Adaptive Uncertainty Estimation via High-Dimensional Testing on Latent Representations ————Supplementary Materials————

Anonymous Author(s)
Affiliation
Address
email

Overview. In the supplementary material, we first present a detailed summary of the datasets in Section 1. Additionally, we provide detailed descriptions of distributions used for BNN training in Section 2. We further present the implementation settings, hyperparameters and settings of baseline methods in Section 3. We also provide the definitions of uncertainty measures used for comparison (Section 4), together with an algorithm of our framework (Algorithm 1).

1 Additional Information on Datasets

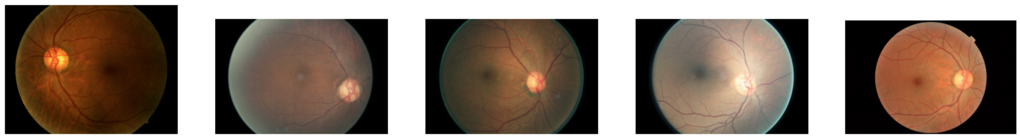
Table 1 presents a summary of the datasets used for experiments. We totally use seven image datasets to evaluate our method, including nature image and medical image.

Table 1: Summary of Datasets

Dataset	No. Classes	No. Training	No. Testing
MNIST	10	60,000	10,000
Fashion-MNIST	10	60,000	10,000
OMNIGLOT	50	13,180	19,280
SVHN	10	73,257	26,032
CIFAR-10	10	60,000	10,000
CIFAR-100	100	60,000	10,000
DRD	2	50	100

Diabetes Retinopathy Detection (DRD). For this experiment, we define normal samples as healthy (no DR; with label 0), and OOD samples as DR (mild, moderate, severe, or proliferative DR; with labels 1–4). We select 50 healthy images to train the encoder, and compute $\boldsymbol{\mu}_1$ and $\boldsymbol{\Sigma}_1$ from these samples using the trained encoder. For testing, we select 50 images as in-distribution data and 50 images as the OOD data. All images are resized to 64×64 for computational convenience. We train the encoder with a task to classify whether the input image is left eye or right eye. The examples of healthy and DR are presented in Figure 1.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.



(a) Healthy samples, $y = 0$.

(b) Unhealthy samples, $y = 1$.

Figure 1: Health and unhealthy (DR) samples from the Diabetes Retinopathy Detection (DRD) dataset [2]

2 Multivariate Gaussian Distribution

The Multivariate Gaussian is crucial for the approximation of vanilla BNN [5], we provide the formal definition and its important properties in this section. The density of a multivariate Gaussian distribution is defined as

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\Big\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big\},$$

where $\boldsymbol{\mu} \in \mathbb{R}^p$ is a p-dimensional mean vector and $\boldsymbol{\Sigma} \in \mathbb{R}^{p\times p}$ is the covariance matrix.

The KL divergence between two multivariate normal distributions $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ is given by

$$\mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) \| \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)) = \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - p + \mathrm{tr}\{\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\} + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)\right].$$

3 Baseline Methods and Implementation Details

Implementation Details. We present additional implementation details and hyperparameter settings. We first provide the key settings and adaptations applied to the baseline methods for reproducibility. We follow the default settings for other fine-grained parameters (e.g., learning rates).

The proposed method is implemented in Python with *Pytorch* library on a server equipped with four NVIDIA TESLA V100 GPUs. The dropout ratio of each dropout layer is selected as 0.2. All models are trained with 100 epochs with possible early stopping. We use the *Adam* optimizer to optimize the model with a learning rate of 5×10^{-5} and a weight decay of 1×10^{-5}. Data augmentations such as color jittering and random cropping and flipping are applied as a regularization measure.

Hyperparameter Settings. The hyperparameter settings for BNN training and ARHT testing are

- Prior of mean of weights — sample from $\mathcal{N}(-3, 0.01)$
- $s = 5$
- $n_2 = 300$
- $\lambda_0 = 0.01$

Additional Settings of Baseline Methods. We further introduce the experimental settings of baseline methods:

- Deep ensembles [7]: Set the number of ensembles as 5.
- MCDropout [3]: Set the dropout ratio as 0.2 for both training and inference.
- Kendall and Gal: Set the number of inference weights samples as 20.
- Detectron [4]: Set the number of runs as 100.
- PostNet [1]: Because the original codes operate on the features extracted from the standard datasets. Their method cannot be generalized to new datasets (e.g., SVHN) as the data processing codes are not provided.

4 Uncertainty Measures

We describe the uncertainty measures used in the OOD misclassification task in this section. The definitions are well-known and summarized by Malinin and Gales [8],

- Entropy:

$$H[p(\boldsymbol{\mu}|\mathcal{D})] = -\int_{S^{K-1}} p(\boldsymbol{\mu}|\mathcal{D}) \ln p(\boldsymbol{\mu}|\mathcal{D}) d\boldsymbol{\mu},$$

  where S^{K-1} is the supporting set, and $\boldsymbol{\mu}$ is the predictive class probability which is assumed to follow Dirichlet distribution.
- Maximum probability: we take the maximum predicted probability $\mathcal{P}$ from all classes as the confidence score,

$$\mathcal{P} = \max_c P(w_c|\mathcal{D}).$$

  where $P(w_c|\mathcal{D})$ is the predictive probability of class c.
- Differential entropy:

$$I[y, \boldsymbol{\mu}|\mathcal{D}] = -\int_{S^{K-1}} p(\boldsymbol{\mu}|\mathcal{D}) \ln p(\boldsymbol{\mu}|\mathcal{D}) d\boldsymbol{\mu}$$

- Accuracy: the fraction of correct predictions to the total number of ground truth labels.
- F-1 score: The F-1 score for each class is defined as

$$\text{F-1 score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

  where 'recall' is the fraction of correct predictions to the total number of ground truths in each class and precision is the fraction of correct predictions to the total number of predictions in each class.
- AUROC: the area under the receiver operating curve (ROC) which is the plot of the true positive rate (TPR/Recall) against the false positive rate (FPR).
- AUPR: the area under the precision-recall curve. Note that the AUPR for binary classification is sensitive to the distribution of positive and negative classes. Hence, the higher AUPR does not necessarily imply a better model performance.

5 Algorithm

Algorithm 1 demonstrates the detailed workflow of our proposed uncertainty estimation framework.

Algorithm 1 Our proposed uncertainty estimation framework

Input:
The prior distribution of weights of BNN encoder $\pi(\boldsymbol{\theta}) \sim \mathcal{N}(0, \boldsymbol{I})$;
Training data $\mathcal{D}_{tr} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N_{tr}}$;
Testing data $\mathcal{D}_{te} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N_{te}}$;
Hyperparameters μ_0, ρ_0, n_2;
Initial variational posterior distribution $q(\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{\mu}, \log(1 + \exp(\boldsymbol{\rho})))$ with initial parameters $\boldsymbol{\mu} = \mu_0 \mathbf{1}$ and $\boldsymbol{\rho} = \rho_0 \mathbf{1}$
Output: The uncertainty scores

for $(\boldsymbol{x}_i, y_i)$ in $\mathcal{D}_{tr}$ **do** ▷ Train BNN encoder
    Draw weight sample $\boldsymbol{\theta}$ from $q(\boldsymbol{\theta})$
    $\hat{y}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ ▷ Forward propagation
    Compute task-specific loss $\mathcal{L}_{\mathrm{obj}}$
    Compute the KL($q \| \pi$) and hence the ELBO
    Backpropagate the ELBO to update $\boldsymbol{\mu}$ and $\boldsymbol{\rho}$
end for
Compute $\mu_1 \in \mathbb{R}^p$, $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{p \times p}$ ▷ Get summary statistics of training
for $(\boldsymbol{x}_i, y_i)$ in $\mathcal{D}_{tr}$ **do** ▷ OOD Detection
    Compute $\mu_2 \in \mathbb{R}^p$, $\boldsymbol{\Sigma}_2 \in \mathbb{R}^{p \times p}$
    Compute the pooled sample covariance matrix by Eq. (1)
    Compute ARHT by Eq. (4) as the uncertainty score
    Detect OOD samples using the uncertainty score under famil-wise adjusted threshold
end for

References

[1] Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. *Advances in Neural Information Processing Systems*, 33:1356–1367, 2020.

[2] Angelos Filos, Sebastian Farquhar, Aidan N Gomez, Tim GJ Rudner, Zachary Kenton, Lewis Smith, Milad Alizadeh, Arnoud de Kroon, and Yarin Gal. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *arXiv preprint arXiv:1912.10481*, 2019.

[3] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[4] Tom Ginsberg, Zhongyuan Liang, and Rahul G Krishnan. A learning based hypothesis test for harmful covariate shift. *arXiv preprint arXiv:2212.02742*, 2022.

[5] Laurent Valentin Jospin. Hands-on bayesian neural networks-a tutorial for deep learning users. 2020.

[6] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[7] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[8] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.