1: Tube loss function

where, $0 < r < 1$ is a user-defined parameter. In the following, we discuss a few properties of the Tube loss function.

First, in Figure 1, plots of the Tube loss are given for the default choice $r = 0.5$. Evidently, $\mathcal{L}^{0.5}_{(1-\alpha)}(u_1, u_2)$ is a symmetric continuous function around the line $u_1 + u_2 = 0$. Additionally, it is differentiable almost everywhere (with respect to the Lebesgue measure) and has non-zero gradients, thus enabling the direct application of the **GD** method.

The default choice of r outputs centered **PIs**. However, by adjusting the value of r, the **PI** bounds can be shifted up or down, allowing the interval to capture the denser regions of the data cloud while maintaining coverage. It helps in reducing **MPIW** significantly, especially when the conditional distribution of y given x is asymmetric. The experimental results presented in Section 4 support this claim.

3.2 Asymptotic properties of the Tube loss function

Let $y_1, y_2, ..., y_n$ be iid following the distribution of a continuous random variable Y and $\alpha \in (0, 1)$. Let us define the following subsets of $\mathbb{R}$: $\Re_1(\mu_1, \mu_2) = \{y : y > \mu_2\}$, $\Re_2(\mu_1, \mu_2) = \{y : \mu_1 < y < \mu_2, y > r\mu_2 + (1-r)\mu_1\}$, $\Re_3(\mu_1, \mu_2) = \{y : \mu_1 < y < \mu_2, y < r\mu_2 + (1-r)\mu_1\}$ and $\Re_4(\mu_1, \mu_2) = \{y : y < \mu_1\}$. Notice that the sets $\Re_i(\mu_1, \mu_2), i = 1, 2, 3, 4$ are so defined that they do not include the points on the boundaries. Suppose (μ_1^*, μ_2^*) is the minimizer of $\frac{1}{n}\sum_{i=1}^n \mathcal{L}^r_{1-\alpha}(y_i, \mu_1, \mu_2)$ with respect to (μ_1, μ_2), and n_k denotes the number of data points in $\Re_k(\mu_1^*, \mu_2^*)$. Then we have the following lemma.

Lemma 1 *As $n \to \infty$, with probability 1 the following results hold: (i) $\frac{n_1}{n_2} \to \frac{\alpha}{1-\alpha}$, (ii) $\frac{n_4}{n_3} \to \frac{\alpha}{1-\alpha}$, and (iii) $\frac{n_1+n_4}{n_2+n_3} \to \frac{\alpha}{1-\alpha}$.*

The proof of the lemma is given in Appendix A.

We now consider the regression setup with training set $\boldsymbol{T} = \{(\boldsymbol{x}_i, y_i),\ i = 1, 2, \ldots, n\}$ where $(\boldsymbol{x}_i, y_i)$, $i = 1, 2, \ldots, n$ are iid following a density $p(\boldsymbol{x}, y)$. Let us rewrite (13) as

$$\mathcal{L}^r_{(1-\alpha)}(y, \mu_1(\boldsymbol{x}), \mu_2(\boldsymbol{x})) = \begin{cases} (1-\alpha)(y - \mu_2(\boldsymbol{x})), & \text{if } y > \mu_2(\boldsymbol{x}), \\ -\alpha(y - \mu_2(\boldsymbol{x})), & \text{if } y \le \mu_2(\boldsymbol{x}), y \ge \mu_1(\boldsymbol{x}) \text{ and } r(y - \mu_2(\boldsymbol{x})) + (1-r)(y - \mu_1(\boldsymbol{x})) \ge 0, \\ \alpha(y - \mu_1(\boldsymbol{x})), & \text{if } y \le \mu_2(\boldsymbol{x}), y \ge \mu_1(\boldsymbol{x}), \text{ and } r(y - \mu_2(\boldsymbol{x})) + (1-r)(y - \mu_1(\boldsymbol{x})) < 0, \\ -(1-\alpha)(y - \mu_1(\boldsymbol{x})), & \text{if } y < \mu_1(\boldsymbol{x}). \end{cases} \tag{14}$$

For the training set T, let $(\hat{\mu}^r_1(\boldsymbol{x}), \hat{\mu}^r_2(\boldsymbol{x}))$ be the minimizer of the empirical risk $\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}^r_{1-\alpha}(y_i, \mu_1(\boldsymbol{x}_i), \mu_2(\boldsymbol{x}_i))$, where $\mu_1(\boldsymbol{x})$ and $\mu_2(\boldsymbol{x})$ belong to an appropriately chosen class of functions. With some abuse of notation, we let n^r_k denote the cardinality of the set $\Re_k (= \Re_k(\hat{\mu}^r_1(\boldsymbol{x}), \hat{\mu}^r_2(\boldsymbol{x})))$ $(k = 1, 2, 3, 4)$ inducing a partition of the feature space (as shown in Figure 2). We then have the following proposition.

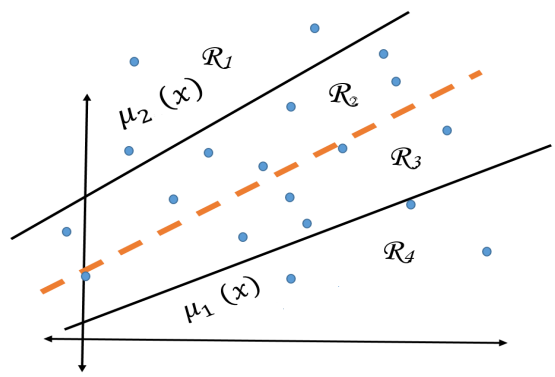


Figure 2: PI tube loss, red line represents the convex combination of $\mu_1(\boldsymbol{x})$ and $\mu_2(\boldsymbol{x})$, i,e., $r\mu_1(\boldsymbol{x}) + (1-r)\mu_2(\boldsymbol{x})$, $0 < r < 1$. Blue dots represent data points $(\boldsymbol{x}_i, y_i)$, $i = 1, 2, , .., n$.

Proposition 1 *For $\alpha \in (0, 1)$ as $n \to \infty$ the following results hold with probability 1, (i) $\frac{n^r_1}{n^r_2} \to \frac{\alpha}{1-\alpha}$, (ii) $\frac{n^r_4}{n^r_3} \to \frac{\alpha}{1-\alpha}$ (iii) $\frac{n^r_1+n^r_4}{n^r_2+n^r_3} \to \frac{\alpha}{1-\alpha}$, provided $(\boldsymbol{x}_i, y_i), i = 1, 2, ..., n$ are iid following a distribution $p(\boldsymbol{x}, y)$ with $p(y|\boldsymbol{x})$ continuous and the expectation of the modulus of absolute continuity of its density [1] satisfies $\lim_{\delta \to 0} E[\epsilon(\delta)] = 0$.*

Proof: The proof follows from the proof of the Lemma stated above and Lemma 3 of Takeuchi et al. (2006).

[1]The modulus of absolute continuity of a function f is defined as the function $\epsilon(\delta) = Sup \sum_i |f(b_i) - f(a_i)|$, where the supremum is taken over all disjoint intervals (a_i, b_i) with $a_i < b_i$ satisfying $\sum_i (b_i - a_i) < \delta$. Loosely speaking, the conditional density $p(y|x)$ is absolutely continuous on average. This ensures that the probability of a point lying on the boundaries of PI vanishes.

3.3 Parameter r and movement of the PI bounds

Now, we show that how the PI bounds can be moved up and down by choosing r appropriately. Consider two values of r, say r_1 and r_2 such that $0 < r_1 < r_2 < 1$. Let us denote the cardinality of the set $\Re_k(\hat{\mu}_1^{r_i}(x), \hat{\mu}_2^{r_i}(x))$ by $n_k^{r_i}$ $(k = 1, 2, 3, 4,\ i = 1, 2.)$

Now, Proposition 1 (iii) entails that asymptotically $1 - \alpha$ fraction of y values should lie inside each of the intervals $[\hat{\mu}_1^{r_1}(\boldsymbol{x}), \hat{\mu}_2^{r_1}(\boldsymbol{x})]$ and $[\hat{\mu}_1^{r_2}(\boldsymbol{x}), \hat{\mu}_2^{r_2}(\boldsymbol{x})]$. Since $r_1 < r_2$, (i) and (ii) of Proposition 1 entail $\frac{n_2^{r_1}}{n_3^{r_1}} > \frac{n_2^{r_2}}{n_3^{r_2}}$, which in turn implies $\frac{n_1^{r_1}}{n_4^{r_1}} > \frac{n_1^{r_2}}{n_4^{r_2}}$ for large n. Thus, changing the value of r from r_1 to r_2 moves the tube down and vice versa.

In Section 4, we demonstrate that, in several experiments, selecting an appropriate value of r results in a significant reduction of **MPIW**. Notice that, $\mathcal{M}_{1-\alpha}^r(u_1, u_2)$ is discontinuous at the separating plane $\{(u_1, u_2) : ru_2 + (1 - r)u_1 = 0\}$ when $r \neq 0.5$. However, while running the experiments, we observe that it has not caused any problem for the implementation of the GD method.

3.4 Recalibrating Prediction Intervals to Balance MPIW and PICP

After choosing an appropriate value of r, if the attained coverage (**PICP**) of the output interval in the validation set exceeds the nominal coverage $1 - \alpha$, then we go for re-calibration, which helps in further minimization of MPIW, and thus, enhances the quality of the interval, keeping the attained coverage more than equal to $1 - \alpha$. For re-calibration, we employ the following loss function,

$$\mathcal{L}_{\mathcal{W}_{(1-\alpha)}^r}(y, \mu_1(\boldsymbol{x}), \mu_2(\boldsymbol{x})) = \mathcal{L}_{(1-\alpha)}^r(y, \mu_1(\boldsymbol{x}), \mu_2(\boldsymbol{x})) + \delta|\mu_2(\boldsymbol{x}) - \mu_1(\boldsymbol{x})|. \tag{15}$$

The default choice of δ is zero, unless there is room for recalibration, i.e, if **PICP** is more than $1 - \alpha$ in the validation data set. Notice that in contrast to RQR, the penalty term in our case uses a linear function of the errors, thus making it less sensitive to outliers. In Section 4, we employ recalibration in some experiments to enhance the quality of the interval.

4 Experiments

In this section, we present the experimental results on generating prediction intervals in three different regression settings: kernel machines, neural networks, and probabilistic forecasting, employing the Tube loss and comparing with existing losses in the literature. The experimental results clearly demonstrate the superiority of the Tube loss compared to its competitors in outputting better quality intervals.

In general, to compare two **PI** estimation methods, say $\mathcal{M}_1$ and $\mathcal{M}_2$, we adopt the following rule in this paper: method $\mathcal{M}_1$ is considered superior to method $\mathcal{M}_2$ if either $\mathbf{PICP}(\mathcal{M}_1) \geq 1 - \alpha$ and $\mathbf{PICP}(\mathcal{M}_2) < 1 - \alpha$, or if $\min\{\mathbf{PICP}(\mathcal{M}_1), \mathbf{PICP}(\mathcal{M}_2)\} \geq 1 - \alpha$ and $\mathbf{MPIW}(\mathcal{M}_1) < \mathbf{MPIW}(\mathcal{M}_2)$.

Furthermore, in experiments conducted with synthetic data—where samples are generated from a known distribution—the true conditional quantiles of y given $\boldsymbol{x}$ are known for all $\boldsymbol{x}$. Consequently, the corresponding true **PI**, denoted by $[\mu_1(\boldsymbol{x}), \mu_2(\boldsymbol{x})]$, is available. In such settings, the accuracy of an estimated PI, $[\hat{\mu}_1(\boldsymbol{x}), \hat{\mu}_2(\boldsymbol{x})]$, is assessed using the *Sum of Mean Squared Errors (**SMSE**)*, in the test set defined as

$$\frac{1}{m}\sum_{i=1}^{m}\left(\hat{\mu}_1(\boldsymbol{x}_i) - \mu_1(\boldsymbol{x}_i)\right)^2 + \frac{1}{m}\sum_{i=1}^{m}\left(\hat{\mu}_2(\boldsymbol{x}_i) - \mu_2(\boldsymbol{x}_i)\right)^2.$$

4.1 Tube loss in kernel machines

In kernel machines, the PI bounds are a pair of functions

$$\mu_1(\mathbf{x}) := \sum_{i=1}^{n} k(\mathbf{x}_i, \mathbf{x})\alpha_i + \alpha_0 \;\text{ and }\; \mu_2(\mathbf{x}) := \sum_{i=1}^{n} k(\mathbf{x}_i, \mathbf{x})\beta_i + \beta_0, \tag{16}$$

where $k(\mathbf{x}, y)$ is positive definite kernel(Mercer (1909)). Using matrix notation, we rewrite $\mu_1(\mathbf{x})$ and $\mu_2(\mathbf{x})$ as

$$\mu_1(\mathbf{x}) := K(A^T, \mathbf{x})\boldsymbol{\alpha} + \alpha_0 \;\text{ and }\; \mu_2(\mathbf{x}) := K(A^T, \mathbf{x})\boldsymbol{\beta} + \beta_0, \tag{17}$$

where A is an $n \times p$ data matrix containing n training points in $\mathbb{R}^p$, $\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ .. \\ \alpha_n \end{bmatrix}$, $\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ .. \\ \beta_n \end{bmatrix}$ and $K(A^T, \mathbf{x}) = \Big[k(\mathbf{x}_1, \mathbf{x}), k(\mathbf{x}_2, \mathbf{x}), .., k(\mathbf{x}_n, \mathbf{x})\Big]$. The Tube loss-based kernel machines consider the following optimization problem for outputting the **PI**:

$$\begin{aligned} \min_{(\boldsymbol{\alpha},\boldsymbol{\beta},\alpha_0,\beta_0)} \frac{\lambda}{2}(\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{\beta}^T\boldsymbol{\beta}) + \sum_{i=1}^{n} \mathcal{L}^r_{(1-\alpha)}(y_i, (K(A^T, \mathbf{x}_i)\boldsymbol{\alpha} + \alpha_0), (K(A^T, \mathbf{x}_i)\boldsymbol{\beta} + \beta_0)\ ) \\ + \delta \sum_{i=1}^{n} \big|(K(A^T, \mathbf{x}_i)(\boldsymbol{\alpha} - \boldsymbol{\beta}) + (\alpha_0 - \beta_0)\big|. \end{aligned} \tag{18}$$

The Gradient descent solution to the above optimization problem is derived in Appendix B.

For obtaining high-quality **PI**s, the parameters r and δ must be appropriately tuned using the validation dataset. The default settings $r = 0.5$ and $\delta = 0$, typically yield good performance when the conditional distribution of y given $\mathbf{x}$ is symmetric, since in that case a centered prediction interval is optimal. However, if the distribution is positively (negatively) skewed, selecting an r value smaller (larger) than 0.5 shifts the bounds downward (upward), thereby minimizing **MPIW** while maintaining **PICP** at the desired level. Once an appropriate r has been determined, if the resulting **PICP** exceeds the target confidence level $1 - \alpha$, recalibrating through δ can further reduce **MPIW** without compromising the intended coverage.

4.1.1 Experimental Results

1. Choice of r

We first show, in the context of kernel-based models, that the choice of the parameter r is crucial for achieving narrow **PI** bounds, especially when the underlying data distribution exhibits asymmetry.

To illustrate this, we construct two synthetic datasets, denoted by $\mathbf{D_1}$ and $\mathbf{D_2}$. Each dataset consists of observations $\{(x_i, y_i) : i = 1, 2, \ldots, 1500\}$, where x_i is sampled from the uniform distribution on the interval $[0, 1]$, i.e., $x_i \sim U(0, 1)$, and the response variable y_i is generated according to

$$y_i = \frac{\sin x_i}{x_i} + \epsilon_i, \tag{19}$$

where ϵ_i represents a random noise component. For dataset $\mathbf{D_1}$, ϵ_i follows a symmetric normal distribution $\mathcal{N}(0, 0.8)$, while for dataset $\mathbf{D_1}$, ϵ_i follows a positively skewed distribution given by $\chi^2(3) - 3$.

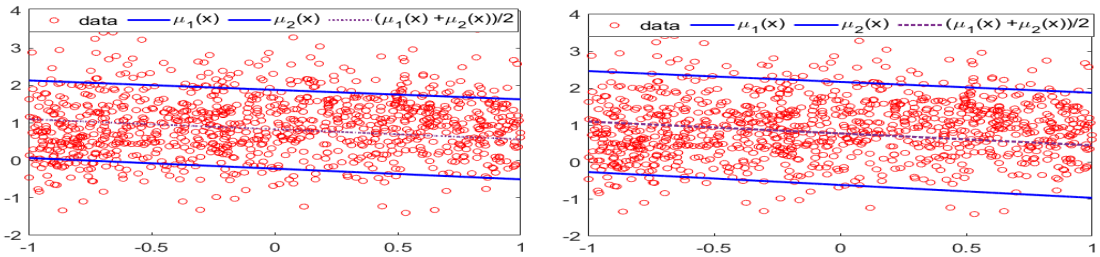
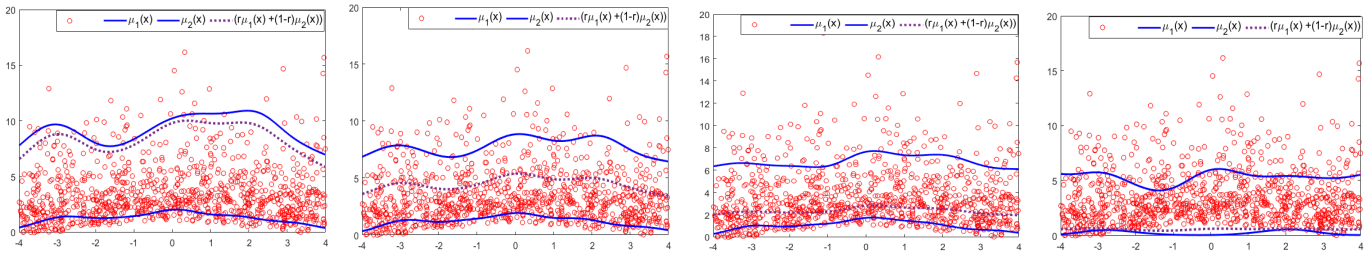


Figure 3: Tube loss-based SVPI estimation for (a)$1-\alpha=0.8$ and (b) $1-\alpha=0.8$.



(a) r= 0.8 (b) r = 0.5 (c) r= 0.3 (d) r= 0.1

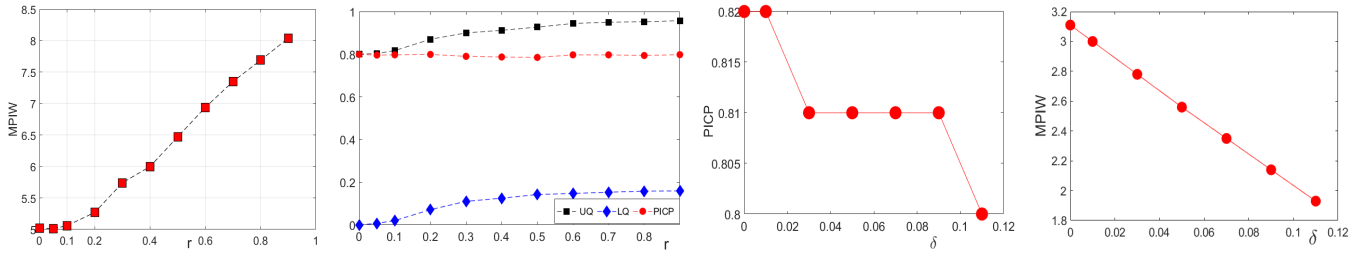
Figure 4: Location of PI tube changes with r values in Tube loss based kernel machine.

The model is trained on 500 data points, with the remaining samples reserved for testing. For constructing linear PI tubes, a linear kernel is employed. For nonlinear PI estimation, we adopt the radial basis function (RBF) kernel defined as

$$k(x_1, x_2) = e^{-\gamma\|x_1-x_2\|^2},$$

where γ is a suitably chosen hyperparameter.

For dataset $\mathbf{D_1}$, the Tube Loss function is applied with a linear kernel to construct **PIs** at nominal confidence levels of $1-\alpha=0.8$ and $1-\alpha=0.9$, using the default parameter settings $r=0.5$ and $\delta=0$. The obtained **PICP** and **MPIW** values for the test dataset corresponding to $1-\alpha=0.8$ (0.9) are 0.799 (0.909) and 2.10 (2.78), respectively. Figure 3 illustrates the estimated PIs on the test data for both target confidence levels, $1-\alpha=0.8$ and $1-\alpha=0.9$.



(a) (b) (c) (d)

Figure 5: Plot of (a) r against MPIW ,(b) PCIP, UQ and LQ for dataset **B**. Plot of (c) δ against PICP,(d) δ against MPIW on the Servo dataset.

	Model	(q+1-α,q)/(r,δ)	PICP	MPIW	Time (s)
$\mathbf{D_1}$	Q Ker M	(0.95,0.15)	0.78 ± 0.017	2.088 ± 0.109	219.38
	Q Ker M	(0.90,0.10)	0.78 ± 0.019	2.002 ± 0.068	221.72
	Q Ker M	(0.85,0.05)	0.79 ± 0.025	2.151 ± 0.078	217.38
	T Ker M	(0.6,0)	0.80 ± 0.012	2.165 ± 0.068	56.93
	T Ker M	(0.5,0)	0.80 ± 0.018	2.156 ± 0.069	61.50
$\mathbf{D_2}$	Q Ker M	(0.90,30)	0.59 ± 0.032	4.6609 ± 0.153	138.05
	Q Kerl M	(0.95,0.35)	0.58 ± 0.023	5.5369 ± 0.272	146.60
	Q Ker M	(0.80,20)	0.59 ± 0.027	3.6196 ± 0.143	146.41
	T Ker M	(0.3,0)	0.60 ± 0.032	3.495 ± 0.168	39.07
	T Ker M	(0.2,0)	0.601 ± 0.028	3.174 ± 0.165	41.55

Table 1: Quantile-based Kernel Machine (Q ker M) and Tube loss-based kernel Machine (T ker M) on dataset $\mathbf{D_1}$ and $\mathbf{D_2}$. Q ker M involves parameters (q+1-α,q) and T ker M involves (r, δ).

For dataset $\mathbf{D_2}$, we construct the **PI** corresponding to $1 - \alpha = 0.8$ using the RBF kernel. The Tube Loss-based kernel machine with $r = 0.5$ and $\delta = 0$ produces a centered interval, as shown in Figure 4(b), yielding **PICP** and **MPIW** values of 0.79 and 6.47, respectively. Owing to the positively skewed nature of the data distribution, such a centered interval is expected to produce a relatively wider **PI** (i.e., higher **MPIW**) compared to a **HQ PI**. To effectively capture the denser regions of the data, the **PI** bounds therefore need to be shifted downward.

Figures 4(a), 4(c), and 4(d) depict the **PIs** obtained from the Tube Loss-based kernel machines for $r = 0.8$, 0.3, and 0.1, respectively. It is evident that as r decreases, the estimated **PI** bounds move downward, covering the denser portions of the data and thereby reducing the **MPIW**.

Figure 5(a) presents the variation of **MPIW** with respect to r, clearly showing that smaller values of r lead to lower **MPIW**, as expected. Figure 5(b) displays the plots of **PICP**, lower quantile (LQ), and upper quantile (UQ) against r, where LQ and UQ denote the proportions of y_i values falling below the upper and lower bounds of the estimated PI tube, respectively. The plots indicate that changing r affects both LQ and UQ but leaves **PICP** largely unchanged. Notably, reducing r from 0.5 to 0.1 results in a significant improvement in **MPIW** by approximately 21.80%, without compromising the desired coverage.

It is important to note that for dataset $\mathbf{D_1}$, where the noise terms are drawn from a symmetric distribution, the default choice of $r = 0.5$ yields a **HQ PI**. In contrast, for dataset $\mathbf{D_2}$, characterized by a positively skewed distribution, obtaining a good quality **PI** requires reducing the value of r from its default setting of 0.5.

Finally, we compare the performance of **PIs** obtained using the Tube Loss with those based on the pinball loss, where the latter estimates the two bounds $\hat{F}_q(\boldsymbol{x})$ and $\hat{F}_{q+(1-\alpha)}(\boldsymbol{x})$ separately. We generate 10 independent replicates each of datasets $\mathbf{D_1}$ and $\mathbf{D_2}$. For every replicate, we compute the **PICP** and **MPIW** of the PIs derived from both methods at nominal confidence levels of $1-\alpha = 0.8$ and 0.6, respectively.

It is worth noting that for quantile-based **PI** estimation, the choice of the parameter q, and for Tube Loss-based estimation, the choice of the parameter r, are both critical for producing **HQ** intervals. Table 1 summarizes the comparative results. The Tube Loss-based method not only offers a significant reduction in overall training complexity, thereby decreasing training time, but also consistently achieves the desired nominal confidence level across both datasets.

For dataset $\mathbf{D_1}$, the centered interval corresponding to $q = 0.1$ under the pinball loss framework is expected to perform best. Although its **PICP** falls marginally short of the target value of 0.8, it yields a slightly narrower interval compared to the Tube Loss-based PI with $r = 0.5$. As expected for symmetric noise distributions, the Tube Loss with $(r = 0.5, \delta = 0)$ provides the most reliable **PI** for dataset $\mathbf{D_1}$.

For dataset $\mathbf{D_2}$, however, the **MPIW** values of the estimated **PIs** can be substantially improved by reducing the value of r, which aligns with expectations since the noise component follows a positively skewed distribution.

2. Choice of δ

As discussed earlier, we now demonstrate how an appropriate choice of the recalibration parameter δ (cf. Equation (18)) can further reduce the **MPIW** on the test data. To illustrate this, we consider the well-known Servo dataset (Karl Ulrich (2016)), which contains 167 observations across five variables. A randomly selected subset comprising 10% of the data points is used as the test set, while the remaining data are utilized for training with a 10-fold cross-validation procedure to obtain a Tube Loss-based **PI** (using the linear kernel) at a nominal confidence level of 0.8, with default settings of r and δ. The mean **PICP** observed on the validation set is 0.82. Since the empirical **PICP** exceeds the target level of 0.8, there exists an opportunity to reduce the **MPIW** on the test set by selecting a positive value of δ. We incrementally increase δ until the validation **PICP** reaches the target level of 0.8, as depicted in Figure 5(c). This adjustment results in approximately a 44% reduction in **MPIW** on the test set, as shown in Figure 5 (d), while preserving the desired coverage.

4.2 Tube Loss in Neural Network (NN)

The Tube Loss-based PI estimation model within the **NN** framework aims to solve the following optimization problem using the training set $T = \{(\boldsymbol{x}_i, y_i) : i = 1, 2, \ldots, n\}$:

$$\min_{(\mu_1,\mu_2)} \left[\sum_{i=1}^{n} \mathcal{L}^{r}_{(1-\alpha)}\big(y_i, \mu_1(\boldsymbol{x}_i), \mu_2(\boldsymbol{x}_i)\big) + \ \delta \sum_{i=1}^{n} \big|\mu_2(\boldsymbol{x}_i) - \mu_1(\boldsymbol{x}_i)\big| \right], \tag{20}$$

where $\mu_1(\boldsymbol{x})$ and $\mu_2(\boldsymbol{x})$ represent the two output neurons of the dense network corresponding to the lower and upper bounds of the **PI**, respectively. The weights of both the hidden and output layers are iteratively updated through gradient-based optimization using backpropagation of the loss function.

4.2.1 Comparison with QD Loss

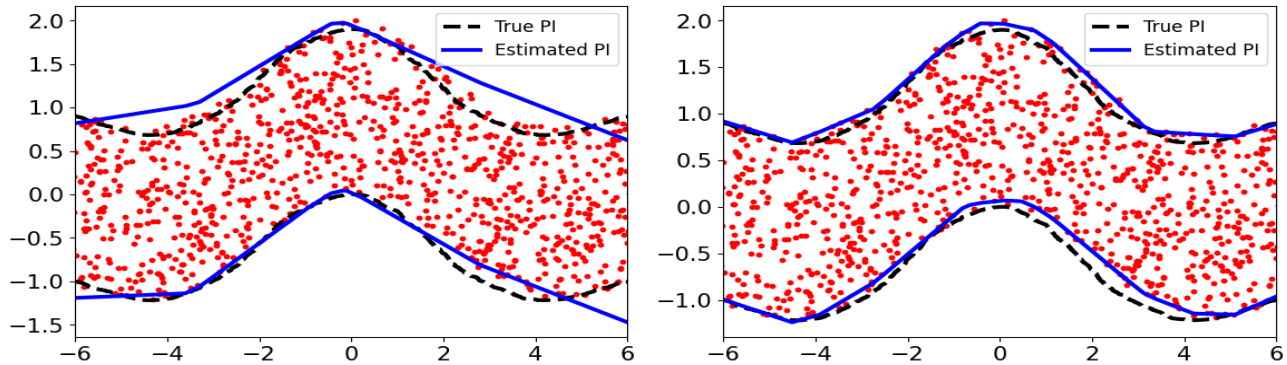
Within the **NN** framework, Pearce et al. (2018) demonstrated that **PIs** obtained using the **QD** loss yield slightly superior performance compared to those derived from the **LUBE** approach proposed by Khosravi et al. (2011). Furthermore, they observed that the **QD** method also outperforms the mean-variance estimation (MVE) approach of Nix & Weigend (1994) on several benchmark datasets.

In this study, we conduct a simple experiment to show that the Tube Loss-based **PI** provides better performance than the **QD**-based **PI** in terms of the *sum of mean squared errors (SMSE)*—a local measure of smoothness and tightness introduced earlier in Section 4. To this end, we generate 1000 data points (x_i, y_i) according to the following model:

$$y_i = \frac{\sin(x_i)}{x_i} + \epsilon_i, \tag{21}$$

where $x_i \sim U(-2\pi, 2\pi)$ and $\epsilon_i \sim U(-1, 1)$.

Both the Tube Loss-based and QD Loss-based NN models are trained to obtain **PIs** at a nominal confidence level of 0.95. After hyperparameter tuning, the **NN** architecture is fixed with 100 ReLU-activated hidden neurons and two output neurons. The *Adam* optimizer is employed for



(a) QD loss-based NN (b) Tube loss-based NN

Figure 6: Comparison between Tube Loss and QD Loss function-based neural networks.

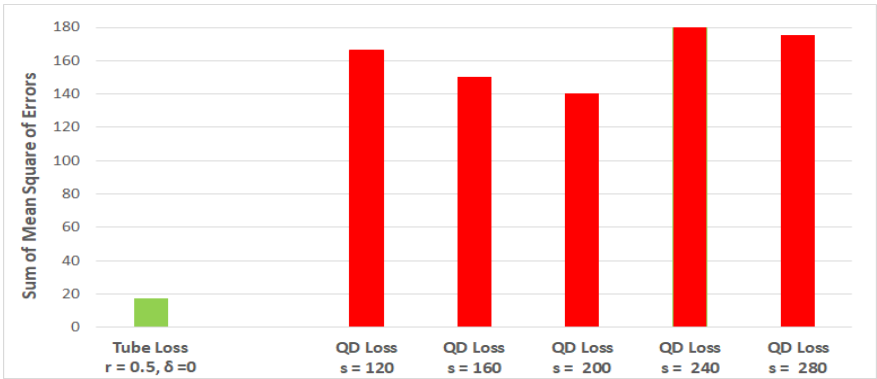


Figure 7: Tube Loss-based NN approximates the true PI more closely than the QD loss-based NN.

model training. Figure 6(a) presents the **PI** estimated using the QD loss, while Figure 6(b) displays the **PI** obtained via the Tube Loss with $r = 0.5$ $\delta = 0$, alongside the true **PI**. It is evident that the **PI** generated by the **Tube Loss** is smoother and more compact compared to that produced by the **QD** loss. The true **PI** is computed from the 97.5% and 2.5% quantiles of the conditional distribution of $y|x$ for various values of x.

The observed (**PICP**, **MPIW**) values for the **PIs** obtained using the **Tube loss** and **QD** loss are $(0.95, 1.78)$ and $(0.98, 2.17)$, respectively. For the **QD** loss, proper adjustment of the softening parameter s—which governs the approximation of the step function via the product of sigmoid functions—is crucial but often cumbersome. Through the tuning process, the optimal parameters for the **QD** loss were identified as $s = 200$ and $\lambda = 0.1$.

Subsequently, we computed the **SMSEs** of the resulting **PIs**. Figure 7 illustrates the comparison of **SMSE** values between the **Tube loss** and **QD** loss for various choices of the softening parameter s. It is evident that the performance of the **QD** loss-based **PI** is highly sensitive to the selection of s, and consequently, to the fidelity of the step function approximation achieved by the sigmoid functions.

4.2.2 Comparison with RQR loss

We write the **RQR** loss function introduced in (13) as follows,

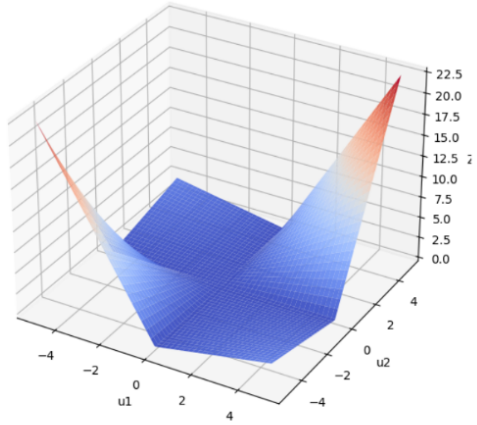
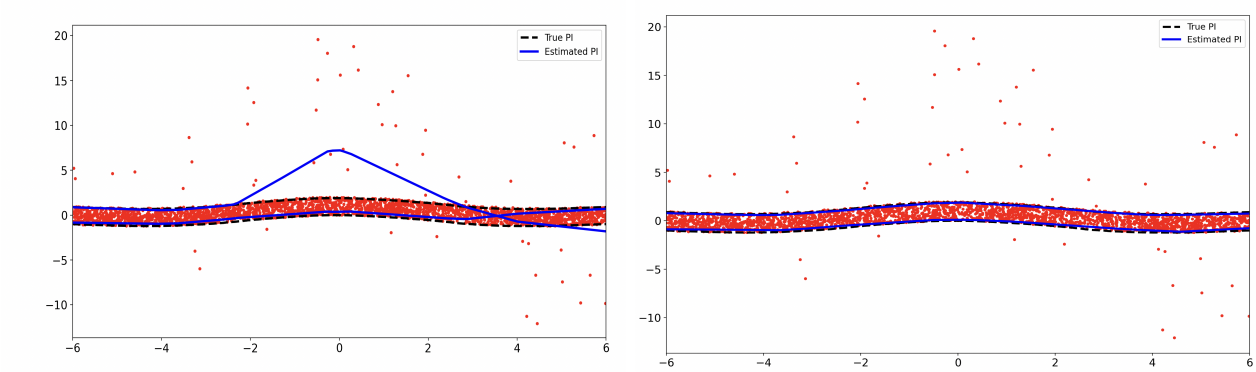


Figure 8: RQR loss function for t=0.9



(a) RQR loss based NN

(b) Tube loss based NN

Figure 9: Comparison of RQR loss and Tube loss function based NNs

$$\mathcal{L}_{1-\alpha}^{RQR} = \begin{cases} (1-\alpha)\kappa, & \text{if} \quad \kappa \geq 0, \\ -\alpha\kappa, & \text{if} \quad \kappa < 0, \end{cases} \tag{22}$$

where, $\kappa = (y - \mu_1)(y - \mu_2) = u_1 u_2$.

Figure 8 shows a plot of the **RQR** loss function for $1 - \alpha = 0.9$. The **RQR** loss function is continuous but non-convex function of $u_1 = (y - \mu_1)$ and $u_2 = (y - \mu_1)$. As stated before, the **RQR PI** bounds are sensitive to outliers. The presence of outliers may trigger the crossing of the two bounds, thus adversely affecting the conditional coverage of the **PI** near the crossing. We perform a simple experiment to show the impact of outliers on **RQR** based **PI** as stated above. It is worth mentioning in this context that for comparison with **Tube loss**, we consider the RQR-Width Minimizing (**RQR-W**) method. We generate an artificial data set $\{(x_i, y_i), i = 1, 2, ...3000\}$ using the relation.

$$y_i = \frac{sin(x_i)}{x_i} + \epsilon_i, \tag{23}$$

where x_i is $U(-2\pi, 2\pi)$ and ϵ_i is from $U(-1, 1)$. Next, we contaminate the dataset by adding 60 (2%) data points $\{(x_i, 10y_i), i = 3001, 3002, ...3060\}$, where $\{(x_i, y_i), i = 3001, 3002, ...3060\}$ are

generated randomly. The **Tube Loss** and **RQR** Loss-based **NNs** were trained using a single-layer **NN** with 100 hidden neurons, ReLU activation, a batch size of 100, 1200 epochs, and the Adam optimizer for a target confidence level of 0.8. For both loss functions, the penalty parameters δ in **Tube Loss** and λ in **RQR** were set to zero. Figure 9 presents the results of **RQR**- and **Tube Loss**–based **NNs** with $r = 0.3$. In 9(a), the **RQR**-based **PI** deviates considerably from the true **PI** (black dotted line), and the **PI** bounds cross each other near $x = 4$. In contrast, the **Tube Loss**-based PI almost overlaps with the true **PI**, effectively handling the asymmetric noise introduced by the outliers. This example clearly supports our apprehension stated above.

Dataset	TUBE	RQR-W	QR	SQR-C	SQR-N	QD loss
miami	89.34 (0.45)	88.89 (0.32)	90.50 (0.57)	87.36 (0.46)	85.97 (0.48)	90.26 (0.49)
	0.49 (0.02)	0.51 (0.01)	**0.51** (0.01)	1.90 (0.05)	1.52 (0.03)	1.74 (0.12)
wine	91.21 (0.37)	90.16 (0.37)	89.97 (0.36)	91.94 (0.67)	88.19 (0.97)	88.03 (0.92)
	0.26 (0.017)	0.33 (0.01)	0.28 (0.00)	0.49 (0.03)	0.48 (0.03)	1.18 (0.12)
power	90.31 (0.54)	89.34 (0.55)	89.92 (0.54)	91.81 (1.34)	89.11 (1.40)	62.47 (13.63)
	0.04 (0.009)	0.07 (0.01)	0.06 (0.01)	0.15 (0.03)	0.15 (0.03)	1.14 (0.23)
yacht	89.32 (0.41)	90.32 (0.42)	91.45 (1.23)	85.97 (1.18)	84.68 (1.69)	90.32 (0.96)
	0.16 (0.019)	0.33 (0.02)	0.32 (0.02)	3.56 (0.38)	2.91 (0.26)	1.66 (0.29)
kin8nm	89.00 (0.32)	89.27 (0.40)	91.89 (0.27)	87.16 (0.49)	85.53 (0.53)	89.52 (0.46)
	0.42 (0.01)	0.42 (0.01)	**0.50** (0.01)	1.14 (0.01)	1.10 (0.01)	1.61 (0.12)
protein	92.16 (0.29)	88.47 (0.30)	90.40 (0.23)	89.01 (0.20)	86.01 (0.32)	90.38 (0.30)
	1.68 (0.02)	1.50 (0.01)	**1.61** (0.01)	2.19 (0.02)	2.05 (0.01)	1.96 (0.09)
boston	91.58 (0.48)	88.14 (0.66)	86.67 (1.06)	82.55 (1.48)	81.18 (1.58)	90.29 (0.77)
	0.44 (0.025)	0.40 (0.02)	0.38 (0.01)	1.09 (0.03)	1.01 (0.02)	1.41 (0.15)
energy	90.08 (0.42)	89.68 (0.70)	93.05 (0.88)	89.42 (0.99)	86.30 (0.99)	89.42 (0.76)
	0.23 (0.016)	0.23 (0.01)	0.29 (0.01)	1.31 (0.01)	1.23 (0.01)	1.39 (0.17)
sulfur	92.00 (0.36)	88.87 (0.20)	88.30 (0.34)	87.83 (0.47)	87.40 (0.50)	89.41 (0.53)
	1.06 (0.027)	1.02 (0.01)	1.05 (0.01)	1.09 (0.03)	1.02 (0.02)	1.36 (0.07)
cpu_act	91.03 (0.35)	89.14 (0.37)	88.94 (0.48)	90.85 (0.75)	86.73 (1.10)	90.32 (0.62)
	0.44 (0.014)	0.44 (0.00)	0.41 (0.01)	0.78 (0.01)	0.76 (0.02)	N/A* (N/A*)
concrete	91.12 (0.52)	88.74 (0.66)	87.77 (1.09)	86.02 (1.29)	85.29 (1.39)	89.37 (0.68)
	0.46 (0.026)	0.47 (0.03)	0.44 (0.01)	1.39 (0.03)	1.33 (0.02)	1.28 (0.16)
naval	91.01 (0.34)	88.81 (0.22)	88.34 (0.25)	90.70 (1.68)	88.86 (1.77)	91.16 (0.35)
	0.02 (0.008)	0.03 (0.00)	0.02 (0.00)	0.26 (0.04)	0.26 (0.04)	2.93 (0.18)

Table 2: Comparisons of Tube loss based intervals against that of based on RQR-W and other existing baseline NN methods on 12 datasets for a PI estimation task with target $1 - \alpha$ =0.90. The first row reports the PICP, while the second row reports MPIW. All results represent the test set mean over 10 runs ($\pm$ standard error). Best results are in bold; the best PI method attains average confidence level at least 0.90 with the lowest MPIW. Tunned parameters are listed in Appendix C.

4.2.3 Comparison With Baseline Methods Using Benchmark Datasets

Next, we assess the performance of the **Tube Loss**-based **PI** in comparison with the baseline approaches discussed in Section 2, using twelve benchmark datasets. In particular, we examine the prediction intervals generated by **RQR-W** (Pouplin et al. (2024)), **QRNN** (Quantile Regression Neural Network), **SQR-C** (centered intervals) (Tagasovska & Lopez-Paz (2019)), **SQR-N** (potentially non-centered and typically the narrowest) (Tagasovska & Lopez-Paz (2019)), and **QD** (Pearce et al. (2018)). The numerical results summarized in Table 2 are produced under the experimental setup described in (Pouplin et al. (2024)). Details regarding the selection of the tuning parameters r and δ are presented in Appendix C.

Among the twelve datasets considered, the **Tube loss**-based intervals exhibit superior performance in nine cases, whereas the QR-based intervals perform best in the remaining three. These results clearly

demonstrate the advantage of **Tube loss**-based intervals over existing approaches in generating high-quality prediction intervals (**PIs**) within the **NN** framework.

4.3 Tube Loss Based deep Probabilistic Forecasting

Further, we use the Tube loss function in Long Short Term Memory (LSTM) Network (Hochreiter & Schmidhuber (1997)) for obtaining the probabilistic forecast upon popular benchmark time-series datasets namely Electric (BP & Ember. (2016)), Sunspots (SIDC & Quandl.), SWH (NDBC),Temperature (machinelearningmastery.com) , Female Birth (datamarket.com) and Beer Production (Australian (1996)). Also, we compare the Tube loss based LSTM (T-LSTM) with Quantile based LSTM (Q-LSTM) on these datasets. The 70% initial data points were considered for training and rest of them were test set. Out of training sets, last 10% of observations were used as validation set. For each dataset, we tuned the LSTM architecture, dropout, and window size based

Dataset	PICP		MPIW		Training Time (SEC)		Improvement	
	Q-LSTM	T- LSTM	Q-LSTM	T-LSTM	Q-LSTM	T-LSTM	in Time(s)	Better
Electric	0.95	0.95	18.23	**17.02**	145	58	60 %	√
Sunspots	0.93	**0.95**	121.09	113.98	839	442	47 %	√
SWH	0.96	0.96	0.52	**0.35**	5119	2733	46 %	√
Temperature	**0.95**	0.94	24.82	15.56	1135	447	60 %	
Female Birth	0.95	0.96	28.20	**28.09**	118	43	63 %	√
Beer Production	0.94	**0.95**	134.8	42.91	132.8	89.6	33 %	√

Table 3: Comparison of Quantile based LSTM (Q-LSTM) and Tube loss based LSTM (T-LSTM) on real-world time-series datasets with target coverage $t = 0.95$. Tunned parameters and plots obtained by the T-LSTM are listed in Appendix C.

on prior literature, then applied Tube Loss and the quantile approach separately to target a 0.95 PI. The T-LSTM model obtains better performance on five cases out of 6 datasets. It also tends to obtain lower MPIW than Q-LSTM model, as the δ parameter facilitates the re-calibration for capturing the narrower PI in T-LSTM model. But, main advantages of T-LSTM over the Q-LSTM is the significant improvement in training time . It is because that the Q-LSTM needs to be trained twice where as T-LSTM requires only single cycle of training.

In a distribution-free setting, quantile-based deep learning models are the most popular choice for probabilistic forecasting. However, we also need to compare the performance of the Tube loss based probabilistic forecasting with that of the deep learning models trained with the QD loss function (Pearce et al. (2018)) . A significant practical challenge in training deep learning models with the QD loss function (Pearce et al. (2018)) is the frequent occurrence of NaN losses during computation. However, the extended Quality Driven (QD^+) loss function (Salem et al. (2020)) solves this problem up to some extent. Consequently, we have implemented the QD^+ loss function based deep forecasting models for obtaining the probabilistic forecast. We tune the parameter values of QD+ loss function based LSTM well and compare its performance with the Tube loss based LSTM model on time-series benchmark dataset at Table 4. Unlike, the Quantile and Tube loss based LSTM models, the QD+ loss function based LSTM model fails to obtain the consistent performance. Out of six datasets, it fails to obtain the target coverage on three datasets. In case of Sunspots and Temperature datasets, the QD^+ loss function based LSTM obtains very poor performance. Additionally, across all datasets, the Tube loss LSTM models outperform the QD+ loss LSTM models.

Dataset	PICP		MPIW		Improvement	
	QD^+-LSTM	T- LSTM	QD^+-LSTM	T-LSTM	in MPIW	Better
Electric	0.96	0.95	54.63	**17.02**	68.84 %	√
Sunspots	0.43	**0.95**	24.06	113.98	NA	√
SWH	0.96	0.96	0.36	**0.35**	2.78%	√
Female Birth	0.94	**0.96**	38.98	28.09	27.94%	√
Temperature	0.79	**0.95**	5.94	15.56	73.13 %	√
Beer Production	0.96	0.95	159.71	**42.91**	NA	√

Table 4: Comparison of Extended Quality Driven loss based LSTM (QD$^+$ LSTM) and Tube loss based LSTM (T-LSTM) on real-world time-series datasets

For numerical results presented in the Table 3, the tuned parameters are detail in the Table 7. Also, the plot obtained by the Tube loss based LSTM for benchmark datasets reported in Table 3, are shown in the 11.

Application to wind probabilistic forecasting

We have also employed the Tube loss function in LSTM, GRU and TCN for probabilistic forecasting of wind speed. Table 5 lists the performance of the Tube loss base deep learning models along with the recent benchmark models used for probabilistic forecasting in energy domain on San Francisco wind dataset containing 26,304 hourly observations.The last 30% of observations were used as the test set, while the final 10% of the training data was reserved for validation.

Rank	**Model**	**PICP**	**MPIW**	**MPIW/PICP**
1	TCN + Tube	0.9543	4.560	4.78
2	LSTM + Tube	0.9561	4.627	4.84
3	GRU + Tube	0.9507	4.857	5.11
4	GRU + Quantile	0.951	4.955	5.21
5	LSTM + Quantile	0.9594	5.407	5.64
6	TCN + QD$^+$	0.9505	5.490	5.78
7	LSTM + QD$^+$	0.9734	6.043	6.21
8	Deep AR	0.9855	6.2023	6.29
9	GRU + QD$^+$	0.9724	6.309	6.49
10	MDN	0.949	4.620	4.87
11	TCN + Quantile	0.9428	4.908	5.21
12	TimeGPT	0.9357	11.235	12.00

Table 5: Ranking of different deep probabilistic forecasting methods including Deep AR (Salinas et al. (2020)), Mixed Density Network (MDN) (Bishop (1994)), QD $^+$ based deep learning models (Salem et al. (2020)) and Time GPT (Garza et al. (2023)) on San Francisco Dataset with target coverage $t = 0.95$. DeepAR (Salinas et al. (2020)) has recently gained traction in probabilistic forecasting, including wind power prediction (Arora et al. (2022)), while studies (Yang et al. (2021), Zhang et al. (2020), Men et al. (2016)) highlight the effectiveness of Mixture Density Networks (MDN) for wind forecasting.

Table 5 presents the performance of the Tube-loss–based deep forecasting models alongside several recent baseline models commonly used in the wind-energy literature for probabilistic forecasting. The Tube-loss–based deep learning models consistently achieve the best overall performance among all compared baselines. We do not include Adaptive Conformal Inference (ACI) (Gibbs & Candes (2021)) and related conformal methods as baselines because they follow a fundamentally different conformal framework rather than a prediction PI based approach. Moreover, these methods are computationally expensive and require retraining from scratch to generate forecasts at each future time step.

5 Future Work

The Tube loss function evidently offers a superior method for PI estimation in kernel machines and NN. In future work, we are planning to explore its potential applications in producing high-quality PIs in other important regression settings, such as conformal regression and adaptive conformal inference under distribution shift. It would also be worthwhile to implement the proposed methodology for probabilistic forecasting across a range of neural architectures employed in diverse application domains, including solar irradiance, cryptocurrency valuation, exchange rate prediction, stock market trends, ocean wave modeling, pollution estimation, and weather forecasting. In many text processing applications, the target variable is inherently multidimensional. A compelling direction for future work would be to extend the proposed prediction interval estimation methodology to accommodate such multidimensional targets.

References

Parul Arora, Seyed Mohammad Jafar Jalali, Sajad Ahmadian, Bijaya K Panigrahi, Ponnuthurai N Suganthan, and Abbas Khosravi. Probabilistic wind power forecasting using optimized deep auto-regressive recurrent neural networks. *IEEE Transactions on Industrial Informatics*, 19(3): 2814–2825, 2022.

Beer Production Australian. Beer production australian. `https://www.kaggle.com/datasets/sergiomora823/monthly-beer-production`, 1996. Accessed: 10-01-2024.

Christopher M Bishop. Mixture density networks. 1994.

BP and Ember. Electricity production by source (world). `https://www.kaggle.com/datasets/prateekmaj21/electricity-production-by-source-world`, 2016. Accessed: 10-01-2024.

George Chryssolouris, Moshin Lee, and Alvin Ramsey. Confidence interval prediction for neural network models. *IEEE Transactions on neural networks*, 7(1):229–232, 1996.

Youngseog Chung, Willie Neiswanger, Ian Char, and Jeff Schneider. Beyond pinball loss: Quantile methods for calibrated uncertainty quantification. *Advances in Neural Information Processing Systems*, 34:10971–10984, 2021.

Peng Cui, Wenbo Hu, and Jun Zhu. Calibrated reliable regression using maximum mean discrepancy. *Advances in Neural Information Processing Systems*, 33:17164–17175, 2020.

datamarket.com. Daily total female births in california, 1959. `https://www.kaggle.com/datasets/dougcresswell/daily-total-female-births-in-california-1959`. Accessed: 10-01-2024.

Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

JT Gene Hwang and A Adam Ding. Prediction intervals for artificial neural networks. *Journal of the American Statistical Association*, 92(438):748–757, 1997.

1986 Karl Ulrich. Servo. `https://archive.ics.uci.edu/dataset/87/servo`, 2016. Accessed: 10-01-2024.

Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE transactions on neural networks*, 22(3):337–346, 2011.

Roger Koenker and Gilbert Bassett Jr. Regression quantiles. *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4): 143–156, 2001.

machinelearningmastery.com. Daily minimum temperatures in melbourne. `https://www.kaggle.com/datasets/paulbrabban/daily-minimum-temperatures-in-melbourne`. Accessed: 10-01-2024.

David JC MacKay. The evidence framework applied to classification networks. *Neural computation*, 4(5):720–736, 1992.

Zhongxian Men, Eugene Yee, Fue-Sang Lien, Deyong Wen, and Yongsheng Chen. Short-term wind speed and power forecasting using an ensemble of mixture density neural networks. *Renewable Energy*, 87:203–211, 2016.

James Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209(441-458):415–446, 1909.

NDBC. Significant wave height, national data buoy center, buoy station 42001 for 21 april 2021 - 25 july 2021. `https://www.ndbc.noaa.gov/station`$_{h}istory.php?station = 42001$.

David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pp. 55–60. IEEE, 1994.

Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *International conference on machine learning*, pp. 4075–4084. PMLR, 2018.

Thomas Pouplin, Alan Jeffares, Nabeel Seedat, and Mihaela Van Der Schaar. Relaxed quantile regression: Prediction intervals for asymmetric noise. *arXiv preprint arXiv:2406.03258*, 2024.

Tárik S Salem, Helge Langseth, and Heri Ramampiaro. Prediction intervals: Split normal mixture from quality-driven deep ensembles. In *Conference on Uncertainty in Artificial Intelligence*, pp. 1179–1187. PMLR, 2020.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3): 1181–1191, 2020.

SIDC and Quandl. Sunspots. `https://www.kaggle.com/datasets/robervalt/sunspots`. Accessed: 10-01-2024.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *Advances in neural information processing systems*, 32, 2019.

Ichiro Takeuchi, Quoc Le, Timothy Sears, Alexander Smola, et al. Nonparametric quantile estimation. 2006.

Luoxiao Yang, Zhong Zheng, and Zijun Zhang. An improved mixture density network via wasserstein distance based adversarial learning for probabilistic wind speed predictions. *IEEE Transactions on Sustainable Energy*, 13(2):755–766, 2021.

Hao Zhang, Yongqian Liu, Jie Yan, Shuang Han, Li Li, and Quan Long. Improved deep mixture density network for regional wind power probabilistic forecasting. *IEEE Transactions on Power Systems*, 35(4):2549–2560, 2020.

A Proof of the Lemma 1

Let us assume that $\#\{y_i|y_i = \mu_2^*\} = k_1$, $\#\{y_i|y_i = r\mu_2^* + (1-r)\mu_1^*\} = k_2$, and $\#\{y_i|y_i = \mu_1^*\} = k_3$, where $\#$ represents the cardinality of a set. In other words, k_1, k_2 and k_3 represent the number of points on the boundary sets.

Thus, we obtain $n_1 + k_1 + n_2 + k_2 + n_3 + k_3 + n_4 = n$.

Let us choose, $\delta_1^* \in [0, \epsilon_1)$, $\delta_2^* \in [0, \epsilon_2)$, where ϵ_1 and ϵ_2 are positive real numbers, such that $\epsilon_1 < \min_{y_i < \mu_1^*}(|\mu_1^* - y_i|)$, $\epsilon_2 < \min_{y_i > \mu_2^*}|(y_i - \mu_2^*)|$, and $r\epsilon_2 + (1-r)\epsilon_1 < \min_{\mu_1^* < y_i < \mu_2^*}|y_i - (r\mu_2^* + (1-r)\mu_1^*)|$, which entails

$$\#\{y_i : \mu_2^* < y_i \leq \mu_2^* + \delta_2^*\} = 0, \quad \#\{y_i : \mu_2^* < y_i \leq \mu_2^* - \delta_2^*\} = 0, \quad \#\{y_i : \mu_1^* < y_i \leq \mu_1^* + \delta_1^*\} = 0,$$
$$\#\{y_i : \mu_1^* < y_i \leq \mu_1^* - \delta_1^*\} = 0, \quad \#\{r\mu_2^* + (1-r)\mu_1^* < y_i \leq r(\mu_2^* + \delta_2^*) + (1-r)(\mu_1^* + \delta_1^*)\} = 0,$$
$$\#\{r\mu_2^* + (1-r)\mu_1^* < y_i \leq r(\mu_2^* - \delta_2^*) + (1-r)(\mu_1^* - \delta_1^*)\} = 0.$$

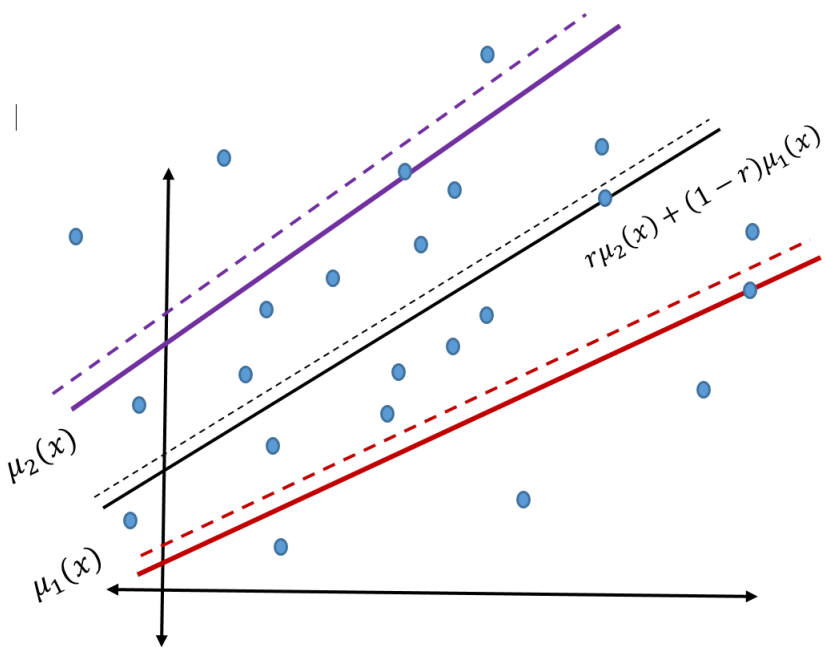


Figure 10

Given that (μ_1^*, μ_2^*) is an optimal solution, we have $\sum_{i=1}^n \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^n \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*) \geq 0$. We now evaluate the difference of the sums for each of the following ten sets inducing a partition of $\mathbb{R}$.

$R_1 = \{y_i : y_i > \mu_2^* + \delta_2^*\}$, $R_2 = \{y_i : \mu_2^* < y_i \leq \mu_2^* + \delta_2^*\}$, $R_3 = \{y_i : y_i = \mu_2^*\}$,
$R_4 = \{y_i : r(\mu_2^* + \delta_2^*) + (1-r)(\mu_1^* + \delta_1^*) < y_i < \mu_2^*\}$, $R_5 = \{y_i : r\mu_2^* + (1-r)\mu_1^* < y_i \leq r(\mu_2^* + \delta_2^*) + (1-r)(\mu_1^* + \delta_1^*)\}$, $R_6 = \{y_i : y_i = r\mu_2^* + (1-r)\mu_1^*\}$, $R_7 = \{y_i : \mu_1^* + \delta_1^* < y_i < r\mu_2^* + (1-r)\mu_1^*\}$,
$R_8 = \{y_i : \mu_1^* < y_i \leq \mu_1^* + \delta_1^*\}$ $R_9 = \{y_i : y_i = \mu_1^*\}$, $R_{10} = \{y_i : y_i < \mu_1^*\}$.

Denote the difference $\sum_{y_i \in R_i} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{y_i \in R_i} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*)$ by Δ_i. The simple calculation leads to the following values of $\Delta_i, i = 1, 2, ..., 10$: $\Delta_1 = -tn_1\delta_2^*, \Delta_2 =$

$0, \Delta_3 = (1-t)k_1\delta_2^*, \Delta_4 = (1-t)n_2\delta_2^*, \Delta_5 = 0, \Delta_6 = k_2(1-t)(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*), \Delta_7 = -(1-t)n_3\delta_1^*, \Delta_8 = 0, \Delta_9 = tk_3\delta_1^*, \Delta_{10} = tn_4\delta_1^*$.

Thus, we obtain

$$\sum_{i=1}^{n} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^{n} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*) = -tn_1\delta_2^* + (1-t)k_1\delta_2^* + (1-t)n_2\delta_2^* + (1-t)k_2(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*) - (1-t)n_3\delta_1^* + tk_3\delta_1^* + tn_4\delta_1^* \geq 0. \quad (24)$$

If we assume $\delta_1^* = 0$ and $\delta_2^* > 0$, the above inequality entails

$$\frac{1-t}{t} \geq \frac{n_1\delta_2^*}{n_2\delta_2^* + k_1\delta_2^* + k_2((2-r)\mu_1^* - (1-r)\mu_2^*)}.$$

Given that y_i's are the realizations of a continuous random variable with no discrete probability mass, $k_1/m, k_2/m \to 0$ with probability 1 as $m \to \infty$. Thus, as $m \to \infty$, with probability 1,

$$\frac{n_1}{n_2} \leq \frac{1-t}{t}. \quad (25)$$

.

Arguing on a similar line, if we consider $\delta_2^* = 0$ and $\delta_1^* > 0$ in 24, then we have the following inequality

$$\frac{(k_3 + n_4)\delta_1^*}{n_3\delta_1^* - k_2(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*)} \geq \frac{1-t}{t}$$

As $m \to \infty$, we can state that the following inequality holds with probability 1,

$$\frac{n_4}{n_3} \geq \frac{1-t}{t}. \quad (26)$$

Again, starting with (μ_1^*, μ_2^*) as an optimal solution, we have $\sum_{i=1}^m \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* - \delta_1^*, \mu_2^* - \delta_2^*) - \sum_{i=1}^m \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*) \geq 0$. For computing this, let us evaluate the difference of the sums for each of the following ten sets inducing a partition of $\mathbb{R}$.

$R_1' = \{y_i : y_i > \mu_2^*\}$, $R_2' = \{y_i : y_i = \mu_2^*\}$, $R_3' = \{y_i : \mu_2^* - \delta_2^* \leq y_i < \mu_2^*\}$, $R_4' = \{y_i : r\mu_2^* + (1-r)\mu_1^* < y_i < \mu_2^* - \delta_2^*\}$, $R_5' = \{y_i : y_i = r(\mu_2^*) + (1-r)\mu_1^*\}$, $R_6' = \{y_i : r(\mu_2^* - \delta_2^*) + (1-r)(\mu_1^* - \delta_1^*) \leq y_i < r\mu_2^* + (1-r)\mu_1^*\}$, $R_7' = \{\mu_1^* < y_i < r(\mu_2^* - \delta_2^*) + (1-r)(\mu_1^* - \delta_1^*)\}$, $R_8' = \{y_i : y_i = \mu_1^*\}$, $R_9' = \{y_i : \mu_1^* < y_i \leq \mu_1^* - \delta_1^*\}$, $R_{10}' = \{y_i : y_i < \mu_1^* - \delta_1^*\}$.

Denoting $\sum_{y_i \in R_i} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* - \delta_1^*, \mu_2^* - \delta_2^*) - \sum_{y_i \in R_i} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*)$ by Δ_i' we obtain the following values of $\Delta_i', i = 1, 2, ..., 10$: $\Delta_1' = tn_1\delta_2^*$, $\Delta_2' = -(1-t)k_1\delta_2^*$, $\Delta_3' = 0$, $\Delta_4' = -(1-t)n_2\delta_2^*$, $\Delta_5' = -(1-t)k_2\delta_2^*$, $\Delta_6' = 0$, $\Delta_7' = (1-t)n_3\delta_1^*$, $\Delta_8' = (1-t)k_3\delta_1^*$, $\Delta_9' = 0$, $\Delta_{10}' = -tn_4\delta_1^*$.

Thus, we obtain

$$\sum_{i=1}^{n} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^{n} \mathcal{L}_{1-\alpha}^r(y_i, \mu_1^*, \mu_2^*) = tn_1\delta_2^* - (1-t)k_1\delta_2^* - (1-t)n_2\delta_2^* - (1-t)k_2\delta_2^* + (1-t)n_3\delta_1^* + (1-t)k_3\delta_1^* - tn_4\delta_1^* \geq 0. \quad (27)$$

Now, if we assume $\delta_1^* = 0$ and $\delta_2^* > 0$ in (27), then we obtain,

$$\frac{n_1}{k_1 + n_2 + k_2} \geq \frac{1-t}{t},$$

which entails

$$\frac{n_1}{n_2} \geq \frac{1-t}{t}, \tag{28}$$

as $n \to \infty$ with probability 1.

Similarly, considering $\delta_2^* = 0$ and $\delta_1^* > 0$ in (27), we obtain as $n \to \infty$,

$$\frac{n_4}{n_3} \leq \frac{1-t}{t}, \tag{29}$$

holds with probability 1.

Thus, (25) and (28) imply that as $n \to \infty$

$$\frac{n_1}{n_2} = \frac{1-t}{t} \tag{30}$$

holds with probability 1.

Similarly, combining (26) and (28), we obtain as $n \to \infty$,

$$\frac{n_4}{n_3} = \frac{1-t}{t} \tag{31}$$

holds with probability 1.

Thus, combining 30 and 31, we obtain, as $m \to \infty$,

$$\frac{n_1 + n_4}{n_2 + n_3} = \frac{1-t}{t} \tag{32}$$

holds with probability 1.

B Gradient Descent method for Tube loss-based kernel machines

The Tube loss-based kernel machine estimates a pair of functions

$$\mu_2(x) := \sum_{i=1}^{n} k(x_i, x)\alpha_i + b_1 \quad \text{and} \quad \mu_1(x) := \sum_{i=1}^{n} k(x_i, x)\beta_i + b_2. \tag{33}$$

where $k(x, y)$ is positive definite kernel. For the sake of simplicity, we rewrite $\mu_1(x)$ and $\mu_2(x)$ in vector form

$$\mu_1(x) := K(A^T, x)\alpha + b_1 \quad \text{and} \quad \mu_2(x) := K(A^T, x)\beta + b_2. \tag{34}$$

where A is the $n \times p$ data matrix containing n training points in $\mathbb{R}^p$, $K(A^T, x) = \Big[k(x_1, x), k(x_2, x), .., k(x_n, x)\Big]$, $\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ .. \\ \alpha_n \end{bmatrix}$ and $\beta \begin{bmatrix} \beta_1 \\ \beta_2 \\ .. \\ \beta_n \end{bmatrix}$. The Tube loss-based kernel machines considers the problem

$$\min_{(\alpha,\beta,b_1,b_2)} J_2(\alpha, \beta, b_1, b_2) = \frac{\lambda}{2}(\alpha^T\alpha + \beta^T\beta) + \sum_{i=1}^{n} \mathcal{L}^r_{1-\alpha}\big(y_i, \big(K(A^T, x_i)\alpha + b_1\big), \big(K(A^T, x_i)\beta + b_2\big)\ \big) + \delta \sum_{i=1}^{n} \big|(K(A^T, x_i)(\alpha - \beta) + (b_1 - b_2)\big|, \tag{35}$$

where $\mathcal{L}^r_{1-\alpha}$ is the Tube loss function as given in (14) with the parameter r.

For a given point (x_i, y_i), let us compute the gradient of $\mathcal{L}^r_{1-\alpha}\big(y_i, \big(K(A^T, x_i)\alpha + b_1\big), \big(K(A^T, x_i)\beta + b_2\big)\ \big)$ first. For this, we compute

$$\frac{\partial \mathcal{L}^r_{1-\alpha}\Big(y_i, \Big(K(A^T,x_i)\alpha+b_1\Big), \Big(K(A^T,x_i)\beta+b_2\Big)\ \Big)}{\partial \alpha} = \begin{cases} (1-t)K(A, x_i), & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i > (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ 0, & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i < (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ 0, & \text{if } K(A^T, x_i)\beta + b_2 > y. \\ -tK(A^T, x_i), & \text{if } K(A^T, x_i)\alpha + b_1 < y. \end{cases}$$

$$\frac{\partial \mathcal{L}^r_{1-\alpha}\Big(y_i, \Big(K(A^T,x_i)\alpha+b_1\Big), \Big(K(A^T,x_i)\beta+b_2\Big)\ \Big)}{\partial b_1} = \begin{cases} (1-t), & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i > (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ 0, & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i < (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ 0, & \text{if } K(A^T, x_i)\beta + b_2 > y. \\ -t, & \text{if } K(A^T, x_i)\alpha + b_1 < y. \end{cases}$$

$$\frac{\partial \mathcal{L}^r_{1-\alpha}\Big(y_i, \Big(K(A^T,x_i)\alpha+b_1\Big), \Big(K(A^T,x_i)\beta+b_2\Big)\ \Big)}{\partial \beta} = \begin{cases} 0, & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i > (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ -(1-t)K(A, x_i), & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i < (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ tK(A, x_i), & \text{if } K(A^T, x_i)\beta + b_2 > y. \\ 0, & \text{if } K(A^T, x_i)\alpha + b_1 < y. \end{cases}$$

$$\frac{\partial \mathcal{L}^r_{1-\alpha}\Big(y_i, \Big(K(A^T,x_i)\alpha+b_1\Big), \Big(K(A^T,x_i)\beta+b_2\Big)\ \Big)}{\partial b_2} = \begin{cases} 0, & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i > (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ -(1-t), & \text{if } (K(A^T, x_i)\beta + b_2) < y_i < (K(A^T, x_i)\alpha + b_1) \text{ and } y_i < (K(A^T, x_i)(r\alpha + (1-r)\beta) + (rb_1 + (1-r)b_2). \\ t, & \text{if } K(A^T, x_i)\beta + b_2 > y. \\ 0, & \text{if } K(A^T, x_i)\alpha + b_1 < y. \end{cases}$$

For data point (x_k, y_k) such that $y_k = K(A^T, x_k)\alpha + b_1$, or $K(A^T, x_k)\beta + b_2$, the unique gradient for Tube loss does not exist and any sub-gradient can be considered for computation. The data point (x_k, y_k) lying exactly upon the surface $r((K(A^T, x_k)\alpha) + b_1) + (1-r)(K(A^T, x_k)\beta + b_2)$ can be ignored.

Now, for given data point (x_i, y_i), we consider the width of PI tube $\delta\big|(K(A^T, x_i, y_i)(\alpha-\beta)+(b_1-b_2)\big|$ and denote it as $J(\alpha, \beta, b_1, b_2, x_i, y_i)$ and compute

$$\frac{\partial J(\alpha,\beta,b_1,b_2,x_i)}{\partial \alpha} = sign((K(A^T,x_i)(\alpha-\beta)+(b_1-b_2))K(A^T,x_i) \quad \text{if } (K(A^T,x_i)(\alpha-\beta)+(b_1-b_2) \neq 0$$
$$\frac{\partial J(\alpha,\beta,b_1,b_2,x_i)}{\partial b_1} = sign((K(A^T,x_i)(\alpha-\beta)+(b_1-b_2)) \quad \text{if } (K(A^T,x_i)(\alpha-\beta)+(b_1-b_2) \neq 0$$
$$\frac{\partial J(\alpha,\beta,b_1,b_2,x_i)}{\partial \beta} = -sign((K(A^T,x_i)(\alpha-\beta)+(b_1-b_2))K(A^T,x_i) \quad \text{if } (K(A^T,x_i)(\alpha-\beta)+(b_1-b_2) \neq 0$$
$$\frac{\partial J(\alpha,\beta,b_1,b_2,x_i)}{\partial b_2} = -sign((K(A^T,x_i)(\alpha-\beta)+(b_1-b_2)) \quad \text{if } (K(A^T,x_i)(\alpha-\beta)+(b_1-b_2) \neq 0$$

For data point (x_i, y_i) satisfying $(K(A^T, x_i)(\alpha - \beta) + (b_1 - b_2) = 0$, a unique gradient of $J(\alpha, \beta, b_1, b_2, x_i, y_i)$ does not exist, and any subgradient can be considered for computation.

Now, we state gradient descent algorithm for the Tube loss based kernel machine problem.
Algorithm 2:-

Input:- Training Set $T = \{(x_i, y_i) : x_i \in \mathbb{R}^p, y_i \in \mathbb{R}, i = 1, 2, ...n\}$, confidence $t \in (0, 1)$ r,δ,η and tol.
Initialize:- α^0,$\beta^0 \in \mathbb{R}^n$ and b_1^0,$b_2^0 \in \mathbb{R}$.

Repeat

$$\beta^{(k+1)} = \beta^{(k)} - \eta_k\big(\lambda\beta^{(k)} + \textstyle\sum_{i=1}^n \frac{\partial \mathcal{L}^r_{1-\alpha}\big(y_i,\big(K(A^T,x_i)\alpha+b_1\big),\big(K(A^T,x_i)\beta+b_2\big)\ \big)}{\partial \beta^{(k)}} + \delta \sum_{i=1}^n \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial \beta^{(k)}}\ \big)$$
$$b_2^{(k+1)} = b_2^{(k)} - \eta_k\big(\textstyle\sum_{i=1}^n \frac{\partial \mathcal{L}^r_{1-\alpha}\big(y_i,\big(K(A^T,x_i)\alpha+b_1\big),\big(K(A^T,x_i)\beta+b_2\big)\ \big)}{\partial b_2^{(k)}} + \delta \sum_{i=1}^n \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial b_2^{(k)}}\ \big)$$
$$\alpha^{(k+1)} = \alpha^{(k)} - \eta_k\big(\lambda\alpha^{(k)} + \textstyle\sum_{i=1}^n \frac{\partial \mathcal{L}^r_{1-\alpha}\big(y_i,\big(K(A^T,x_i)\alpha+b_1\big),\big(K(A^T,x_i)\beta+b_2\big)\ \big)}{\partial \alpha^{(k)}} + \delta \sum_{i=1}^n \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial \alpha^{(k)}}\ \big)$$
$$b_1^{(k+1)} = b_1^{(k)} - \eta_k\big(\textstyle\sum_{i=1}^n \frac{\partial \mathcal{L}^r_{1-\alpha}\big(y_i,\big(K(A^T,x_i)\alpha+b_1\big),\big(K(A^T,x_i)\beta+b_2\big)\ \big)}{\partial b_1^{(k)}} + \delta \sum_{i=1}^n \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial b_1^{k}}\ \big)$$

Until $|| \begin{bmatrix} \alpha^{(k+1)} - \alpha^{(k)} \\ b_1^{(k+1)} - b_1^{(k)} \\ \beta^{(k+1)} - \beta^{(k)} \\ b_2^{(k+1)} - b_2^{(k)} \end{bmatrix} || \geq tol.$

In our implementation [1], the gradient descent algorithm for the Tube loss kernel machine initially utilizes only the gradient of the Tube loss function. After a certain number of iterations, once we confirm that the upper bound of the PI, $K(A^T x_i)\alpha + b_1$, has moved above the lower bound, $K(A^T x_i)\alpha + b_2$, of PI, we incorporate the gradient of the PI tube width into the training of the Tube loss kernel machine.

C Additional Experimental Details

Tuning r parameter:- Ideally, the tuning of the r parameter in the tube loss function should align with the skewness of the distribution $y|x$. However, real-world datasets often exist in high dimensions. Consequently, for a specific value of x, there may be only a few of y_i values in the

[1] A MATLAB implementation of gradient descent with Tube loss is provided in supplementary material code.

training dataset, leading to potential inaccuracies in estimating the skewness of the distribution $y|x$. In practical benchmark experiments, we have tuned the r values for the tube loss machine from the set $\{0.1, 0.2, ..., 0.9\}$.

Tuning δ parameter:- Initially, we initialize δ to zero in the Tube loss-based machine for benchmark dataset experiments and aim to achieve the narrowest PI by adjusting the r parameter, either moving the PI tube upwards or downwards. Once the r parameter is set and if the Tube loss-based machine achieves a coverage higher than the target t on the validation set, we gradually fine-tune the δ parameter from the set $\{0.001, 0.005, 0.1, 0.15, 0.2\}$ to minimize the tube width while maintaining the desired target coverage t.

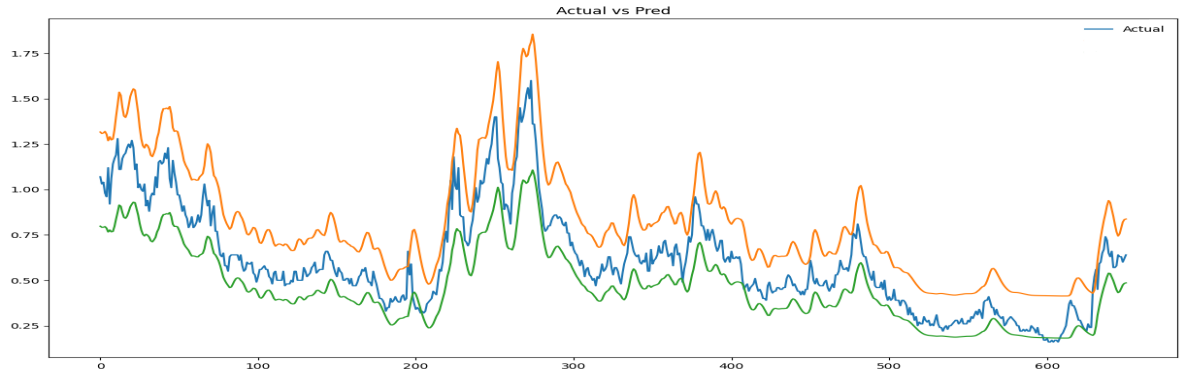
For the numerical results in Table 2, the tuned parameters for the Tube Loss are summarized in Table 6.

dataset	r (tube_r)	delta (penalty)	lr	batch	dropout	epochs
boston	0.5	0.03	0.05	10000	0.2	400
concrete	0.25	0.05	0.015	64	0.25	150
cpu_act	0.5	0.03	0.05	10000	0.2	400
energy	0.9	0.01	0.05	10000	0.2	400
kin8nm	0.1	0.01	0.05	10000	0.2	400
miami	0.3	0.05	0.05	10000	0.2	400
naval	0.3	0.05	0.05	10000	0.2	400
power	0.3	0.1	0.05	10000	0.2	400
protein	0.1	0.01	0.05	10000	0.2	400
sulfur	0.1	0.01	0.05	10000	0.2	400
wine	0.5	0.05	0.05	10000	0.2	400
yacht	0.5	0.03	0.01	10000	0.1	400

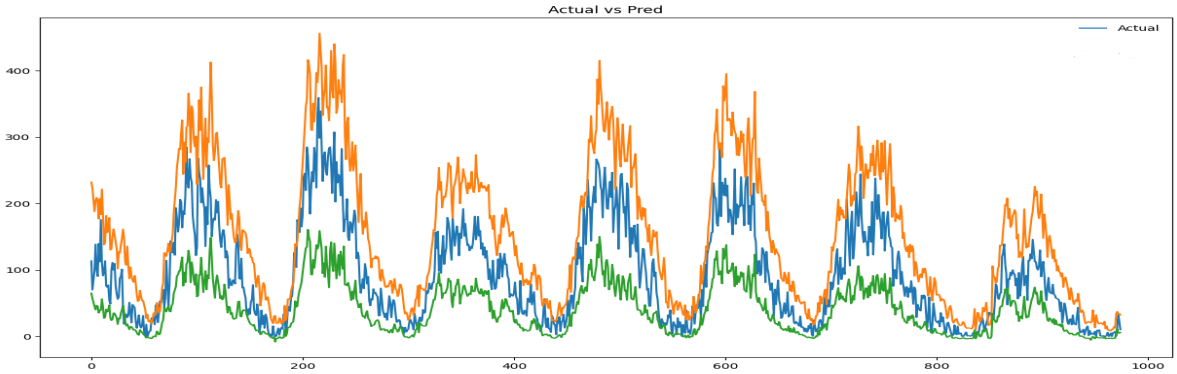
Table 6: Best Hyperparameters for Tube Loss based NN for results generated in Table 2.

Dataset (Size)	LSTM Structure	Batch Size		Up,Low	r,δ	Window Size	Learning Rate	
		Q- LSTM	T-LSTM	Q- LSTM	T-LSTM		Q-LSTM	T-LSTM
Sunspots (3265)	[256(0.3) ,128(0.2)]	128	128	0.98,0.03	0.5,0	16	0.001	0.001
Electric Production (397)	[64]	32	16	0.98,0.03	0.5,0.01	12	0.01	0.001
Daily Female Birth (365)	[100]	64	128	0.98,0.03	0.5,0.01	12	0.01	0.005
SWH (2170)	[128(0.4),64(0.3),32(0.2)]	300	300	0.98, 0.03	0.5, 0.01	100	0.001	0.001
Temperature (3651)	[16,8]	300	300	0.98, 0.03	0.5, 0.01	100	0.001	0.0001
Beer Production (464)	[64,32]	64	64	0.98, 0.03	0.1, 0.01	12	0.001	0.0001

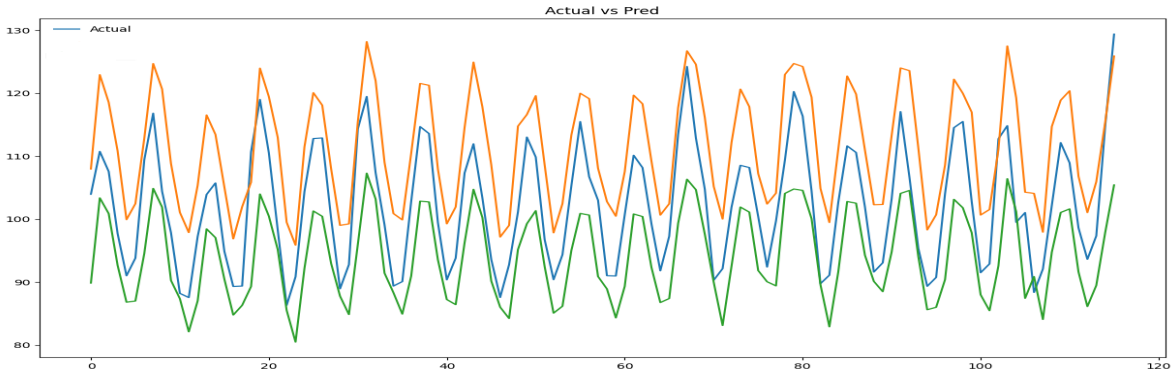
Table 7: Tuned parameter values for Q-LSTM and T-LSTM model for considered benchmark datasets in Table 3. The LSTM architecture [128(0.4),64(0.3),32(0.2)] means that three hidden layers with neurons 128, 64 and 32 respectively and each hidden layer is followed by a drop out layer which are 0.4,0.3 and 0.2 respectively.



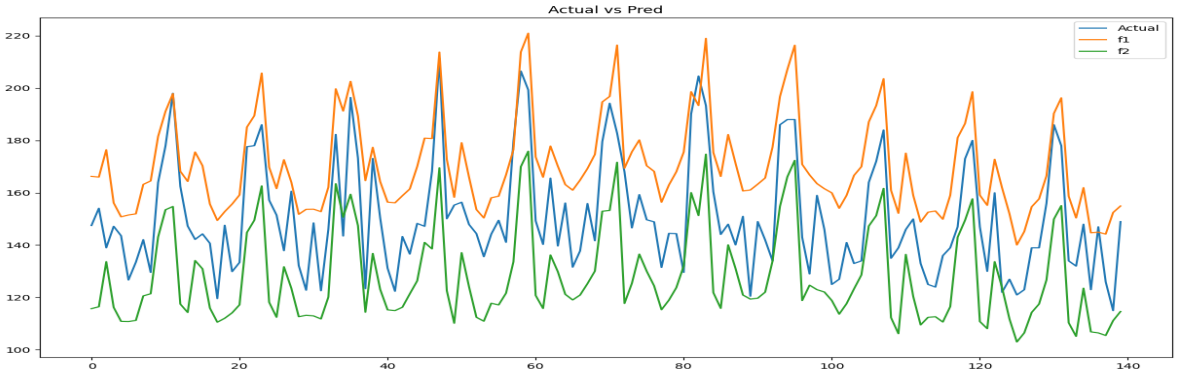
(a) SWH



(b) Sunspot



(c) Electric Production



(d) Beer Production

Figure 11: Probabilistic forecast of LSTM with proposed Tube loss function.