Exploring Uncertainty in Deep Learning for Construction of Prediction Intervals

Yuandu Lai, Yucheng Shi, Yahong Han, Member, IEEE, Yunfeng Shao, Meiyu Qi, Bingshuai Li

Abstract—Deep learning has achieved impressive performance on many tasks in recent years. However, it has been found that it is still not enough for deep neural networks to provide only point estimates. For high-risk tasks, we need to assess the reliability of the model predictions. This requires us to quantify the uncertainty of model prediction and construct prediction intervals. In this paper, We explore the uncertainty in deep learning to construct the prediction intervals. In general, We comprehensively consider two categories of uncertainties: aleatory uncertainty and epistemic uncertainty. We design a special loss function, which enables us to learn uncertainty without uncertainty label. We only need to supervise the learning of regression task. We learn the aleatory uncertainty implicitly from the loss function. And that epistemic uncertainty is accounted for in ensemble form. Our method correlates the construction of prediction intervals with the uncertainty estimation. Impressive results on some publicly available datasets show that the performance of our method is competitive with other state-of-the-art methods.

Index Terms—Prediction Intervals, Uncertainty Estimation, Deep learning, Neural Networks.

1 INTRODUCTION

With the rapid development of artificial intelligence, deep learning has attracted the interest of many researchers [1]. As a representative tool of deep learning, deep neural networks has achieved impressive performance in many tasks. At present, deep neural networks have important applications in computer vision [2], speech recognition [3], natural language processing [4], bioinformatics [5] and other fields. Although deep neural networks have achieved high accuracy for many tasks, they still perform poorly in quantifying the uncertainty of predictions and tend to be overconfident. For many real-world applications, it is not enough for a model to be accurate in its predictions. It must also be able to quantify the uncertainty of each prediction. This is especially importance in tasks where wrong prediction has a great negative impact, such as: machine learning auxiliary medical diagnosis [6], automatic driving [7], finance and energy system, etc. These high-risk applications require not only point prediction, but also the precise quantification of the uncertainty. So it can be said that in many application scenarios, uncertainty is very important, more important than precision. Moreover, uncertainty plays an important role in determining when to abandon the prediction of the model. Abandoning the inaccurate predictions of the model can handle exceptions and hand over high-risk decisions to humans [8]. As more and more autonomous systems based on deep learning are deployed in our real life that may cause physical or economic harm, we need to better understand when we can have confidence in the prediction of deep neural networks and when we should not be so sure.

The development of neural networks precise prediction intervals (PIs) [9] is a challenging work to explore the uncertainty of prediction, which has just aroused the interest of researchers. PIs directly conveys uncertainty, providing a lower bound and an upper bound for prediction [10], but traditional deep neural networks can only provide a point estimation. PIs is of great use in practical applications, and can help people make better decisions. However, although there are some researches on uncertainty estimation [11] [12] [13] [14] [15], there are few researches on PIs. In this paper, We will explore the uncertainty in deep learning to construct the prediction intervals.

Uncertainty comes from different sources in various forms, and most literature considers two sources of uncertainty: aleatory uncertainty and epistemic uncertainty [16] [17] [18] [19] [20] [21]. Aleatory uncertainty describes the irreducible inherent noise of the observed data. This is due to the complexity of the data itself, which can be sensor noise, label noise, class overlap, etc. Aleatory uncertainty is also caused by hidden variables that are not observed or errors in the measurement process, and this type of uncertainty cannot be reduced by collecting more data. Aleatory uncertainty is also known in some literature as data uncertainty, random uncertainty [22], stochastic uncertainty [23]. Epistemic uncertainty describes the model errors due to lack of experience in some regions of the feature space [24]. In some feature spaces, if the training samples do not cover them, the uncertainty will increase in those places. Therefore, the epistemic uncertainty is inversely proportional to the density of the training samples, which can be reduced by collecting more data in the low-density areas. In regions with high-density training samples, epistemic uncertainty will decrease, and aleatory uncertainty caused by data noise will play a major role. Epistemic uncertainty is also known as model uncertainty [19], because it is caused by the limitations of the model.

By analyzing the sources and categories of uncertainty, we can deal with and estimate each kind of uncertainty well in practice.
The precise prediction intervals of neural network is a difficult task to solve, and its development is closely related to uncertainty estimation. The prediction interval directly reflects the uncertainty. Many literatures have proposed that the loss function can be modified to implicit learning prediction intervals [9] [10] [12] [25] [26]. These methods have obvious disadvantages, such as loss function can not be optimized by gradient descent-based algorithm, which is a modern machine learning technology. Or the accuracy of PIs is low. Although there are many confidence interval construction methods with good performance in statistical literature, these methods can not be effectively combined with deep learning.

In this paper, we propose a prediction interval construction method from the perspective of statistics which can be used effectively only by slight modification of deep neural networks. We modify the last layer of the general deep neural network so that the network no longer outputs only a point estimation, but the upper bound and lower bound of the interval. And a loss function that can be used to train such a network is derived from the perspective of probability theory. Because most modern deep neural networks can use our loss function with only a few modifications, our method will have more practical use than other methods.

The contributions of this work are summarized as follows: i) In this paper, the construction of optimal prediction intervals is associated with uncertainty estimation. We design a novel loss function based on the theory of statistics. We think our work bridges the gap between the construction of prediction intervals and the uncertainty estimation; ii) Unlike bayesian deep learning, our loss function can be seamlessly integrated with current deep learning frameworks such as TensorFlow and PyTorch. Existing deep learning networks can use our loss function with only minor modifications. And it can be optimized efficiently using standard optimization techniques. We believe that our method promotes the development of non-bayesian method to construct prediction interval. iii) Experimental results show that this method can construct compact optimal prediction intervals. It can sensitively capture the epistemic uncertainty. In the low-density data distribution area, the uncertainty increases obviously. Experimental results on some publicly available regression datasets show the effectiveness of the method.

The rest of the paper is organized as follows. In the section 2, we summarize the related work of uncertainty estimation and prediction interval. And we introduce the method of constructing prediction interval based on exploring uncertainty in section 3. In the section 4, we organize some convincing experiments to show the effectiveness of our method. Finally, in the section 5, we make a brief summary of this paper.

2 RELATED WORK

Although it is impossible to eliminate the uncertainty completely, it is worthwhile to reasonably estimate and deal with the uncertainty to avoid the cost caused by the error of high-risk prediction. In this section, current research in this field will be presented.
certainty can be quantified by calculating the entropy or variance of the predictions of the model. However, this approach has obvious limitation on the network model structure, which must include the dropout layer.

Since dropout has been largely replaced by batch normalization \cite{22,23}. Teye et al. \cite{45}, demonstrate how to use the Monte Carlo batch normalization (MCBN) to make meaningful estimation of model uncertainty. Further, this discovery enables us to make appropriate estimation of model uncertainty for traditional network architectures without modifying the network or training process. Although these methods avoid the disadvantages of bayesian methods, they can only measure one uncertainty, which is not complete.

**Prediction Intervals.** Prediction intervals is a widely used tool to quantify the uncertainty of predictions. PIs directly conveys uncertainty. A representative work is proposed by Pearce et al. \cite{10}, which is based on the improvement of khosravi et al. \cite{45}. The author thinks that high quality PIs should be as narrow as possible and cover the data points of specified proportion. The author calls it the high quality criterion. According to this criterion, the author deduces a loss function, which is used to predict PIs without distribution assumption. In addition, a common method for prediction of interval range is quantile regression \cite{46}. It was first proposed by Koenker et al. \cite{47}. On the other hand, Rosenfeld et al. put forward the method of building PIs in the setting of batch learning, which is called IntPred \cite{48}, that is, building PIs of a group of test points at the same time. An obvious disadvantage of these methods is that the PIs they constructed have nothing to do with uncertainty or have little relevance.

**Applications of uncertainty.** Firstly, uncertainty plays an important role in deciding when to abandon the prediction of the model and let humans make decisions to avoid the loss caused by mistakes \cite{5}. Secondly, the study of uncertainty can promote the development of active learning \cite{49}, that is, deciding which data people should label to maximize the performance of the model \cite{50}. Thirdly, uncertainty estimation is very important when constructing the prediction intervals \cite{25}.

3 CONSTRUCTION OF OPTIMAL PREDICTION INTERVALS BASED ON UNCERTAINTY ESTIMATION

In this paper, we propose a method of constructing prediction intervals based on quantifying two kind of uncertainty in deep neural networks. We call our method uncertainty-based PI (UBPI).

3.1 Notation and Problem Setup

Let $x \in \mathbb{R}^d$ represent the d-dimensional features. For the regression problem, the label is assumed to be real-valued, that is $y \in \mathbb{R}$. And we assume that a training dataset $D$ consists of $N$ i.i.d. samples $D = \{(x_n, y_n)\}_{n=1}^N$.

In order to explain the uncertainty in the regression scenario conveniently, let’s consider the following formula:

$$ y = f(x) + \sigma_{\text{noise}}, \quad (1) $$

where statistically $f(\cdot)$ is some data generating function, and $\sigma_{\text{noise}}$ is the irreducible noise, also named the data noise. The reason it exists is because there are unobserved explanatory variables or due to the inherent randomness in nature. In general, the goal of regression is to produce an estimate $\hat{f}(x)$ that can be used to make a prediction of point estimation. The two terms in the eq.(1) correspond to the sources of epistemic uncertainty and aleatory uncertainty respectively. Therefore, when we estimate the uncertainty of $y$, assuming that the two uncertainties are independent, then the total uncertainty is:

$$ \sigma^2 = \sigma_{ep}^2 + \sigma_{al}^2, \quad (2) $$

where $\sigma_{ep}^2$ denotes the epistemic uncertainty, $\sigma_{al}^2$ denotes the aleatory uncertainty.

3.2 Construct the Optimal Prediction Interval

Now we introduce how to construct optimal prediction intervals without the ‘uncertainty labels’. We assume that the size of a mini-batch is $n$, and with $x_i$ denoting the $i$th input corresponding to $y_i$. The philosophy of uncertainty quantification is to predict a prediction interval (PI) $[\hat{y}_L, \hat{y}_U]$ to bound $y_i$ to satisfy:

$$ Pr[\hat{y}_L \leq y_i \leq \hat{y}_U] \geq P_c, \quad (3) $$

where $P_c$ is the predefined confidence level.

3.2.1 Hybrid Loss Function for Prediction Intervals

In order to equip UBPI with the capability of quantifying uncertainty and providing optimal prediction intervals, the training loss function $L$ has two terms:

$$ L = L_{UE} + \lambda L_{PI}, \quad (4) $$

where hyper-parameter $\lambda$ balances the importance of two terms.

We now derive $L_{UE}$ from estimating the aleatory uncertainty in the regression task. For regression task, we often define model likelihood as a Gaussian with mean given by the model output and an observation noise scalar $\sigma$: $\mathcal{N}(\hat{f}(x), \sigma^2)$. We fix this Gaussian likelihood to model aleatior uncertainty in regression tasks. For a regression problem, its likelihood is the product of the corresponding value of the probability density function (PDF). Combined with the PDF of Gaussian distribution, the negative log likelihood (NLL) \cite{51} of regression can be derived as follows:

$$ \text{NLL} = - \sum_{i=1}^{n} \log \left[ \frac{1}{\sqrt{2\pi} \sigma} e^{-\frac{(y_i - \hat{y}_i)^2}{2\sigma^2}} \right] \quad (5) $$

$$ = n \left( \frac{1}{2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \log(\sigma^2) + \log(2\pi) \right), \quad (6) $$

with $\sigma^2$ capturing how much uncertainty we have in the outputs.

Here, we use the width of the interval as an estimate of the uncertainty. In practice, we simplify eq. (6), and $L_{UE}$ is designed in the following form:

$$ L_{UE} = \frac{n}{2} \left[ \frac{MSE}{MPIW} + \log MPIW \right], \quad (7) $$
where MSE is Mean Squared Error of a mini-batch, and we use the midpoint of the interval $[\hat{y}_{L,i}, \hat{y}_{U,i}]$ as the point estimation:

$$\hat{y}_i = \frac{\hat{y}_{L,i} + \hat{y}_{U,i}}{2},$$  \hspace{1cm} (8)

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2.$$  \hspace{1cm} (9)

And $MPIW$ is Mean Prediction Interval Width \cite{52}, which is defined as:

$$MPIW = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_{U,i} - \hat{y}_{L,i}).$$  \hspace{1cm} (10)

We have replaced the $\sigma^2$ in eq. (6) with $MPIW$. We think the interval width is a good indicator of the uncertainty. That is to say, the wider the interval, the greater the uncertainty. On the contrary, the narrower the interval, the smaller the uncertainty. The $\mathcal{L}_{UE}$ loss consists of two components: a residual regression term and an uncertainty regularization term. We do not need uncertainty labels to learn uncertainty. Instead, we only need to supervise the learning of the regression task. We learn the aleatory uncertainty implicitly from the loss function. The second regularization term prevents the network from generating too wide prediction intervals, which is meaningless.

To learn prediction interval, $\mathcal{L}_{PI}$ is inspired from the study \cite{10} \cite{45} and constructed as below:

$$\mathcal{L}_{PI} = \max(0, P_c - PICP)^2,$$  \hspace{1cm} (11)

**Algorithm 1 Construction of hybrid loss function**

**Input:** Predictions of lower bound $\hat{y}_L$ and upper bound $\hat{y}_U$, ground truth label $y$, predefined confidence level $P_c$, hyper-parameter factor for softing $s$, hyper-parameter $\lambda$ balances the importance of two terms, $n$ denotes the size of a mini-batch, $\sigma$ denotes sigmoid activation function, $\odot$ denotes the element-wise product.

**Output:** Loss

$$\hat{y} = (\hat{y}_L + \hat{y}_U)/2$$

$$MSE = \text{reduce}_\text{mean}(y - \hat{y})^2$$

$$MPIW = \text{reduce}_\text{mean}(\hat{y}_U - \hat{y}_L)$$

$$c = \sigma(s \cdot (\hat{y}_U - y)) \odot \sigma(s \cdot (y - \hat{y}_L))$$

$$PICP = \text{reduce}_\text{mean}(c)$$

$$\mathcal{L}_{UE} = \frac{n}{\sqrt{MPIW}} \log MPIW$$

$$\mathcal{L}_{PI} = \max(0, P_c - PICP)^2$$

$$\mathcal{L} = \mathcal{L}_{UE} + \lambda \mathcal{L}_{PI}$$

where $PICP$ is PI coverage probability. $PICP$ is the spontaneous measure related to the quality of the constructed PIs, and is denoted as:

$$PICP = \frac{1}{n} \sum_{i=1}^{n} c_i,$$  \hspace{1cm} (12)

where $c_i = 1$ if $y_i \in [\hat{y}_{L,i}, \hat{y}_{U,i}]$, otherwise $c_i = 0$. The advanced intuition of eq. (11) is that if the $PICP$ is lower than the predefined confidence level $P_c$ during the training, penalty loss will be imposed.

Due to the existence of operation $\max(\cdot)$ in eq. (11),
Fig. 2: Results on a toy regression task. The red dots are observed noisy training data points and the gray shade correspond to the 95% prediction interval. This figure shows the compact PIs.

applying the boolean $c_i$ in eq.(12) can cause the $L_{PI}$ loss not to be differentiable for back-propagation algorithm. So, following the previous work [10] [53] [54] [55], we also replace $c_i$ with a softened version as follows:

$$c_i = \sigma(s \cdot (\hat{y}_{U_i} - y_i)) \cdot \sigma(s \cdot (y_i - \hat{y}_{L_i})),$$

where $\sigma$ is the sigmoid activation function and $s$ is a hyper-parameter factor for softing. Algorithm 1. shows the process of constructing our hybrid loss function.

3.2.2 Comparison of Various Loss Functions for Uncertainty Prediction

In this section, we compare our loss function with two other loss functions that have been proposed to predict PIs. The key idea of these methods is to construct the PIs by minimizing the loss function based on eq.(3) and eq.(10). These loss functions typically contain two terms: $MPIW$ and $PICP$. This phenomenon shows that the recent work is to find the basic balance between coverage probability and PI width [45]. For example, the loss function proposed by Khosravi et al. [45] is:

$$Loss_{LUBE} = \frac{MPIW}{r} \left(1 + \exp(\lambda max(0, (1 - \alpha) - PICP))\right),$$

where $r$ is the numerical range of the target variable $y$, and $\lambda$ controls the importance between $PICP$ and $MPIW$. And another loss function proposed by hu et al. [56] is called $MBPEP$ and its form is as follows:

$$Loss_{MBPEP} = MPIW + \lambda \cdot ReLU((1 - \alpha) - PICP),$$

where $(1 - \alpha)$ is the predefined confidence level and $ReLU(\cdot)$ denotes the Rectified Linear Unit (ReLU).

The proposed loss function differs from the existing work of PIs estimation in many aspects. Firstly, where our loss function differs from them is in the use of the term $MPIW$. $Loss_{MBPEP}$ adds $MPIW$ as a single term to the loss function, while we integrate $MPIW$ into the form of NLL as an estimate of uncertainty. That is, our loss function bridges the gap between the width of the PIs and the uncertainty. And intuitively, $MBPEP$ loss function is not strongly related to uncertainty estimation. Secondly, ours loss function is differentiable and can be effectively minimized using standard optimization techniques like Stochastic Gradient Descent (SGD). However, the $Loss_{LUBE}$ loss function is not differentiable, which makes it difficult to optimize.

3.2.3 Ensemble Methodology

Ensembles of neural networks have long been used to estimate predictive uncertainty [9] [57]. A group of neural networks are trained with different parameter initializations [12] or with different sub-sampling training dataset. The variance or entropy of the predictions is interpreted as its epistemic uncertainty. Related algorithms include bootstrapping [58], and bagging from statistics [59], etc. Following the work [10], we have adopted a simple but effective ensemble principle: each individual model is a UBPI model initialized with different parameters. We assume that the size of the ensemble of neural networks is $m$ as shown in Figure 1. The ensemble PIs is the averaged PIs of all individual models,

$$\mu_{\hat{y}_{L_i}} = \frac{1}{m} \sum_{j=1}^{m} \hat{y}_{L_{ij}}, \quad \text{(16)}$$

$$\mu_{\hat{y}_{U_i}} = \frac{1}{m} \sum_{j=1}^{m} \hat{y}_{U_{ij}}. \quad \text{(17)}$$

Table 1: The UCI datasets and shows summary statistics. In the table below, $S$ represents the size of the dataset and $D$ represents the dimension of the data.

| Dataset                  | S  | D  |
|--------------------------|----|----|
| Boston Housing           | 506| 13 |
| Wine Quality             | 1,599| 11 |
| Forest Fires             | 517| 13 |
| Concrete Compression Strength | 1,030| 9  |
| Energy Efficiency        | 768| 8  |
| Naval Propulsion         | 11,934| 16 |
| Combined Cycle Power Plant | 9,568| 4  |
| Protein Structure        | 45,730| 9  |
| Online News Popularity   | 39,797| 61 |

Fig. 3: Results on a toy regression task. The red dots are observed noisy training data points and the gray shade correspond to the 95% prediction interval. This figure shows capturing epistemic uncertainty.

TABLE 1: The UCI datasets and shows summary statistics. In the table below, $S$ represents the size of the dataset and $D$ represents the dimension of the data.
### 4 EXPERIMENT

In order to demonstrate the effectiveness of our method in capturing epistemic uncertainty, we generated some one-dimensional toy data for experiments. The function used for data point generation is:

\[ y = 1.5 \sin(x) + \epsilon, \quad (23) \]

where \( \epsilon \sim N(0, x^2) \).

The results is shown in Figure 3. From the figure, we can see that our method is very sensitive to the density of data distribution. The constructed PIs is in line with our expectation, that is, where there is less training data, the epistemic uncertainty increases, which is reflected in the increase of interval width. As we can see, where the data points are dense, the PIs is very narrow.

#### 4.3 Regression on Real World Datasets

In our next experiment, we compare our method with several baseline methods for constructing PIs. The details of the experiment are described below.

##### 4.3.1 Experimental Setup and Baseline Methods

The neural network structure we use is one hidden layer of 50 neurons with ReLU activation. The exceptions to the experimental setup are the two largest datasets: Online News Popularity and Protein Structure. For these two datasets, the network structure is one hidden layer of 100 neurons.

Several baseline methods are used for comparison, including Quantile Regression (QR) [46], Gradient Boosting Decision Tree with Quantile Loss (QLEGBDT) implemented in the Scikit-learn package [60] and quality-based PI construction method IntPred [48]. For all datasets, the ensemble size of neural networks is five.
TABLE 3: The performance of different values of hyperparameter $\lambda$ on Protein Structure and Online News Popularity datasets

| $\lambda$ | Protein Structure | Online News Popularity |
|-----------|-------------------|------------------------|
|           | PICP MPIW         | PICP MPIW              |
| $\lambda = 5$ | 0.947 2.212     | 0.925 0.668             |
| $\lambda = 10$ | 0.950 2.271     | 0.935 0.761             |
| $\lambda = 20$ | 0.953 2.377     | 0.948 1.050             |
| $\lambda = 30$ | 0.957 2.510     | 0.951 1.155             |
| $\lambda = 40$ | 0.965 2.708     | 0.953 1.213             |
| $\lambda = 50$ | 0.969 2.907     | 0.958 1.462             |
| $\lambda = 60$ | 0.973 3.067     | 0.960 1.583             |

4.3.2 Benchmark Datasets and Evaluation Metrics

To measure the quality of PIs, we conduct experiments on nine UCI regression datasets as shown in Table 1. The datasets are randomly divided into training set and test set, accounting for 90% and 10% respectively.

In this experiment, we used two quality assessment metrics: PICP and MPIW. The results are shown in Table 2. We repeated the experiments 20 times except for OnlineNewsPopularity and ProteinStructure. For these two datasets, the experiment is repeated five times. And the results are presented as averages.

4.3.3 Performance Comparison and Discussion

As we can see from Table 2, our method achieves the best performance on most datasets. And overall, the average performance is better than all other methods. Average PICP reached the predefined confidence level $P_c$, which is slightly better than other methods. At the same time, our method can obtain narrower PIs. Compared with QR and QR GBDT, our method can significantly reduce the PI width by 30.1% and 24.1% respectively. And compared with IntPred, the width was narrowed by 20.1%.

At the same time, as shown in Figure 4, we construct and visualize the prediction intervals of some test samples. The effectiveness of our method can be seen intuitively from the diagram.

4.3.4 The Effect of Different Values of Hyper-parameter $\lambda$

In Table 3, we show the effect of hyper-parameter $\lambda$ with different values. We can see $\lambda$ can controls the trade-off between MPIW and PICP. When we want the high coverage probability of PIs, the average PI width will increase. On the contrary, when we want the average PI width to be narrower, the coverage probability of PIs will decrease. This phenomenon is in line with our logic. In practical application, we can get the desired PIs coverage probability by adjusting $\lambda$.

5 CONCLUSION AND FUTURE WORKS

In this paper, the problem of estimating uncertainty associate with constructing optimal prediction intervals. We first give a comprehensive introduction to two kinds of uncertainties and the construction of prediction interval. And we give an unified literature review on uncertainty estimation. We estimate the uncertainty in deep learning from the perspective of prediction intervals. In general, we design a special loss function based on the theory of uncertainty estimation. We do not need uncertainty labels to learn uncertainty. Instead, we only need to supervise the learning of the regression task. We learn the aleatory uncertainty implicitly from the loss function. The experiments show that our method can construct compact optimal prediction intervals. And it can capture the epistemic uncertainty sensitively. In the area of sparse data distribution, the uncertainty increases obviously. In addition, the experimental results on some public regression datasets show the effectiveness of our method.

For future works, we believe that more innovative neural network ensembles method can be designed. In addition, there is a lack of quantitative evaluation metrics of model uncertainty currently, which should be a research direction to be paid attention to.
[50] B. Settles, “Active learning literature survey,” University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[51] J. Quinonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf, “Evaluating predictive uncertainty challenge,” in Machine Learning Challenges Workshop. Springer, 2005, pp. 1–27.

[52] A. Khosravi, S. Nahavandi, and D. Creighton, “Construction of optimal prediction intervals for load forecasting problems,” IEEE Transactions on Power Systems, vol. 25, no. 3, pp. 1496–1503, 2010.

[53] B. Wang, T. Li, Z. Yan, G. Zhang, and J. Lu, “Deeppipe: A distribution-free uncertainty quantification approach for time series forecasting,” Neurocomputing, 2020.

[54] L. Yan, R. H. Dodier, M. Mozer, and R. H. Wolniewicz, “Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic,” in Proceedings of the 20th International Conference on Machine Learning (ICML-03), 2003, pp. 848–855.

[55] L. Yan, D. Verbel, and O. Saidi, “Predicting prostate cancer recurrence via maximizing the concordance index,” in Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004, pp. 479–483.

[56] R. Hu, Q. Huang, S. Chang, H. Wang, and J. He, “The mbpep: a deep ensemble pruning algorithm providing high quality uncertainty prediction,” Applied Intelligence, vol. 49, no. 8, pp. 2942–2955, 2019.

[57] R. Tibshirani, “A comparison of some error estimates for neural network models,” Neural Computation, vol. 8, no. 1, pp. 152–163, 1996.

[58] R. W. Johnson, “An introduction to the bootstrap,” Teaching Statistics, vol. 23, no. 2, pp. 49–54, 2001.

[59] L. Breiman, “Bagging predictors,” Machine learning, vol. 24, no. 2, pp. 123–140, 1996.

[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., “Scikit-learn: Machine learning in python,” Journal of machine learning research, vol. 12, no. Oct, pp. 2825–2830, 2011.