Efficient Scaling of Bayesian Neural Networks

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

While Bayesian neural networks (BNNs) have gained popularity for their theoretical guarantees and robustness, they have yet to see a convincing implementation at scale. This study investigates a variational inference-based neural architecture called Variational Density Propagation (VDP) that boasts noise robustness, self-compression and improved explanations over traditional (deterministic) neural networks. Due to the large computational burden associated with BNNs, however, these methods has yet to scale efficiently for real-world problems. In this study, we simplify the VDP architecture reducing its time and space requirements allowing for efficient scaling to ImageNet level problems. Additionally, we evaluate the inherent properties of the VDP method in order to validate the simplified method. Across all datasets and architectures, our method exhibits exceptional self-compression capabilities, retaining performance even with over 90% of its parameters pruned. The method also presents improved visual explanations via saliency maps, suggesting superior explanation quality compared to deterministic models. Lastly, we employ the VDP method to train a vision transformer on ImageNet-1k, something that was previously impossible due to the inherent computational constraints of the method.

1 Introduction

The state of machine learning research has seen tremendous growth, with increasingly complex and large-scale models being developed [1]. This increase in size and complexity, however, has led to a notable concern regarding overfitting [2]. As models become larger, they may inadvertently memorize training data instead of learning to generalize, resulting in poor performance on unseen or novel data samples. A guiding principle against complexity increase has been the The Minimum Description Length Principle which states that the best model is one which minimizes the distance between the model and data and the models description of itself [3]. This principle is the foundation of stochastic modeling and a benefit of using Bayesian inference.

Bayesian inference provides a principled approach for dealing with uncertainty by combining prior knowledge and observed data to update beliefs about model parameters [4]. In the context of machine learning, Bayesian Neural Networks (BNNs) extend this framework by incorporating Bayesian principles into the architecture and training of neural networks [5]. BNNs estimate the posterior distribution over model parameters, allowing them to capture and quantify uncertainties in both predictions and model structures. By leveraging Bayesian inference, BNNs offer a more robust and adaptive learning approach, which can potentially alleviate overfitting and improve generalization performance in complex machine learning tasks [6].

BNNs have demonstrated successful implementations across a wide range of tasks, including computer vision [7–9], speech [10], and natural language processing [11]. BNNs have shown particular promise in critical decision-making tasks [12–14], due to their ability to capture and model uncertainty in both parameters and predictions [7]. Several classical techniques have been used to approximate

Bayesian inference for neural networks, such as Laplacian approximation [15], Hamiltonian Monte Carlo [16], and Variational Inference (VI) [6]. Specifically, estimating the parameters of the variational distribution that approximates the posterior of the latent variables given an observation is analogous to averaging model parameters via Monte Carlo sampling [17]. This model averaging inherently reduces variance in computed parameters, thus mitigating overfitting [18]. Moreover, the regularization term of the VI algorithm can lead to extreme self-compression of the model's parameters [19], further decreasing overfitting.

A major drawback of BNNs is their limited scalability in terms of both model and data size [20]. A high-quality implementation of a VI-based training scheme called Variational Density Propagation (VDP) has emerged, offering state-of-the-art performance on various computer vision tasks, reduced epistemic uncertainty, and an estimate of aleatoric uncertainty [8, 9]. However, the primary limitation of the VDP algorithm is the substantial computational load due to the propagation of the full covariance matrix through layers of the neural network. Propagating these large matrices quickly becomes infeasible in terms of space and time for all but the smallest ResNets, significantly limiting the applicability of the VDP method for problems that require training on datasets larger than CIFAR-10 [21].

In this work, we propose a modification to the VDP method, in which we disregard the off-diagonal terms of the variance-covariance matrix and only propagate the diagonal terms. This simplification enables efficient scaling of these models to larger and more complex datasets, with a theoretical memory requirement only twice that of a traditional deep neural network, while retaining all the benefits of BNNs. Our contributions include:

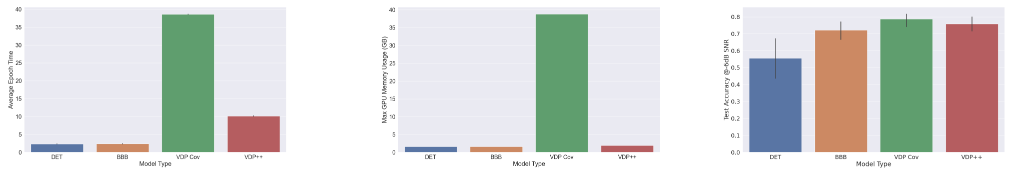
1. A theoretical justification for the novel variance-only VDP implementation (referred to as VDP++).
2. A comparison of runtime and memory requirements between VDP++ and traditional models.
3. Validation of inherent BNN properties, including uncertainty quantification, self-compression, and robust interpretations or explainability.
4. Implementation of Bayesian VDP Vision Transformers (VDP-ViT) and scaling the model to ImageNet-1k.

2 Related Work

Self-compression: Since their inception in the early 1990s, BNNs have been developed with the goal of minimizing the information content in the weights of a neural network. This approach is based on the principle that smaller model sizes often lead to more generalizable models [6, 15]. However, contemporary BNN methods such as Bayes by Backprop (BBB) [7] and Dropout CNNs [22] appear to lack this property. In the case of BBB, parameter histograms suggest a less efficient use of parameters compared to traditional deterministic neural networks [7]. To the best of our knowledge, the only modern framework that exhibits this self-compression property is Variational Density Propagation (VDP) [8, 9, 19]. The self-compression characteristic of VDP can be attributed to the propagation of variance information through the network layers [19]. By quantifying parameter uncertainty, the model can selectively target less important parameters during training. In the absence of uncertainty quantification, the network's utilization of its parameter space is less efficient.

Uncertainty Quantification: The total uncertainty in a machine learning model can be characterized by two distinct components. The first is epistemic uncertainty, which arises from the parameters of the model. The second is aleatoric uncertainty, which stems from the data or environment. Uncertainty quantification is becoming increasingly important in high-stakes decision-making, as establishing trust between users and models is crucial for the widespread adoption of such models [23]. To date, the implementation of machine learning models in high-stakes domains, such as medicine, has not achieved substantial user acceptance and confidence [24, 25]. This is primarily due to high false alarm rates [26] and suboptimal test characteristics [27]. VI has been demonstrated to alleviate the impact of epistemic uncertainty by effectively averaging predictions during inference [6, 7, 18]. Furthermore, the output variance associated with a single prediction obtained from a BNN can quantify the level of aleatoric uncertainty in that prediction [8, 28].

Explainability: A common approach to interpreting predictions in computer vision models involves calculating sensitivity maps. These gradient-based methods are used to determine the contribution



(a) (b) (c)

Figure 1: In order to motivate the needs for a simplification of the VDP algorithm, we measured the: (a) average epoch time, (b) maximum GPU memory usage for each model type training on MNIST with a batch size of 512 and (c) test accuracy on the MNIST dataset with a high level of noise added (higher is better). All measurements were taken using a Quadro RTX 8000 GPU.

of each image pixel to the classification output [29]. However, interpretations derived from deep neural networks, particularly sensitivity maps, are often fragile—meaning they are sensitive to small perturbations in the input or model [30–32]. One potential solution to this issue involves sampling the surrounding input space, adding Gaussian noise, and averaging the resulting interpretations [33]. This averaging technique for reducing uncertainty aligns with our hypothesis on BNNs and weight averaging. It has been demonstrated that the explanations generated by the VDP model outperform those from traditional models [34].

3 Methods

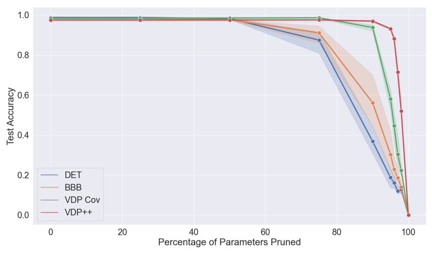
Variance-only Variational Density Propagation (VDP++): Our work presents a novel simplification of the VDP algorithm developed in [8]. This method utilizes VI and assumes Tensor Normal Distributions (TNDs) defined over the parameters of the neural network in order to propagate the first two moments (mean and covariance) of these TNDs through the layers of a neural network [8]. However, in an experimental setting, propagating covariance proves to be computationally expensive in terms of both time and space. As these networks scale, calculating such large matrices becomes increasingly burdensome, rendering the method impractical. In this work, we have streamlined the VDP algorithm by propagating only the diagonal elements of the covariance matrix, i.e., the variance. This simplification, originally discussed by Hinton [6], is akin to the assumptions of the Naive Bayes classifier [35].

For the sake of brevity, we have omitted the original derivation of the VDP algorithm and only provide the modifications needed to compute the variance-only version [8]. To describe our method, we use a 2-layer fully-connected neural network as an example. In a traditional or deterministic neural network, Equation 1 describes the forward pass of the model.

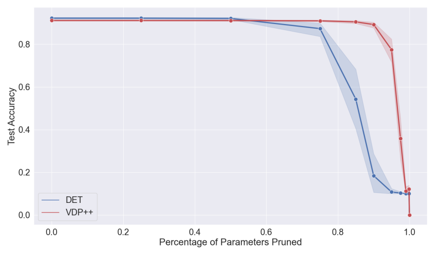
$$\begin{aligned} z &= Wx + b^{[1]}, \\ a &= f(z), \\ \hat{y} &= g(Va + b^{[2]}), \end{aligned} \tag{1}$$

where $W \in \mathbb{R}^{j\times k}$ is the weight matrix of layer 1, $x \in \mathbb{R}^{k\times 1}$ is the input vector, $b^{[1]} \in \mathbb{R}^{j\times 1}$ is the bias vector in layer 1, $z \in \mathbb{R}^{j\times 1}$ is the result of the linear operation in layer 1, f is an arbitrary element-wise non-linear function, g is an arbitrary non-linear activation function that does not operate element-wise, $a \in \mathbb{R}^{j\times 1}$ is the result after applying the non-linear activation function in layer 1, $V \in \mathbb{R}^{l\times j}$ is the weight matrix in layer 2, $b^{[2]} \in \mathbb{R}^{j\times 1}$ is the bias vector in layer 2, $\hat{y} \in \mathbb{R}^{l\times 1}$ is the predicted output, k is the dimensionality of the input vector, j is the number of nodes in layer 1 and l is the number of classes to predict.

To propagate the first two moments, several assumptions must be made. First, let us consider $w_m^\top = m^{\text{th}}$ row of W, $m = 1, 2, ..., j$ and $z_m = w_m^\top x + b_m^{[1]}$, $m = 1, 2, ..., j$. Next, consider the following assumptions: the input vector x is deterministic, $a \sim \mathcal{N}(\mu_a, \Sigma_a)$, $w_m \sim \mathcal{N}(\mu_{w_m}, \Sigma_{w_m})$, $m = 1, 2, ..., j$, $b_m^{[1]} \sim \mathcal{N}(\mu_{b_m^{[1]}}, \sigma^2_{b_m^{[1]}})$, $m = 1, 2, ..., j$, $v_n \sim \mathcal{N}(\mu_{v_n}, \Sigma_{v_n})$, $n = 1, 2, ..., l$, $b_n^{[2]} \sim \mathcal{N}(\mu^{[2]}_{b_n}, \sigma^2_{b_n^{[2]}})$, $n = 1, 2, ..., l$ and the weight vectors w_m, a, bias $b^{[1]}$ and $b^{[2]}$ are mutually uncorrelated



(a)



(b)

Figure 2: Global unstructured pruning was performed on each model type tested on: (a) MNIST and (b) CIFAR-10. This process removes parameters from the model from all layers in by their L1 norm. Here we show that the VDP model types have superior self-compressing properties as they can maintain their performance when greater than 90% of the parameters of the model are removed. Five models were trained for each model and dataset and the 95% confidence interval is displayed as the shaded region.

with each other for $m = 1, 2, ..., j$. Based on these assumptions, the elements of μ_z and σ_z^2 are given in Equations 2 and 3.

$$\begin{aligned} \mu_{z_m} &= \mathbb{E}[w_m^\top x + b_m^{[1]}], \\ &= \mathbb{E}[w_m^\top]x + \mathbb{E}[b_m^{[1]}], \\ &= \mu_{w_m}^\top x + \mu_{b_m^{[1]}}. \end{aligned} \tag{2}$$

$$\begin{aligned} \sigma_{z_m}^2 &= \text{Var}[w_m^\top x + b_m^{[1]}], \\ &= x_m^\top \Sigma_{w_m}^2 x_m + \sigma_{b_m^{[1]}}^2, \\ &= x_m^2 [\sigma_{w_m}^2]^\top + \sigma_{b_m^{[1]}}^2. \end{aligned} \tag{3}$$

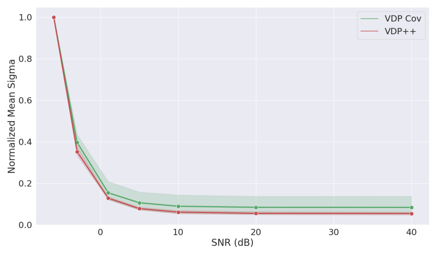
Since we have assumed the weight vectors and the elements of the bias vector to be uncorrelated, $\Sigma_{w_p w_q} = 0$ and $\sigma_{b_p b_q} = 0$ for $p \neq q$, where $p, q = 1, 2, ..., j$. Hence, the covariance is zero. To propagate the first two moments through an arbitrary element-wise non-linear function, we utilize the first-order Taylor series approximation shown in Equations 4 and 5.

$$\begin{aligned} a &= f(z), \\ &= f(\mu_z) + f'(\mu_z)(z - \mu_z) + ... \\ &\approx f(\mu_z) + f'(\mu_z)(z - \mu_z), \\ \mathbb{E}[a] &= \mu_a \approx f(\mu_z), \\ \mu_a &= f(\mu_z). \end{aligned} \tag{4}$$

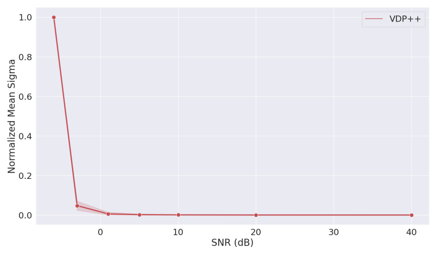
$$\sigma_a^2 = \sigma_z^2 \odot (f'(\mu_z))^2. \tag{5}$$

For the second layer of the network, we can no longer consider the incoming vector, μ_a, to be deterministic. Additionally, we now need to propagate the incoming variance, σ_a^2. Again, we assume the off-diagonal elements of the covariance Σ_a to be 0 and choose to only propagate the variance. The mean and variance propagated through the second fully-connected layer are given in Equations 6 and 7.

$$\begin{aligned} \mu_{\tilde{y}} &= \mathbb{E}[v_n^\top a + b_n^{[2]}], \\ &= \mathbb{E}[v_n^\top]\mathbb{E}[a] + \mathbb{E}[b_n^{[2]}], \\ &= \mu_{v_n}^\top \mu_a + \mu_{b_n^{[2]}}. \end{aligned} \tag{6}$$



(a)



(b)

Figure 3: To validate our model's robustness to noise, a procedure designed by Dera *et al*. [8] was performed. Gaussian noise was added to the test set of: (a) MNIST and (b) CIFAR-10 to generate sets of images with varying Signal to Noise Ratio (SNR). The output variance of the predicted class was extracted for each prediction and was aggregated over the test set. The output variances were normalized by dividing the variance by the mean variance of the lowest SNR. Five models were trained for each model and dataset and the 95% confidence interval is displayed as the shaded line.

$$\begin{aligned}\sigma_{\tilde{y}}^2 &= \text{Var}[v_n^\top a + b_n^{[2]}],\\ &= \text{Var}[v_n^\top a] + \text{Var}[b_n^{[2]}],\\ &= \text{Tr}(\Sigma_v^2 \Sigma_a^2) + \mu_v^\top \Sigma_a^2 \mu_v + \mu_a^\top \Sigma_v^2 \mu_a + \Sigma^2_{b_n^{[2]}},\\ &= \sigma_v^2[\sigma_a^2]^\top + \mu_v^2[\sigma_a^2]^\top + \mu_a^2[\sigma_v^2]^\top + \sigma^2_{b_n^{[2]}}.\end{aligned} \tag{7}$$

Since the diagonals of the covariance matrix are along the rows of the variance matrix, the trace of the product of two vectors is equivalent to their inner product.

Next, we use a non-linear activation function such as softmax to obtain the predictions of our model. Since g does not operate element-wise on our mean and variance, we use a slightly different Taylor-series to obtain Equations 8 and 9 [36].

$$\mu_{\hat{y}} \approx g(\mu_{\tilde{y}}), \tag{8}$$

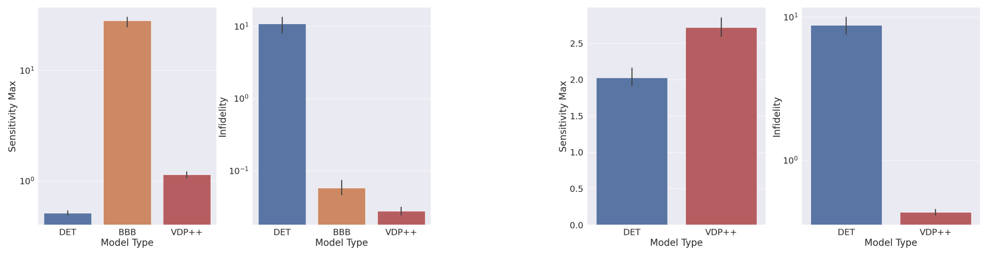
$$\sigma_{\hat{y}}^2 \approx J_g^2 \sigma_{\tilde{y}}^2, \tag{9}$$

where J_g is the Jacobian matrix of the softmax function g with respect to $\tilde{y}$ and calculated at $\mu_{\tilde{y}}$.

Finally, we use the Evidence Lower Bound (ELBO) function, $\mathcal{L}(\phi, D)$, which consists of two parts: the expected log-likelihood of the training data given the weights, and a regularization term, $D = \{x^{(i)}, y^{(i)}\}_{i=1}^N$ given by Equation 10.

$$\mathcal{L}(\phi, D) = \mathbb{E}_{q(\phi)}[\log p(D|\phi)] - \text{KL}[q(\phi)|p(\phi)]. \tag{10}$$

In Equation 10. ϕ represents the weights W, V, and biases $b^{[1]}$, $b^{[2]}$. The expected log-likelihood is given in Equation 11 and the KL term is given in Equation 12.



(a) (b)

Figure 4: To evaluate the effect if any on the quality of explanations produced by BNNs, two metrics were used: Sensitivity Max [32] and Infidelity [37] (lower is better). Perturbations were added to all instances of the test sets of (a) MNIST and (b) CIFAR-10 and the Sensitivity and Infidelity scores were computed and averaged. Five models were trained for each model and dataset and the 95% confidence interval is displayed as the error bar.

$$
\begin{aligned}
E_{q(\phi)}[\log p(D|\phi)] &\approx \frac{1}{M}\sum_{m=1}^{M}\log p(D|\phi),\\
&\approx -\frac{Nl}{2}\log(2\pi) - \frac{1}{M}\sum_{m=1}^{M}\Bigg[\frac{N}{2}\log(|\Sigma_{\hat{y}}|)\\
&\quad + \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \mu_{\hat{y}}^{(m)})^{\top}(\Sigma_{\hat{y}}^{(m)})^{-1}(y^{(i)} - \mu_{\hat{y}}^{(m)})\Bigg],\\
&\approx -\frac{Nl}{2}\log(2\pi) - \frac{1}{M}\sum_{m=1}^{M}\Bigg[\sum_{k=1}^{l}\log\sigma_{\hat{y}_k}^{2}\\
&\quad + \frac{1}{2}\sum_{i=1}^{N}\left(y^{(i)} - \mu_{\hat{y}}^{(m)}\right)^{2}\mathrm{diag}(\sigma_{\hat{y}}^{2})^{-1}\Bigg].
\end{aligned}
\tag{11}
$$

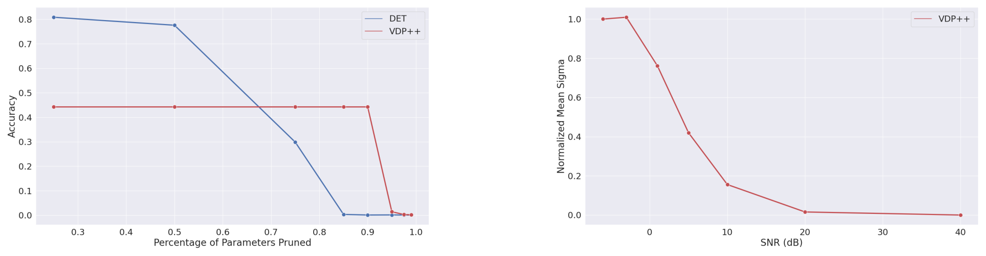
In Equation 11, $y^{(i)}$ is the true label of the i^{th} data point, N is the number of data points and M is the number of Monte Carlo samples needed to approximate the expectation by summation.

Recall that we have assumed the weight vectors and the elements of the bias vector to be uncorrelated, $\Sigma_{w_p w_q} = 0$ and $\sigma_{b_p b_q} = 0$ for $p \neq q$, where $p, q = 1, 2, ..., j$. It follows that $\Sigma_{\hat{y}}$ is a diagonal matrix with elements $\sigma_{\hat{y}}^2$. The log determinant of a diagonal matrix simplifies to the product of the elements that can be implemented as a sum of the log of the elements, $\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{k=1}^{l}\log\sigma_{\hat{y}_k}^2\right)$ to prevent numerical overflow. Additionally, the inverse of a diagonal matrix is equivalent to the reciprocal of each element, $\mathrm{diag}(\sigma_{\hat{y}}^2)^{-1}$.

$$
\mathrm{KL}[q(\phi)|p(\phi)] = -\frac{1}{2}\sum_{n=1}^{l}(j\log\sigma_{v_n}^2 - ||\mu_{v_n}||_F^2 - j\sigma_{v_n}^2). \tag{12}
$$

VDP++ for Convolutional Kernels: In practice, convolution operations are implemented as matrix multiplications. Therefore, no additional derivations are needed to propagate the first two moments through convolutional kernels. For max-pooling, we cannot take the maximum variance. Instead, we utilize the co-pooling operation. The co-pooling operation is the same for the first moment: the maximum of the means in the kernel is passed forward. For the variance, we keep only the elements of the variance that correspond to the maximum means [8].

VDP++ for Residual Connections: Residual connections are a module used in deep architectures to resolve the issue of vanishing gradients. These connections propagate parameters from previous



(a) (b)

Figure 5: Replication of the pruning experiments (a), and robustness experiments (b) on ImageNet-1k using a VDP-ViT-S/16 architecture. Due to the large hyperparameter selection space and compute time, we were unable to achieve a top-1 accuracy greater than 45%. Although the performance of these models are not high, the VDP retains its self-compressing and noise robustness properties.

layers in the neural network by concatenating the output of one layer to the input of the next layer [38]. The residual function is effectively a non-elementwise, non-linear function. Therefore, we can use the same formulation for Softmax as in Equation 9 [9].

$$\begin{aligned} \mu_{\mathbf{x}_{l+1}} &\approx \mu_{\mathbf{x}_l} + \mathcal{F}(\mu_{\mathbf{x}_l}), \\ \sigma^2_{\mathbf{x}_{l+1}} &\approx J^2 \sigma^2_{\mathbf{x}_l}. \end{aligned} \tag{13}$$

In Equation 13, J is the Jacobian of x_{l+1} with respect to x_l and $\mathcal{F}$ is the residual function [9].

VDP for Vision Transformers: A novel contribution in this work, we extend the VDP framework to Vision Transformers (ViTs). Given the simple nature of the ViT architecture, to implement models like ViT-S/16 and ViT-B/16 [39, 40], we need only implement the layer normalization operation [41] for VDP. Similar to the batch normalization formulation for VDP [9], layer normalization for VDP++ is given in Equation 14 where x is the input and y is the output of the layer.

$$\begin{aligned} \mu_y &= \frac{\mu_x - \mu_{LN}}{\sqrt{\text{Var}(\mu_x) + \epsilon}} \odot \gamma + \beta, \\ \sigma^2_y &= \left(\frac{\mu_{LN}}{\sqrt{\text{Var}(\mu_x) + \epsilon}} \right)^2 \odot \sigma^2_x. \end{aligned} \tag{14}$$

Validation and Robustness: To validate our approach, we will assess the VDP++ method on the MNIST dataset using various back-ends: a deterministic CNN, VDP [8], Bayes by Backprop (BBB) [7], and VDP++. On the CIFAR-10 dataset, we will evaluate the deterministic model and VDP++. For the MNIST dataset, we employ a LeNet architecture, while for CIFAR-10, we use ResNet-18. In order to determine whether any of the backends result in significantly different test accuracy, we will repeat training five times and perform a one-way Analysis of Variance (ANOVA) on the results. As our approach aims to simplify the original method while preserving its properties, we would like to show no significant differences in performance among the methods, i.e., $p > \alpha$.

To evaluate the robustness properties of VDP, we will replicate an experiment from the original implementation, which entails adding zero-mean Gaussian noise to the test set at varying magnitudes and examining the pattern of the normalized output variance [8]. We will use the Spearman rank-order correlation measure to compare the two implementations. Additionally, we will compare the maximum GPU memory allocation and training time for each formulation at a fixed batch size of 512 to benchmark each method's performance.

To examine the self-compression properties of VDP++ [19], we will evaluate the performance of each backend on the MNIST test set as each model undergoes global pruning by weight magnitude (L1 unstructured pruning).

Lastly, we will assess the explainability of saliency maps produced by the deterministic model and VDP++. The metrics we will utilize to evaluate the quality of the explanations are Infidelity and Sensitivity Max [37]. We will employ pair-wise t-tests to determine statistical significance, if any. For all statistical analyses, we choose $\alpha = 0.01$.

Scaling to Large Datasets: Remember that the primary limitation of using Bayesian neural networks lies in the original formulation's inability to scale to model sizes larger than ResNet-18. Overcoming this drawback would enable the application of BNNs to more complex problems and larger datasets, which is essential for advancing the state-of-the-art in machine learning. To that end, we will evaluate the performance of VDP++ on the ImageNet-1k dataset using Vision Transformers (ViTs). This analysis involves examining the ViT-B/16 [40] and ViT-S/16 [39] architectures. To our knowledge, this is the first analysis of BNNs on ImageNet-1k using the ViT architecture.

4 Results

Time-space Improvements: We began first by examining the advantages of our method in both time and space. On the MNIST dataset, Figure 1 shows a comparison of average epoch time and maximum GPU allocation for each method. The original VDP method [8] denoted "VDP Cov" takes nearly 10x longer to complete one epoch of training as well as 10x the GPU memory requirement as compared to traditional networks (DET) and Bayes-by-Backprop (BBB) [7]. Training on MNIST with a batch size of 512 requires nearly 40GB of GPU memory. Our proposed method, VDP++, reduces the average epoch time by about 4x over the prior method and reduces the GPU memory requirement by 10x.

Self-compression: The first of the inherent properties of VDP that we investigated was self-compression. We began by training 5 models from each method on MNIST (DET, BBB, VDP Cov, VDP++) and CIFAR-10 (DET, VDP++). A one-way ANOVA statistic was computed on the test statistics of each method and it was found that the performances were not significantly different from one another ($p > 0.01$). Next, we iteratively pruned each model using global L1 unstructured pruning and computed test statistics. Figure 2 shows the result of this procedure. The VDP Cov and VDP++ approaches perform similarly on MNIST (Figure 2a). However, VDP++ is able to prune $> 90\%$ of its parameters before the performance begins to drop significantly. This is verified by using a t-test of the statistics that compares $< 90\%$ pruning and$> 90\%$ pruning. This trend continues to hold for the CIFAR-10 experiments (Figure 2b).

Noise Robustness and Uncertainty Quantification: Figure 1c shows the result of adding zero-mean Gaussian noise to the test set of MNIST and examining the test accuracy at the highest level of noise (SNR of -6dB). Here the Bayesian models (BBB, VDP Cov, VDP++) all significantly outperform the deterministic network (DET). Pair-wise t-tests were performed on deterministic versus each Bayesian method at this level of noise and all p-values were significant ($p < 0.01$). Figure 3 shows the output variance as a function of SNR normalized by the -6dB value for both MNIST (Figure 3a) and CIFAR-10 (Figure 3b). When comparing the behavior of VDP-Cov and VDP++ in the presence of varying amount of noise (Figure 3a), it was found that there was no significant difference (Spearman correlation $p < 0.01$) between the methods.

Explanation Sensitivity: Due to the high regularization of the network, we postulated that explanations via saliency maps from Bayesian Networks would be less sensitive to perturbations. Figure 4 shows the average Sensitivity Max and Infidelity on MNIST (Figure 4a) and CIFAR-10 (Figure 4b) (lower is better). Pair-wise t-tests indicate significant differences between methods for all experiments ($p < 0.01$). The deterministic method narrowly outperformed VDP++ by Sensitivity Max. However, VDP++ significantly outperformed all methods by Infidelity.

ImageNet and VDP-ViT: To first validate our implementation of VDP-ViT, we trained a small ViT on MNIST and confirmed noise robustness using the variance vs SNR experiment. We then began parameter sweeps for ViT-B/16 and ViT-S/16 but ultimately decided on only optimizing ViT-S/16 due to long training times. The training was split across 6 Quadro RTX 8000 GPUs following the regime outlined by Beyer *et al.* [39]. Ultimately, we were only able to obtain a model with 45% top-1 training accuracy given the long training times and large parameter space to search. Figure 5 shows the same analyses as above using the underfitted ImageNet-1k model. Even in this state, the VDP++ model still retains noise robustness and self-compressing properties, remaining consistent with the smaller models. The results for explanation sensitivity were not consistent with our prior results most likely due to the low performance of the model.

5 Discussion

Our study aimed to investigate the benefits and properties of the proposed novel VDP++ method compared to the original VDP method and other traditional networks. The results show that VDP++ offers several key advantages, including significant time-space improvements, self-compression, noise robustness, and uncertainty quantification. The time-space improvements demonstrated by VDP++ enable more efficient training and reduced GPU memory requirements, which are crucial in large-scale applications. This addresses the primary limitations of the original VDP method, which required much longer training times and higher memory consumption.

In terms of self-compression and noise robustness, VDP++ performs similarly to the original VDP method. Additionally, we show that it is able to prune more than 90% of its parameters before performance significantly declines. This ability to maintain performance despite substantial pruning suggests that VDP++ learns more efficient representations and could lead to more compact models that may be beneficial for edge computing.

The evaluation of the explanations produced by the various backends revealed unexpected results. Our original hypothesis was that due to the averaging effect of BNNs, we would have expected to see an improvement in Sensitivity Max (measures the stability and robustness of the explanation with respect to small perturbations in the input data) over deterministic models. An improvement in infidelity would suggest that the explanation approximates the underlying model's decision-making process better in BNNs than in deterministic models. The difference in Sensitivity Max, while significant ($p < 0.01$), was quite small. According to the original work from which it was derived, this score can be reduced by modifying the formulation of the saliency map, while infidelity cannot [37]. This yields promising results for the use of BNNs in areas where interpretability and explainability are desirable.

During our testing, we found that the balance between the two terms of the loss function in Equation 10 had the largest effect on the convergence and subsequent properties of our models. Since the KL term of Equation 12 is largely dominated by matrix norms, in larger models especially, this term tends to dominate. When this happens, the model compresses itself at the cost of performance. Conversely, when the KL term is scaled down too much (i.e. $\mathbb{E}_{q(\phi)}[\log p(D|\phi)] \gg \text{KL}[q(\phi)|p(\phi)]$), the model overfits and the resulting models lack the inherent properties like self-compression and noise robustness, effectively becoming a deterministic model. To mitigate this behavior, we provided scaling terms to our hyperparameter optimizer and searched outright. This proved to be the fastest and most reliable method for training VDP++ models with high performance while retaining their inherent properties.

For classification problems such as ImageNet-1k, the Negative Log-Likelihood (NLL) term in the ELBO loss function became a bottleneck. In practice, the NLL term uses one-hot encoded labels. This led to vanishing gradients and low model performance. To alleviate this problem, we changed the NLL term to categorical cross-entropy for our ImageNet-1k experiments. We confirmed that this change did not negatively affect our prior results.

6 Limitations and Future Work

Our limitation in this study was a lack of computing resources. Given the increased parameter space we needed to search for our ImageNet-1k experiment, it became infeasible to expect high model performance in a short period of time. It is also worth noting that the original ViT formulation made use of the JFT-300M dataset for pretraining, which is not publicly available. We have substantially reduced the computational burden of the original VDP method while preserving its desirable properties. In tandem with our preliminary results on ImageNet-1k, scaling this approach to larger datasets is both possible and feasible given adequate computing resources and time. The robustness and improved explanatory capabilities of the VDP framework make it highly attractive for high-risk domains such as healthcare. Furthermore, the self-compressing properties render this technique suitable for applications with varying resource constraints, such as edge computing and large language models where computational resources are highly valuable.

References

[1] OpenAI, “Gpt-4 technical report,” 2023.

[2] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. B. Brown, D. Song, U. Erlingsson, *et al.*, “Extracting training data from large language models.,” in *USENIX Security Symposium*, vol. 6, 2021.

[3] J. Rissanen, “Stochastic complexity and modeling,” *The annals of statistics*, pp. 1080–1100, 1986.

[4] G. E. Box and G. C. Tiao, *Bayesian inference in statistical analysis*. John Wiley & Sons, 2011.

[5] D. J. MacKay, “The evidence framework applied to classification networks,” *Neural computation*, vol. 4, no. 5, pp. 720–736, 1992.

[6] G. E. Hinton and D. Van Camp, “Keeping the neural networks simple by minimizing the description length of the weights,” in *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13, 1993.

[7] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, “Weight uncertainty in neural network,” in *International conference on machine learning*, pp. 1613–1622, PMLR, 2015.

[8] D. Dera, G. Rasool, and N. Bouaynaya, “Extended variational inference for propagating uncertainty in convolutional neural networks,” in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2019.

[9] D. Dera, N. C. Bouaynaya, G. Rasool, R. Shterenberg, and H. M. Fathallah-Shaykh, “Premium-cnn: Propagating uncertainty towards robust convolutional neural networks,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 4669–4684, 2021.

[10] X. Xie, X. Liu, T. Lee, S. Hu, and L. Wang, “Blhuc: Bayesian learning of hidden unit contributions for deep neural network speaker adaptation,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5711–5715, IEEE, 2019.

[11] Y. Xiao and W. Y. Wang, “Quantifying uncertainties in natural language processing tasks,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 7322–7329, 2019.

[12] A. Bate, M. Lindquist, I. R. Edwards, S. Olsson, R. Orre, A. Lansner, and R. M. De Freitas, “A bayesian neural network method for adverse drug reaction signal generation,” *European journal of clinical pharmacology*, vol. 54, no. 4, pp. 315–321, 1998.

[13] Y. Kwon, J.-H. Won, B. J. Kim, and M. C. Paik, “Uncertainty quantification using bayesian neural networks in classification: Application to ischemic stroke lesion segmentation,” in *Medical Imaging with Deep Learning*, 2018.

[14] Y. Kwon, J.-H. Won, B. J. Kim, and M. C. Paik, “Uncertainty quantification using bayesian neural networks in classification: Application to biomedical image segmentation,” *Computational Statistics & Data Analysis*, vol. 142, p. 106816, 2020.

[15] D. J. MacKay, “A practical bayesian framework for backpropagation networks,” *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[16] R. M. Neal, *Bayesian learning for neural networks*, vol. 118. Springer Science & Business Media, 2012.

[17] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, “Variational inference: A review for statisticians,” *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[18] E. Hüllermeier and W. Waegeman, “Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods,” *Machine Learning*, vol. 110, no. 3, pp. 457–506, 2021.

[19] G. Carannante, D. Dera, G. Rasool, and N. C. Bouaynaya, “Self-compression in bayesian neural networks,” in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2020.

[20] J. M. Hernández-Lobato and R. Adams, “Probabilistic backpropagation for scalable learning of bayesian neural networks,” in *International conference on machine learning*, pp. 1861–1869, PMLR, 2015.

[21] A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,” 2009.

[22] Y. Gal and Z. Ghahramani, “Bayesian convolutional neural networks with bernoulli approximate variational inference,” *arXiv preprint arXiv:1506.02158*, 2015.

[23] E. Begoli, T. Bhattacharya, and D. Kusnezov, “The need for uncertainty quantification in machine-assisted medical decision making,” *Nature Machine Intelligence*, vol. 1, no. 1, pp. 20–23, 2019.

[24] A. D. Bedoya, M. E. Clement, M. Phelan, R. C. Steorts, C. O’Brien, and B. A. Goldstein, “Minimal impact of implemented early warning score and best practice alert for patient deterioration,” *Critical care medicine*, vol. 47, no. 1, p. 49, 2019.

[25] J. L. Guidi, K. Clark, M. T. Upton, H. Faust, C. A. Umscheid, M. B. Lane-Fall, M. E. Mikkelsen, W. D. Schweickert, C. A. Vanzandbergen, J. Betesh, *et al.*, “Clinician perception of the effectiveness of an automated early warning and response system for sepsis in an academic medical center,” *Annals of the American Thoracic Society*, vol. 12, no. 10, pp. 1514–1519, 2015.

[26] J. C. Ginestra, H. M. Giannini, W. D. Schweickert, L. Meadows, M. J. Lynch, K. Pavan, C. J. Chivers, M. Draugelis, P. J. Donnelly, B. D. Fuchs, *et al.*, “Clinician perception of a machine learning–based early warning system designed to predict severe sepsis and septic shock,” *Critical care medicine*, vol. 47, no. 11, pp. 1477–1484, 2019.

[27] M. Dewan, N. Muthu, E. Shelov, C. P. Bonafide, P. Brady, D. Davis, E. S. Kirkendall, D. Niles, R. M. Sutton, D. Traynor, *et al.*, “Performance of a clinical decision support tool to identify picu patients at high risk for clinical deterioration,” *Pediatric Critical Care Medicine*, 2019.

[28] A. Y. Foong, Y. Li, J. M. Hernández-Lobato, and R. E. Turner, “’in-between’uncertainty in bayesian neural networks,” *arXiv preprint arXiv:1906.11537*, 2019.

[29] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” *arXiv preprint arXiv:1312.6034*, 2013.

[30] D. Alvarez-Melis and T. S. Jaakkola, “On the robustness of interpretability methods,” *arXiv preprint arXiv:1806.08049*, 2018.

[31] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, “Sanity checks for saliency maps,” *Advances in neural information processing systems*, vol. 31, 2018.

[32] A. Ghorbani, A. Abid, and J. Zou, “Interpretation of neural networks is fragile,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, pp. 3681–3688, 2019.

[33] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, “Smoothgrad: removing noise by adding noise,” *arXiv preprint arXiv:1706.03825*, 2017.

[34] I. E. Nielsen, R. P. Ramachandran, N. Bouaynaya, H. M. Fathallah-Shaykh, and G. Rasool, “Evalattai: A holistic approach to evaluating attribution maps in robust and non-robust models,” *arXiv preprint arXiv:2303.08866*, 2023.

[35] I. Rish *et al.*, “An empirical study of the naive bayes classifier,” in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, pp. 41–46, 2001.

[36] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[37] C.-K. Yeh, C.-Y. Hsieh, A. Suggala, D. I. Inouye, and P. K. Ravikumar, “On the (in) fidelity and sensitivity of explanations,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[38] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[39] L. Beyer, X. Zhai, and A. Kolesnikov, “Better plain vit baselines for imagenet-1k,” *arXiv preprint arXiv:2205.01580*, 2022.

[40] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[41] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.