TUBE LOSS: A NOVEL APPROACH FOR PREDICTION INTERVAL ESTIMATION AND PROBABILISTIC FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper introduces a new loss function termed Tube Loss, designed for simultaneously estimating the lower and upper bounds of a Prediction Interval (PI) in regression tasks, while also facilitating probabilistic forecasting in auto-regressive settings. The PIs derived by minimizing the empirical risk with Tube Loss demonstrate superior performance compared to those from existing approaches. Notably, the method ensures that the generated PI asymptotically achieve a user-defined target confidence level $t \in (0, 1)$, a result supported by theoretical proof. Additionally, Tube Loss offers flexibility by allowing users to shift the PI vertically using a tunable parameter r, which enables the intervals to better capture high-density regions of the response variable's distribution, particularly useful for skewed distributions, resulting in narrower and more informative intervals. The approach also facilitates a balance between interval coverage and average width within the same optimization process through parameter δ that allows for further width reduction via recalibration. Unlike some existing techniques, the empirical risk under Tube Loss can be efficiently minimized using gradient descent. Comprehensive experiments validate the effectiveness of the proposed method across various models, including kernel machines, neural networks, deep learning architectures, and in probabilistic forecasting applications.

1 INTRODUCTION

In regression setting, machine learning (ML) algorithms predict the value of a variable y, often called dependent variable, given the value of an independent variable, say, x. Merely giving the predicted value of y without attaching a measure of uncertainty with it, may not be useful in real world applications. Uncertainty quantification (**UQ**) is especially important, when the cost of incorrect prediction is high. For example, in planning replacement of a critical and expensive component of a nuclear reactor, failure of which may lead to a nuclear disaster, the information that the predicted value of its life is 2.5 years without attaching a measure of uncertainty may not be useful. In contrast, if it is predicted that the life of critical component is between 2 and 3 years with, say, 99% confidence, it throws out useful information. Such an interval, predicted for values of y, is often called a prediction interval (PI) with a pre-specified confidence (in this case 99%).

Let us assume $(x_i, y_i), i = 1, 2, ..., m$, denote m independent copies of the random variables (x, y) having a joint distribution $p(x, y)$. With some abuse of notation, we denote by (x, y), a pair of random variables as well as its values whenever the distinction is obvious. A standard regression model provides an estimate of $\mathbb{E}(y|x)$ as the predicted value of y. However, for uncertainty quantification of the output of a Neural Network (NN) in regression setting, researchers have obtained PI of y given x with a certain confidence (Khosravi et al. (2011b), Khosravi et al. (2011a), Nix & Weigend (1994), Pearce et al. (2018), Chung et al. (2021)). The limits of a PI are appropriately chosen quantiles of the distribution of NN outputs y given x, and are thus, naturally, functions of x. Given x, let's denote a PI of y with confidence t by $[\mu_1(x), \mu_2(x)]$ where $\mu_1(x)$ and $\mu_2(x)$ are the lower and the upper bounds of the interval, respectively, satisfying $P[\mu_1(x) \leq y \leq \mu_2(x)] \geq t$ where the probability is calculated based on the conditional distribution of y given x.

Notice that, in the set-up described above, there are infinitely many choices of the bounds $\mu_1(x)$ and $\mu_2(x)$ satisfying the coverage constraint. Among all choices, the shortest PI, the PI having minimum average width $\mathbb{E}[\mu_2(x) - \mu_1(x)]$ is preferable, where the expectation is based on the marginal distribution of x. Suppose $[\hat{\mu}_1(x), \hat{\mu}_2(x)]$ denotes an estimate of PI $[\mu_1(x), \mu_2(x)]$ based on the data $(x_i, y_i), i = 1, 2, ..., m$, then we judge the quality of an estimated PI by two metrics. First, PI Coverage Probability (**PICP**) defined as the proportion of y_is lying inside of the estimated interval. Naturally, we expect it to be close to t. Second, the Mean PI width (**MPIW**) defined as $\frac{1}{m}\sum_{i=1,..,m}[\hat{\mu}_2(x_i) - \hat{\mu}_1(x_i)]$. Among all intervals satisfying the constraint that **PICP** is greater than equal to t, the interval which minimizes **MPIW** is considered as the optimal choice of a PI. Following Pearce et al. (2018) we may call it **High Quality (HQ)** principle to be followed for choosing a PI.

In the auto-regressive setting, the problem of estimating PIs is commonly referred to as probabilistic forecasting. Given the time-series data $\{y_t, t = 1, 2, ..., m\}$ on a variable y over m time points and a lag window $p \leq m$, the probabilistic forecasting models obtain the PI for y_{i+1} using the features $z_i = (y_i, y_{i-1}, ..., y_{i-p+1})$ for $i \geq m$ with given target coverage t.

All of the these UQ tasks require the estimation of a pair of quantile functions. The PI estimation task requires the estimation of the q^{th} and $(t+q)^{th}$ quantile of the distribution $y|x$ for some $0 \leq q, q + t \leq 1$. For probabilistic forecasting tasks, the bound functions are the q^{th} and $(t+q)^{th}$ quantile of the distribution $y|z$.

In non-parametric framework, the most popular, simple and effective method for the quantile estimation is through the pinball loss function. The estimate of the q^{th} quantile function say, $\hat{F}_q(x)$ (Takeuchi et al. (2006)) is obtained by minimizing the empirical loss using pinball loss function with parameter q. Thus UQ tasks in regression require to obtain the $[\hat{F}_q(x), \hat{F}_{q+t}(x)]$.

However, this quantile based PI approach presents two major limitations. First, for finding the $[\hat{F}_q(x), \hat{F}_{q+t}(x)]$, the pinball loss minimization problem is to be solved twice independently for q and $q + t$. Also, for finding a **HQ** PI, one needs to search for a q among all feasible choices that minimizes the **MPIW** $\frac{1}{m}\sum_{i=1,..,m}[\hat{F}_{q+t}(x_i) - \hat{F}_q(x_i)]$. This amounts to solving two separate pinball loss problems repeatedly searching across all feasible choices of q. It significantly increases computational complexity of the UQ tasks, particularly when learning with complex neural network architectures and large-scale datasets. Second, this approach does not allow for explicit minimization of the PI width during optimization, which limits its ability to produce narrower and more informative intervals in practice.

To address these limitations, (Khosravi et al. (2011a)) proposed the Lower and Upper Bound Estimation (LUBE) loss function in the Neural Network (NN) setting. This approach enables the simultaneous estimation of both bounds of the PI by optimizing a single NN problem. The LUBE loss is formulated as a nonlinear function of PICP and MPIW. However, minimizing this loss using Gradient Descent (GD) is not feasible, since PICP is a step function whose derivative is zero almost everywhere. This is inconvenient considering that GD is the standard method for training NN.

Pearce et al. (2018) simplify the LUBE loss function by considering a linear combination of **MPIW** and $max(0, t - PICP)^2$ which they name Quality-Driven (QD) distribution-free loss function . However, to enable the use of GD method in training NN, they propose approximating **PICP**, a step function, by the product of two sigmoid functions.

The above loss functions simplify the PI estimation task by allowing both bounds to be estimated simultaneously by solving single optimization problem. However, unlike quantile based approach, they fail to guarantee the target coverage t asymptomatically. Also, another important limitation is that they fail to facilitate the movement of the PI tube. The PI movement is very important, as it enables the model to better capture high-density regions of the response variable's distribution, particularly useful for skewed distributions, resulting in narrower and more informative PIs.

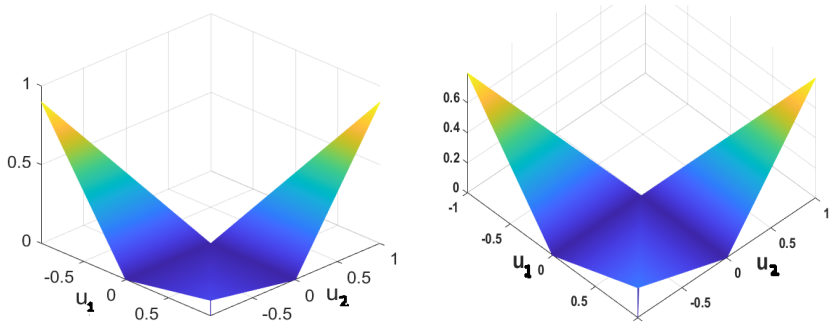
Recently, Pouplin et al. (2024) develop a novel loss function termed as 'Relaxed Quantile Regression (RQR)' for simultaneous estimation of both bounds of the PI in their work (Pouplin et al. (2024)) . Although the minimizer of the RQR loss function asymptotically guarantees that the resulting PI achieves the target coverage, but it does not facilitate the movement of the PI explicitly, through a user-controlled parameter. Apart from this, the RQR loss function considers the product of deviation

of true values from both bounds of the PI, that makes them sensitive to the presence of outliers similar to the case of expectile estimation through asymmetric least square loss (Newey & Powell (1987)). Also, the presence of outliers is more likely to trigger the crossings of the bounds, and thus, it may affect the conditional coverage of the PI near the crossings.

Taking motivation from this, we design the 'Tube loss', a new class of loss functions, for simultaneous estimation of the quantile bounds of PI, a sort of true two dimensional extension of pinball loss.

For a given $t \in (0, 1)$ and $u_2 \leq u_1$, we define the Tube loss function as

$$\rho_t^r(u_2, u_1) = \begin{cases} tu_2, & \text{if } \ u_2 > 0, \\ -(1-t)u_2, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \ ru_2 + (1-r)u_1 \geq 0, \\ (1-t)u_1, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \ ru_2 + (1-r)u_1 < 0, \\ -tu_1, & \text{if } \ u_1 < 0, \end{cases} \tag{1}$$



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

Figure 1: Tube loss function

where $0 < r < 1$ is a user-defined parameter and (u_2, u_1) are errors, representing the deviations of y values from the bounds of PI. We describe the novel features of the proposed Tube loss function along with their significance as follows.

(i) For minimizing the average width of a PI with target coverage t, the PI tube needs to capture denser regions of y values. By changing r in Tube loss, the PI tube can be moved up or down. Thus, a proper choice of r enables capturing denser region of the response values of y for each x. This is effective in reducing the width of the PI especially when the distribution of $y|x$ is skewed. For $r = 0.5$, the Tube loss estimate the centered PI ideal for symmetric noise distribution.

(ii) The proposed Tube loss function is differentiable almost everywhere (re Lebesgue measure) with non-zero gradients enabling application of GD method for direct PI estimation. The minimizer of the Tube loss for a given x is a pair of functions $(\mu_2(x), \mu_1(x))$ yielding the PI $[\mu_2(x), \mu_1(x)]$, which attains the target coverage asymptotically. We have provided a theoretical proof for this.

(iii) In Tube loss problems, the PI width can also be traded-off against empirical coverage by a user-defined parameter, say δ, into the optimization problem explicitly. It enables achieving narrower PI further in practice, particularly when the empirical coverage obtained on the validation set significantly exceeds the target level t.

The Tube loss problem is a natural choice for the loss function to be used in PI estimation of an output of Machine Learning (ML) and deep learning models. The Tube loss approach is model agnostic. In this paper, we have implemented it in the kernel machine and NN for PI estimation task to improve the PI quality. For probabilistic forecasting task, we have trained different popular deep-auto regressive architectures with the Tube loss function and shown its efficacy over existing recently developed baseline methods.

The rest of this paper is organized as follows. Appendix A summaries the literature for PI estimation and probabilistic forecasting in detail. Section 2 details the Tube loss function and its application in

different ML frameworks. Section 3 presents an extensive comparison of the Tube loss function-based PI estimation method with existing PI estimation methods in the kernel, NN and sequential deep learning framework, followed by the Future work.

2 TUBE LOSS FUNCTION

The Tube loss function is a function of two variables as shown in Figure 1. For $r = 0.5$, $\rho_t^r(u_1, u_2)$ is symmetrically located around the line $u_1 + u_2 = 0$ and continuous function of (u_1, u_2). The default choice of r in the Tube loss function is 0.5 which targets to obtain the centered PI.

To provide better intuition, we replace u_1 by $y - \mu_1$ and u_2 by $y - \mu_2$ in the Tube loss function (1) that reduces it to

$$\rho_t^r(y, \mu_1, \mu_2) = \begin{cases} t(y - \mu_2), & \text{if } y > \mu_2. \\ (1-t)(\mu_2 - y), & \text{if } \mu_1 \le y \le \mu_2 \text{ and } y \ge r\mu_2 + (1-r)\mu_1, \\ (1-t)(y - \mu_1), & \text{if } \mu_1 \le y \le \mu_2 \text{ and } y < r\mu_2 + (1-r)\mu_1, \\ t(\mu_1 - y), & \text{if } y < \mu_1, \end{cases} \tag{2}$$

For given (x_i, y_i), $i = 1, 2, ..m$, and a confidence level $t \in (0, 1)$, the PI $[\hat{\mu}_1(x), \hat{\mu}_2(x)]$ is the solution of the following optimization problem

$$arg \min_{\mu_1, \mu_2} \sum_{i=1}^{m} \rho_t^r(y_i, \mu_1(x_i), \mu_2(x_i)), \tag{3}$$

where, $\mu_1(.)$ and $\mu_2(.)$ belong to a suitably chosen class of functions and $\rho_t^r(y, \mu_1(x), \mu_2(x)$, is given by

$$\rho_t^r(y, \mu_1(x), \mu_2(x)) = \begin{cases} t(y - \mu_2(x)), & \text{if } y > \mu_2(x). \\ (1-t)(\mu_2(x) - y), & \text{if } \mu_1(x) \le y \le \mu_2(x) \text{ and } y \ge r\mu_2(x) + (1-r)\mu_1(x). \\ (1-t)(y - \mu_1(x)), & \text{if } \mu_1(x) \le y \le \mu_2(x) \text{ and } y < r\mu_2(x) + (1-r)\mu_1(x). \\ t(\mu_1(x) - y), & \text{if } y < \mu_1(x). \end{cases} \tag{4}$$

By controlling the value of the user parameter $0 < r < 1$, the upper and lower bounds of the PI could be moved up and down simultaneously.

2.1 ASYMPTOTIC PROPERTIES OF THE TUBE LOSS FUNCTION

Let $y_1, y_2, ..., y_m$ be iid following the distribution of a continuous random variable Y and $t \in (0, 1)$. 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 $\frac{1}{m}\sum_{i=1}^{m} \rho_t^r(y_i, \mu_1, \mu_2)$ with respect to (μ_1, μ_2), and m_k denotes the number of data points in $\Re_k(\mu_1^*, \mu_2^*)$. Then we have the following lemma.
As $m \to \infty$, with probability 1 the following results hold:
(i) $\frac{m_1}{m_2} \to \frac{1-t}{t}$, (ii) $\frac{m_4}{m_3} \to \frac{1-t}{t}$, and (iii) $\frac{m_1+m_4}{m_2+m_3} \to \frac{1-t}{t}$. The proof of the lemma is given in Appendix B.

We now extend the above lemma to the regression setup. For the training set T, let $(\hat{\mu}_1(x), \hat{\mu}_2(x))$ be the minimizer of the empirical risk $\frac{1}{m} \sum_{i=1}^{m} \rho_t^r(y_i, \mu_1(x_i), \mu_2(x_i))$, where $\mu_1(x)$ and $\mu_2(x)$ belong to an appropriately chosen class of functions. With some abuse of notation, we let m_k denote the cardinality of the set $\Re_k(\hat{\mu}_1(x), \hat{\mu}_2(x))$ $(k = 1, 2, 3, 4)$ inducing a partition of the feature space (as shown in Figure 2). We then have the following proposition.

For $t \in (0, 1)$ as $m \to \infty$ the following results hold with probability 1,
(i) $\frac{m_1}{m_2} \to \frac{1-t}{t}$, (ii) $\frac{m_4}{m_3} \to \frac{1-t}{t}$ (iii) $\frac{m_1+m_4}{m_2+m_3} \to \frac{1-t}{t}$, provided $(x_i, y_i), i = 1, 2, ..., m$ are iid following a distribution $p(x, y)$ with $p(y|x)$ continuous and the expectation of the modulus of absolute continuity of its density satisfies $\lim_{\delta \to 0} E[\epsilon(\delta)] = 0$.

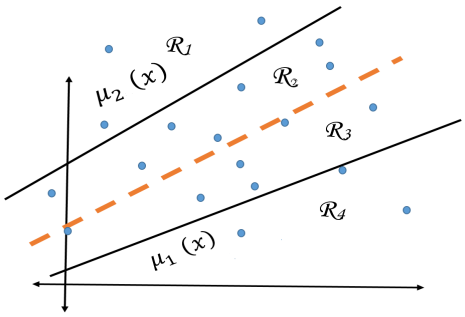


Figure 2: PI tube loss, red line represents the convex combination of $\mu_1(x)$ and $\mu_2(x)$, i,e., $r\mu_1(x) + (1-r)\mu_2(x)$, $0 < r < 1$. Blue dots represent data points (x_i, y_i), $i = 1, 2, , .., m$.

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

The r parameter and movement of the PI bounds:- Now, we show that how the PI tube can be moved up and down by choosing r appropriately in the Tube loss function.

Let us suppose that $(\hat{\mu}_1^{r_2}(x), \hat{\mu}_2^{r_2}(x))$ is the minimizer of the average loss for $r = r_2$ i,e. of $\frac{1}{m}\sum_{i=1}^{m}\rho_t^{r_2}(y_i, \mu_1(x_i), \mu_2(x_i))$. Also, we assume that there are $m_k^{r_2}$ points in the set $\Re_k(\hat{\mu}_1^{r_2}(x), \hat{\mu}_2^{r_2}(x)), (k = 1, 2, 3, 4)$. Similarly, for $r = r_1 (< r_2)$, we assume that $(\hat{\mu}_1^{r_1}(x), \hat{\mu}_2^{r_1}(x)$ is the minimizer of the average loss, and there are $m_k^{r_1}$ points in the set $\Re_k(\hat{\mu}_1^{r_1}(x), \hat{\mu}_2^{r_1}(x)), (k = 1, 2, 3, 4)$. Now, Proposition 1 (iii) entails that asymptotically t fraction of y values should lie inside each of the PIs $[\hat{\mu}_1^{r_1}(x), \hat{\mu}_2^{r_1}(x)]$ and $[\hat{\mu}_1^{r_2}(x), \hat{\mu}_2^{r_2}(x)]$. Since $r_1 < r_2$, (i) and (ii) of Proposition 1 entail $\frac{m_2^{r_1}}{m_3^{r_1}} > \frac{m_2^{r_2}}{m_3^{r_2}}$, which in turn implies $\frac{m_1^{r_1}}{m_4^{r_1}} > \frac{m_1^{r_2}}{m_4^{r_2}}$ asymptotically for large m. In other words, changing r from r_1 to r_2 moves the tube down. Thus, the PI bounds can be moved up and down by changing r.

As stated above, by controlling r MPIW can be reduced. But, the Tube loss function $\rho_t^r(u_1, u_2)$ is discontinuous at the separating plane $\{(u_1, u_2) : ru_2 + (1-r)u_1 = 0\}$ when $r \neq 0.5$. It is worth mentioning here that this discontinuity is not found to cause any problem in the implementation of GD method in the practical experiments as the Tube loss function is differentiable everywhere (re Lebesgue measure) with non-zero gradients.

Tube loss in kernel machine and Neural Network:- The Tube loss PI estimation models enjoy the explicit minimization of the PI width against the empirical coverage with a user defined positive parameter δ as follows.

$$\min_{(\mu_1,\mu_2)} [\sum_{i=1}^{m} \rho_t^r(y_i, \mu_1(x_i), \mu_2(x_i)) + \delta \sum_{i=1}^{m} |(\mu_2(x_i) - \mu_1(x_i)|]. \tag{5}$$

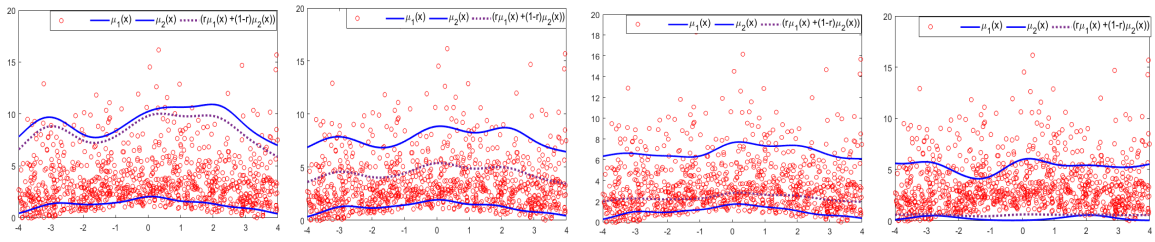
For NN training, $(\mu_1(x), \mu_2(x))$ corresponds to two distinct neurons in the output layer of the dense network. The weights of the hidden and output layers are optimized by back-propagating the error defined in (5). In kernel machine, the Tube loss PI model estimates pair of functions

$$\mu_2(x) := \sum_{i=1}^{m} k(x_i, x)\alpha_i + b_1 \text{ and } \mu_1(x) := \sum_{i=1}^{m} k(x_i, x)\beta_i + b_2. \tag{6}$$

where $k(x, y)$ is positive definite kernel(Mercer (1909)). The Gradient descent solution for the Tube loss based kernel machine is derived at the Appendix C.

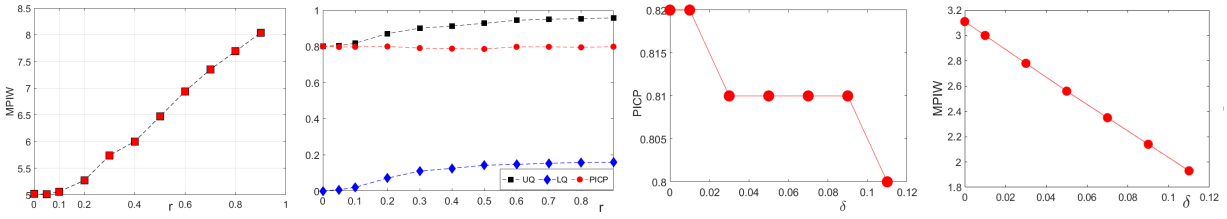
For the High Quality PI, the value of the parameter r and δ of problem (5) should be tuned efficiently on validation set. A practical approach for selecting the value of r is to choose it from the discrete set $\{0.1, 0.2, \ldots, 1\}$. The default choice of the r parameter is 0.5 which targets to estimate the centered PI.

We state a heuristic approach for tuning the parameters of the Tube loss optimization problem (5). The parameter δ in (5) is initially set to zero, based on the theoretical justification that the minimizer of the Tube loss problem (3) ensures the desired calibration for sufficiently large values of m.



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

Figure 3: Location of PI tube changes with r values in Tube loss based kernel machine.



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

Figure 4: 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 Servo dataset.

Recalibration of PI:-To identify the narrowest PI that meets the target coverage t, we adjust the PI through local movement by tuning the parameter r on the validation set. If the empirical coverage achieved exceeds the target t by a significant margin, this indicates potential to further narrow the PI. In such cases, we increment the parameter δ slightly and retrain the PI model (5). This process is repeated until the resulting PI on the validation set is well-calibrated to the target coverage. We refer to this iterative refinement as *recalibration.* Our numerical results show that the Tube loss-based model can obtain a significant decrease in MPIW values on test set through *recalibration.*

3 Experimental Results

We perform experiments to observe the effectiveness of the proposed Tube loss function in NN, kernel machine and deep learning for UQ task. While comparing the two PI methods, we prefer the PI method with lower MPIW, if it approximately manages to obtain the target calibration. In case, the target true PI is known for the given dataset (it is possible when we know the distribution $y|x$ apriori), the least RMSE decides the best.

3.1 Tube loss in kernel machine:-

We simulate the two artificial datasets by generating the independent variable x_i from $U(0,1)$ and response values y_i using the relation

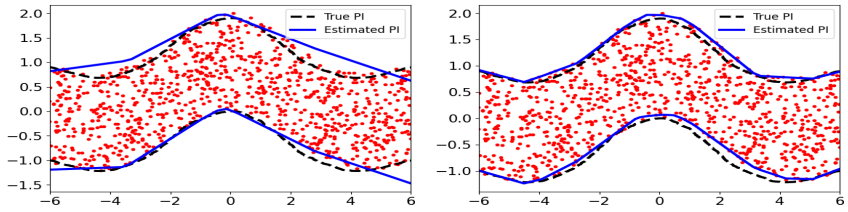
$$y_i = \frac{sin(x_i)}{(x_i)} + \epsilon_i. \tag{7}$$

In dataset **A**, the noise ϵ_i is added from $\mathcal{N}(0, 0.8)$. In dataset **B**, the noise ϵ_i is added from asymmetric $\chi^2(3)$. We generated 500 data points for the training set and 1000 data points for the testing set for both datasets.

In dataset **B**, we target to estimate the PI for $t = 0.8$ with RBF kernel. In Figure 3 (b), the Tube loss-based kernel machine with parameters $r = 0.5$ and $\delta = 0$ obtains the PICP 0.79 and MPIW 6.47. However, the estimated PI tube does not capture the densest region of response values leading to large MPIW values as noise in the data is from the asymmetric χ^2 distribution. For this, we need

	Model	(Uq,Lq)/(r,δ)	PICP	MPIW	Time (s)
A	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
B	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 **A** and **B**. Q kernel M and T kernel M involves two parameters (Uq,Lq) and (r, δ) respectively.



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

Figure 5: Comparison of Tube loss and QD loss function based NNs

to shift the estimated PI tube downwards. Figure 3 (a) (c) and (d) shows the PI obtained by the tube loss-based kernel machine for $r = 0.8, 0.3$, and 0.1. As we decrease the r values, the estimated PI tube moves downward, capturing denser regions of y values with lower MPIW. We have plotted the MPIW, PICP, LQ, and UQ obtained by the tube loss-based kernel machine against different r values in Figure 4 (a) and (b). The LQ (Lower Quantile) and UQ (Upper Quantile) are the fractions of y_i lying below the upper and lower bound functions of our estimated PI tube. The LQ and UQ values decrease with the decrease in r values leading to a downward movement of the PI tube with lesser MPIW values without compromising the PICP of estimate. The MPIW improves significantly with the decrease in r values. In fact, MPIW values improve by 21.80%, if we decrease the r from 0.5 to 0.1.

Comparison with quantile based kernel machine:- We generate the training and testing data sets **A** and **B** in ten different trials. The target PI was set to 0.80 and 0.60 for dataset **A** and **B** respectively. In Table 1, we compare the performance of the Tube loss- based kernel machine and quantile based kernel machine in different parameter settings. The Tube loss-based kernel machine looks bit more promising than quantile based kernel machine. But, main advantage of the Tube loss- based kernel machine over quantile based kernel machine is significant improvement in training time. It is because, that the Tube loss-based kernel machine requires solution of the single optimization problem, where as, the quantile based kernel machine needs to be trained twice for obtaining τ and $t + \tau$ quantiles.

Re-calibration effect of parameter δ:- The parameter δ in the tube loss-based kernel machine is practically useful for improving the MPIW on the test set. To show this, we have considered the popular Servo (167×5) Karl Ulrich (2016) dataset and separated the 10% of data points as a testing set. The remaining data points were trained with a 10-fold cross-validation method for target $t = 0.80$ PI with $r = 0.5$ and $\delta = 0$. The mean PICP reported on the validation set was 0.82. Since the observed PICP was significantly larger than the target PICP on the validation set, therefore, there was a scope to obtain lesser MPIW on the test set by increasing the value of δ. We have increased the value of δ till the observed PICP on the validation set decreases to the target $t = 0.8$, shown in Figure 4 (c). It causes about 44% improvement in test MPIW as shown in Figure 4 (d) while maintaining the required coverage. However, this facility remains missing in the quantile based kernel machine.

3.2 TUBE LOSS IN NEURAL NETWORK

We perform a simple experiment to clearly observe the advantages of the proposed Tube loss over QD loss function in NN framework. For this, we generate 1000 data points by adding uniform noise to the sinc function as

$$y_i = \frac{sin(x_i)}{x_i} + \epsilon_i, \tag{8}$$

where x_i is $U(-2\pi, 2\pi)$ and ϵ_i is from $U(-1, 1)$.

We train the Tube Loss-based NN model and the QD loss-based NN model to obtain the best PI with confidence 0.95 by tunning their parameter cautiously.

Figure 5 shows the PI obtained by Tube loss with $r = 0.5$ and QD loss based NN models along with the true target PI. The true target PI was computed by obtaining the 97.5% and 2.5% quantiles of the distribution of y_i/x_i. The boundaries of the Tube loss based NN approximates the sinc function (true PI) well, far better than QD loss based NN. The observed (RMSE, PICP, MPIW/PICP) for Tube loss and QD loss were $(0.092, 0.95, 1.97)$ and $(0.171, 0.97, 2.11)$ respectively. An extended experiment along with a detailed discussion regarding the parameter effects is presented in Appendix D.1.

Benchmark Datasets: Now, we compare the performance of the Tube loss with the recently developed RQR Pouplin et al. (2024) loss function along with the other NN baselines used by them including the Quantile Regression (QR), Simultaneous Quantile Regression (Tagasovska & Lopez-Paz (2019)) with Centered PI (SQR-C), Simultaneous Quantile Regression with Centered Interval with non-centered PI. The numerical results presented in Table 2 show that Tube loss manages to obtain the best performance on 9 datasets out of 12 datasets. All numerical results were generated following the experimental setup in Pouplin et al. (2024) using their public code. More details about the experiments along with the parameter tuning details are at the Appendix D.4. A detail comparison of the RQR loss and Tube loss in NN setting is also presented at the Appendix D.2.

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

Table 2: Comparisons of the Tube loss against RQR loss and other existing baseline NN methods on 12 datasets for PI estimation task with target t =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 0.90 coverage approximately well with the lowest MPIW.

3.3 TUBE LOSS FOR 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

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 parameter details and plots are at Appendix D.4.

tuned the LSTM architecture, dropout, and window size based 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.

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 4 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 4: 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 $t = 0.95$.

Please see the Appendix D.3 for more numerical results for probabilistic forecasting, experimental, parameter and datasets details. In Appendix E, our Tube loss function is extended in Conformal Regression setting.

4 Future Works

TheTube loss function can be applied to key probabilistic forecasting tasks, such as electric load, pollution rate, wave height, financial, and solar radiance forecasting, to quantify uncertainty and enhance decision-making. The future requires the PI loss function which can also ensure the individual calibration Chung et al. (2021) in place of average calibration.

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.

Yitian Chen, Yanfei Kang, Yixiong Chen, and Zizhuo Wang. Probabilistic forecasting with temporal convolutional neural network. *Neurocomputing*, 399:491–501, 2020.

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.

Mingjian Cui, Jie Zhang, Hongyu Wu, and Bri-Mathias Hodge. Wind-friendly flexible ramping product design in multi-timescale power system operations. *IEEE Transactions on Sustainable Energy*, 8(3):1064–1075, 2017.

Wenkang Cui, Can Wan, and Yonghua Song. Ensemble deep learning-based non-crossing quantile regression for nonparametric probabilistic forecasting of wind power generation. *IEEE Transactions on Power Systems*, 38(4):3163–3178, 2022.

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.

Jan Gasthaus, Konstantinos Benidis, Yuyang Wang, Syama Sundar Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski. Probabilistic forecasting with spline quantile function rnns. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1901–1910. PMLR, 2019.

Jiani Heng, Yongmiao Hong, Jianming Hu, and Shouyang Wang. Probabilistic and deterministic wind speed forecasting based on non-parametric approaches and wind characteristics information. *Applied Energy*, 306:118029, 2022.

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

Jianming Hu, Yingying Lin, Jingwei Tang, and Jing Zhao. A new wind power interval prediction approach based on reservoir computing and a quality-driven loss function. *Applied Soft Computing*, 92:106327, 2020a.

Jianming Hu, Jingwei Tang, and Yingying Lin. A novel wind power probabilistic forecasting approach based on joint quantile regression and multi-objective optimization. *Renewable Energy*, 149:141–164, 2020b.

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, 2011a.

Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on neural networks*, 22(9):1341–1356, 2011b.

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

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

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_history.php?station=42001.

Whitney K Newey and James L Powell. Asymmetric least squares estimation and testing. *Econometrica: Journal of the Econometric Society*, pp. 819–847, 1987.

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.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

Adnan Saeed, Chaoshun Li, and Zhenhao Gan. Short-term wind speed interval prediction using improved quality-driven loss based gated multi-scale convolutional sequence model. *Energy*, 300: 131590, 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.

Can Wan, Jin Lin, Jianhui Wang, Yonghua Song, and Zhao Yang Dong. Direct quantile regression for nonparametric probabilistic forecasting of wind power generation. *IEEE Transactions on Power Systems*, 32(4):2767–2778, 2016.

Yi Wang, Dahua Gan, Mingyang Sun, Ning Zhang, Zongxiang Lu, and Chongqing Kang. Probabilistic individual load forecasting using pinball loss guided lstm. *Applied Energy*, 235:10–20, 2019.

Chongchong Xu and Guo Chen. Interpretable transformer-based model for probabilistic short-term forecasting of residential net load. *International Journal of Electrical Power & Energy Systems*, 155:109515, 2024.

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.

Luren Yang, Tom Kavli, Mats Carlin, Sigmund Clausen, and Paul FM De Groot. An evaluation of confidence bound estimation methods for neural networks. *Advances in Computational Intelligence and Learning: Methods and Applications*, pp. 71–84, 2002.

Yixiao Yu, Ming Yang, Xueshan Han, Yumin Zhang, and Pingfeng Ye. A regional wind power probabilistic forecast method based on deep quantile regression. *IEEE Transactions on Industry Applications*, 57(5):4420–4427, 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.

Wenjie Zhang, Hao Quan, and Dipti Srinivasan. An improved quantile regression neural network for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, 10(4):4425–4434, 2018.

Jianhua Zhu and Yaoyao He. A large-scale multi-objective evolutionary quantile estimation model for wind power probabilistic forecasting. *IEEE Transactions on Evolutionary Computation*, 2024.

Zhenglin Zhu, Yusen Xu, Junzhao Wu, Yiwen Liu, Jianwei Guo, and Haixiang Zang. Wind power probabilistic forecasting based on combined decomposition and deep learning quantile regression. *Frontiers in Energy Research*, 10:937240, 2022.

A Related Works

For a given target confidence t, a PI is actually a pair of functions $\hat{F}_q(x)$ and $\hat{F}_{q+t}(x)$ which are estimates of the q^{th} and $(q+t)^{th}$ quantiles of y given x for some fixed $0 < q, q+t < 1$.

PI through quantiles:-In non-parametric framework, for given training set T, Takeuchi et al. (2006) obtains the τ^{th} quantile function $\hat{F}_\tau(x)$ by solving the problem

$$\min_{(\alpha,b)} [\frac{\lambda}{2}\alpha^T\alpha + \frac{1}{m}\sum_{i=1}^{m} L_\tau\big(y_i - \big(K(A^T, x_i)\alpha + b\big)\big)] \tag{9}$$

where $K(A^T, x) = [k(x_1, x), k(x_2, x), .., k(x_m, x)]$, $k(x, y)$ is a positive semi-definite kernel (Mercer (1909)) and $L_\tau(u)$ is the pinball loss function Koenker & Bassett Jr (1978) given by $L_\tau(u) = \begin{cases} \tau u, & \text{if } u \geq 0, \\ -(1-\tau)u, & \text{otherwise,} \end{cases}$.

Clearly, for the estimation of PI with confidence t, the problem (9) needs to be solved independently for $\tau = q$ and $\tau = q + t$, respectively.

Direct PI estimation in NN:- For reducing the complexity of the PI estimation task, LUBE NN Khosravi et al. (2011a) simultaneously obtains a pair of NN generated functions $\mu_2(x)$ and $\mu_1(x)$ by solving the problem

$$\min_{\bar{\mu_1},\bar{\mu_2}} \frac{1}{mR}\sum_{i=1}^{m}(\bar{\mu_1}(x_i) - \bar{\mu_2}(x_i))\big(1 + \gamma\,(P)e^{-\eta(P-t)}\big), \tag{10}$$

where P is empirical coverage computed upon the training set by $P = \frac{1}{m}\sum_{i=1}^{m} L_{01}(y_i, \bar{\mu_1}(x_i), \bar{\mu_2}(x_i))$ and $L_{01}(y_i, \bar{\mu_1}(x_i), \bar{\mu_2}(x_i))$ is the step function given by

$$L_{01}(y_i, \bar{\mu_1}(x_i), \bar{\mu_2}(x_i)) = \begin{cases} 1, & \text{if } y_i \in [\bar{\mu_2}(x), \bar{\mu_1}(x)]. \\ 0, & \text{Otherwise.} \end{cases} \tag{11}$$

Also, the $\gamma(P) = \begin{cases} 0, & \text{if } P \geq t, \\ 1, & \text{otherwise,} \end{cases}$, R is the range of response values y_i and η is the user-defined parameter. But, the derivatives of the step function in (11) is zero almost everywhere, which makes the problem (10) impossible to be solved with the gradient descent method.

To improve the LUBE method, Pearce et al., have proposed the QD loss function Pearce et al. (2018) which minimizes

$$\frac{1}{m}\sum_{i=1}^{m}(\bar{\mu_1}(x_i) - \bar{\mu_2}(x_i))k_i + \lambda\frac{l}{t(1-t)}\,max\big(0, (t-P)\big)^2, \tag{12}$$

where k_i is also computed using the $L_{01}(y_i, \bar{\mu_1}(x_i), \bar{\mu_2}(x_i))$ and λ is the user-defined trade-off parameter. Clearly, like LUBE, GD method cannot be used to minimize the QD loss in (5). To get round this problem, Pearce et al. propose approximating the step function $L_{01}(y_i, \bar{\mu_1}(x_i), \bar{\mu_2}(x_i))$ by the product of two sigmoidal functions $\sigma((\mu_1 - y).s)\sigma((y - \mu_2).s)$, where s is the soften parameter. With this approximation the derivatives of the step function in (12) become non-zero and thus, enabling the application of the GD method for training the NN.

Probabilistic Forecasting in deep network:- Given the time-series data $\{y_t, t = 1, 2, ..., m\}$ on a variable y over m time points and a lag window $p \leq m$, the probabilistic forecasting of y_{m+1} with coverage t provides an interval $[\hat{y}^q_{m+1}, \hat{y}^{q+t}_{m+1}]$ where, $\hat{y}^q_{m+1}$ and $\hat{y}^{q+t}_{m+1}$ are the estimates of q^{th} and $(t+q)^{th}$ quantiles of the distribution $(y_{m+1}|y_m, ..., y_{m-p+1})$, $t+q \leq 1$. In the parametric framework, the joint distribution of $\{y_t, t = 1, 2, ..., m\}$ is assumed to be multivariate normal with some kind of dependence structure. Consequently, the conditional distribution of $(y_{m+1}|y_m, ..., y_{m-p+1})$ is normal with mean $\mu(y_m, ..., y_{m-p+1})$ and standard deviation $\sigma(y_m, ..., y_{m-p+1})$. The output layer of the deep network produces the maximum likelihood estimates of the mean and the standard deviation of the conditional distribution. Often, the normality assumption is based on mathematical convenience rather than evidence, which is often not adequate Gasthaus et al. (2019) for addressing

the PI estimation problem. In the distribution-free setting, for obtaining probabilistic forecasts, the pinball loss based deep networks are used in different application contexts Zhang et al. (2018) Yu et al. (2021) Wang et al. (2019). These models require training of two independent deep networks for estimating the q^{th} and $(t+q)^{th}$ quantiles using the pinball loss function Chen et al. (2020) Yang et al. (2002) Xu & Chen (2024).

B 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 $m_1 + k_1 + m_2 + k_2 + m_3 + k_3 + m_4 = m$.

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 $\epsilon_1 \text{¡} \min_{y_i < \mu_1^*}(|\mu_1^* - y_i|)$, $\epsilon_2 \text{¡} \min_{y_i > \mu_2^*}|(y_i - \mu_2^*)|$, 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

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

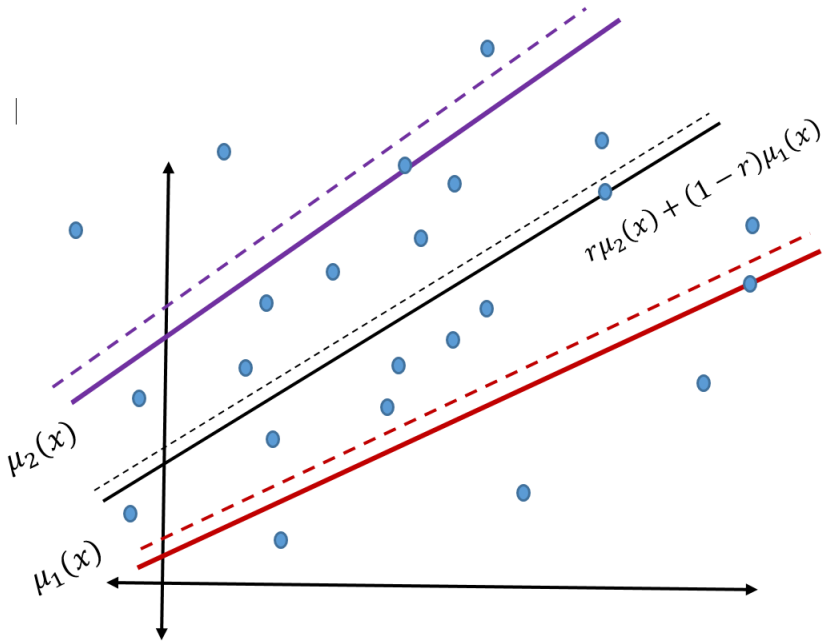


Figure 6

Given that (μ_1^*, μ_2^*) is an optimal solution, we have $\sum_{i=1}^m \rho_t^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^m \rho_t^r(y_i, \mu_1^*, \mu_2^*) \geq 0$. 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 $\sum_{y_i \in R_i} \rho_t^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{y_i \in R_i} \rho_t^r(y_i, \mu_1^*, \mu_2^*)$ by Δ_i. The simple calculation leads to the following values of $\Delta_i, i = 1, 2, ..., 10$:

$\Delta_1 = -tm_1\delta_2^*, \Delta_2 = 0, \Delta_3 = (1-t)k_1\delta_2^*, \Delta_4 = (1-t)m_2\delta_2^*, \Delta_5 = 0, \Delta_6 = k_2(1-t)(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*), \Delta_7 = -(1-t)m_3\delta_1^*, \Delta_8 = 0, \Delta_9 = tk_3\delta_1^*, \Delta_{10} = tm_4\delta_1^*.$

Thus, we obtain

$$\sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^*, \mu_2^*) = -tm_1\delta_2^* + (1-t)k_1\delta_2^* + (1-t)m_2\delta_2^* + (1-t)k_2(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*) - (1-t)m_3\delta_1^* + tk_3\delta_1^* + tm_4\delta_1^* \geq 0. \quad (13)$$

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

$$\frac{1-t}{t} \geq \frac{m_1\delta_2^*}{m_2\delta_2^* + k_1\delta_2^* + k_2((2-r)\mu_1^* - (1-r)\mu_2^*)}.$$

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,

$$\frac{m_1}{m_2} \leq \frac{1-t}{t}. \quad (14)$$

.

Arguing on a similar line, if we consider $\delta_2^* = 0$ and $\delta_1^* > 0$ in 13, then we have the following inequality

$$\frac{(k_3+m_4)\delta_1^*}{m_3\delta_1^* - k_2(\delta_1^* + (2-r)\mu_1^* - (1-r)\mu_2^*)} \geq \frac{1-t}{t}$$

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

$$\frac{m_4}{m_3} \geq \frac{1-t}{t}. \quad (15)$$

Again, starting with (μ_1^*, μ_2^*) as an optimal solution, we have $\sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^* - \delta_1^*, \mu_2^* - \delta_2^*) - \sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^*, \mu_2^*) \geq 0$. 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 $\sum_{y_i \in R_i} \rho_t^r(y_i, \mu_1^* - \delta_1^*, \mu_2^* - \delta_2^*) - \sum_{y_i \in R_i} \rho_t^r(y_i, \mu_1^*, \mu_2^*)$ by Δ_i' we obtain the following values of $\Delta_i', i = 1, 2, ..., 10$: $\Delta_1' = tm_1\delta_2^*$, $\Delta_2' = -(1-t)k_1\delta_2^*$, $\Delta_3' = 0$, $\Delta_4' = -(1-t)m_2\delta_2^*$, $\Delta_5' = -(1-t)k_2\delta_2^*$, $\Delta_6' = 0$, $\Delta_7' = (1-t)m_3\delta_1^*$, $\Delta_8' = (1-t)k_3\delta_1^*$, $\Delta_9' = 0$, $\Delta_{10}' = -tm_4\delta_1^*$.

Thus, we obtain

$$\sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^* + \delta_1^*, \mu_2^* + \delta_2^*) - \sum_{i=1}^{m} \rho_t^r(y_i, \mu_1^*, \mu_2^*) = tm_1\delta_2^* - (1-t)k_1\delta_2^* - (1-t)m_2\delta_2^* - (1-t)k_2\delta_2^* + (1-t)m_3\delta_1^* + (1-t)k_3\delta_1^* - tm_4\delta_1^* \geq 0. \quad (16)$$

Now, if we assume $\delta_1^* = 0$ and $\delta_2^* > 0$ in (16), then we obtain,

$$\frac{m_1}{k_1 + m_2 + k_2} \geq \frac{1-t}{t},$$

which entails

$$\frac{m_1}{m_2} \geq \frac{1-t}{t}, \quad (17)$$

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

Similarly, considering $\delta_2^* = 0$ and $\delta_1^* > 0$ in (16), we obtain as $m \to \infty$,

$$\frac{m_4}{m_3} \leq \frac{1-t}{t}, \tag{18}$$

holds with probability 1.

Thus, (14) and (17) imply that as $m \to \infty$

$$\frac{m_1}{m_2} = \frac{1-t}{t} \tag{19}$$

holds with probability 1.

Similarly, combining (15) and (17), we obtain as $m \to \infty$,

$$\frac{m_4}{m_3} = \frac{1-t}{t} \tag{20}$$

holds with probability 1.

Thus, combining 19 and 20, we obtain, as $m \to \infty$,

$$\frac{m_1 + m_4}{m_2 + m_3} = \frac{1-t}{t} \tag{21}$$

holds with probability 1.

C GRADIENT DESCENT METHOD FOR TUBE LOSS BASED KERNEL MACHINE

The Tube loss based kernel machine considers the problem

$$\min_{(\alpha,\beta,b_1,b_2)} J_2(\alpha,\beta,b_1,b_2) = \frac{\lambda}{2}(\alpha^T\alpha + \beta^T\beta) + \sum_{i=1}^{m} \rho_t^r\big(y_i, \big(K(A^T,x_i)\alpha + b_1\big), \big(K(A^T,x_i)\beta + b_2\big)\ \big) \\ +\delta \sum_{i=1}^{m} \big|(K(A^T,x_i)(\alpha-\beta) + (b_1-b_2)\big|, \tag{22}$$

where ρ_t^r is the Tube loss function as given in (4) with the parameter r and A is the $m \times n$ data matrix containing m training points in $\mathbb{R}^n$, Also, $\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ .. \\ \alpha_m \end{bmatrix}$, $\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ .. \\ \beta_m \end{bmatrix}$ are m-dimensional vectors and $K(A^T,x) = [k(x_1,x), k(x_2,x), .., k(x_m,x)]$ is the kernel vector containing the kernel evaluation of x with each of training points.

For a given point (x_i, y_i), let us compute the gradient of $\rho_t^r\big(y_i, \big(K(A^T, x_i)\alpha + b_1\big), \big(K(A^T, x_i)\beta + b_2\big)\ \big)$ first. For this, we compute

$$\frac{\partial \rho_t^r\left(y_i, \left(K(A^T,x_i)\alpha+b_1\right), \left(K(A^T,x_i)\beta+b_2\right)\ \right)}{\partial \alpha} =$$

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

$$\frac{\partial \rho_t^r\left(y_i, \left(K(A^T,x_i)\alpha+b_1\right), \left(K(A^T,x_i)\beta+b_2\right)\ \right)}{\partial b_1} =$$

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

$$\frac{\partial \rho_t^r\left(y_i, \left(K(A^T,x_i)\alpha+b_1\right), \left(K(A^T,x_i)\beta+b_2\right)\ \right)}{\partial \beta} =$$

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

$$\frac{\partial \rho_t^r\left(y_i, \left(K(A^T,x_i)\alpha+b_1\right), \left(K(A^T,x_i)\beta+b_2\right)\ \right)}{\partial b_2} =$$

$$\begin{cases} 0, & \text{if } (K(A^T,x_i)\beta+b_2) < y_i < (K(A^T,x_i)\alpha+b_1) \text{ and } y_i > (K(A^T,x_i)(r\alpha+(1-r)\beta)+(rb_1+(1-r)b_2). \\ -(1-t), & \text{if } (K(A^T,x_i)\beta+b_2) < y_i < (K(A^T,x_i)\alpha+b_1) \text{ and } y_i < (K(A^T,x_i)(r\alpha+(1-r)\beta)+(rb_1+(1-r)b_2). \\ t, & \text{if } K(A^T,x_i)\beta+b_2 > y. \\ 0, & \text{if } K(A^T,x_i)\alpha+b_1 < y. \end{cases} \quad (2$$

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

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

Algorithm 2:-

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

0: Initialize:- α^0,$\beta^0 \in \mathbb{R}^m$ and b_1^0,$b_2^0 \in \mathbb{R}$.

0: Repeat

0: $\beta^{(k+1)} = \beta^{(k)} - \eta_k\big(\lambda\beta^{(k)} + \sum_{i=1}^m \frac{\partial \rho_t^r\left(y_i, \left(K(A^T,x_i)\alpha+b_1\right), \left(K(A^T,x_i)\beta+b_2\right)\ \right)}{\partial \beta^{(k)}} + \delta \sum_{i=1}^m \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial \beta^{(k)}}\ \big)$

$$0:\ b_2^{(k+1)} = b_2^{(k)} - \eta_k\Big(\sum_{i=1}^{m} \frac{\partial \rho_t^r\left(y_i,\left(K(A^T,x_i)\alpha+b_1\right),\left(K(A^T,x_i)\beta+b_2\right)\right)}{\partial b_2^{(k)}} + \delta \sum_{i=1}^{m} \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial b_2^{(k)}}\Big)$$

$$0:\ \alpha^{(k+1)} = \alpha^{(k)} - \eta_k\Big(\lambda\alpha^{(k)} + \sum_{i=1}^{m} \frac{\partial \rho_t^r\left(y_i,\left(K(A^T,x_i)\alpha+b_1\right),\left(K(A^T,x_i)\beta+b_2\right)\right)}{\partial \alpha^{(k)}} + \delta \sum_{i=1}^{m} \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial \alpha^{(k)}}\Big)$$

$$0:\ b_1^{(k+1)} = b_1^{(k)} - \eta_k\Big(\sum_{i=1}^{m} \frac{\partial \rho_t^r\left(y_i,\left(K(A^T,x_i)\alpha+b_1\right),\left(K(A^T,x_i)\beta+b_2\right)\right)}{\partial b_1^{(k)}} + \delta \sum_{i=1}^{m} \frac{\partial J(\alpha,\beta,b_1,b_2,x_i,y_i)}{\partial b_1^{k}}\Big)$$

=0

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

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.

D Extended numerical experiments

D.1 Tube loss versus QD loss

Pearce et al. clearly shows that the QD loss-based NN Pearce et al. (2018), is more efficient than LUBE Khosravi et al. (2011a) and MVE Nix & Weigend (1994) method in terms of smoothness, coverage probability, and width of PI as it uses the gradient-based technique for solving the optimization problem.

Unlike Tube loss, QD loss neither guarantees asymptotic coverage nor facilitates the movement of PIs. Moreover, it requires approximating PICP using sigmoidal functions, making PI quality dependent on the accuracy of this approximation.

We perform a simple experiment to clearly observe the advantages of the proposed Tube loss over QD loss function in NN framework. For this, we generate 1000 data points by adding uniform noise to the sinc function as

$$y_i = \frac{sin(x_i)}{x_i} + \epsilon_i, \tag{24}$$

where x_i is $U(-2\pi, 2\pi)$ and ϵ_i is from $U(-1, 1)$.

We train the Tube Loss-based NN model and the QD loss-based NN model to obtain the best PI with confidence 0.95 by tunning the parameter efficiently. After tuning, we fix the NN model, containing 100 neurons in hidden layer with 'RELU' activation function and two neurons in output layer. The 'Adam' optimizer was used for training the NNs.

Figure 5 shows the PI obtained by Tube loss with $r = 0.5$ and QD loss based NN models along with the true target PI. The true target PI was computed by obtaining the 97.5% and 2.5% quantiles of the distribution of y_i/x_i. The boundaries of the tube loss based NN approximates the sinc function (true PI) well, far better than QD loss based NN. The observed (RMSE, PICP, MPIW/PICP) for tube loss and QD loss were $(0.092, 0.95, 1.97)$ and $(0.171, 0.97, 2.11)$ respectively. We have tuned soften parameter s and λ parameters of QD loss as possible with best estimates at $s = 200$ and $\lambda = 0.1$. For QD loss, the tuning of the soften parameter s is important as the quality of approximation depends upon s and impact the quality of PI significantly.

Figure 7 lists the RMSE comparison of Tube loss with QD loss based NN model for different values of softening parameter s. The performance of the QD loss based NN model depends upon the quality of the approximation with sigmoidal function, controlled by the s parameter.

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

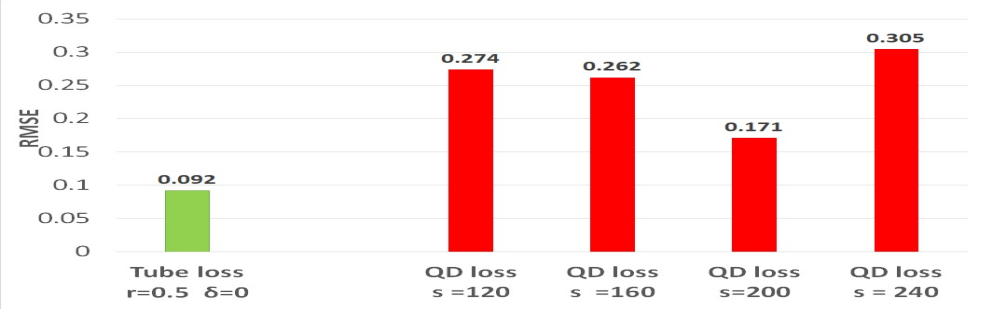


Figure 7: Tube loss based NN approximates true PI better than QD loss.

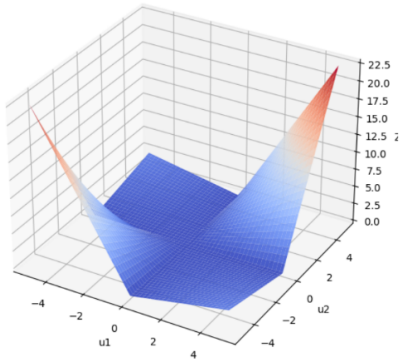


Figure 8: RQR loss function for t=0.9

D.2 TUBE LOSS VERSUS RQR LOSS

At first, we detail about the RQR loss to provide an insight of its limitation. The objective of the RQR is given by

$$L_t((\mu_1,\mu_2),x,y) = \begin{cases} t\big(y-\mu_1(x)\big)\big(y-\mu_2(x)\big), & \text{if } \big(y-\mu_1(x)\big)\big(y-\mu_2(x)\big) \geq 0, \\ (t-1)\big(y-\mu_1(x)\big)\big(y-\mu_2(x)\big), & \text{otherwise,} \end{cases} \tag{25}$$

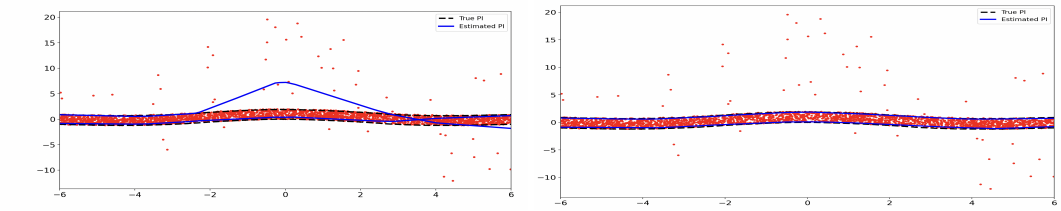
If we consider $u_1 = (y-\mu_1(x))$ and $u_2 = (y-\mu_2(x))$, then the RQR objective (1) can be rewritten as

$$L_t(u_1,u_2) = \begin{cases} tu_1u_2, & \text{if } u_1u_2 \geq 0, \\ (t-1)u_1u_2, & \text{otherwise.} \end{cases} \tag{26}$$

Figure 8 shows the plot of the RQR objective for $t = 09$. The RQR loss function is the continuous loss function but remains non-convex.

We outline the limitations of the RQR loss and advantages of the Tube loss function as follows.

1. The RQR loss function does not involve any inherent mechanism for movement of the PIs so that it can cross through the densest region of the data cloud. In the Tube loss function, there is an implicit mechanism to enable the movement of the PI with parameter r which causes it to obtain the narrow PI than RQR approach in real-world problems.
2. As detail in the (26), the RQR objective provides asymmetric penalization based on the product of the two errors, calculated from the distances between the predicted bounds of the PI and the true value. However, minimizing the square of the error is not supposed to obtain a robust estimate in the literature. Therefore, the bounds of the PI estimated by the RQR objective is much sensitive to the presence of the outliers.
3. The original pinball loss function for quantile estimation does not consider the product of the error, while it is minimized to obtain the estimate for expectile regression. For $q = 0.5$,



(a) RQR loss based NN (b) Tube loss based NN

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

the pinball loss estimate is the median function, while the expectile regression estimate is the mean function. Our Tube loss function is the true extension of the pinball loss function for the simultaneous estimation of the PI bounds.

We perform a simple experiment to show the limitation of the RQR objective. We generate an artificial data set $\{(x_i, y_i)i = 1, 2, ...3000\}$ using the relation.

$$y_i = \frac{sin(x_i)}{x_i} + \epsilon_i, \tag{27}$$

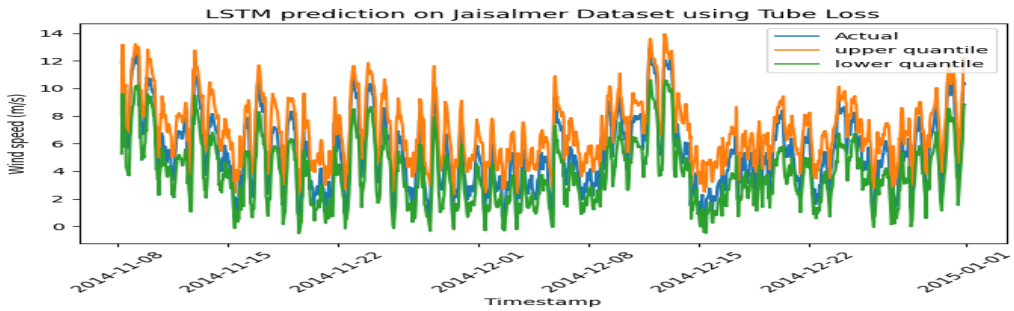
where x_i is $U(-2\pi, 2\pi)$ and ϵ_i is from $U(-1, 1)$. Further, we pollute the dataset by outliers by considering the 2% of randomly selected points and multiply them by 10. The Tube Loss and RQR Loss based NNs were trained using a single-layer NN with 100 hidden neurons, ReLU activation, batch size of 100, 1200 epochs, and the Adam optimizer for target $t = 0.8$. 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 neural networks with $r = 0.3$. In 9(a), the RQR loss estimates deviate considerably from the true PI (black dotted line), and the PI bounds even cross each other. In contrast, the Tube Loss with $r = 0.3$ closely follows the true PI, effectively handling the asymmetric noise introduced by the outlier.

D.3 Additional experiments for probabilistic forecasting with Tube loss function

Due to the unpredictable nature of wind, it is very difficult to obtain the accurate forecast of wind speed. Probabilistic forecasting of wind speed is crucial for effective decision-making in the wind power industry. It helps evaluate potential risks and outcomes, supporting better planning for grid operations and more reliable participation in the electricity market. We systematically details the different steps involved in our probabilistic forecasting experiments as follows.

Dataset Collection:- We selected two datasets collected from two different locations: Jaisalmer and San Francisco. The Jaisalmer dataset contains hourly wind speed measurements at a height of 120 meters for 8,760 hours, while the San Francisco dataset includes 26,304 hourly observations at the same height. To demonstrate the effectiveness of the Tube loss methodology, we applied it using three popular deep auto-regressive models: LSTM, GRU, and TCN.

Baseline Methods:- Quantile regression using the pinball loss function is one of the most widely adopted approaches for probabilistic wind speed forecasting. Several studies have explored neural and deep learning models based on this technique, including works such as (Heng et al. (2022), Cui et al. (2017), Hu et al. (2020b), Wan et al. (2016), Yu et al. (2021), Zhu et al. (2022), Cui et al. (2022)), and (Zhu & He (2024)). Additionally, QD loss-based deep architectures have been used for wind forecasting in (Saeed et al. (2024)) and (Hu et al. (2020a)). Accordingly, we use LSTM, GRU, and TCN architectures trained with pinball loss functions as baseline methods in our experiments. For avoiding the occurrence of NaN losses during the computation, we use the QD^+ loss function in place of QD loss function in LSTM, GRU, and TCN architectures for obtaining the probabilistic forecast of wind speed. Recently, the DeepAR model (Salinas et al. (2020)) has gained popularity for probabilistic forecasting and has been applied in various time series applications, including wind power forecasting (Arora et al. (2022)). Moreover, several recent studies, such as (Yang et al. (2021), Zhang et al. (2020)), and (Men et al. (2016)), demonstrate the effectiveness of Mixture Density



(a)

Figure 10: Probabilistic forecast of the Tube loss based LSTM on Jaisalmer wind dataset.

Networks (MDN) in wind forecasting. Therefore, we also include DeepAR and MDN as baseline models in our evaluation. In addition, we incorporate the pre-trained Time-GPT model (Garza et al. (2023)), a foundation model for time series forecasting, as another baseline for comparison.

We have trained different deep forecasting architectures such as GRU, LSTM and TCN with the Tube loss function on Jaisalmer and San Franciso datasets and compared their performance with existing baseline methods for probabilistic forecast at Table 5 and 4 respectively. In these tables, the probabilistic forecasting models are ranked according to the comparison rules outlined at the beginning of the experimental section.

Rank	Model	PICP	MPIW	MPIW/PICP
1	GRU + Tube	0.9550	3.392	3.55
2	TCN + Tube	0.9581	3.453	3.60
3	TCN + QD$^+$	0.9589	3.511	3.66
4	LSTM + Tube	0.9589	3.697	3.86
5	GRU + QD$^+$	0.9666	3.966	4.10
6	TCN + Quantile	0.9674	4.094	4.23
7	LSTM + QD$^+$	0.9689	4.470	4.61
8	MDN	0.955	4.626	4.84
9	LSTM + Quantile	0.979	4.937	5.04
10	GRU + Quantile	0.9891	5.365	5.43
11	Deep AR	0.9969	6.3553	6.38
12	Time GPT	0.9417	10.608	11.27

Table 5: Ranking of different deep probabilistic forecasting methods on Jaisalmer Dataset with target coverage $t = 0.95$.

Analysis:- In Tables 5 and 4, we observe that the Tube loss-based probabilistic forecasting models consistently achieve the target coverage for future wind observations by calibrating narrower PIs. The Tube loss-based deep forecasting models secure top ranks in both tables and outperform recent wind probabilistic forecasting models from the literature. Figure 10 shows the probabilistic forecast obtained by the Tube loss LSTM model on Jaisalmer dataset.

D.4 Tunning of the parameters in the Tube loss

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 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.

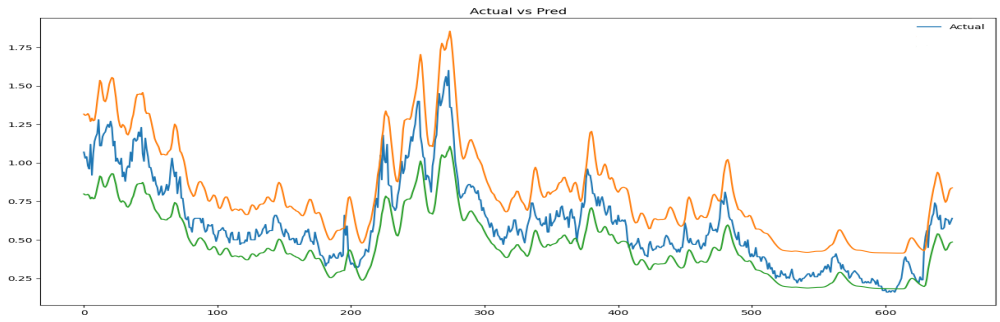
dataset	r (tube_r)	delta (penalty)	lr	batch	dropout	epochs
boston	0.5	0.03	0.05	10000	0.2	400
concrete	0.25	0.05	0.015	64	0.25	150
cpu_act	0.5	0.03	0.05	10000	0.2	400
energy	0.9	0.01	0.05	10000	0.2	400
kin8nm	0.1	0.01	0.05	10000	0.2	400
miami	0.3	0.05	0.05	10000	0.2	400
naval	0.3	0.05	0.05	10000	0.2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

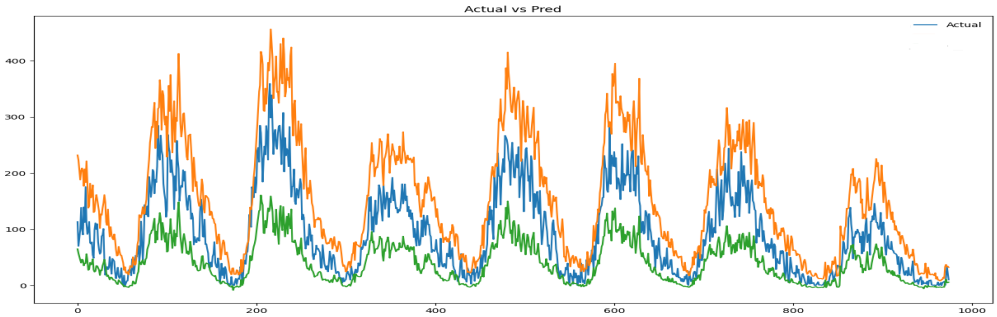
For numerical results presented in the Table 3, the tuned parameters are detail in the Table 7. Also, the plot obtained by the Tube loss based LSTM for benchmark datasets reported in Table 3, are shown in the 11.

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

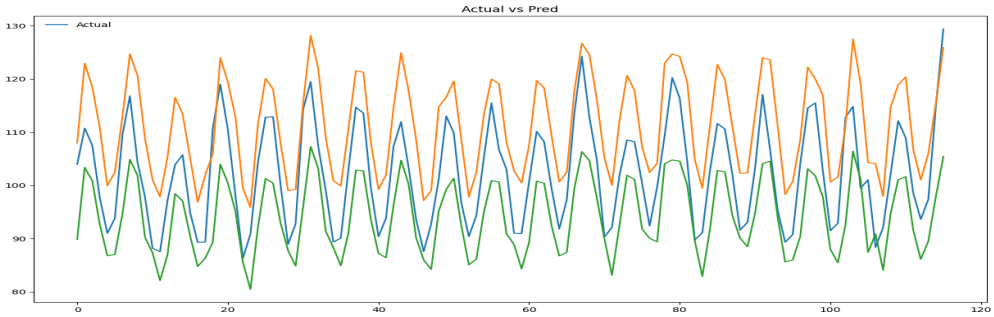
Table 7: Tuned parameter values for Q-LSTM and T-LSTM model for considered benchmark datasets in Table 3. The LSTM architecture [128(0.4),64(0.3),32(0.2)] means that three hidden layers with neurons 128, 64 and 32 respectively and each hidden layer is followed by a drop out layer which are 0.4,0.3 and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



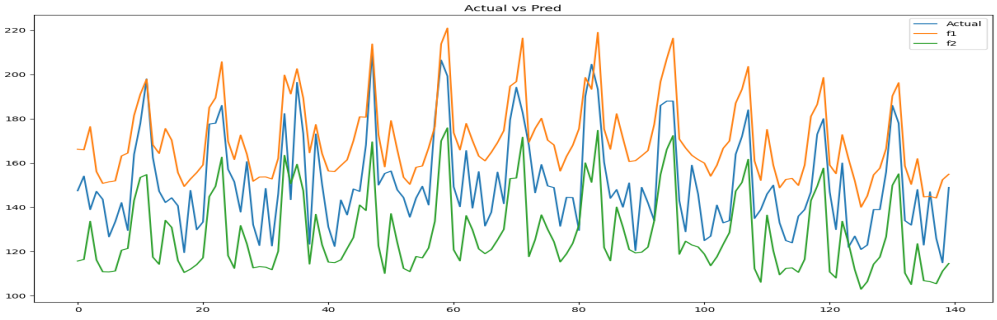
(a) SWH



(b) Sunspot



(c) Electric Production



(d) Beer Production

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

E EXTENSION OF THE TUBE LOSS IN CONFORMAL REGRESSION

At last, we compare the Tube loss based Conformal Regression (TCR) model with the CQR model (Romano et al. (2019)) to show the efficacy of the Tube loss approach over the quantile approach in conformal regression setting. To conduct this evaluation, we use popular benchmark datasets and randomly split each into training, calibration, and test sets in a ratio of 3:1:1, repeated across ten different trials. We test the performance of the CQR and TCR in each trail and report the mean of PICP and MPIW in Table 8 by setting the target calibration $t = 0.9$. We also record the number of trials in which a given CR model fails to achieve the target calibration level of $t = 0.9$, denoted as "Error" and report these values for both the CQR and TCR models in Table 8 for the comparison. Across 90 trials conducted on the 9 benchmark datasets listed in Table 8, the CQR model failed to achieve the target calibration on the test set in 29 cases, while the TCR model failed in 21 cases. The TCR model achieves a significant reduction in total training time compared to the CQR model,

Dataset	PICP		Error		MPIW		Time (s)	
	CQR	TCQR	CQR	TCQR	CQR	TCQR	CQR	TCQR
Concrete	0.90 ± 0.04	0.91 ± 0.03	2	3	17.11 ± 1.76	19.89 ± 1.46	34.04 ± 2.66	17.32 ± 1.99
Bike	0.90 ± 0.01	0.90 ± 0.01	3	3	192.51 ± 43.23	171.77 ± 9.12	74.36 ± 7.05	40.21 ± 4.31
Star	0.90 ± 0.02	0.90 ± 0.02	3	3	1022.19 ± 40.99	982.83 ± 43.13	32.06 ± 2.36	16.28 ± 1.16
Community	0.90 ± 0.02	0.91 ± 0.02	5	1	0.43 ± 0.02	0.47 ± 0.02	17.30 ± 3.61	9.01 ± 0.18
Facebook	0.91 ± 0.01	0.90 ± 0.00	0	1	18.51 ± 4.46	17.78 ± 1.63	182.81 ± 5.48	99.35 ± 3.53
Boston Housing	0.89 ± 0.03	0.92 ± 0.03	6	2	9.38 ± 0.73	13.12 ± 3.70	4.73 ± 0.06	2.66 ± 0.07
Naval	0.90 ± 0.01	0.90 ± 0.01	1	4	0.02 ± 0.00	0.02 ± 0.00	83.10 ± 6.92	46.70 ± 4.28
Yatch Hydro	0.86 ± 0.04	0.90 ± 0.05	7	4	2.37 ± 0.71	2.47 ± 0.56	41.72 ± 8.54	20.39 ± 3.39
Auto MPG	0.92 ± 0.04	0.93 ± 0.04	2	1	10.03 ± 0.94	9.90 ± 1.24	53.21 ± 4.64	28.11 ± 4.18

Table 8: Comparison of the TCR method with CQR method on different benchmark datasets

with an improvement of approximately 47%. This is primarily because TCR estimates both quantile bounds simultaneously by training a single network.