Estimation and Applications of Quantiles in Deep Binary Classification

Anuj Tambwekar, Anirudh Maiya, Soma Dhavala, and Snehanshu Saha, *Senior Member, IEEE*

***Abstract*—Conditional quantiles obtained via regression are used as a robust alternative to classical conditional means in econometrics and statistics, as they can capture the uncertainty in a prediction, and model tail behaviors, while making very few distributional assumptions. In this work, we extend the notion of conditional quantiles to the binary classification setting—allowing us to quantify the uncertainty in the predictions, increase resilience to label noise, and provide new insights into the functions learnt by the models. We accomplish this by defining a new loss called binary quantile regression loss. We compute the Lipschitz constant of the proposed loss and show that its curvature is bounded under some regularity conditions. These properties are later used to characterize the error rates of the learning algorithms and to accelerate the training regime with using Lipschitz adaptive learning rates. We leverage the estimated quantiles to obtain individualized confidence scores that provide an accurate measure of a prediction being misclassified. We aggregate these scores to provide two additional metrics, namely, confidence score and retention rate, which can be used to withhold decisions and increase model accuracy. We also study the robustness of the proposed nonparametric binary quantile classification framework, and finally, we demonstrate that quantiles aid in explainability as they can be used to obtain several univariate summary statistics that can be directly applied to existing explanation tools.**

***Impact Statement*—While machine learning models have made significant strides in improving healthcare by detecting diseases, there is still hesitancy among both patients and doctors in adopting ML diagnostic tools and trusting their predictions. This is primarily due to the fact that the confidence scores reported by a model are not true representations of the likelihood of a diagnosis being correct. Our method solves this issue by providing an absolute measure of the chance that a sample was classified incorrectly by computing the uncertainty in the function learnt by the ML model. In addition, it provides additional resilience to mislabeled training data, as well as a visualization to both doctors and patients as to why the decision was made. This method serves as an add-on to any existing classifier, and thus augments any deep learning model that aims to answer a yes-or-no question.**

***Index Terms*—Backpropagation, interpretability, predictive models, robust predictions, uncertainty quantification.**

Manuscript received May 23, 2021; revised August 9, 2021 and August 30, 2021; accepted September 18, 2021. Date of publication September 27, 2021; date of current version March 24, 2022. This paper was recommended for publication by Associate Editor P. D'Urso upon evaluation of the reviewers' comments. *(Corresponding author: Snehanshu Saha.)*

Anuj Tambwekar and Anirudh Maiya are with the PES University, Bengaluru 560085, India (e-mail: anujstam@gmail.com; maiyaanirudh@gmail.com).

Soma Dhavala is with the Wadhwani Institute for Artificial Intelligence, Bengaluru 560085, India (e-mail: soma@mlsqaure.org).

Snehanshu Saha is with the Department of CSIS and APPCAIR, Birla Institute of Technology and Science, K K Birla Goa Campusi, Zuarinagar 403726, India (e-mail: snehanshus@goa.bits-pilani.ac.in).

Digital Object Identifier 10.1109/TAI.2021.3115078

I. Introduction

DEEP learning (DL) has seen tremendous success over the last few years in the fields of computer vision, speech and natural language processing [1]. As DL makes its way into numerous real-world applications, the focus is now shifting from achieving state-of-the-art performance to questions about explainability, robustness, trustworthiness, fairness, and training efficiency, among others. Uncertainty quantification (UQ) is, therefore, witnessing a renewed interest within the DL community. All of the aforementioned aspects need to be tackled in a holistic manner to democratize artificial intelligence (AI) and make AI equitable for all [2]. In a seminal paper, Parzen [3] provides a foundation for exploratory and confirmatory data analysis using quantiles, and later argues for unification of the theory and practice of statistical methods with them [4]. In this work, we take these ideas forward and show how some of the problems mentioned before in the DL context can be solved using a quantile-centric approach.

Quantile regression (QR) generalizes the traditional mean regression to model the relationship between the quantiles of the response to the dependent variables, median regression being the special case [5]. QR inherits many desirable properties of quantiles: they are robust to noise in the response variable, have a clear probabilistic interpretation, and are equivariant under monotonic transformations. They also have appealing asymptotic properties under mild assumptions both in the parametric [6] and the nonparametric [7] settings. QR has found many successful applications in econometrics [8] and statistics [9], such as modeling growth curves, extreme events, and even in the robust regression contexts as seen in [10], [11]. Its introduction to the machine learning community is relatively recent, where Ichiro *et al.* [12] showed the relationship between ν-support vector machines and QR, for example. Tagasovska and Lopez-Paz [13] applied QR for modeling *aleatoric* uncertainty in deep learning via prediction intervals (PIs).

Unlike previous works, we study QR in the binary classification. We make this choice for the following reasons.

- Much of the earlier work on QR can be extended to the deep learning context, with very minimal effort, which is not the case with classification tasks.
- Despite the dominance of classification tasks in the DL space, reliance on the popular but problematic binary cross-entropy (BCE) is still prevalent, and there is a need to find viable alternatives. Both Refs. [14] and [15] show how BCE is not a true measure of classification success

likelihood, while Ref. [16] demonstrates how it is vulnerable to adversarial attacks.
- We also want to study several problems together, as mentioned before, with binary classification as a test bed. We hope that, our findings can be extended to multiclass settings in future.

In the rest of this work, first, we set up the problem, along with notations, and derive the binary quantile regression (BQR) loss. We derive some properties of the loss function and provide the learning rates under the regularity assumptions. Later, for each of the subproblems, namely, UQ, robustness, explainability, and adaptive learning rates, we provide the necessary background, develop the idea, and provide the results. Finally, we discuss our findings, scope for improvements, and new opportunities. The code can be found:[1]

Our contributions are as follows.
- We establish the BQR loss that allows a binary classification network to learn the conditional quantiles of the underlying latent function it is attempting to learn. We verify the legitimacy of our BQR loss by showing that the quantiles generated provide accurate coverage in cases where the underlying distribution is known.
- We define two new metrics *Confidence Score (δ)* and *Retention Rate (r_r)* by using the generated quantiles. We show that δ is a true measure of UQ by showing a linear relationship between δ and likelihood of misclassification. These metrics can be used to withhold low confidence decisions at run-time, which improves model performance without any additional effort.
- We show that the median classifier obtained via BQR is more resilient to label noise. We also show that traditional model explainability techniques such as `Shapely` can be applied to the quantiles we generate.
- Finally, we show that BQR loss can be enhanced using Lipschitz adaptive learning rates, thereby speeding up convergence.

II. Binary Quantile Regression

In this section, we provide the necessary background needed to derive the BQR loss, and also show some of its important mathematical properties.

A. Setup

Definition: For any real-valued random variable Z, with distribution function $F(z)$, with $F(z) = P(Z \leq z)$, the quantile function $Q(\tau)$ is given as $Q(\tau) = F^{-1}(\tau) = \inf\{r : F(r) \geq \tau\}$ for any $0 < \tau < 1$.

We collect *n* independent and identically distributed (i.i.d) samples $\{x_i, y_i\}_{i=1}^n$, where $x \in [-1, 1]^d$ is continuously distributed and represents the d-dimensional input features, and $y \in \{0, 1\}$ represents the class label. Given the input features x, we aim to learn a classifier that maps the inputs to the class labels. Let $Q_x(\tau) = f_\tau(x), \tau \in (0, 1)$ be a continuous, smooth, conditional (on x) quantile learnt by the deep neural network (DNN). We consider an architecture of the form $Q_x(\tau) = g_\tau(g_c(x))$ where g_τ is the quantile-specific network and g_c is a layer shared by all quantiles. Zou and Yuan [17] showed that, sharing parameters across quantiles generally leads to better statistical efficiency. It is akin to multitask learning, where each quantile estimation is a task, and our architecture is inspired by this observation.

[1]https://github.com/anujstam/BinaryQuantileRegression

B. Background

Mansky[18] considered the median regression for thresholded binary response models of the form $Z = x\beta + U, Y = I(Z \geq 0)$, where Z is the latent response, β is a $d \times 1$ vector of unknowns, $U \sim F(.)$ are i.i.d errors from a continuous distribution, and $I(.)$ is an indicator function. Later, in [19], he proved the consistency and asymptotic properties of the maximum score estimator and also noted that it can be extended to model other quantiles as a solution to the optimization problem: $\arg\min_{\beta|\beta|=1} \sum_{i=1}^n \rho_\tau(y_i - I(x\beta \geq 0))$. Here, ρ_τ is the check loss or pinball loss, defined as $\rho_\tau(e) = (\tau - I(e < 0))e$ [5]. It is well known that check loss is a generalized version of the mean absolute error (MAE), often used in robust regression settings, and that quantiles minimize the check loss. Ref. [20] and its extension [21] provided efficient estimators by replacing the indicator function with smooth kernels. Benoit and Van den Poel [22] considered the Bayesian counterpart by noting that check loss is the kernel of asymmetric Laplace density (ALD). Below, we extend this to the nonparametric settings and derive the loss function suitable for DNNs.

C. Binary Quantile Regression Loss

Let us reconsider thresholded binary response model

$$y = I(z \geq 0),\ z = Q_x(\tau) = f_\tau(x) + \epsilon$$

where $\epsilon \sim ALD(0, 1, \tau)$ and

$$ALD(y; \mu, \sigma, \tau) \equiv \tau(1 - \tau)\sigma^{-1} \exp(-\rho_\tau((y - \mu)\sigma^{-1})).$$

It can be shown that

$$P(y = 1|f_\tau(x)) \equiv \begin{cases} 1 - \tau \exp((\tau - 1)f_\tau(x)) & 0 < f_\tau(x) \\ (1 - \tau) \exp(\tau f_\tau(x)) & 0 \geq f_\tau(x) \end{cases}.$$

The empirical loss, under the settings defined earlier, can now be defined as the negative of the log-likelihood function, given as

$$L_{\text{BQR}}(y; f_{\tau(x)}) = y_i \log((P(y = 1|f_\tau(x_i))^{-1})) + (1 - y_i) \log((1 - P(y = 1|f_\tau(x_i)))^{-1}).$$

It is to be noted that, we can recover logistic and probit models, when the error distributions are logistic and normal distributions, respectively. Next, we analyze the learnability of latent functions. Before we do that, we provide two Lemmas.

Lemma II.1: The Lipschitz constant of the BQR loss is $\max(\tau, 1 - \tau)$.

Proof: Recall that the empirical risk under the BQR loss is $L(y,z) = -(y \log p_z + (1-y)\log(1-p_z))$ where

$$p_z \equiv \begin{cases} 1-\tau\exp((\tau-1)z) & z \geq 0 \\ (1-\tau)\exp(\tau z)) & z < 0 \end{cases}.$$

For convenience, let us define $\Delta_z L(y) \equiv \frac{|L(y,z_2)-L(y,z_1)|}{|z_2-z_1|}$. Let us consider the following cases:

Case-1a: $0 < z_1 < z_2, y = 1$

$$\Delta_z L(1) = \frac{\log(1-\tau e^{(\tau-1)z_2}) - \log(1-\tau e^{(\tau-1)z_1})}{z_2 - z_1}.$$

The RHS approaches maximum as $z_2, z_1 \to 0$. Taking the limit with respect to z_1 first, we get

$$\lim_{z1\to 0} \Delta_z L(1) = \frac{\log(1-\tau e^{(\tau-1)z_1}) - \log(1-\tau)}{z_2}$$

and then taking the limit with respect to z_2 later, we get $\lim_{z_2,z_1\to 0} \Delta_z L(1) = \tau$. Therefore, $\Delta_z L(1) \leq \tau$

Case-1b: $0 < z_1 < z_2, y = 0$

$$\Delta_z L(0) = \frac{\log(\tau e^{(\tau-1)z_2}) - \log(\tau e^{(\tau-1)z_1})}{z_2 - z_1}.$$

In this case, the RHS simplifies to $\Delta_z L(0) = \frac{(z_2-z_1)(1-\tau)}{z_2-z_1}$. Therefore, $\Delta_z L(0) \leq 1-\tau$.

Case-2a: $z_1 < 0 < z_2, y = 1$

$$\Delta_z L(1) = \frac{\log(1-\tau e^{(\tau-1)z_2}) - \log((1-\tau)e^{\tau z_1})}{z_2 - z_1}.$$

The RHS approaches maximum as $z_2, z_1 \to 0$. Taking the limit with respect to z_1 first, we get

$$\lim_{z1\to 0} \Delta_z L(1) = \frac{\log(1-\tau e^{(\tau-1)z_2}) - \log(1-\tau)}{z_2}$$

and then taking the limit with respect to z_2, we get $\lim_{z_2,z_1\to 0} \Delta_z L(1) = \tau$. Therefore, $\Delta_z L(1) \leq \tau$

Case-2b: $z_1 < 0 < z_2, y = 0$

$$\Delta_z L(0) = \frac{\log(\tau e^{(\tau-1)z_2}) - \log(1-(1-\tau)e^{\tau z_1})}{z_2 - z_1}.$$

The RHS approaches maximum as $z_2, z_1 \to 0$. Taking the limit with respect to z_1 first, we get

$$\lim_{z1\to 0} \Delta_z L(0) = \frac{\log(1-\tau e^{(\tau-1)z_2}) - \log(\tau)}{z_2}$$

and then taking the limit with respect to z_2, we get $\lim_{z_2,z_1\to 0} \Delta_z L(0) = 1-\tau$. Therefore, $\Delta_z L(0) \leq 1-\tau$.

Case-3a: $z_1 < z_2 < 0, y = 1$

$$\Delta_z L(1) = \frac{\log((1-\tau)e^{\tau z_2}) - \log((1-\tau)e^{\tau z_1})}{z_2 - z_1}.$$

The RHS simplifies to $\Delta_z L(1) = \frac{\tau(z_2-z_1)}{z_2-z_1}$. Therefore, $\Delta_z L(1) \leq \tau$.

Case-3b: $z_1 < z_2 < 0, y = 0$

$$\Delta_z L(0) = \frac{\log(1-(1-\tau)e^{\tau z_2}) - \log(1-(1-\tau)e^{\tau z_1})}{z_1 - z_2}.$$

The RHS approaches maximum as $z_1, z_2 \to 0$. Taking the limit with respect to z_2 first, we get

$$\lim_{z_1\to 0} \Delta_z L(0) = \frac{\log(1-\tau) - \log(1-\tau e^{(\tau-1)z_1})}{z_1}$$

and then taking the limit with respect to z_1, we get $\lim_{z_1,z_2\to 0} \Delta_z L(0) = 1-\tau$. Therefore, $\Delta_z L(0) \leq 1-\tau$.

Hence, $\forall z_1, z_2 \in R, y \in \{0,1\}$

$$\Delta_z L(y) \equiv \frac{|L(y,z_2) - L(y,z_1)|}{|z_2 - z_1|} \leq \max(1-\tau, \tau).$$

Implication of Lemma II.1: This lemma implies that the rate of change of the BQR loss function is bounded, and that this bound is determined by the quantile being fit. For instance, the rate of change of BQR for the median (τ=0.5) will never exceed 0.5. Similarly, the rate of change of BQR for the 20% quantiles (τ=0.2) will never exceed 0.8. This allows us to tune the learning rate of each quantile in order to reach convergence faster. This is further discussed in Section VI.

Lemma II.2: BQR also admits a bound in terms of the curvature of a bounded function f^*. That is

$$c_1 E((f-f^*)^2) \leq E(L(y,f) - L(y,f^*)) \leq c_2 E((f-f^*)^2)$$

where $|f^*| < M$, c_1, c_2, M are finite constants, bounded away from 0.

Proof: The proof is similar to the proof of Lemma 8 in [23]. Let M represent the bound of f, i.e., $0 < |f| < M$. Define $h_a(b) \equiv L(b,y) - L(a,y)$ with $a = f, b = f^*$. We can write its Taylor series expansion, up to quadratic terms, as

$$h_a(b) = h_a(a) + h'_a(a)(b-a) + \frac{1}{2}h''_a(a)(b-a)^2.$$

We will be looking at h''_a to determine the bounds for the curvature of the loss function. Let us consider the following cases.

Case-1: $b \geq 0, a \geq 0$

$$h''_a(b) = (1-\tau e^{-(1-\tau)a})\frac{\tau(1-\tau)^2 e^{-(1-\tau)b}}{(1-\tau e^{-(1-\tau)b})^2}.$$

$h''_a(b) \equiv A_1$ is maximum at $a = 0, b = 0$, and minimum at $a = M, b = M$, therefore

$$\frac{\tau(1-\tau)^2 e^{-(1-\tau)M}}{1-\tau e^{-(1-\tau)M}} \leq A_1 \leq \tau(1-\tau).$$

Case-2: $b \leq 0, a \leq 0$

$$h''_a(b) = \tau^2(1-\tau)(1-(1-\tau)e^{\tau a})\frac{e^{\tau b}}{(1-(1-\tau)e^{\tau b})^2}.$$

$h''_a(b) \equiv A_2$ is maximum at $a = 0, b = 0$, and minimum at $a = -M, b = -M$, therefore

$$\frac{\tau^2(1-\tau)e^{-\tau M}}{1-(1-\tau)e^{-\tau M}} \leq A_2 \leq \tau(1-\tau).$$

Case-3: $b \geq 0, a \leq 0$

$$h''_a(b) = \tau(1-\tau)^3 e^{\tau a}\frac{e^{-(1-\tau)b}}{(1-\tau e^{-(1-\tau)b})^2}.$$

$h''_a(b) \equiv A_3$ is maximum at $a = 0, b = 0$, and minimum at $a = -M, b = M$, therefore

$$\frac{\tau(1-\tau)^3 e^{-M}}{(1-\tau e^{-(1-\tau)M})^2} \leq A_3 \leq \tau(1-\tau).$$

Case-4: $b \leq 0, a \geq 0$

$$h''_a(b) = \tau^3(1-\tau)e^{-(1-\tau)a}\frac{e^{\tau b}}{(1-(1-\tau)e^{\tau b})^2}.$$

$h''_a(b) \equiv A_4$ is maximum at $a = 0, b = 0$, and minimum at $a = M, b = -M$, therefore

$$\frac{\tau^3(1-\tau)e^{-M}}{(1-(1-\tau)e^{-M})^2} \leq A_4 \leq \tau(1-\tau).$$

Therefore,

$$c_1 = 0.5 \min(A_1, A_2, A_3, A_4)$$
$$c_2 = 0.5\tau(1-\tau).$$

Due to Lemmas II.1 and II.2, the BQR loss satisfies (2.1) of [23]. If, in addition, function $f_\tau(x)$ is smooth, bounded and is restricted to lie in Sobolev space (see Assumptions 1–3, [23]), all the results of their paper are directly applicable. In particular, we restate their major result, Theorem 2.

Theorem II.3: Let f be a smooth, bounded function in Sobolev space, learnt by a deep rectified linear unit (ReLU) network with W number of parameters. Under BQR, with probability at least $1 - e^{-\gamma}$, for large enough n, for some $C > 0$,

$$\|f - f^*\|^2_{L_2(x)} = E((f - f^*)^2) \leq B$$

for $B = C(\frac{W \log(W)}{n} \log n + \frac{\log \log n + r}{n} + \epsilon^2_{f^*})$.

Implications of Lemma II.2 and Theorem II.3: These lemmas combined allow us to place bounds on the function curvature and the error. The above nonasymptotic error bounds can even be used to tune and optimize the architectures as a function of the DNN architecture complexity W, sample size n, confidence γ, and approximation error ϵ_{f^*}. Most refreshingly, it allows us to look at the DNN as a decompressor: given quantized outputs y and inputs x, we can train a DNN to decompress the signal $f(x)$.

In Fig. 1, we show the estimated conditional quantiles of the latent response. It is fascinating to see that the original signal is recovered, despite observing only quantized labels at the time of training. In addition, it is worth noting that multiple quantiles are simultaneously estimated. In order to prevent quantile crossing, we add a regularization term

$$L_{\text{BQC}} = \sum_{i=1}^{n} \sum_{p=1}^{m-1} \max(0, Q_{x_i}(\tau_p) - Q_{x_i}(\tau_{p+1}))$$

giving us a regularized BQR: $L_{\text{BQ}} = L_{\text{BQR}} + \lambda L_{\text{BQC}}$. In this work, we find that for most cases, a $\lambda = 1$ was sufficient to prevent crossing. Next, we quantify how well the latent functions are learnt in terms of coverage, where coverage is an estimate of $P_{x,y}(x < Q_x(\tau))$ that should be close to the nominal value τ.

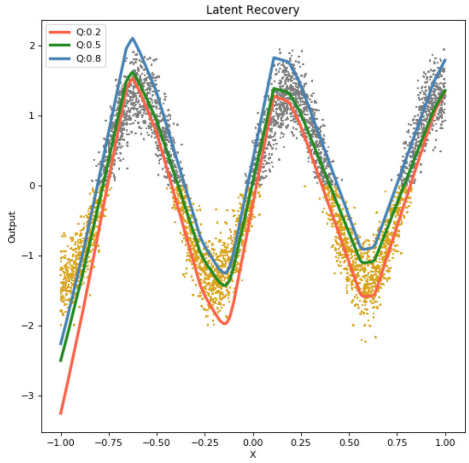


Fig. 1. Recovering the latent response $f_\tau(x)$ for D1.

TABLE I
COVERAGE VALUES FOR SIMULATED DATASETS

	Coverage for τ								
Dataset	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
D1	0.10	0.18	0.28	0.40	0.51	0.59	0.69	0.81	0.92
D2	0.14	0.23	0.33	0.45	0.51	0.60	0.69	0.79	0.84
D3	0.10	0.19	0.28	0.39	0.51	0.58	0.69	0.81	0.86
D4	0.04	0.09	0.22	0.37	0.50	0.67	0.75	0.82	0.91
D5	0.08	0.20	0.36	0.46	0.53	0.61	0.70	0.82	0.89
D6	0.05	0.18	0.32	0.40	0.49	0.55	0.69	0.81	0.90

D. Validation Via Coverage

Any of the quantiles generated via BQR must possess the fundamental coverage property described earlier. We compute the coverage by scaling the true latent distribution and the obtained quantiles and verify that the coverage obtained is close to the nominal value τ. The threshold value is subtracted from the true latent, and the resulting distribution is then normalized to have $\mu = 0$ and $\sigma = 1$. For the quantiles, the mean and standard deviation of the median are obtained, and all the quantiles are normalized using these terms.

Coverage in Simulated Datasets: We created a collection of datasets as per the distributions listed below, sampling X from $U(-1, 1)$ and classified the points as class 0 if $y_i \leq \mu$. We computed the coverage for the generated quantiles. The datasets used are formulated as follows, with D5 and D6 being variants of the dataset proposed in [24]. The results can be seen in Table I.

- D1 : $y_i = 5 \sin 8x_i + \zeta_i$, where $\zeta_i \sim N(0, 1)$
- D2 : $y_i = {}^{(4x_i)^2}/_2 + \zeta_i$, where $\zeta_i \sim N(0, 0.5)$
- D3 : $y_i = \sqrt{(4x_i)^2 + 5} - 2.5 + \zeta_i$, where $\zeta_i \sim U(-0.3, 0.3)$
- D4 : $y_i = \zeta_i + \begin{cases} 2x_i \sin(1/2x_i) & x \neq 0 \\ 0 & x = 0 \end{cases}$, where $\zeta_i \sim N(0, 0.5)$
- D5 : $y_i = 2((1 - 3x_i + 2(3x_i)^2) \exp -0.5(3x_i)^2 - 1.5) + \zeta_i$, where $\zeta_i \sim N(0, 0.25)$
- D6 : $y_i = 2((1 - 3x_i + 2(3x_i)^2) \exp -0.5(3x_i)^2 - 1.5) + {}^{\zeta_i}/_4$, where $\zeta_i \sim \chi^2(2)$.

TABLE II
COVERAGE RESULTS FOR BINARY CLASSIFICATION USING THRESHOLDED UCI REGRESSION DATASETS

				Coverage for τ								
Dataset	t	Acc.	RMSE	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Abalone	9	0.81	0.83	0.08	0.19	0.27	0.35	0.55	0.68	0.79	0.88	0.97
	7	0.89	0.89	0.08	0.19	0.31	0.43	0.54	0.65	0.75	0.84	0.96
Boston	22	0.88	0.74	0.17	0.22	0.30	0.39	0.50	0.62	0.71	0.80	0.90
	18	0.90	0.82	0.09	0.20	0.34	0.42	0.53	0.67	0.74	0.80	0.86
California	180K	0.80	0.77	0.05	0.15	0.26	0.36	0.51	0.62	0.75	0.84	0.93
	200K	0.79	0.83	0.10	0.25	0.37	0.48	0.57	0.66	0.73	0.79	0.87
Concrete	35	0.88	0.62	0.09	0.17	0.26	0.36	0.50	0.64	0.76	0.88	0.94
	50	0.91	0.66	0.11	0.18	0.28	0.41	0.50	0.65	0.79	0.83	0.87
Energy	20	0.99	0.40	0.13	0.18	0.27	0.39	0.50	0.66	0.79	0.85	0.91
	15	0.94	0.52	0.07	0.16	0.29	0.37	0.51	0.65	0.74	0.81	0.90
Protein	5	0.82	0.82	0.10	0.22	0.34	0.44	0.53	0.63	0.73	0.83	0.93
	9	0.81	0.84	0.09	0.16	0.30	0.42	0.53	0.62	0.72	0.82	0.92
Redshift	0.65	0.91	0.83	0.09	0.18	0.26	0.37	0.48	0.61	0.81	0.86	0.92
	0.9	0.92	0.88	0.07	0.10	0.15	0.32	0.45	0.70	0.77	0.88	0.96
Wine	5	0.82	0.82	0.08	0.18	0.27	0.38	0.49	0.61	0.72	0.83	0.92
	6	0.93	0.93	0.03	0.12	0.24	0.37	0.51	0.64	0.73	0.80	0.86
Yacht	2	0.98	0.63	0.17	0.27	0.35	0.43	0.49	0.55	0.64	0.81	0.89
	7.5	0.98	0.60	0.19	0.34	0.41	0.45	0.51	0.69	0.84	0.91	0.98

Coverage in Real-World Datasets: We use regression datasets taken from the UCI machine learning repository [25], and convert them into classification tasks by thresholding the target, and converting it into a binary label. We use two different thresholds, one to simulate a balanced classification task, and the other to simulate an imbalanced problem. The results can be seen in Table II. The scaling methodology is the same as the method described for the simulated datasets.

Analyzing the Results: For both simulated and real-world datasets, we observe that reported coverages are very close to their nominal values around the median, and the precision decreases as the nominal quantile moves away from the median. While we do not know the distribution of the estimators, from the classical QR perspective, it suggests that, the precision at the lower quantiles is more dominated by the density terms, than by the $\tau(1-\tau)$ factor [9]. Nevertheless, this indicates that BQR is able to recover the generalized latent function as well as capture the range of possible values the latent function may take at an input. In the next section, we describe how we can use this range of values to form a discrete uncertainty measure for the network's output.

E. Computational Overhead

A straightforward way to assess the computational overhead of BQR is to contrast it with the standard binary cross-entropy (BCE) loss used in binary classification. BQR takes the number of quantiles as an argument. If the number of quantiles desired is k, then the number of outputs in the model is $k+1$ (1 additional for BCE, but this can be omitted). Barring these k additional outputs, the rest of the architecture remains entirely the same. If there are h hidden neurons in the penultimate layer, then the BQR model will possess $(h+1)\times k$ additional parameters—hk additional weights and k additional biases. When coming to the complexity of the loss function itself, the individual formulation for each quantile output is very similar to that of BCE. If we have k quantiles, then the computation cost becomes $k+1$ that of BCE. The computational complexity of BCE itself is determined by the data and the architecture as opposed to the loss. As such, BQR produces negligible additional overhead in terms of computation time as well as space. The maximum overhead we observed was 2.8 s/epoch and 0.01 MB extra space during backpropagation for the Asirra [26] dataset. The overhead per model can be found in Table IX in the Appendix.

III. ESTABLISHING UNCERTAINTY IN CLASSIFICATION WITH QUANTILES

The quantiles obtained via BQR capture the uncertainty present in the latent function learnt by the model. In this section, we discuss how to quantify this uncertainty to provide per-prediction uncertainty estimates. By withholding predictions below a prescribed confidence threshold, we demonstrate that it is possible to increase model accuracy by choosing not to make a decision when uncertainty is high. We also contrast our methodology with a popular uncertainty measurement system known as TrustScore (TS).

A. Formulation and Verification

Nguyen *et al.* [27] studied how DL models can be fooled easily, despite the high confidence in the predictions. Sur and Candès [15] discuss why the classical asymptotic results are not reliable and offer some practical tools. Nar *et al.* [16] show that estimates based on BCE are not consistent. One way to approach the problem is by UQ. Gal and Ghahramani [14] estimate the uncertainty via Monte–Carlo drop-out, while Lakshminarayanan *et al.* [28] propose deep ensembles with a Bayesian justification to report Monte–Carlo estimates of prediction variance. Liu *et al.* [29] pose UQ as a min–max problem, where a single model,

instead of an ensemble, is input-distance aware. Tagasovska and Lopez-Paz [13] propose using quantiles to report the PIs in the regression setting. It is straightforward to establish PIs using the conditional quantiles even in the binary classification setting since $P_{x,y}(x < Q_x(\tau)) = \tau$. It follows then that $[Q_x(0.5\tau), Q_x(1 - 0.5\tau)]$ is a $100(1-\tau)\%$ PI at x. Any monotonic transformation, such as a sigmoid function or an indicator function, can be used to produce PIs in the class probabilities space or the label space. Along with measuring the precision, it is sometimes helpful to know when to withheld from a making prediction. Recently, Jiang *et al.* [30] proposed *TrustScore* based on how close a sample is to a set of high trustworthy samples to that affect. In addition to reporting the precision (via PIs), we can also measure the confidence via confidence score δ defined as follows.

Definition III.1: Confidence score (δ) is defined for a sample x as

$$\delta = \inf_{d \in (0,0.5)} \{d : \text{ s.t } Q_x(0.5 - d) \leq 0 \leq Q_x(0.5 + d)\}.$$

The confidence score δ is thus a metric of how close to the decision boundary the latent function for x_i is. As δ increases, the likelihood of the point being misclassified reduces, as the quantiles for the latent response move further away from the decision boundary. The relationship between misclassification rate and confidence can be explicitly stated as follows.

Theorem III.1: An instance with confidence score δ has a misclassification rate of $0.5 - \delta$.

Proof: Let μ be the median of the latent response z, i.e., $\mu = Q_x(0.5)$, and that $\mu \geq 0$. Note that

$$P(z <= \mu) = P(z <= 0) + P(0 < z \leq \mu).$$

By definition $P(z <= \mu) = 0.5$, $\delta = P(0 < z <= \mu)$, and $P(z <= 0)$ is the misclassification rate. Using the same reasoning, we can show that, when $\mu < 0$, the misclassification rate $P(z > 0)$ is $0.5 - \delta$. Hence, the misclassification rate is $0.5 - \delta$.

To verify the relationship between misclassification and confidence, we use the same datasets and thresholds used in our coverage computation tests, and compute the goodness-of-fit (R^2) score of the expected misclassification rate vs. δ curve on the obtained values misclassification rate per delta. The results can be seen in Table III.[2] In addition, by omitting samples whose δ-score is below a certain threshold, more confident predictions can be obtained, as per the theorem described above. We term this threshold as the *model confidence*. However, it is important to keep in mind that as this tolerance for a certain δ becomes more rigid, the number of acceptable decisions will also reduce. We define the *retention rate* for a given confidence threshold as the ratio of number of points having a δ-score less than the confidence score to the samples available for decision. Table IV shows the retention (r_r) and misclassification rates (m_r) for some standard binary classification datasets such as those from the UCI repository [25], the IMDB movie review dataset [31], and the high-dimensional image datasets—the Asirra Dogs vs. Cats dataset [26] and the Pneumonia X-ray datasets [32].

TABLE III
MISCLASSIFACTION RATE PER δ-SCORE IN ARTIFICIALLY CREATED CLASSIFICATION TASKS

			δ-Score					
Dataset	t	Rate	0.1	0.2	0.3	0.4	0.5	R^2
Abalone	9	m_r	0.40	0.35	0.25	0.15	0.04	0.89
		r_r	1.00	0.84	0.68	0.52	0.33	–
	7	m_r	0.45	0.33	0.23	0.13	0.02	0.95
		r_r	1.00	0.94	0.89	0.80	0.63	–
Boston	22	m_r	0.44	0.33	0.23	0.13	0.02	0.96
		r_r	1.00	0.96	0.91	0.86	0.74	–
	18	m_r	0.30	0.27	0.18	0.10	0.02	0.78
		r_r	1.00	0.97	0.92	0.87	0.78	–
California	1.8	m_r	0.41	0.35	0.24	0.13	0.03	0.94
		r_r	1.00	0.92	0.84	0.74	0.60	–
	2.0	m_r	0.43	0.37	0.26	0.15	0.03	0.88
		r_r	1.00	0.92	0.84	0.76	0.61	–
Concrete	35	m_r	0.42	0.31	0.23	0.11	0.04	0.97
		r_r	1.00	0.94	0.87	0.79	0.66	–
	50	m_r	0.46	0.34	0.20	0.15	0.01	0.94
		r_r	1.00	0.97	0.92	0.88	0.81	–
Energy	20	m_r	0.38	0.21	0.23	0.08	0.00	0.89
		r_r	1.00	0.99	0.99	0.99	0.97	–
	15	m_r	0.42	0.41	0.39	0.21	0.00	0.54
		r_r	1.00	0.96	0.93	0.89	0.84	–
Protein	5	m_r	0.37	0.36	0.24	0.13	0.04	0.90
		r_r	1.00	0.88	0.76	0.61	0.40	–
	9	m_r	0.41	0.37	0.26	0.15	0.04	0.87
		r_r	1.00	0.88	0.75	0.61	0.40	–
Redshift	0.65	m_r	0.40	0.32	0.20	0.12	0.02	0.99
		r_r	1.00	0.96	0.92	0.87	0.78	–
	0.9	m_r	0.39	0.28	0.19	0.19	0.01	0.89
		r_r	1.00	0.97	0.93	0.87	0.81	–
Wine	5	m_r	0.43	0.39	0.29	0.17	0.07	0.70
		r_r	1.00	0.87	0.74	0.57	0.33	–
	6	m_r	0.47	0.36	0.26	0.18	0.03	0.83
		r_r	1.00	0.95	0.90	0.83	0.74	–
Yacht	2	m_r	0.13	0.10	0.03	0.00	0.00	-9.6
		r_r	1.00	1.00	0.99	0.97	0.89	–
	7.5	m_r	0.24	0.10	0.00	0.00	0.00	-1.6
		r_r	1.00	0.99	0.98	0.94	0.86	–

Analyzing the Results: As seen in Tables III and IV, one can clearly see the trend of a lower δ score corresponding to high misclassification rates. The correlation between misclassification rates and δ matches the results as expected by Theorem III.1. This phenomenon is consistent between the artificial binary classification tasks as well as native classification tasks. This clearly shows that δ is an accurate measure of true model classification uncertainty. In addition, we can use the tried and tested classifier metrics, namely AUC, RoC, and precision–recall curves in order to evaluate the classifier per confidence score level. We simply compute the True Positive Rate (FPR)–False Positive Rate (FPR) and precision–recall curves for the classifier, only considering points that have a specific confidence level. Fig. 2 shows an example of the same. As one can note, the classifier performance improves when low confidence labels are withheld.

[2]The R^2 score for yacht is correct. The value is computed using Scikit learn's https://scikit-learn.org/stable/modules/model_evaluation.html#r2-scorer2_score, which ranges from $-\infty$ to 1.0.

TABLE IV
CONFIDENCE SCORE-BASED METRICS FOR BINARY CLASSIFICATION DATASETS

			Confidence Scores(δ)				
Dataset	Acc.	Metric	0.1	0.2	0.3	0.4	0.5
Asirra	0.95	m_r	0.39	0.33	0.24	0.12	0.01
		r_r	1.00	0.98	0.96	0.92	0.86
Banknote	1.00	m_r	0.10	0.00	0.01	0.00	0.00
		r_r	1.00	0.96	0.92	0.86	0.78
Haberman	0.76	m_r	0.43	0.39	0.36	0.19	0.12
		r_r	1.00	0.88	0.77	0.66	0.26
Heart Disease	0.88	m_r	0.46	0.36	0.23	0.12	0.04
		r_r	1.00	0.92	0.85	0.74	0.58
ILP	0.75	m_r	0.41	0.42	0.32	0.22	0.03
		r_r	1.00	0.87	0.58	0.42	0.32
IMDB	0.92	m_r	0.45	0.29	0.20	0.11	0.01
		r_r	1.00	0.98	0.97	0.95	0.91
Ionosphere	0.94	m_r	0.50	0.32	0.17	0.16	0.02
		r_r	1.00	0.97	0.95	0.90	0.82
Pima	0.80	m_r	0.45	0.37	0.30	0.16	0.04
		r_r	1.00	0.88	0.74	0.58	0.37
Pneumonia	0.89	m_r	0.72	0.31	0.24	0.20	0.04
		r_r	1.00	0.99	0.97	0.95	0.92
Sonar	0.94	m_r	0.45	0.38	0.22	0.13	0.03
		r_r	1.00	0.96	0.92	0.87	0.80
Titanic	0.87	m_r	0.41	0.30	0.27	0.15	0.06
		r_r	1.00	0.94	0.88	0.78	0.65
WBC	0.97	m_r	0.36	0.28	0.20	0.11	0.01
		r_r	1.00	0.99	0.97	0.95	0.91

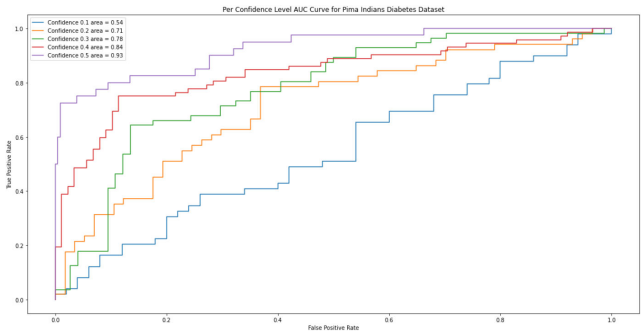


Fig. 2. Per δ-score (confidence score) AUC-ROC curves for the Pima Indians dataset. Each curve represents the AUC–ROC score of all the points with a particular δ. Blue ($\delta = 0.1$) has the lowest, while purple ($\delta = 0.5$) has the highest.

The individualized δ-score performance can be evaluated and used as another metric when deciding whether or not a prediction should be rejected.

B. Comparison With TrustScore

TS was recently proposed in [30] in order to provide a relative measure of how likely a classifier's prediction on sample is correct. We benchmark δ_x with TS by computing them on all the samples in a dataset. Following this, we rank the samples based on TS and create 10 equi-distributed bins, one bin per decile. We the compute the average δ_x and TS of all points in each bin. According to [30], low ranking points are the ones likely to be misclassified.

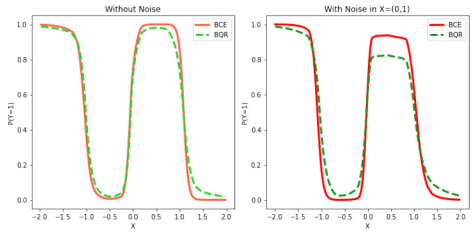


Fig. 3. BRQ vs. BCE class boundaries under label noise. In the region with 20% label noise (X = 0 to 1), BQR (green, dotted) shows a more accurate 80% confidence, while BCE (red, solid) shows an inaccurate 90%.

Analysis of δ vs. TrustScore: As seen in Table V, as TS bin decile increases, the average δ_x score also increases as expected, indicating that our method captures the expected trend. However, note that, the average TS per bin can increase dramatically, as TS is not calibrated and is not intended to be used in isolation, but rather as a relative metric. Further, the real benefit of our approach can be seen in datasets such as Ionosphere, Banknote, and Sonar, wherein even the lowest TS bins have very high δ_x values. This makes perfect sense, as these are easy to classify datasets. However, by construction, TS involves ranking the samples, and, as a result, it is possible for some samples with low confidence to get high TS and vice versa, which is undesirable. Unlike TS, δ_x is well calibrated due to Theorem 3.2, and requires no other models.

IV. ROBUSTNESS TO LABEL NOISE

As noted previously in Section III, BCE can be overconfident or can be fooled easily in the presence of noise. While confidence scores may help detect such spurious cases, is it possible to reduce them in the first place? That brings us to developing robust estimation techniques. It is widely established that quantiles are a type of robust estimators to noise in the response variable (vertical noise). In the classification setting, Ghosh *et al.* [33] showcase the ability of the MAE of the class probabilities being more robust to label noise than BCE but note that training under MAE could be slow. In our case, having shown the promise of BQR in UQ, we are interested in seeing whether median class probabilities are robust to label noise.

To test the hypothesis, we take the D1 dataset, and add normal noise in between X=0 and X=1, resulting in the majority of the labels in the region retaining the same label, but a few labels changing. As one can see in Fig. 3, the confidence exhibited by BCE in the corrupted region is misleading, whereas BQR's is more conservative, while still resulting in the same predicted class. Also, note that the class transitions are smooth in the case of BQR, implying that BQR-based class probabilities can withstand label noise, and somewhat surprisingly are smooth to horizontal noise (contamination of covariates). This observation is stated in the following theorem.

Theorem IV.1: Let $f\tau(y|x^*)$ be L-Lipschitz continuous in x^* for every τ, and let R be the radius of an L_2 ball around x^*. Let $F(y|x^*)$ be the conditional Cumulative Distribution Function

TABLE V
AVERAGE δ_x-SCORE AND TRUSTSCORE VALUES PER TRUSTSCORE BIN

		Trustscore Bin									
Dataset	Avg.	1	2	3	4	5	6	7	8	9	10
Banknote	δ_x	0.49	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	TS	3.69	5.61	6.95	8.22	9.50	11.12	13.01	15.69	20.08	1.79E11
Haberman	δ_x	0.22	0.24	0.25	0.28	0.32	0.36	0.37	0.41	0.44	0.45
	TS	0.60	0.94	1.13	1.39	1.71	2.20	2.83	3.73	5.45	5.3E11
Heart	δ_x	0.35	0.40	0.45	0.45	0.46	0.47	0.47	0.49	0.49	0.49
	TS	0.86	1.08	1.19	1.27	1.36	1.46	1.59	1.81	2.19	2.3E11
ILP	δ_x	0.16	0.19	0.25	0.28	0.29	0.30	0.34	0.37	0.41	0.43
	TS	0.51	0.81	1.00	1.14	1.26	1.42	1.64	1.89	2.38	5.1E11
Iono	δ_x	0.46	0.49	0.49	0.48	0.49	0.50	0.50	0.50	0.50	0.50
	TS	0.91	1.05	1.13	1.37	1.77	2.34	3.11	4.64	6.46	2.5E11
Pima	δ_x	0.23	0.26	0.30	0.37	0.38	0.40	0.41	0.45	0.47	0.48
	TS	0.79	0.99	1.11	1.22	1.33	1.45	1.59	1.78	2.09	3.22
Sonar	δ_x	0.44	0.48	0.49	0.49	0.48	0.49	0.49	0.49	0.49	0.49
	TS	0.94	1.06	1.12	1.20	1.30	1.39	1.50	1.63	2.03	2.54
Titanic	δ_x	0.27	0.38	0.41	0.44	0.45	0.46	0.45	0.46	0.46	0.47
	TS	0.78	1.39	2.32	4.25	6.34	10.97	25.55	1.1E9	6.3E10	8.6E11
WBC	δ_x	0.43	0.48	0.49	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	TS	1.19	1.61	1.95	2.49	3.55	8.3E10	1.6E12	1.8E12	2.1E12	2.4E12

(CDF) of y at x^* obtained via quantities. Then, ∇, the change in the classification probability is at most $\max(|F(y \pm LR|x^*) - 0.5|)$.

Proof: Since $f\tau(.)$ is L-Lipschitz continuous for every τ, it follows that $|\nabla f(x)| \leq L|\nabla x|$. It implies that $|\nabla f(x)| \leq LR$ since $|\nabla x| \leq R$. Therefore, $P(Y < y \pm LR|x) = F(y \pm LR|x)$ and given that $P(Y \geq f(x)) = 0.5$ by reference, it immediately follows that $\nabla p \leq \max(|F(y \pm LR|x) - 0.5|)$.

Choosing a certain type of loss functions such as BQR, and generally speaking, robust estimators [34], [35] is one of the many possible ways to develop robust classifiers. Another recent approach is to constrain the functions that are locally Lipschitz-continous during training [36], [37]. Such restricted functional class can be learnt by constraining the gradients. In the below lemma, we claim that the gradients can be bounded if the functional class is bounded.

Lemma IV.2: Let $f\tau(x)$ be L-Lipschitz continuous in x for a given τ. Then, the gradient of the loss with respect to x $\nabla_x L(x)$ is at most $L \max(\tau, 1 - \tau)$.

Proof: Since $f\tau(.)$ is L-Lipschitz continuous, and the BQR loss is also $\max(\tau, 1 - \tau)$-Lipschitz continuous, it follows from the compositionality of Lipschitz continuous functions that $\nabla L(x^*, \tau = 0.5) \leq L \max(\tau, 1 - \tau)$.

Therefore, by imposing regularity constraints on the functions, BQR networks can generalize well.

To investigate further, we use the same networks as before, with one trained with BCE, and the other with BQR. For each dataset, we vary the percentage of wrongly labeled samples in the training set, and compare the accuracy of the model on the entire real dataset. The results can be seen in Table VI.

Analysis of Results: At no noise, the BQR-based median classifier is usually equivalent to or slightly less accurate than the BCE one. However, as noise levels increase, we observe that the degradation in accuracy is lower for the median classifier as compared to the BCE one. Crossing 20% noise results in BQR outperforming BCE for a majority of the classes, due to the median-based classifier's ability to hold on to its classification performance better than the mean-based BCE classifier as it is less sensitive to outliers. Aside from better tolerance to harsh label noise, BQR allows us to gain insight into the data that we are using. If the median-based classifier is outperforming traditional BCE, it indicates that the data being used likely contains a large amount of outliers.

TABLE VI
BCE VS. BQR LOSS ON LABEL NOISE

		% of Flipped Labels				
Dataset	Loss	0%	10%	20%	30%	40%
Banknote	BCE	1.000	1.000	0.998	0.986	0.925
	BQR	1.000	0.999	0.997	0.989	0.939
Haberman	BCE	0.767	0.766	0.744	0.733	0.642
	BQR	0.764	0.763	0.749	0.735	0.688
Heart	BCE	0.919	0.881	0.822	0.723	0.645
	BQR	0.899	0.859	0.836	0.785	0.700
Ionosphere	BCE	0.962	0.923	0.881	0.799	0.666
	BQR	0.950	0.916	0.887	0.841	0.706
Pima	BCE	0.817	0.803	0.776	0.714	0.618
	BQR	0.802	0.792	0.776	0.735	0.683
Sonar	BCE	0.957	0.880	0.786	0.700	0.586
	BQR	0.946	0.875	0.801	0.718	0.607
Titanic	BCE	0.874	0.868	0.858	0.828	0.756
	BQR	0.872	0.866	0.859	0.845	0.805
WBC	BCE	0.978	0.970	0.964	0.930	0.841
	BQR	0.975	0.970	0.967	0.951	0.917

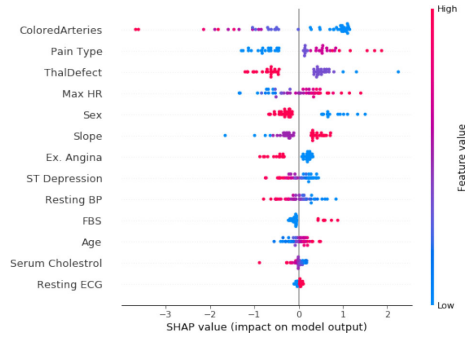


Fig. 4. An example of the obtained `Shapely` statistics of the mean response of the Heart Disease Dataset. Each point in the figure represents one datapoint. The larger the spread of the points for a variable, the more the impact of that variable. The quantiles allow us to obtain linear statistics that can be used to view statistics such as these.

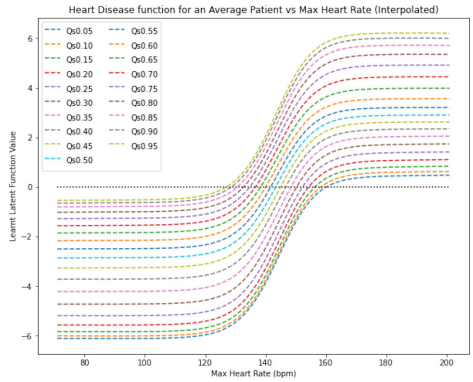


Fig. 5. Heart disease latent vs. max heart rate for an average patient (Interpolated). Each curve represents one quantile. Max heart rate is allowed to vary while other parameters are kept constant. The region of uncertainty is captured and indicates that a change in classification can occur at any point above a heart rate of ĩ30.

V. Explaining the Predictions

A strong criticism of DL models is that they are black-boxes in nature [38]. A wide variety of techniques that attempt to explain the model predictions in terms of activations [39], saliency maps [40], and counter-factuals based on gradient propagation [41] are proposed and are actively being developed. In majority of the cases, the same techniques can be applied to DNNs fit with BQR as well. There are additional classes of explanations for mean predictions like `shapley` [42] and `LIME` [43]. Below, we show, how the conditional quantiles can be used to estimate conditional effects, as well as report conditional means. Recall that $Q_x(\tau)$ is the conditional quantile at the covariate x, where τ is chosen over a set of discrete values $T \in \{\tau_0, \tau_1, \tau_2, \ldots, \tau_m, \tau_{m+1}\}$ with $\tau_0 = 0, \tau_{m+1} = 1$, and there are m outputs available from the DNN corresponding to the remaining values of τ. We can get smooth versions of the quantiles by

$$Q_x^s(\tau)_{\tau\in(0,1)} = \sum_{i=0}^{m} Q_x(\tau_i) \int_{p=\tau_i}^{\tau_{i+1}} \frac{1}{h} K\left(\frac{\tau - p}{h}\right) dp$$

where $K(.)$ is the suitable kernel with bandwidth parameter h [3]. In our examples, we used a Gaussian kernel with bandwidth set to 0.1. Now, one can immediately compute any univariate statistic. In particular, the mean response can be computed as $E(f(x)) = \int_{\tau=0}^{1} Q_x^s(\tau) d\tau$. Likewise, $Var(f(x))$ can also be computed. In fact, any quantity of interest can be computed simply by postprocessing the smoothed full distribution [4]. The smoothed quantiles also allow for more fine-grained values of the confidence metric δ, while keeping the number of prediction quantile outputs manageable.

Visualizations Enabled by Binary Quantile Regression Loss to Aid in Explainability: For our example, we use the Shapely values from the SHAP package [42], which is a game theoretic approach to model the importance of an attribute. Our BQR technique allows us to obtain the underlying latent function, which in turn allows us to use shapely to determine the magnitude of change in the latent function, and its dependency on the inputs. BQR even enables us to use explanation techniques intended for a regression setting in a classification task. Fig. 4 showcases the `shapely` summary statistics of the mean response of the latent on the test data of the heart disease dataset via quantile interpolation. It allows us to capture information about which parameters will influence the decision. In addition to this, we also show through Fig. 5 that we can view the effects of a single variable on the latent function—instead of just observing how the labels switch, we can actually view the function learnt. The figure graphically showcases the region of uncertainty, something which cannot be obtained from conventional binary classifiers as they provide only a single threshold value.

VI. Lipschitz Adaptive Learning Rates for Binary Quantile Regression Loss

A critical parameter in training DNNs via stochastic gradient descent is the learning rate. One of the early approaches to adapt the learning rate is by recognizing the inverse relationship between the step size of gradient descent update and the Lipschitz constant of the function being optimized [44]. Since the Lipschitz constant is generally unknown *a priori*, Palgianakos *et al.* [45] estimate a local approximation during training the feedforward networks. Yedida and Saha [46] derived adaptive learning rates by exploiting the gradient properties of deep ReLU networks, and Saha *et al.* [47] verified the same both theoretically and experimentally on numerous datasets. These Lipschitz constant-based adaptive learning rates have also been applied on models for large datasets [48]. We summarize the key results of the Lipschitz adaptive learning rate (LALR), in the following proposition.

Proposition VI.1: In a deep ReLU network, let constant k_z be the supremum of gradients with respect to the function, and let L be the Lipschitz constant of the Loss. Then, the adaptive learning rate η is $\eta = (k_z L)^{-1}$, where the weight update rule is $w^t = w^{t-1} - \eta \nabla L(f(x))$.

This particular choice of LALR under the assumption that gradients cannot change arbitrarily fast ensures a convex quadratic upper bound, minimized by the descent step.

To show the efficacy of the Lipschitz constant-based adaptive learning rate, we compared the performance of the adaptive learning rate versus fixed learning rates of 0.01 and 0.1, and tested how quickly they were able to reach a specified target accuracy in terms of number of epochs. The results can be found

TABLE VII
CONVERGENCE COMPARISON BETWEEN THE ADAPTIVE AND FIXED LEARNING RATES FOR SGD

Dataset	Accuracy	$N_{0.01}$	$N_{0.1}$	$N_{1/L}$
Banknote	0.99	945	104	14
Haberman	0.80	N/A (0.775)	773	78
Heart	0.85	221	161	4
ILP	0.75	N/A (0.733)	317	28
IMDB	0.90	N/A (0.716)	106	27
Ionosphere	0.90	1841	104	6
Pima	0.80	4099	417	53
Sonar	0.97	2199	320	40
Titanic	0.87	982	152	17
Winsconsin	0.97	1577	101	8

TABLE VIII
ADAPTIVE LR PERFORMANCE IN DEEP BINARY IMAGE CLASSIFICATION

Dataset	Arch.	Target Acc.	LR	N_E	T_E (min)
Asirra	Resnet18	0.76	Fixed (0.01)	16	2.5
			Fixed (0.1)	15	2.5
			Adaptive	6	4.2
	Resnet50	0.70	Fixed (0.01)	20	4.7
			Fixed (0.1)	16	4.7
			Adaptive	5	7.0
	Densenet	0.86	Fixed (0.01)	20	4.9
			Fixed (0.1)	8	4.9
			Adaptive	9	12.1
Pneumonia	Resnet18	0.83	Fixed (0.01)	18	1.2
			Fixed (0.1)	7	1.2
			Adaptive	6	2.3
	Resnet50	0.82	Fixed (0.01)	20	1.9
			Fixed (0.1)	15	1.9
			Adaptive	9	3.2
	Densenet	0.81	Fixed (0.01)	20	1.9
			Fixed (0.1)	10	1.9
			Adaptive	3	3.2

in Table VII. N/A indicates that the classifier was unable to reach the accuracy threshold within 5000 epochs (500 for IMDB)—if this occurs, the maximum accuracy reached is provided as well. For IMDB, we use an embedding dimension of size 100, and a two-layer Long Short-Term Memory network (LSTM) of dimension 256 which feeds a linear layer.

For our image datasets, we compared the efficacy of the LALR on various deep architectures. The Resnet [49] implementations are Pytorch's default Resnet18 and Resnet50 implementations of 18 and 50 layers each, while the Densenet [50] implementation is Pytorch's Densenet121 architecture consisting of 10 layers, out of which 4 are Dense blocks, for a total of 121 layers. For all models, the optimizer was Stochastic Gradient Descent (SGD), and each test was run for 20 epochs. We obtained the best validation accuracy of LR=0.01 and found the number of epochs for the other learning rates to achieve both a training and validation accuracy equal to or greater than LR=0.01.

Analysis of the Results: As seen in Tables VII and VIII, adaptive learning rates are able to substantially improve the performance of BQR networks that use SGD. In the case of data with fewer dimensions (Table VII), we observe that there is a reduction of an order of magnitude in the number of epochs needed to converge, despite each epoch taking an equivalent amount of time. In the cases of high-dimensional data, however (Table VIII), we notice less improvement. This is likely due to the increased complexity of the network architecture, which results in individual epochs taking longer when using LALR owing to the computation of k_z. This can be seen very evidently in the Asirra dataset.

VII. KEY LIMITATIONS AND SCOPE FOR FUTURE WORK

One important question still remains: how to handle a multi-class setting? The proposed implementation of BQR is unable to handle problems in this domain. One obvious choice is to adopt a one-vs.-all strategy, and develop appropriate voting techniques. When the classes are ordinal, one could extend the thresholded model to include more cut-points. In fact, this would be a more appropriate modeling choice than using softmax-classifier for ordinal classification. Otherwise, it is not trivial to extend BQR to multiclass setting because there is no unique way to define multivariate quantiles. But it is extremely interesting to consider depth quantiles as described by [51] and [52] and extend the BQR loss in that direction. Additionally, applying a weighting strategy akin to [34] or [35], coupled with an adversarial training regime, could make BQR more resilient to adversarial attacks.

VIII. CONCLUSION

In this work, we propose a method to obtain the conditional quantiles of the underlying latent function learnt by a binary classifier using the BQR Loss. Thanks to few properties of BQR, we can derive nonasymptotic errors rates as a function of sample size, architecture complexity, approximation errors, and confidence. Based on both simulated and real datasets, we empirically showed that estimated quantiles have good coverage. Later, we showed how individual confidence scores can be obtained, besides being able to report the precision of the class probabilities in terms of PIs. We also extend this UQ technique to the sample confidence score (δ_x) that are well calibrated in terms of misclassification rate. Classifier accuracy can be improved by choosing not to decide the label based on wether or not the δ score is below a threshold. We quantified this notion in terms of *model confidence* and *retention rate*. We showed how a median classifier learnt with BQR is more robust to label noise compared to BCE. We also show how one could report several univariate summaries from quantiles, allowing off-the-shelf explanation techniques like `Shapley` to be applied. Finally, we also demonstrate that advancements in adaptive learning rates via LALR are directly applicable to BQR.

Two themes emerge out of our work.

- Focusing on the smooth functional spaces offers a tremendous potential to understand the task learnability in terms of approximation errors, sample size, architecture complexity, and consistency, among other theoretical aspects.
- Quantile-inspired loss functions provide a natural framework to address many seemingly disparate problems in DL that, until now, were tackled in isolation.

Hopefully, our work convinces readers that both of these share a synergistic relationship. We believe that this encourages the development of a coherent set of technologies to advance the state-of-the-art.

TABLE IX
COMPARISON BETWEEN TIME PER EPOCH AND MEMORY UTILIZATION OF MODELS TRAINED ON BQR AND BCE LOSSES. THE FIRST TWO ROWS ARE RESNET18 NETWORKS, WHILE THE REMAINING ARE DNNS WITH TWO HIDDEN LAYERS

Dataset	Time/Epoch (sec)		Model Size (MB)		Max Memory (MB)	
	BQR	BCE	BQR	BCE	BQR	BCE
Asirra	143.437	140.613	46.3	46.29	5211.59	5211.58
Pneumonia	75.974	79.694	46.27	46.27	5160.19	5160.17
Banknote	0.277	0.153	0.01	0.01	0.07	0.06
Haberman	0.17	0.088	0.01	0.01	0.07	0.06
Heart	0.175	0.09	0.01	0.01	0.07	0.07
ILP	0.172	0.086	0.01	0.01	0.07	0.07
Iono	0.191	0.091	0.02	0.02	0.09	0.08
Pima	0.302	0.121	0.01	0.01	0.07	0.07
Sonar	0.142	0.085	0.03	0.03	0.11	0.1
Titanic	0.256	0.146	0.01	0.01	0.07	0.07
WBC	0.177	0.104	0.01	0.01	0.07	0.07

TABLE X
LIST OF SYMBOLS AND NOTATIONS USED

Symbol/Notation/Abbreviation	Meaning
I	Identity Function
ALD	Asymmetric Laplace Distribution
μ	Mean
σ	Standard Deviation
τ	Quantile control variable
$Q(\tau)$	The Quantile Function for τ
$L()$	Loss function
ρ	Check Loss
$P()$	Probability
BQR	Binary Quantile Regression (Loss)
BCE	Binary Cross Entropy (Loss)
exp	Exponential Function
$f*$	Smooth function learnt by minimizing the loss function
ϵ_{f*}	Approximation error of the DNN
δ	Confidence Score
m_r	Model Confidence
r_r	Retention Rate
TS	Trustscore
UQ	Uncertainty Quantification
$K()$	Kernel Function
h	Kernel Bandwidth Parameter
η	Learning Rate
∇	Gradient
LALR	Lipschitz Adaptive Learning Rate
MAE	Mean Absolute Error
AUC-ROC	Area under the Curve - Receiver Operating Characteristics

APPENDIX

A. Computational Overhead for Various Datasets

We provide a comparison between BCE and BQR (nine quantiles) in terms of computation time and memory across various data sets below. The Model size column denotes the total size of the model, i.e., the sum of all its parameters. The Max memory column refers to the total memory used by the model during backward pass.

As Table IX makes evident, the memory usage and computation time are far more dependent on the model architecture than on the usage of BQR. BQR results in minimal overhead, with the highest observed being just 2.8 s.

B. Symbols, Notations, and Abbreviations

Table X provides a list of the various symbols, notations, and abbreviations used in this article.

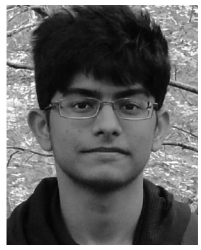
ACKNOWLEDGMENT

The authors would like to thank the Science and Engineering Research Board (SERB), Department of Science and Technology (EMR/05687), Government of India, for supporting our research by providing us with resources to conduct our experiments. The authors are indebted to Prof. Probal Choudhury, ISI Kolkata for suggestions and critical insights which helped the manuscript immensely. Anuj and Anirudh would like to thank Inumella Sricharan and Prof. K.S. Srinivas from PES University, for their help. The authors would like to thank the Science and Engineering Research Board (SERB), Department of Science and Technology, Government of India, for supporting our research by providing us with resources to conduct our experiments.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, May 2015.
[2] S. Ahmed, R. S. Mula, and S. S. Dhavala, "A framework for democratizing AI," 2020, *arXiv:2001.00818*.
[3] E. Parzen, "Nonparametric statistical data modeling," *J. Amer. Stat. Assoc.*, vol. 74, no. 365, pp. 105–121, 1979.
[4] E. Parzen, "Quantile probability and statistical data modeling," *Stat. Sci.*, vol. 19, pp. 652–662, Nov. 2004.
[5] R. Koenker and G. B. Bassett, "Regression quantiles," *Econometrica*, vol. 46, pp. 33–50, 1978.
[6] S. Portnoy and R. Koenker, "Adaptive l-estimation for linear models," *Ann. Statist.*, vol. 17, no. 1, pp. 362–381, 1989.
[7] P. Chaudhuri, K. Doksum, and A. Samarov, "On average derivative quantile regression," *Ann. Statist*, vol. 25, no. 2, pp. 715–744, 1997.
[8] A. C. Cameron and P. K. Trivedi, *Microeconometrics Using Stata*. 2nd ed. College Station, TX, USA: Stata Press, 2010.
[9] R. Koenker, *Quantile Regression, Ser. Econometric Society Monographs*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[10] R. Maronna, D. Martin, and V. Yohai, *Robust Statistics: Theory and Methods*. Hoboken, NJ, USA: Wiley, Jun. 2006.
[11] P. Chaudhuri, "Generalized regression quantiles: Forming a useful toolkit for robust linear regression," in *L1 Stat. Anal. Related Methods–Proc. 2nd Int. Conf. L1 Norm Related Methods*. Amsterdam, The Netherlands: North Holland, 1992, pp. 169–185.
[12] T. Ichiro, V. L. Quoc, D. S. Timothy, and J. S. Alexander, "Nonparametric quantile estimation," *J. Mach. Learn. Res.*, vol. 7, pp. 1231–1264, Jul. 2006.
[13] N. Tagasovska and D. Lopez-Paz, "Single-model uncertainties for deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 6417–6428.
[14] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.
[15] P. Sur and E. J. Candès, "A modern maximum-likelihood theory for high-dimensional logistic regression," *Proc. Nat. Acad. Sci.*, vol. 116, no. 29, pp. 14516–14525, 2019.
[16] K. Nar, O. Ocal, S. S. Sastry, and K. Ramchandran, "Cross-entropy loss and low-rank features have responsibility for adversarial examples," 2019, *arXiv:1901.08360*.
[17] H. Zou and M. Yuan, "Composite quantile regression and the Oracle model selection theory," *Ann. Statist.*, vol. 36, no. 3, pp. 1108–1126, 2008.
[18] C. F. Mansky, "Maximum score estimation of the stochastic utility model of choice," *J. Econ.*, vol. 3, pp. 205–228, 1975.
[19] C. Mansky, "Semiparametric analysis of discrete response: Asymptotic properties of the maximum score estimator," *J. Econ.*, vol. 3, pp. 205–228, 1985.
[20] J. L. Horowitz, "A smoothed maximum score estimator for the binary response model," *Econometrica*, vol. 60, pp. 505–531, 1992.
[21] G. Kordas, "Smoothed binary regression quantiles," *J. Appl. Econ.*, vol. 21, pp. 387–407, 2006.
[22] D. F. Benoit and D. Van den Poel, "Binary quantile regression: A Bayesian approach based on the asymmetric laplace distribution," *J. Appl. Econometrics*, vol. 27, no. 7, pp. 1174–1188, 2012.

[23] M. H. Farrell, T. Liang, and S. Misra, "Deep neural networks for estimation and inference: Application to causal effects and other semiparametric estimands," *Econometrica*, vol. 89, no. 1, pp. 182–213, 2021.
[24] P. Anand, R. Rastogi, and S. Chandra, "A new asymmetric ϵ-insensitive pinball loss function based support vector quantile regression model," *Appl. Soft Comput.*, vol. 94, 2019, Art. no. 106473.
[25] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[26] J. Elson, J. J. Douceur, J. Howell, and J. Saul, "Asirra: A captcha that exploits interest-aligned manual image categorization," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 366–374.
[27] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 427–436.
[28] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6405–6416.
[29] J. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax Weiss, and B. Lakshminarayanan, "Simple and principled uncertainty estimation with deterministic deep learning via distance awareness," in *Adv. Neural Inf. Proc. Syst.*, Curran Associates, Inc., 2020, pp. 7498–7512.
[30] H. Jiang, B. Kim, M. Guan, and M. Gupta, "To trust or not to trust a classifier," *Adv. Neural Inf. Process. Syst.*, vol. 31, pp. 5546–5557, 2018.
[31] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics: Hum. Lang. Technol.*, Portland, OR, USA: Association for Computational Linguistics, 2011, pp. 142–150.
[32] D. S. Kermany, M. H. Goldbaum, W. Cai, C. C. S. Valentim, and K. Zhang, "Identifying medical diagnoses and treatable diseases by image-based deep learning," *Cell*, vol. 172, pp. 1122–1131.e9, 2018.
[33] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1919–1925.
[34] M. Neykov, P. Cizek, P. Filzmoser, and P. Neytchev, "The least trimmed quantile regression," *Comput. Statist. Data Anal.*, vol. 56, no. 6, pp. 1757–1770, 2012.
[35] P. Čížek and P. Čížek, "General trimmed estimation: Robust approach to nonlinear and limited dependent variable models," *Econometric Theory*, vol. 24, no. 6, pp. 1500–1529, 2008.
[36] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. Salakhutdinov, and K. Chaudhuri, "A closer look at accuracy vs. robustness," *Adv. Neural Inf. Proc. Syst.*, Curran Associates, Inc., 2020, pp. 8588–8601.
[37] C. Finlay, J. Calder, B. Abbasi, and A. Oberman, "Lipschitz regularized deep neural networks generalize and are adversarially robust," 2019, *arXiv:1808.09540*.
[38] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Mach. Intell.*, vol. 1, no. 5, pp. 206–215, May 2019.
[39] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.
[40] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2014, *arXiv:1312.6034*.
[41] M. Sundararajan, A. Taly, and Q. Yan, "Gradients of counterfactuals," 2016, *arXiv:1611.02639*.
[42] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nature Mach. Intell.*, vol. 2, no. 1, pp. 56–67, Jan. 2020, doi: 10.1038/s42256-019-0138-9.
[43] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, New York, NY, USA: Association for Computing Machinery, 2016, pp. 1135–1144.
[44] L. Armijo, "Minimization of functions having Lipschitz continuous first partial derivatives," *Pacific J. Math.*, vol. 16, no. 1, pp. 1–3, 1966.
[45] V. Palgianakos, M. Vrahatis, and G. Magoulas, "Nonmonotone methods for backpropagation training with adaptive learning rate," *Int. Joint Conf. Neural Netw.*, vol. 3, pp. 1762–1767, Feb. 1999.
[46] R. Yedida and S. Saha, "Lipschitzlr: Using theoretically computed adaptive learning rates for fast convergence," *Appl. Intell.*, vol. 51, no. 3, pp. 1460–1478, 2021.
[47] S. Saha, T. Prashanth, S. Aralihalli, S. Basarkod, T. S. B. Sudarshan, and S. S. Dhavala, "LALR: Theoretical and experimental validation of Lipschitz adaptive learning rate in regression and neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.
[48] S. Sridhar, S. Saha, A. Shaikh, R. Yedida, and S. Saha, "Parsimonious computing: A minority training regime for effective prediction in large microarray expression data sets," in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.
[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[50] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.
[51] P. Chaudhuri, "On a geometric notion of quantiles for multivariate data," *J. Amer. Stat. Assoc.*, vol. 91, pp. 862–872, 1996.
[52] M. Hallin, D. Paindaveine, and M. Šiman, "Multivariate quantiles and multiple-output regression quantiles: From l 1 optimization to halfspace depth," *Ann. Statist.*, vol. 38, no. 2, pp. 635–669, Apr. 2010.



Anuj Tambwekar received the B.Tech. degree in computer science and engineering from PES University, Bengaluru, India, in 2021.

He is currently with Microsoft, Hyderabad, India. His research interests include applied machine learning and deep learning under data constraints.

Anirudh Maiya received the B.Tech. degree in computer science and engineering from PES University, Bengaluru, India, in 2021.

He is currently with Commvault, Bengaluru, India. His research interests include interpreting remote sensing data through machine learning.

Soma S. Dhavala received the Ph.D. degree in statistics from Texas A&M University, College Station, TX, USA, in 2010.

He is the Founder of `mlsquare`, an open-source initiative to democratize AI. He is a Principal Researcher with the Wadhwani Institute for Artificial Intelligence, Mumbai, India, working on AI for social good in public health space. His research interests include developing theory and tools for holistic machine learning, and applying them in AI for social good.

Snehanshu Saha (Senior Member, IEEE) received the Ph.D. degree in mathematical sciences from the University of Texas at Arlington, Arlington, TX, USA, in 2008.

He is a Professor of Artificial Intelligence with BITS Pilani K K Birla Goa Campus, Zuarinagar, India. His research interests include the theory of optimization, learning theory, activation functions in deep neural networks, and AstroInformatics.

Dr. Saha is a Senior Member of ACM and a Fellow of IETE.