Variational Bayes Ensemble Learning Neural Networks With Compressed Feature Space

Zihuan Liu, Shrijita Bhattacharya, and Tapabrata Maiti

Abstract*— We consider the problem of nonparametric classification from a high-dimensional input vector (small n large p problem). To handle the high-dimensional feature space, we propose a random projection (RP) of the feature space followed by training of a neural network (NN) on the compressed feature space. Unlike regularization techniques (lasso, ridge, etc.), which train on the full data, NNs based on compressed feature space have significantly lower computation complexity and memory storage requirements. Nonetheless, a random compression-based method is often sensitive to the choice of compression. To address this issue, we adopt a Bayesian model averaging (BMA) approach and leverage the posterior model weights to determine: 1) uncertainty under each compression and 2) intrinsic dimensionality of the feature space (the effective dimension of feature space useful for prediction). The final prediction is improved by averaging models with projected dimensions close to the intrinsic dimensionality. Furthermore, we propose a variational approach to the afore-mentioned BMA to allow for simultaneous estimation of both model weights and model-specific parameters. Since the proposed variational solution is parallelizable across compressions, it preserves the computational gain of frequentist ensemble techniques while providing the full uncertainty quantification of a Bayesian approach. We establish the asymptotic consistency of the proposed algorithm under the suitable characterization of the RPs and the prior parameters. Finally, we provide extensive numerical examples for empirical validation of the proposed method.

***Index Terms*— Intrinsic dimensionality, model averaging, random compression, variational inference (VI).**

I. Introduction

Bayesian neural networks (BNNs) are widely used, especially in complex and high-dimensional data analysis, including image recognition, biomedical diagnosis, and others. One of the major limitations of neural networks (NNs) and deep networks (DNNs) is the need for a large training dataset due to a large number of inherent parameters [1], [2]. Thus, high-dimensional NNs are widely applied with regularization, dropout techniques, or early stopping to prevent overfitting [3], [4]. Regularization techniques which allow for dimension reduction in the feature space include lasso [5], ridge [6], elastic net [7], sparse group lasso [8], Bayesian lasso [9], and horseshoe prior [10]. Although regularization/dropout can set the weights to zero or small, they still train on the full data, which severely increases the cost of both computation and memory storage. Quoting [11], in the context of linear regression, the compression of the feature space in contrast to regularization approaches, dramatically reduces storage and computational bottlenecks and performs exceedingly well when the predictors can be projected to a low-dimensional linear subspace ([11, Fig. 3] shows the gain in computational time from lasso to Bayesian compressed regression as a function of log number of predictors). This motivates us to adopt random projection (RP) techniques for high-dimensional nonparametric classification. Another popular dimension reduction technique, principal component analysis [12], is an unsupervised learning approach and cannot be directly adapted to learn the effective dimension of the feature space in the context of the prediction task. We thereby propose to compress the feature space to a smaller dimension with an RP matrix, followed by training with a BNN. The use of RP in high-dimensional statistics is motivated by Johnson–Lindenstrauss Lemma [13] by which for $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, $\epsilon \in (0, 1)$, $d > 8 \log n/\epsilon^2$, there exists a linear map $f : \mathbb{R}^p \to \mathbb{R}^d$ such that $(1-\epsilon)||x_i - x_j||_2^2 \leq ||f(\mathbf{x}_i) - f(\mathbf{x}_j)||_2^2 \leq (1+\epsilon)||x_i - x_j||_2^2$ for $i, j = 1, \ldots, n$. The properties of RPs and their applications to statistical problems are further explored in [14], [15].

To reduce sensitivity to the choice of RP, existing literature aggregates information from varying RPs by ensembling multiple estimators with bagging (see [16]–[18]). There are three drawbacks to this frequentist approach of ensembling: 1) it does not provide the uncertainty quantification for each RP let alone for the ensembled prediction; 2) it assigns equal weights to all models, thereby one may aggregate information from poor models, and 3) exact determination of the optimal projected dimension relies on a cross-validation approach and is therefore computationally intensive. To circumvent these drawbacks, we innovatively adopt a Bayesian model averaging (BMA) [19] approach for combining information across multiple RP-based BNNs. Guhaniyogi and Dunson [11], [20] applied BMA in the context of RP-based linear regression and Gaussian process regression, respectively. However, they used the computationally intensive Markov Chain Monte Carlo (MCMC). In the context of NNs, there are two main challenges of implementing BMA: 1) due to the convoluted structure of the NN likelihood, a closed-form expression of posterior distribution under each model does not exist and 2) the posterior distribution of model weights is intractable. Thus, the use of MCMC is next to impossible. Furthermore, the computation cost associated with MCMC is humongous since each posterior model weight depends on the remaining models' posterior model weights.

To address the challenges of MCMC implementation, we provide a variational inference (VI) [21], [22] based approximate solution to BMA for combining BNNs with multiple instances of compressed feature space. First, the proposed VI solution can be parallelized across all RPs and thus preserves the uncertainty quantification of BMA while ensuring the scalability of bagging. Second, for a fixed choice of the RP, the proposed VI solution is equivalent to the variational Bayes (VB) solution for BNNs on the corresponding projected feature space. This reduces the instability and increased cost of optimization associated with VI implementation of BNNs [23] based on the entire feature space. Furthermore, for a given RP, we establish posterior contraction rates of variational posterior for classification. The proof underscores the conditions on the RP to ensure the universal approximation property of compressed feature space-based VBNN. We also detail the characterization of the prior,

Manuscript received August 6, 2021; revised January 31, 2022; accepted April 26, 2022. This work was supported in part by the National Science Foundation (NSF) under Grant DMS-1952856 and Grant DMS-1924724. *(Corresponding author: Shrijita Bhattacharya.)*

Zihuan Liu is with the Collaborative Center for Statistics in Science, Yale University School of Public Health, New Haven, CT 06520 USA.

Shrijita Bhattacharya and Tapabrata Maiti are with the Department of Statistics and Probability, Michigan State University, East Lansing, MI 48824 USA (e-mail: bhatta61@msu.edu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2022.3172276.

Digital Object Identifier 10.1109/TNNLS.2022.3172276

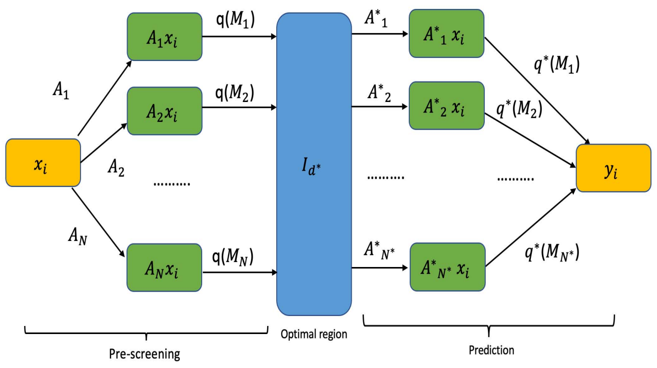

Fig. 1. Overview of the RPVBNN.

variational family, and the RP matrix to guarantee consistency of VBNN under the compressed feature space. To the best of our knowledge, no prior work provides theoretical guarantees and computation algorithms of VBNNs on compressed feature space.

Another main advantage of the proposed VI-based BMA is that it gives the posterior model weights under each RP of the feature space which further induces a probability distribution on projected dimensions of the feature space. The mode of this probability distribution centers around the intrinsic dimension (*the optimal dimension of the feature space useful for prediction*). VI-based BMA is then applied to a pool of RP matrices with projected dimensions around the intrinsic dimension to improve prediction. Thus, we provide a scalable Bayesian solution to learn the intrinsic dimension of the feature space using BNNs on RPs of the inputs together with full uncertainty quantification arising out of averaging across multiple RPs.

A. Related Work

Early works on BNNs are comprehensively discussed in [24]–[26]. Recent developments with higher efficient BNNs can be found in [27]–[30] and the references therein. For high-dimensional BNN, penalization and sparse network-based approach have been studied in [1], [2], [31]–[33], and so on. The idea of using RPs to overcome the curse of dimensionality was popularized by [13] and is now widely used in many statistical problems [16], [34]–[38]. To ensemble information across projections, [17] and [39] use a bagging approach for the classification task. On the other hand, [11] and [20] use BMA in the context of linear regression and Gaussian processes.

Several works discuss VI [22] to overcome the drawbacks of a full MCMC implementation. The majority of black-box VI methods for BNNs are based on pathwise gradient estimator [40]–[45] or score-function estimator [46], [47]. Theoretical properties of variational posterior in the context of individual models have been studied in [33], [48]–[51]. The works of Kejzlar *et al.* [52] and Latouche and Robin [53] explore VI for BMA in the context of generalized linear models and graph-on functions. However, VI for BMA for BNNs with compressed feature space remains unexplored.

B. Our Contributions

First, to handle a high-dimensional feature space for classification, we introduce the RP idea to reduce the dimension of the input to an NN model. Second, we apply the BMA to reduce sensitivity to each compression, followed by the development of its VB solution to allow for parallelization across RPs without compromising the uncertainty quantification of BMA (see also Fig. 1). As a by-product, we leverage the posterior model weights to learn the intrinsic dimension of the feature space and improve classification accuracy by using RPs with projected dimensions around the intrinsic dimension. Third, we validate our method from a statistical perspective: 1) develop the theoretical foundation, that is, the posterior contraction of the variational posterior for BNNs under a compressed feature space and 2) provide numerical results to enunciate that our proposed approach learns well the intrinsic dimension in simulated examples and performs competitively on real datasets.

II. BNN With Compressed Feature Space

A. BNN Model

For a binary random variable Y, representing the class levels 0 or 1, and a feature vector $X \in \mathbb{R}^p$ with some marginal distribution P_X, consider the following classification problem:

$$P(Y=1|X=\boldsymbol{x}) = \varphi(\eta_0(\boldsymbol{x})) = 1 - P(Y=0|X=\boldsymbol{x}) \quad (1)$$

where $\eta_0(\cdot) : \mathbb{R}^p \to \mathbb{R}$ is some continuous function and $\varphi(.) = e^{(.)}/(1+e^{(.)})$ is the sigmoid function. Following [17] and [54], the test error of a classifier C, $R(C) = \int_{\mathbb{R}^p \times \{0,1\}} I_{\{C(X)\neq Y\}} dP_{X,Y}$ (joint density $P_{X,Y}$ is a product of (1) and P_X) is minimized by the Bayes classifier is $C^{\text{Bayes}}(\boldsymbol{x}) = \boldsymbol{I}[\eta_0(\boldsymbol{x}) \geq 0]$. Since $\eta_0(\boldsymbol{x})$ is unknown, we instead use single-layer NN with k nodes

$$\eta_{\boldsymbol{\theta}}(\boldsymbol{x}) = \beta_0 + \sum_{j=1}^{k_n} \beta_j \psi(\gamma_{j0} + \boldsymbol{\gamma}_j^T \boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^\top \psi(\boldsymbol{\gamma}_0 + \boldsymbol{\Gamma}\boldsymbol{x}) \quad (2)$$

where ψ is an activation function, $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_k]$, $\boldsymbol{\gamma}_0 = [\gamma_{10}, \ldots, \gamma_{k0}]$, $\boldsymbol{\Gamma} = [\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_k]$, and $\boldsymbol{\theta} = (\beta_0, \boldsymbol{\beta}, \boldsymbol{\gamma}_0, \text{vec}(\boldsymbol{\Gamma}))$ is the set of all the parameters. Note that $\boldsymbol{\theta}$ is a $K \times 1$ vector where $K = 1 + k + k(p+1)$. Both k and $p(\gg n)$ grow as a function of n. We then use the following model for the problem in (1):

$$P(Y=1|X=\boldsymbol{x}) \approx \varphi(\eta_{\boldsymbol{\theta}}(\boldsymbol{x})) \approx 1 - P(Y=0|X=\boldsymbol{x}). \quad (3)$$

B. Compression in the Feature Space With RPs

There exists several choices for compressing the feature space $\boldsymbol{X}$ using RP matrices as in [13]–[15], [17], [55]. For a given compression matrix A, our model in (2) becomes

$$\eta_{\boldsymbol{\theta}}(A\boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^\top \psi(\boldsymbol{\gamma}_0 + \boldsymbol{\Gamma}(A\boldsymbol{x})) \quad (4)$$

where A is a $d_A \times p$ projection matrix, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}_0$ are $k \times 1$ vectors, and $\boldsymbol{\Gamma}^\top = [\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_k]$ is $d_A \times k$ matrix. In the projected space, the number of parameters reduces from $K = kp + 2k + 1$ to $K_A = kd_A + 2k + 1$ which then gives the following model:

$$P(Y=1|X=\boldsymbol{x}) \approx \varphi(\eta_{\boldsymbol{\theta}}(A\boldsymbol{x})) \approx 1 - P(Y=0|X=\boldsymbol{x}). \quad (5)$$

Experiments with different RP matrices suggested the use of one in [11]. In this method, we draw elements A_{ij} independently as $A_{ij} = -1/(a_*)^{1/2}$, 0, $1/(a_*)^{1/2}$ with probabilities a_*^2, 0, $(1-a_*)^2$, respectively. The rows of A are then normalized by Gram–Schmidt orthogonalization. Note that $a_* \in (0.1, 1)$ provides a handle on the sparsity of the RP matrix. The algorithm can be generalized to any arbitrary class of RP matrices.

C. Prior Choice

For the NN, $\eta_{\boldsymbol{\theta}_A}(A\boldsymbol{x})$ with projected input $A\boldsymbol{x}$, assume $p(\boldsymbol{\theta}_A|M_A) = \text{MVN}(\boldsymbol{\mu}_A, \text{diag}(\boldsymbol{\sigma}_A))$, that is, an independent Gaussian prior on $\boldsymbol{\theta}_A$. Here $\boldsymbol{\theta}_A$ is a $K_A \times 1$ vector of parameters for the model (4). For this prior and the likelihood in (5), the posterior based on compressed dataset $(y_i, A\boldsymbol{x}_i)_{i=1}^n$ is

$$\pi(\boldsymbol{\theta}_A|M_A) = \frac{L(\boldsymbol{\theta}_A|M_A)p(\boldsymbol{\theta}_A|M_A)}{\int L(\boldsymbol{\theta}_A|M_A)p(\boldsymbol{\theta}_A|M_A)d\boldsymbol{\theta}_A} \quad (6)$$

where M_A is the model induced by random matrix A with corresponding likelihood $L(\boldsymbol{\theta}_A|M_A) = \prod_{i=1}^{n} \exp(y_i \eta_{\boldsymbol{\theta}_A}(A\boldsymbol{x}_i) - \log(1 + \exp(\eta_{\boldsymbol{\theta}_A}(A\boldsymbol{x}_i))))$. The denominator in (6) is free of $\boldsymbol{\theta}_A$.

III. VB Model Averaging for Pooling RPs

A. Bayesian Model Averaging

Ensemble methods are most widely used in machine learning literature to pool across varying classifiers [56]. Here, we address the same problem from a Bayesian perspective. Let $\mathcal{A}$ denote a subspace of the space of all RP matrices. We assume each RP matrix induces a separate model M_A, $A \in \mathcal{A}$ on the data $\mathcal{D} = (y_i, \boldsymbol{x}_i)_{i=1}^n$. Thus, the predictive distribution of a new observation y_{n+1} given $\boldsymbol{x}_{n+1}$, denoted by $p(y_{n+1}|\boldsymbol{x}_{n+1}, \mathcal{D})$, is

$$\int p(y_{n+1}|\boldsymbol{x}_{n+1}, M_A, \boldsymbol{\theta}_A, \mathcal{D})\pi(M_A, \boldsymbol{\theta}_A|\mathcal{D})d\mu(M_A, \boldsymbol{\theta}_A) \qquad (7)$$

where μ is the product measure of counting and Lebesegue measure. In (7), the most difficult quantity to compute is $\pi(M_A, \boldsymbol{\theta}_A|\mathcal{D})$. In [11], explicit forms can be obtained for linear regression model which is impossible for a convoluted NN likelihood. In Section III-B, we circumvent this by VI.

B. ELBO Derivation

Let $\pi(M_A, \boldsymbol{\theta}_A|\mathcal{D})$ be the joint density of parameter and model conditional on data. We posit a variational distribution $q(M_A, \boldsymbol{\theta}_A)$, where $q(\boldsymbol{\theta}_A|M_A) \sim \text{MVN}(\boldsymbol{m}_A, \text{diag}(\boldsymbol{s}_A))$ and $q(M_A)$ are the model weights. Thus, our variational family is $\mathcal{Q}_n = \{q(M_A, \boldsymbol{\theta}_A) = q(M_A)q(\boldsymbol{\theta}_A|M_A)\}$. Let q^* be minimizer of Kullback–Leibler distance (d_{KL}) between $\pi(.|\mathcal{D})$ and $\mathcal{Q}_n$

$$\begin{aligned} d_{\text{KL}}(q, \pi(.|\mathcal{D})) &= E_Q(\log \pi(\boldsymbol{\theta}_A, M_A|\mathcal{D}) - \log q(\boldsymbol{\theta}_A, M_A)) \\ &= -\log \pi(\mathcal{D}) + \text{ELBO} \end{aligned}$$

where ELBO $= E_Q(\log \pi(\boldsymbol{\theta}_A, M_A, \mathcal{D}) - \log q(\theta_A, M_A))$. Since $-\log \pi(\mathcal{D})$ is independent of $\boldsymbol{\theta}_A$ and M_A, therefore $q^*(M_A, \boldsymbol{\theta}_A)$ is minimizer of the ELBO. Furthermore,

$$\begin{aligned} & E_Q(\log \pi(\boldsymbol{\theta}_A, M_A, \mathcal{D}) - \log q(\boldsymbol{\theta}_A, M_A)) \\ & \quad \sum_{A\in\mathcal{A}} q(M_A)E_{Q(\boldsymbol{\theta}_A|M_A)}(\log \pi(\mathcal{D}|M_A, \boldsymbol{\theta}_A) + \log \pi(\boldsymbol{\theta}_A|M_A) \\ & \quad + \log \pi(M_A) - \log q(\boldsymbol{\theta}_A|M_A) - \log q(M_A)) \\ & = \sum_{A\in\mathcal{A}} q(M_A)E_{Q(\boldsymbol{\theta}_A|M_A)}(\log L(\boldsymbol{\theta}_A|M_A) + \log p(\boldsymbol{\theta}_A|M_A) \\ & \quad + \log \pi(M_A) - \log q(\boldsymbol{\theta}_A|M_A) - \log q(M_A)) \\ & = \sum_{A\in\mathcal{A}} q(M_A)(\mathcal{L}(.|M_A) + \log \pi(M_A) - \log q(M_A)) \end{aligned}$$

$\mathcal{L}(.|M_A) = E_{Q(\boldsymbol{\theta}_A|M_A)}(\log L(\boldsymbol{\theta}_A|M_A) + \log p(\boldsymbol{\theta}_A|M_A) - \log q(\boldsymbol{\theta}_A|M_A))$ is the ELBO under model M_A. The derivative of ELBO with respect to variational parameters $\boldsymbol{m}_A, \boldsymbol{s}_A$[1] is

$$\nabla_{\boldsymbol{m}_A, \boldsymbol{s}_A} \text{ELBO} = q(M_A)\nabla_{\boldsymbol{m}_A, \boldsymbol{s}_A}\mathcal{L}(.|M_A).$$

Since $q(M_A)$ is just constant, thus gradient update for model-specific variational parameters is same as the gradient update of each individual model. Setting $\nabla_{q(M_A)}\text{ELBO} = 0$, we get

$$\begin{aligned} & \log \pi(M_A) - \log q(M_A) + \mathcal{L}(.|M_A) - 1 = 0 \\ & \qquad \implies q(M_A) \propto \exp(\log \pi(M_A) + \mathcal{L}(.|M_A)). \end{aligned}$$

The optimal model weights are $q^*(M_A) = \exp(\log \pi(M_A) + \mathcal{L}^*(.|M_A))/\sum_A \exp(\log \pi(M_A) + \mathcal{L}^*(.|M_A))$, where $\mathcal{L}^*(.|M_A)$ is the optimal ELBO under model A. The above derivation allows individual models to be trained in parallel as final model weights depend only on optimized ELBO of each model.

[1]To ensure that the variance parameter $s_{Aj} > 0$, we shall use the reparameterization $s_{Aj} = \log(1 + e^{\tilde{s}_{Aj}})$ and update $\tilde{s}_{Aj}$ instead.

IV. Optimal Dimensionality and Prediction

A. Optimal Dimension Neighborhood Selection

Let $d_A \times p$ be the dimension of an RP, $A \in \mathcal{A}$. Using Section III, we obtain the posterior model weights $q^*(M_A)$. The values of d_A with largest values of $q^*(M_A)$ tend to concentrate around optimal dimension of the feature space. Let $d_1 \leq d_2 \leq \cdots$ be an enumeration of unique values of d_A, $A \in \mathcal{A}$. Define average posterior probability of each d_i as $q_i^* = (1/|\mathcal{A}^i|)\sum_{A\in\mathcal{A}^i} q(M_A)$, where $\mathcal{A}^i = \{A \in \mathcal{A} : d_A = d_i\}$.

The plot of (i, q_i^*) expectedly peaks near the optimal dimension of feature space for prediction. Let $d^* = \text{argmax}_i\, q_i^*$, then for some $\nu_1, \nu_2 > 0$, $\mathcal{I}_{d^*} = [\lfloor d^*(1-\nu_1)\rfloor, \lceil d^*(1+\nu_2)\rceil]$ is the optimal dimension neighborhood used in the final classification. Lastly, $\mathcal{A}_{d^*}$ which is a subspace of RPs with dimension $d_A \times p$ for $d_A \in \mathcal{I}_{d^*}$, is used for final prediction.

B. Classification Based on Optimal Neighborhood Choice

For $A \in \mathcal{A}_{d^*}$, obtain $q^*(M_A, \boldsymbol{\theta}_A)$ using Section III. Let $\widehat{\eta}_A = \int \eta_{\boldsymbol{\theta}_A}(A\boldsymbol{x}_{n+1})q^*(\boldsymbol{\theta}_A|M_A)d\theta_A$ be the VB estimator under model M_A. Define $\widehat{\eta}(\boldsymbol{x}_{n+1}) = \sum_{A\in\mathcal{A}_{d^*}} q^*(M_A)\widehat{\eta}_A(\boldsymbol{x}_{n+1})$, then

$$\widehat{y}_{n+1} = \boldsymbol{I}[\widehat{\eta}(\boldsymbol{x}_{n+1}) \geq 0]. \qquad (8)$$

Although $\widehat{\eta}(\boldsymbol{x}_{n+1})$ is not the exact VB estimator of $\eta(\boldsymbol{x}_{n+1}) = \log(P(y_{n+1} = 1|\boldsymbol{x}_{n+1})/P(y_{n+1} = 0|\boldsymbol{x}_{n+1}))$, it is a good enough approximator for large train sizes and way faster to compute.

C. Gradient Update

For $q(\boldsymbol{\theta}_A|M_A) = \text{MVN}(\boldsymbol{m}_A, \text{diag}(\boldsymbol{s}_A))$ and $p(\boldsymbol{\theta}_A|M_A) = \text{MVN}(\boldsymbol{\mu}_A, \text{diag}(\boldsymbol{\sigma}_A))$, $d_{\text{KL}}(q(.|M_A), p(.|M_A)))$, the Kullback–Leibler distance between two Gaussian densities has a closed-form expression. To further simplify $\mathcal{L}(.|M_A) = E_{Q(\boldsymbol{\theta}_A|M_A)}(\log L(\boldsymbol{\theta}_A|M_A)) - d_{\text{KL}}(q(.|M_A), p(.|M_A)))$, we need to obtain $E_Q(\log L(\boldsymbol{\theta}_A|M_A))$. In this direction, we generate W samples $\boldsymbol{\theta}_A[1], \ldots, \boldsymbol{\theta}_A[W]$ from $q(\boldsymbol{\theta}_A|M_A)$ and let $\widehat{L}(.|M_A) = (1/W)\sum_{w=1}^{W} \log L(\boldsymbol{\theta}_A[w]|M_A)$. The final function to be optimized with respect to $\boldsymbol{m}_A$ and $\boldsymbol{s}_A$ is

$$\widehat{\mathcal{L}}(.|M_A) = \widehat{L}(.|M_A) - d_{\text{KL}}(q(.|M_A), p(.|M_A))). \qquad (9)$$

V. Theoretical Results

We now study the convergence properties of the variational posterior for a given projection matrix A (without model averaging). The results presented are similar in the spirit of [11].

Let $f_0(y, \boldsymbol{x})$ and $f_{\boldsymbol{\theta}}(y, \boldsymbol{x})$ be joint density of $\mathcal{D}$ under truth and model, respectively. Without loss of generality, let $\boldsymbol{x}_i \sim U[0, 1]^p$, thus $f_0(\boldsymbol{x}) = f_{\boldsymbol{\theta}}(\boldsymbol{x}) = 1$. For model indexed by A

$$\begin{aligned} f_{\boldsymbol{\theta}}(y, \boldsymbol{x}) &= f_{\boldsymbol{\theta}}(y|\boldsymbol{x})f(\boldsymbol{x})f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \ell_{\boldsymbol{\theta}}(y, A\boldsymbol{x}) \\ f_0(y, \boldsymbol{x}) &= f_0(y|\boldsymbol{x})f_0(A\boldsymbol{x}) = \ell_0(y, \boldsymbol{x}) \end{aligned} \qquad (10)$$

where $\ell_{\theta}(y, A\boldsymbol{x}) = \exp(y\eta_{\boldsymbol{\theta}}(A\boldsymbol{x}) - \log(1 + \exp(\eta_{\boldsymbol{\theta}}(A\boldsymbol{x}))))$ and $\ell_0(y, \boldsymbol{x}) = \exp(y\eta_0(\boldsymbol{x}) - \log(1 + \exp(\eta_0(\boldsymbol{x}))))$. With $2d_{\text{H}}^2(\ell_0, \ell_{\boldsymbol{\theta}}) = \int_{\boldsymbol{x}} \sum_y \left((\ell_0(y, \boldsymbol{x}))^{1/2} - (\ell_{\boldsymbol{\theta}}(y, A\boldsymbol{x}))^{1/2}\right)^2 d\boldsymbol{x}$, the Hellinger neighborhood of the true density function $f_0 = \ell_0$ is

$$\mathcal{U}_\varepsilon = \{\boldsymbol{\theta} : d_{\text{H}}(\ell_0, \ell_{\boldsymbol{\theta}}) \leq \varepsilon\}. \qquad (11)$$

For the posterior in (6), let $q_A^* = \text{argmin}_{q\in\mathcal{Q}_n^A} d_{\text{KL}}(q, \pi(.|\mathcal{D}, M_A))$, where $\mathcal{Q}_n^A = \{q(\boldsymbol{\theta}_A) = \text{MVN}(\boldsymbol{m}_A, \text{diag}(\boldsymbol{s}_A))\}$. For a fixed A, one can obtain q_A^* by step 2, in Algorithm 1. We give the conditions to ensure

Algorithm 1 RPVBNN

```
1) Initialization: $(\boldsymbol{m}_A^0, \boldsymbol{s}_A^0, \rho_A^{\{t\}})_{A\in\mathcal{A}}$ where $\rho_A^{\{t\}}$, $t \geq 0$ is step size sequence for model $A$.
2) Parallelization:
    a) Set $t = 1$,
    b) For $A \in \mathcal{A}$, calculate the derivative of the gradient $\widehat{\mathcal{L}}(.|M_A)$ in (9) with respect to $\boldsymbol{m}_A$ and $\boldsymbol{s}_A$.
    c) Update the parameters $\boldsymbol{m}_A^t$ and $\boldsymbol{s}_A^t$ as
       $\boldsymbol{m}_A^{t+1} = \boldsymbol{m}_A^t + \rho_A^t \nabla_{\boldsymbol{m}_A} \widehat{\mathcal{L}}(.|M_A)|_{\boldsymbol{m}_A=\boldsymbol{m}_A^t}$
       $\boldsymbol{s}_A^{t+1} = \boldsymbol{s}_A^t + \rho_A^t \nabla_{\boldsymbol{s}_A} \widehat{\mathcal{L}}(.|M_A)|_{\boldsymbol{s}_A=\boldsymbol{s}_A^t}$
    d) Set $t = t + 1$.
    e) Repeat steps (b)-(d) till convergence.
3) Model averaging:
    a) For the optimized values $(\boldsymbol{m}_A^*, \boldsymbol{s}_A^*)_{A\in\mathcal{A}}$, compute $(\widehat{\mathcal{L}}^*(.|M_A))_{A\in\mathcal{A}}$ using (9).
    b) Compute the model weights $q^*(M_A) = \frac{\exp(\log\pi(M_A)+\widehat{\mathcal{L}}^*(.|M_A))}{\sum_{A\in\mathcal{A}}\exp(\log\pi(M_A)+\widehat{\mathcal{L}}^*(.|M_A))}$
4) Optimal neighborhood selection: Using the values $(q^*(M_A))_{A\in\mathcal{A}}$ compute using Section IV-A
    a) The optimal neighborhood $\mathcal{I}_{d^*}$.
    b) The subspace $\mathcal{A}_{d^*}$ using based on $\mathcal{I}_{d^*}$.
5) Classification:
    a) Repeat steps (1)-(3) for $A \in \mathcal{A}_{d^*}$.
    b) Compute $\widehat{y}_{n+1}$ using relation (8)
```

that asymptotically q_A^* concentrates around true density f_0. Recall p and k are number of covariates and nodes, respectively. If A has dimension $d_A \times p$, then total number of parameters in model indexed by A are $K_A = 1 + k + k(d_A + 1)$.

With ψ as the sigmoid activation function, let $\eta_{\boldsymbol{\theta}^*}(\boldsymbol{x}) = \beta_0^* + \sum_{j=1}^k \beta_j^* \psi(\boldsymbol{\gamma}_j^{*\top}\boldsymbol{x})$ be the NN which can approximate the true function $\eta_0(\boldsymbol{x})$ in L_∞ norm. The existence of such an NN is guaranteed by [57]. Let A be an orthonormal matrix, $p(\boldsymbol{\theta}_A) = \text{MVN}(\boldsymbol{\mu}_A, \text{diag}(\boldsymbol{\sigma}_A))$ and the following conditions hold.

(C1): $k_A \log n = o(n\epsilon_n^2)$, $p = o(e^{n\epsilon_n^2})$.

(C2): $||\boldsymbol{\mu}_\beta||_1^2 = o(n\epsilon_n^2)$, $\log||\boldsymbol{\sigma}_\beta||_\infty = O(\log n)$, $||\boldsymbol{\sigma}_\beta^{-1}||_\infty = O(1)$, $\sum_{j=1}^k ||A^\top \boldsymbol{\mu}_{j\gamma}||_1 = O(1)$, $\sup_{j=1,\ldots,k} \log||\boldsymbol{\sigma}_{j\gamma}||_\infty = O(\log n)$, $\sup_{j=1,\ldots,k} ||\boldsymbol{\sigma}_{j\gamma}^{-1}||_\infty = O(1)$.

(C3): $||\eta_0 - \eta_{\boldsymbol{\theta}^*}||_\infty = o(\epsilon_n^2)$, $||\boldsymbol{\beta}^*||_1^2 = o(n\epsilon_n^2)$, $\sup_{j=1,\ldots,k} ||(I - A^\top A)\gamma_j^*||_1 = o(n^{-1})$, $\sum_{j=1}^k ||\boldsymbol{\gamma}_j^*||_1^2 = O(1)$.

(C4): $\log||A\boldsymbol{x}|| = O(\log n)$, $1/||A\boldsymbol{x}|| = o(n\epsilon_n^2)$.

Condition (C1) restricts the number of effective parameters ($\sim kd_A$) and covariates ($\sim p$). Condition (C2) puts restrictions on the prior parameters. Although $\sum_{j=1}^k ||A^\top \boldsymbol{\mu}_{j\gamma}||_1 = O(1)$ depends on A, it holds if $\boldsymbol{\mu}_{j\gamma} = \boldsymbol{0}$. Condition (C4) for A relates to [11, Theorem 3.1, condition 3)]. Condition (C3) quantifies how fast the NN solution converges to the true function with control on the coefficients' magnitude. By the universal approximation of single-layer NNs [57], [63], it can be shown that for any continuous function η_0, there exists an NN $\eta_1 = \beta_0^* + \sum_{j=1}^k \beta_j^* \psi((\gamma_j^*)^\top \boldsymbol{x})$ such that $||\eta_1 - \eta_0||_\infty \leq \epsilon$. Consider the RP-based NN $\eta_2(A x) = \beta_0^* + \sum_{j=1}^k \beta_j^* \psi((A\gamma_j^*)^\top A\boldsymbol{x})$, then by Lemma 1.1 in the supplement, it can be established that $||\eta_2 - \eta_1||_\infty \leq 2\epsilon$ provided $||(I - A^\top A)\gamma_j^*||_1 \leq \epsilon$. This in turn $||\eta_2 - \eta_0||_\infty \leq 3\epsilon$ which further implies that RP-based NNs $\eta_{\boldsymbol{\theta}}(A\boldsymbol{x})$ enjoy universal consistency properties (the proof of Lemma 1.5 in the supplement in fact relies on this universal consistency property). Although the condition $\sup_{j=1,\ldots,k} ||(I - A^\top A)\gamma_j^*||_1 = o(n^{-1})$ seems restrictive, it holds for any γ_j^* in the row space of the matrix A. However, explicit verification is technically involved and is not explored here.

TABLE I
DATA SUMMARY: n, p, AND c ARE THE NUMBER OF SAMPLES, FEATURES, AND CLASSES, RESPECTIVELY. OPTIMAL REGION IS SELECTED BY RPVBNN

Data	Source	n	p	c	Optimal region
SimulateI	[31]	3000	20	2	(4,12)
SimulateII	[31]	3000	200	2	(1,20)
SimulateIII	[58]	5000	150	2	(7,16)
SimulateIV	[58]	300	500	2	(8,17)
ADNI	[59]	264	278	2	(1,30)
GLI_85	[60]	85	22283	2	(20,40)
SMK_187	[60]	187	19993	2	(45,61)
MNIST	[61]	70000	784	10	(580,600)
Fashion-MNIST	[62]	70000	784	10	(725,740)

Theorem 1: Suppose $\epsilon_n \to 0$, $n\epsilon_n^2 \to \infty$ and (C1)–(C4) hold, then

$$q_A^*(\mathcal{U}_{\varepsilon\epsilon_n}^c) \xrightarrow{P_0^n} 0, \ \forall\varepsilon > 0$$

where $P_0^{(n)}$ be the joint distribution of $\mathcal{D}$ under (1).

The proof is presented in the supplement. The above proof shows that q_A* concentrates on shrinking ϵ_n- Hellinger neighborhoods of the true density function f_0.

VI. Numerical Results

A. Problem Setup

We mimic the RP generation of Section II-B with $a_* = 0.3$. The RPVBNN is summarized in Algorithm 1. There is a negligible impact of the tuning parameters, learning rate, number of projections, and batch size on its performance (see the supplement for more details). For the number of projections (N), we use the following rule: let the projected dimension $d_A \in [p_1, p_2]$, then $N = (2^\upsilon)u$, $\upsilon = \text{argmin}_i\{i : 2^i - (p_2 - p_1) > 0\}$. This ensures nearly u number of d_A values are drawn from a unit sub-interval of $[p_1, p_2]$. For optimal dimension selection (step 2 of RPVBNN), we take $(p_1, p_2) = (1, \min(n, p))$ in accordance with condition (C1) of Section V and $u = 1/4$. For final accuracy (step 4 of RPVBNN), (p_1, p_2) is the pre-selected optimal dimension and $u = 4$. Parallel programming is used across projections to reduce computational time.

B. Datasets

We consider nine datasets, four simulated and five real data examples (see Table I). The intrinsic dimensionality is known in simulated examples and unknown in real data. All input variables are z-normalized. We first use RPVBNN to learn the intrinsic dimensionality of the dataset. With optimal dimensionality at hand, we compare RPVBNN to other traditional algorithms [logistic regression (LR), random forest (RF), and gradient boosting (GB)] and the standard VBNN based on whole feature space. For simulated data, Alzheimer's Disease Neuroimaging Initiative (ADNI), GLI_85, and SMK_187, to prevent optimistically biased estimates of model performance, we consider ten different splits of the data into train and test with a ratio of 7:3. We report the mean and standard deviation of the train and test accuracy and an area under the curve (AUC) score over the splits.

C. Simulated Datasets

For Simulates I and II, we use $y = \boldsymbol{I}[e^{x_1} + x_2^2 + 5\sin(x_3 x_4) - 3 > 0]$ as in [31] to generate data. Since the number of active variables is 4, the intrinsic dimensionality of feature space is 4. We set $p = 20$, $n = 3000$ (Simulate I), and $p = 200$, $n = 3000$ (Simulate II) and generate $\boldsymbol{x} \sim \text{MVN}(0, I)$. For Simulates III and IV, we generate $\boldsymbol{x}$

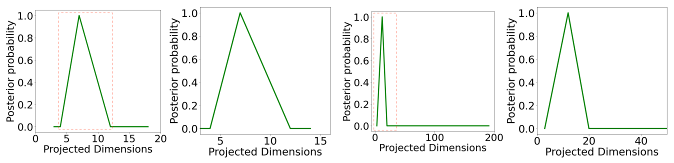

Fig. 2. *Left two:* Optimal region for Simulate I (eight projections). *Right two:* Optimal region for Simulate II (64 projections).

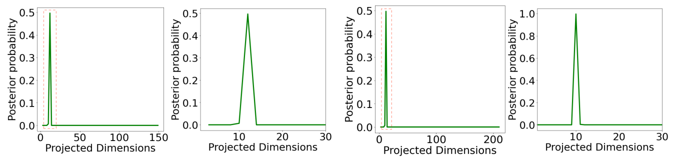

Fig. 3. *Left two:* Optimal region for Simulate III (64 projections). *Right two:* Optimal region for Simulate IV (64 projections).

using the hyper-ball function in [58] and add a "small noise" to all the entries. Let $y = I[\sin(\beta^\top \boldsymbol{x}) - e^{\beta^T \boldsymbol{x}} + 1.05\cos(\beta^T \boldsymbol{x}) > 0]$ with $\beta \sim \mathcal{U}(-0.5, 0.5)$. We set $n = 5000$, $p = 150$ (Simulation III), and $n = 300$, $p = 500$ (Simulation IV). The intrinsic dimension of the hyperball is set at 10. Simulate IV exemplifies the small n large p problem.

D. Real Datasets

We use three biological datasets and two image datasets. The first biological dataset, ADNI, was collected from ADNI database http://www.loni.ucla.edu/ADNI. The final sample had $n = 265$ and $p = 277$ [64]. The other two biological datasets based on Glioma Tumor, GLI_85 and Smokers with Suspected Lung Cancer, SMK_CAN_187 (denoted by SMK_187 in this brief for brevity) were collected from the feature selection datasets available at https://jundongl.github.io/scikit-feature/datasets.html. While GLI_85 had $n = 85$ and $p = 22283$, SMK_187 had $n = 187$ and $p = 19993$. For these ultrahigh-dimensional datasets, we prescreened 100 features using [65].

The two image datasets presented include the Modified National Institute of Standards and Technology (MNIST) [61], a collection of handwritten digits, and Fashion MNIST [62], a dataset of labeled fashion images. Both MNIST and Fashion MNIST contain $n = 70000$ images with 60000 in train and 10000 in test and a feature dimension of $p = 28 \times 28 = 784$.

E. Optimal Dimensional Region

Using Section IV-A, we obtain the average posterior probabilities of the projected dimensions. To learn the intrinsic dimension, we employ RPVBNN with $k = 32$ nodes. Since obtaining the optimal dimension is a preprocessing step, we avoided experimentation with k in this step. Figs. 2 and 3 give the average probability density curve as a function of projected dimensions for simulated datasets (the right panel's zoomed-in view on the region where the density curve peaks). The mode of the density gives an estimate of intrinsic dimensionality and a small interval around the mode characterizes the optimal dimension neighborhood. For Simulate I, Fig. 2 shows that the graph peaks between 6 and 10 and stabilizes after 12. With a mode of 8, the optimal dimension neighborhood was (4, 12). Similarly, for Simulate II, the optimal neighborhood was (1, 20) (see Fig. 2). For both Simulates I and II, the true intrinsic dimensionality was 4. Since the average posterior probability curve peaks around 4, it corroborates that RPVBNN can learn the intrinsic dimension of feature space

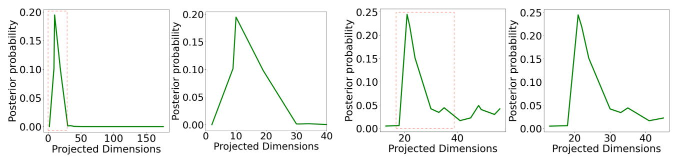

Fig. 4. *Left two:* Optimal region for ADNI (64 projections). *Right two:* Optimal region for GLI_85 (16 projections).

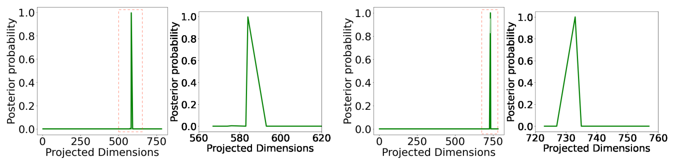

Fig. 5. *Left two:* Optimal region for MNIST (256 projections). *Right two:* Optimal region for Fashion-MNIST (256 projections).

TABLE II
SIMULATED DATA PERFORMANCE

Dataset	Method	Train Acc(%)	Test Acc(%)	AUC(%)
SimulateI	LR-l_1	67.32 ± 0.77	67.44 ± 0.97	68.47 ± 1.51
	RF	68.75 ± 1.91	67.92 ± 1.84	78.30 ± 3.67
	GB	97.62 ± 0.62	87.80 ± 1.95	94.84 ± 0.86
	VBNN	95.28 ± 0.45	90.28 ± 0.65	90.88 ± 0.56
	RPVBNN	95.88 ± 0.32	94.89 ± 0.77	95.83 ± 0.42
SimulateII	LR-l_1	65.75 ± 0.86	66.07 ± 1.09	70.62 ± 1.41
	RF	62.93 ± 3.93	61.69 ± 3.07	70.24 ± 3.43
	GB	99.40 ± 0.04	85.06 ± 2.55	92.98 ± 1.72
	VBNN	63.61 ± 1.13	61.42 ± 1.11	66.21 ± 1.06
	RPVBNN	96.83 ± 0.29	94.92 ± 0.86	96.73 ± 0.41
SimulateIII	LR-l_1	61.49 ± 0.38	61.03 ± 1.53	50.0 ± 1.01
	RF	79.00 ± 0.46	74.12 ± 0.76	78.29 ± 1.38
	GB	61.59 ± 0.34	61.03 ± 1.53	66.91 ± 1.13
	VBNN	100 ± 0.00	51.71 ± 2.63	51.62 ± 2.53
	RPVBNN	99.78 ± 0.11	99.27 ± 0.17	99.84 ± 0.07
SimulateIV	LR-l_1	59.33 ± 2.88	48.11 ± 3.29	44.21 ± 6.97
	RF	83.14 ± 2.99	55.44 ± 3.95	55.04 ± 4.10
	GB	82.85 ± 8.53	54.88 ± 2.59	50.72 ± 7.01
	VBNN	61.21 ± 6.32	56.66 ± 9.03	56.11 ± 7.63
	RPVBNN	97.95 ± 0.91	93.55 ± 2.03	94.72 ± 1.31

for prediction. For Simulates III and IV, the optimal regions were (7, 16) and (8, 17), respectively (see Fig. 3). For both Simulates III and IV, the true intrinsic dimension was 10 (dimension of the hyperball). Since the average posterior probability curve peaks around 10, it corroborates that RPVBNN can learn intrinsic dimensions for prediction. Figs. 4 and 5 provide the optimal regions for ADNI, GLI_85, MNIST, and Fashion MNIST, respectively. For lack of space, the optimal region graph for SMK_187 is provided in the supplement.

F. Comparative Baselines

For the simulated examples, ADNI, GLI_85, and SMK_187, with the pre-selected optimal dimension neighborhood, we use steps 4) and 5) of Algorithm 1 to produce results for RPVBNN (see Tables II and III). We also give results from LR with L_1 penalty (LR-L_1), RF, GB as comparative baseline [66]. For LR, RF, and GB, we report the results corresponding to the best test accuracy. The sensitivity analysis with respect to the hyperparameters for all these methods is given in the supplement. Finally, we report the results of VBNN on the whole feature space (it is RPVBNN with $N = 1$, $d_A = p$, and $A = I$). For RPVBNN and VBNN, the results for the optimal number of nodes are reported. The sensitivity with respect to a number of nodes is given in the supplement. For MNIST and

TABLE III
BIOLOGICAL DATA PERFORMANCE

Dataset	Method	Train Acc(%)	Test Acc(%)	AUC(%)
ADNI	LR-l_1	100 ± 0.00	65.75 ± 4.07	68.71 ± 3.75
	RF	83.67 ± 1.79	72.00 ± 3.88	78.89 ± 4.90
	GB	99.78 ± 0.35	74.87 ± 4.95	81.16 ± 4.17
	VBNN	82.51 ± 2.30	71.75 ± 4.50	79.95 ± 4.45
	RPVBNN	78.97 ± 1.59	76.05 ± 3.76	82.33 ± 1.89
GLI_85	LR-l_1	100 ± 0.00	87.30 ± 5.41	94.08 ± 3.61
	RF	100 ± 0.00	83.84 ± 6.61	85.18 ± 7.54
	GB	99.48 ± 0.71	90.00 ± 4.61	96.39 ± 3.81
	VBNN	100 ± 0.00	90.00 ± 4.31	96.45 ± 3.51
	RPVBNN	100 ± 0.00	96.22 ± 2.13	98.33 ± 1.67
SMK_187	LR-l_1	93.76 ± 1.59	72.85 ± 4.35	81.64 ± 2.72
	RF	84.38 ± 0.77	73.57 ± 4.50	80.68 ± 5.35
	GB	100 ± 0.00	70.18 ± 4.23	78.61 ± 3.99
	VBNN	98.53 ± 0.87	73.75 ± 4.91	81.12 ± 5.47
	RPVBNN	100 ± 0.00	73.75 ± 5.59	81.21 ± 5.32

TABLE IV
IMAGE DATA PERFORMANCE

Dataset	Method	Train Acc(%)	Test Acc(%)	Time(s)
MNIST	VBNN	99.11	97.80	2640
	RPVBNN	98.06	97.32	2350
Fashion-MNIST	VBNN	92.75	88.15	3123
	RPVBNN	93.82	88.07	2535

Fashion MNIST, we report the results of RPVBNN and only VBNN for comparison (see Table IV).

G. Experimental Results

For all the simulated datasets, the performance of RPVBNN supersedes the performance of other algorithms (see Table II). For Simulate I, the second-best performer is VBNN, although its performance deteriorates for Simulate II. This is because, for $p = 200$, the train size of 2100 is way smaller. Since RPVBNN works with compressed feature space, $d_A \in [1, 20]$, it still has the best test accuracy and AUC. Thus, RPVBNN is an effective solution to the small n large p problem. Similarly, for Simulates III and IV, RPVBNN which works with the optimal dimension $d_A \in [7, 16]$ and [8, 17], respectively, outperforms its competitors.

ADNI is another example of a small n and large p problem with $p = 277$ and train size 180. RPVBNN still outperforms its competitors and VBNN suffers from the curse of dimensionality (see Table III). Overall GB is the second-best performer. Similarly GLI_85 is another example of a small n and large p with $n = 70$ and $p = 100$. Similar to ADNI, RPVBNN still outperforms its competitors and VBNN suffers from the curse of dimensionality. For SMK_187, the performance of VBNN and RPVBNN are fairly similar. For MNIST (see Table IV), VBNN with its best test accuracy slightly outperforms the RPVBNN with its best test accuracy. Here, the train size 60 000 is large enough for $p = 784$. However, the average run time based on 500 epochs with VBNN is 2640 s on a 2.3 GHz 8-core Intel Core i9 MacBook Pro workstation while the same for RPVBNN with d_A in optimal dimension neighborhood is 2350 s. For Fashion-MNIST, VBNN and RPVBNN demonstrate the same behavior as under MNIST.

VII. CONCLUSION

To address the curse of dimensionality (small n large p) in a BNN model, we compress the feature space using RPs. To reduce sensitivity to the RP, we propose a BMA approach to base prediction on the most relevant models. We provided a VI technique to estimate model-specific parameters and model weights. As a by-product, the learned posterior model weights of the projected dimensions are used to obtain the intrinsic dimensionality of the feature space. The VB model averaging has two advantages: 1) offers computation gain of frequentist ensemble approaches by allowing parallelization and 2) provides uncertainty quantification due to RP. The approach can be generalized to deep learning with a limited training sample.

REFERENCES

[1] K. Yang and T. Maiti, "On the classification consistency of high-dimensional sparse neural network," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2019, pp. 173–182.

[2] J. Feng and N. Simon, "Sparse-input neural networks for high-dimensional nonparametric regression and classification," 2017, *arXiv:1711.07592*.

[3] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014.

[4] Y. Yao, L. Rosasco, and A. Caponnetto, "On early stopping in gradient descent learning," *Construct. Approx.*, vol. 26, no. 2, pp. 289–315, 2007.

[5] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. B, Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.

[6] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[7] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Statist. Soc. B, Stat. Methodol.*, vol. 67, no. 2, pp. 301–320, 2005.

[8] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *J. Comput. Graph. Statist.*, vol. 22, no. 2, pp. 231–245, May 2012.

[9] T. Park and G. Casella, "The Bayesian Lasso," *J. Amer. Stat. Assoc.*, vol. 103, no. 482, pp. 681–686, 2008.

[10] C. M. Carvalho, N. G. Polson, and J. G. Scott, "Handling sparsity via the horseshoe," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*, vol. 5, D. van Dyk and M. Welling, Eds., 2009, pp. 73–80.

[11] R. Guhaniyogi and D. B. Dunson, "Bayesian compressed regression," *J. Amer. Stat. Assoc.*, vol. 110, no. 512, pp. 1500–1514, 2015.

[12] J. Shlens, "A tutorial on principal component analysis," 2014, *arXiv:1404.1100*.

[13] S. Dasgupta and A. Gupta, "An elementary proof of a theorem of Johnson and Lindenstrauss," *Random Struct. Algorithms*, vol. 22, no. 1, pp. 60–65, 2003.

[14] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Jan. 2006.

[15] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207–1223, Feb. 2005.

[16] T. L. Marzetta, G. H. Tucci, and S. H. Simon, "A random matrix-theoretic approach to handling singular covariance estimates," *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 6256–6271, Sep. 2011.

[17] T. I. Cannings and R. J. Samworth, "Random-projection ensemble classification," *J. Roy. Stat. Soc. B, Stat. Methodol.*, vol. 79, no. 4, pp. 959–1035, Sep. 2017.

[18] R. Durrant and A. Kaban, "Random projections as regularizers: Learning a linear discriminant ensemble from fewer observations than dimensions," in *Proc. 5th Asian Conf. Mach. Learn.*, vol. 29, C. S. Ong and T. B. Ho, Eds. Canberra, ACT, Australia: Australian National Univ., Nov. 2013, pp. 17–32.

[19] A. E. Raftery, D. Madigan, and J. A. Hoeting, "Bayesian model averaging for linear regression models," *J. Amer. Stat. Assoc.*, vol. 92, no. 437, pp. 179–191, Mar. 1997.

[20] R. Guhaniyogi and D. Dunson, "Compressed Gaussian process for manifold regression," *J. Mach. Learn. Res.*, vol. 17, no. 69, pp. 1–26, 2016.

[21] T. S. Jaakkola and M. I. Jordan. (1996). *A Variational Approach to Bayesian Logistic Regression Models and Their Extensions*. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.210

[22] D. M. Blei and J. D. Lafferty, "A correlated topic model of science," *Ann. Appl. Statist.*, vol. 1, no. 1, pp. 17–35, 2007.

[23] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1613–1622.

[24] D. J. C. MacKay. (1992). *A Practical Bayesian Framework for Backpropagation Networks*. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.274

[25] R. M. Neal, *Bayesian Learning for Neural Networks*. Berlin, Germany: Springer-Verlag, 1996.

[26] J. Lampinen and A. Vehtari, "Bayesian approach for neural networks—Review and case studies," *Neural Netw.*, vol. 14, no. 3, pp. 257–274, Apr. 2001.

[27] S. Sun, C. Chen, and L. Carin, "Learning structured weight uncertainty in Bayesian neural networks," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, A. Singh and J. Zhu, Eds., 2017, pp. 1283–1292.

[28] V. Mullachery, A. Khera, and A. Husain, "Bayesian neural networks," 2018, *arXiv:1801.07710*.

[29] A. Hubin, G. Storvik, and F. Frommlet, "Deep Bayesian regression models," 2018, *arXiv:1806.02160*.

[30] K. Javid, W. Handley, M. Hobson, and A. Lasenby, "Compromise-free Bayesian neural networks," 2020, *arXiv:2004.12211*.

[31] F. Liang, Q. Li, and L. Zhou, "Bayesian neural networks for selection of drug sensitive genes," *J. Amer. Stat. Assoc.*, vol. 113, no. 523, pp. 955–972, Jul. 2018.

[32] S. Ghosh, J. Yao, and F. Doshi-Velez, "Model selection in Bayesian neural networks via horseshoe priors," *J. Mach. Learn. Res.*, vol. 20, no. 182, pp. 1–46, 2019.

[33] J. Bai, Q. Song, and G. Cheng, "Efficient variational inference for sparse deep learning with theoretical guarantee," in *Advances in Neural Information Processing Systems*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 466–476.

[34] N. Ailon and B. Chazelle, "Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform," in *Proc. 38th Annu. ACM Symp. Theory Comput. (STOC)*, 2006, pp. 557–563.

[35] M. Lopes, L. Jacob, and M. J. Wainwright, *Advances in Neural Information Processing Systems*, vol. 24, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2011, pp. 1206–1214.

[36] R. Durrant and A. Kaban, "Sharp generalization error bounds for randomly-projected classifiers," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28, no. 3, S. Dasgupta and D. McAllester, Eds., 2013, pp. 693–701.

[37] R. J. Durrant and A. Kabán, "Random projections as regularizers: Learning a linear discriminant from fewer observations than dimensions," *Mach. Learn.*, vol. 99, no. 2, pp. 257–286, May 2015.

[38] N. Goel, G. Bebis, and A. Nefian, "Face recognition experiments with random projection," in *Proc. SPIE*, vol. 5776, Mar. 2005, pp. 426–437.

[39] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[40] G. E. Hinton and D. van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proc. 6th Annu. Conf. Comput. Learn. Theory (COLT)*, 1993, pp. 5–13.

[41] B. A. Logsdon, G. E. Hoffman, and J. G. Mezey, "A variational Bayes algorithm for fast and accurate multiple locus genome-wide association analysis," *BMC Bioinf.*, vol. 11, no. 1, p. 58, Dec. 2010.

[42] P. Carbonetto and M. Stephens, "Scalable variational inference for Bayesian variable selection in regression, and its accuracy in genetic association studies," *Bayesian Anal.*, vol. 7, pp. 73–108, Mar. 2012.

[43] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. ICML*, vol. 37, 2015, pp. 1613–1622.

[44] S. Sun, G. Zhang, J. Shi, and R. B. Grosse, "Functional variational Bayesian neural networks," in *Proc. 7th Int. Conf. Learn. Represent. (ICLR)*, 2019. [Online]. Available: https://openreview.net/forum?id=rkxacs0qY7

[45] R. Price, "A useful theorem for nonlinear devices having Gaussian inputs," *IRE Trans. Inf. Theory*, vol. 4, no. 2, pp. 69–72, Jun. 1958.

[46] R. Ranganath, S. Gerrish, and D. Blei, "Black box variational inference," in *Proc. 17th Int. Conf. Artif. Intell. Statist.*, vol. 33, 2014, pp. 814–822.

[47] J. Paisley, D. Blei, and M. Jordan, "Variational Bayesian inference with stochastic search," in *Proc. 29th Int. Conf. Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 1363–1370.

[48] S. Bhattacharya and T. Maiti, "Statistical foundation of variational Bayes neural networks," *Neural Netw.*, vol. 137, pp. 151–173, May 2021.

[49] Y. Wang and D. M. Blei, "Frequentist consistency of variational Bayes," *J. Amer. Stat. Assoc.*, vol. 114, no. 527, pp. 1147–1161, Jul. 2019.

[50] D. Pati, A. Bhattacharya, and Y. Yang, "On statistical optimality of variational Bayes," in *Proc. 21st Int. Conf. Artif. Intell. Statist.*, vol. 84, A. Storkey and F. Perez-Cruz, Eds., 2018, pp. 1579–1588.

[51] F. Zhang and C. Gao, "Convergence rates of variational posterior distributions," *Ann. Statist.*, vol. 48, no. 4, pp. 2180–2207, Aug. 2020.

[52] V. Kejzlar, S. Bhattacharya, M. Son, and T. Maiti, "Black box variational Bayes model averaging," *Amer. Stat. Assoc.*, pp. 1–12, 2022.

[53] P. Latouche and S. Robin, "Variational Bayes model averaging for graphon functions and motif frequencies inference in W-graph models," *Statist. Comput.*, vol. 26, no. 6, pp. 1173–1185, Nov. 2016.

[54] K. Yang and T. Maiti, "Statistical aspects of high-dimensional sparse artificial neural network models," *Mach. Learn. Knowl. Extraction*, vol. 2, no. 1, pp. 1–19, Jan. 2020.

[55] S. Dasgupta, "Experiments with random projection," in *Proc. 16th Conf. Uncertainty Artif. Intell.*, Jun. 2000, pp. 143–151.

[56] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers Comput. Sci.*, vol. 14, no. 2, pp. 241–258, 2020.

[57] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jul. 1989.

[58] J. Bac, E. M. Mirkes, A. N. Gorban, I. Tyukin, and A. Zinovyev, "Scikit-dimension: A Python package for intrinsic dimension estimation," *Entropy*, vol. 23, no. 10, p. 1368.

[59] (2003). *Alzheimer's Disease Neuroimaging Initiative*. Accessed: Jul. 27, 2021. [Online]. Available: http://adni.loni.usc.edu

[60] J. Li *et al.*, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, p. 94, 2016.

[61] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[62] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[63] H. K. H. Lee, "Consistency of posterior distributions for neural networks," *Neural Netw.*, vol. 13, no. 6, pp. 629–642, Jul. 2000.

[64] Z. Liu, T. Maiti, and A. R. Bender, "A role for prior knowledge in statistical classification of the transition from mild cognitive impairment to Alzheimer's disease," *J. Alzheimer's Disease*, vol. 83, no. 4, pp. 1859–1875, Oct. 2021.

[65] J. Fan and Y. Fan, "High-dimensional classification using features annealed independence rules," *Ann. Statist.*, vol. 36, no. 6, pp. 2605–2637, Dec. 2008, doi: 10.1214/07-AOS504.

[66] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," 2018, *arXiv:1201.0490*.