Uncertainty Quantification with Deep Learning through Variational Inference with applications to Synthetic Aperture Sonar

Marko Orescanin[1], Brian Harrington[2], Derek Olson[3], Marc Geilhufe[4], Roy E. Hansen[5], and Narada Warakagoda[6]

[1,2,3]Naval Postgraduate School, Monterey, CA, USA
[4,5,6]Norwegian Defence Research Establishment (FFI), 2027 Kjeller, Norway

Marko Orescanin, Department of Computer Sciences, Naval Postgraduate School, Monterey, CA, 93943 USA (e-mail: marko.orescanin@nps.edu)

Abstract: *Deep learning (DL) has gained popularity within the active sonar community due to the ability to learn complex non-linear relationships between the input features and the labels through a data driven approach. The DL models have led to significant improvements in automatic target recognition and seafloor texture understanding with synthetic aperture sonar (SAS). Most of the DL models reported in literature are deterministic and do not provide estimates of uncertainty of their predictions limiting the utility for the downstream tasks such as ATR and change detection. In this work, we demonstrate the ability to quantify uncertainty in deep learning predictions by utilizing Bayesian Neural Networks, in this case via variational inference. We introduce and compare several state-of-the art Bayesian methods (including variational inference) on the task of classifying imaging artifacts in SAS. We conduct this on a novel dataset developed for this classification task through introduction of physical perturbations in the image formation stage, namely : 1) sound speed error of 40 m/s, 2) navigation error through perturbation in yaw of 0.35° and 3) Gaussian noise over the imaging channels prior to pulse compression (lowering the average image SNR to 5 dB). Overall, we demonstrate that our best model, a mean-field variational inference via flipout ResNet architecture, achieves 92% accuracy with calibrated uncertainty. By rejecting 10% of the data with highest uncertainty we achieve additional 4% improvement in accuracy.*

Keywords: *Synthetic Aperture Sonar, Bayesian Deep Learning, Machine Learning, Active Sonar, Imaging Artifacts*

1. INTRODUCTION

Naval mines pose a significant threat to both civilian and military vessels. To mitigate this risk, current mine countermeasure operations rely on underwater unmanned vehicles (UUVs) to map the ocean floor with acoustic sensors, such as synthetic aperture sonar (SAS) and incorporate automatic target recognition (ATR) to detect and classify mine-like objects.

SAS image quality is often affected by the dynamic nature of the ocean environment and the imprecision of navigational instruments. Additionally, high-quality labeled data is scarce, with much of the data stored in databases being of unknown quality and unsuitable for algorithmic consumption. In this work, we introduce a novel approach to image quality assessment using Bayesian neural network classification, along with a new dataset specifically designed for this application.

Deep learning (DL) models, built using neural networks, have seen increasing use in ATR systems in recent years, thanks to their ability to model complex, non-linear relationships between input features and labels. However, most DL models reported in the literature are deterministic, providing limited utility for downstream tasks such as ATR and change detection, as they do not offer estimates of uncertainty.

To overcome this limitation, we propose the use of Bayesian neural networks with variational inference (VI), which introduces parameters for uncertainty in addition to the usual parameters used for classification. By analyzing uncertainty, operators can identify areas for improvement and prioritize data samples that require human verification. Furthermore, by monitoring trends in uncertainty and model correctness, Bayesian neural networks can become more explainable than deterministic neural networks, increasing reliability and trust in the system.

Our proposed method was evaluated using physically perturbed SAS data, using three different image artifacts, and achieved high accuracy in classification. By incorporating uncertainty estimates, we were able to identify the most problematic areas of the dataset, leading to the development of new labeling strategies. Furthermore, our approach provides an interpretable and explainable framework, allowing for increased trust and confidence in the system.

2. BACKGROUND

One of the major limitations of deterministic machine learning techniques is overconfidence in inference. From a practical perspective, regardless of whether these models were trained on a specific class distribution or not, they will predict a class for out-of-distribution samples (class not present in initial training). The user has no indication from the model that it has encountered an out-of-distribution sample and that a result should not be trusted. In contrast, Bayesian machine learning techniques provide an estimate of uncertainty, along with a classification of the samples presented on the input of the model.

Bayesian neural networks incorporate an *a priori* probability distribution, $p(\omega)$, over the weights $p(\omega \mid \mathcal{D})$, $\omega \in \Omega$, where Ω represents the set of all weights (including biases) of a neural network architecture and $\mathcal{D}$ is the dataset. A neural network model, $\mathcal{M}$, is parameterized by the neural network weights ω, and a supervised learning dataset is $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in \mathcal{R}^d$ and $y_i \in \{1, 2, \ldots, C\}$ are the i^{th} input and label respectively. N denotes sample size, d is the dimension of the input feature space and C is the number of classes. The optimization problem to solve with Bayesian inference is:

$$p\left(y^{*}|x^{*},\mathcal{D}\right)=\int_{\Omega}p\left(y^{*}|x^{*},\omega\right)p\left(\omega|\mathcal{D}\right)d\omega, \tag{1}$$

where x^* is a new input feature vector (e.g. test data), and y^* is the model prediction or inference [1].

The solution to Eq. 1 is intractable because it would require integration over the infinite feature space [2, 3]. VI focuses on an optimization technique to approximate $p\left(\omega|\mathcal{D}\right)$, the posterior, by finding an approximation, $q_\theta(\omega)\approx p\left(\omega|\mathcal{D}\right)$ indexed by a variational parameter θ via an optimization process.

We therefore try to minimize the Kullback-Leibler divergence between $q_\theta(\omega)$ and $p\left(\omega|\mathcal{D}\right)$. It can be shown that the resulting loss function for VI becomes the objective of minimizing the negative ELBO loss [2]:

$$\mathcal{L}_q=\mathrm{KL}(q_\theta(\omega)\ ||\ p(\omega))-\mathbb{E}_q\left[\log p\left(\mathcal{D}|\omega\right)\right], \tag{2}$$

where $\mathbb{E}$ represents the expected value and KL is the Kullback–Leibler (KL) divergence. The $\mathbb{E}_q\left[\log p\left(\mathcal{D}|\omega\right)\right]$ term represents the negative log likelihood loss and the KL term is a penalty imposed based on distance to the posterior. For this work, a factorized Gaussian distribution is assumed for each weight, characterized with the mean and standard deviation per weight.

During inference neural network weight's distributions are sampled several times to produce a prediction. Inferences are combined through Bayesian model averaging [1]:

$$\bar{p}(y^*\mid x^*)=E_{q(\theta)}p(y^*\mid x^*)\approxeq\frac{1}{T}\sum_{t=1}^{T}p(y^*\mid x^*,\omega_t), \tag{3}$$

where $\omega_t\sim q_\theta(\omega)$ to yield a final prediction. However, analysis of the ensemble of inferences can produce important information about the uncertainty of the model. Uncertainty can be quantified by predictive entropy which measures the average amount of information contained in the predictive distribution and is given by:

$$H_p(y^*\mid x^*)=-\sum_{c}\bar{p}(y^*=c\mid x^*)\frac{\log\bar{p}(y^*=c\mid x^*)}{\log C} \tag{4}$$

where C represents all possible classes [1].

There are many different Bayesian methods and only a few were chosen for analysis and comparison. First, we used a deterministic model base as a reference solution. For the Bayesian methods we showcase performance of: MC Dropout [3], flipout [4], reparametrization [2], and Laplace approximation [5]. These methods were selected for their ease of implementation, popularity in literature, and their unique features.

MC Dropout is a technique to approximate distributions over a model's weights, ω, through the use of common regularization techniques: dropout and L2 regularization [3]. Intuitively, dropout introduces a Bernoulli distribution over weights with probability p_{drop} that was set in all of our experiments to 0.2. Unlike deterministic models, dropout layers remain active for inference. The use of a Bernoulli variable over the dropout probability creates a variance of inferences that can be averaged using Eq. 3 to predict the class [3].

The Laplace approximation [5] uses a Gaussian distribution for a trained deterministic model's weights, θ_0. The mean of this distribution is equal to the vector of the deterministic weights itself and the variance of the distribution is approximated using a second-order Taylor series approximation of the log posterior, $log(p(\theta|D))$, around θ_0. Assuming a Gaussian distribution over weights, the covariance matrix Σ can be approximated through Taylor series expansion as [5]:

$$\Sigma = f''(\theta_0)^{-1}. \tag{5}$$

where f is the neural network transfer function. Practically, the curvature of the weight distribution is equivalent to the inverse of the average Hessian matrix. The form of the Gaussian is $N(\theta_0, \bar{H}^{-1})$ where $\bar{H}$ is the average Hessian matrix. The Hessian is challenging to compute, so it is often derived from the Fisher matrix, F. The Fisher matrix is the expectation of the squared gradients and the Hessian can be approximated with the diagonal Fisher matrix [5] . Once the Gaussian for θ is formed, we use Bayesian Model Averaging (Eq. 3) to compute the uncertainties [5].

Rather than sampling from each weight, the local reparameterization technique [2] samples from the pre-activation neurons rather than the neurons themselves. This reduces the variance of the weight distributions and significantly reduces the computational power required. Assuming the weights are a factorized Gaussian, then the posterior of the activation functions are also a factorized Gaussian [2]. We can sample directly from the activations by using their implied Gaussian distribution and utilize Eq. 3.

To create the Gaussian weight term, we add ϵ such that $\epsilon = N(0, 1)$ as below:

$$w_{i,j} = \mu_{i,j} + \sigma_{i,j}\epsilon_{i,j}. \tag{6}$$

This creates a Gaussian distribution over the weights and can be sampled and modeled with uncertainty. Assuming an input feature matrix A and an activation matrix $B = WA$, where W is the corresponding weight matrix, we can see

$$q_\theta(w_{i,j}) = N(\mu_{i,j}, \sigma_{i,j}^2), \ \ \forall w_{i,j} \in W \longrightarrow q_\theta(b_{m,j}|A) = N(\gamma_{m,j}, \delta_{m,j}), \tag{7}$$

where $\gamma_{m,j} = \sum_{i=1}^{L} a_{m,i}\mu_{i,j}$ and $\delta_{m.j} = \sum_{i=1}^{L} a_{m,i}^2\sigma_{i,j}^2$ with $a_{m,i}$ and $b_{m,j}$ being the elements of the matrix A and B respectively. From this, we can sample directly from the activations by using their implied Gaussian distribution: $b_{m,j} = \delta_{m,j} + \sqrt{\gamma_{m,j}}\zeta_{m,j}$ where $\zeta = N(0, 1)$ [6].

This is a more computationally efficient route than sampling from each of the weights, but it also leads to a lower variance. This is because there is only one sampled noise term (the neuron activation) vs a different random noise term for each of the weights influencing the gradient [2]. Because we are taking $\zeta_{m,j}$ from a normal distribution, we can use Bayesian model averaging as shown in Eq. 3 to obtain probabilistic results from the model.

The local reparameterization trick is also used in flipout, but without the pseudo-random variable technique used in flipout, the resulting Hessian matrix from the summation of all the neuron values tends to be highly correlated. This can lead to poor optimization and slower training.

When adding stochasticity to a model's weights or activation function, it is often computationally easier to apply the same perturbation to all weights in a mini-batch. This is convenient, and necessary for deeper models, but it violates one of the key assumptions that each weight is uncorrelated from each other. Flipout attempts to resolve this issue by decorrelating the perturbations in the weight values in each mini-batch [7], and this effectively leads to a much lower variance across averages of a mini-batch in optimization. In this work we use Gaussian distribution assumption over weights of the neural network for both reparametrization and flipout models. The flipout technique makes two key assumptions: 1) the perturbations of each weight are independent, and 2) The perturbation distribution is symmetric around zero.

For a distribution $q(\theta)$ that follows these assumptions, we can assign the base perturbation rate as $\Delta\hat{W} = \sigma_{i,j}\epsilon_{i,j}$ from Eq. 6. As a result of these assumptions, we can change the sign of

$\Delta\hat{W}$ without changing the loss gradients. This allows us to compute a new perturbation rate, ΔW_n where r_n and s_n are random vectors sampled uniformly from ± 1 [7].

$$\Delta W_n = \Delta\hat{W} * r_n s_n^T \tag{8}$$

By using Eq. 8 to alter the individual weights of a neural network within a mini-batch, a much lower variance can be achieved across averages of a mini-batch because the noise is not all in the same [7]. Because flipout relies on matrix multiplication, it incurs additional computational cost, but this operation can be performed in parallel since each matrix multiplication is independent [7].

For this project, we used a ResNet 20 architecture [8]. Applications have demonstrated that both ResNet architectures [8] and custom CNN models can work well in Bayesian configurations [1] with both MC Dropout and flipout approaches. Flipout convolution and reparametrization convolution layers were implemented utilizing the TensorFlow Probability library [9]. We constructed our MC Dropout models by placing dropout layers after each activation layer within the residual block in the original deterministic model.

3. EXPERIMENTS

The classification task consisted of recognizing whether an image was of good quality (no artefact), or was corrupted by one of three three types of errors introduced during beamforming, which are described below. For this classification task we developed a novel dataset through introduction of physical perturbations in the image formation stage. The data was collected by FFI's HUGIN-HUS AUV [10], equipped with a HISAS interferometric SAS. To ensure diversity of the collected data, experiments were conducted over five different areas with varying terrain profiles and varying unknown seafloor types. The SAS center frequency in the experiments was 100 kHz with a bandwidth of 30 kHz, vehicle altitude was between 15 and 30 meters and theoretical resolution was approximately 3.5$\times$3.5cm. The SAS images were constructed following the approach outlined in [11] using the back-projection algorithm taking into account sidescan bathymetry and navigation correction.

In order to introduce realistic artifacts in SAS imagery, we introduced physical perturbations in the back-projection stage of image formation to produce a degraded image. In this work we considered the following perturbations: 1) sound speed error of 40 m/s, 2) navigation error through perturbation in yaw of 0.35° and 3) Gaussian noise over the imaging channels prior to pulse compression (lowering the average image SNR to 5 dB). A description of how these perturbations impact image quality can be found in [12].

In this study we utilized a total of 25 SAS scenes. The data creation strategy is as follows: 1) Break SAS image into 1500x1500 non-overlapping regions. 2) Assign the first two regions as test, assign 3^{rd} and 4^{th} region as validation and assign the rest of the regions to train data. 3) Break test and validation sub-regions into 300$\times$300 non-overlapping pixel patches. 4) Break training region per image into 300$\times$300 patches such that they are 50% overlapped in each dimension to maximize the amount of data available for training. 5) Combine all of the test, validation and training patches into test, validation and training datasets. Overall, such an approach produced a training dataset per strategy of 84k samples and 5k validation and testing samples each.

For a multi-class classification task, we used accuracy (Acc) and F_1-score as the main metrics. For Bayesian deep learning models the commonly used model selection metric is the mean negative log likelihood (MNLL):

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \quad F_1 = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec}, \quad MNLL = -\frac{1}{N} \sum \log(p(x_c)) \tag{9}$$

where *TP* and *TN* are true-positives and true-negatives, *FP* and *FN* are false-positives and false-negatives, N is the number of samples in the test dataset, and $p(x_c)$ is the probability the model assigns to the true class label. The MNLL is used for Bayesian model selection and provides an indication of how close a trained model is to the true (underlying) distribution.

4. RESULTS

We show performance results of trained models in Table 1 in terms of Accuracy and MNLL. If we use the deterministic model as a baseline performance reference, the model has achieved good skill on the test dataset as quantified by 90.2% accuracy. Relative to the deterministic model, Bayesian model architectures vary $\pm$3% in accuracy where the lowest performing model was Laplace with MC Dropout (accuracy of 87.1%) and the best performing model was flipout (accuracy of 92.7%). After analyzing the classification report results, Laplace approximation simply creates a distribution around the weights of an already trained model and, unsurprisingly, closely aligned with the results of those respective models used for the approximation. Additionally, flipout is the best performing model in terms of MNLL with a value of 0.245.

Bayesian Method	**Acc**	**MNLL**
Deterministic	90.2%	N/A
MC Dropout	87.2%	0.433
Flipout	92.7%	0.245
Reparameterization	87.1%	0.561
Laplace Approximation	90.2%	0.324
Laplace with MC Dropout	87.1 %	0.435

Table 1: Bayesian Method Results.

Based on the results of that study and effectiveness of the methods, flipout was the best representation of Bayesian methods, as it directly samples from the activation weights, and was the highest performing model. The primary benefit to using Bayesian techniques is the inclusion of uncertainty quantification analysis. In addition to comparing an MNLL metric as an indicator of how close a model is to the true underlying data distribution, Bayesian models can also be compared with respect to the quality of the estimated uncertainty by analyzing an uncertainty calibration graph. We illustrate uncertainty calibration graphs for some of the evaluated models in Fig. 1, where we evaluate how the F_1 score changes as the most uncertain data is removed (filtered) from the test dataset in evaluations. On the right of the graph, if all data from the test set is kept, accuracy is as reported in Table 1. If uncertain test samples are removed such that all entropy values are below 0.50, performance increases significantly. The change to F_1 drops off rapidly as entropy approaches zero. The ideal case, or the most calibrated uncertainty graph, would be the curve that hugs the upper right corner; that is data percentage remains high even as entropy drops, indicating fewer uncertain data samples. Under this selection criteria flipout is again the best performing method to develop Bayesian models for this application.

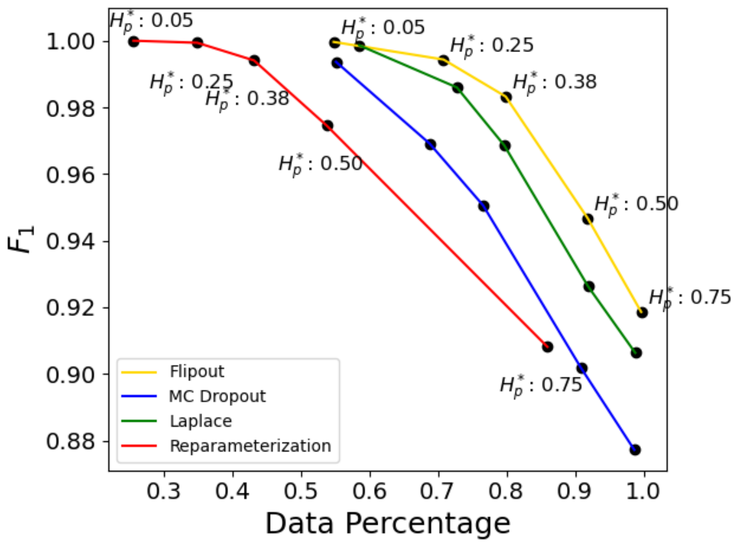

Figure 1: Calibrated Uncertainty

5. CONCLUSION

Assessing the quality of SAS images offers a distinct advantage of identifying the source of a poor performing classification system. When sensors or systems begin to fail, they may produce artifacts in images that go unnoticed by the human eye but can be identified by a machine learning algorithm. With the uniquely developed dataset in this work we introduced this new machine learning task for SAS and establish feasibility of such an approach and a baseline performance, the results of which are summarized in Table 1. We demonstrated that we can obtain high quality uncertainty estimates as quantified by the uncertainty calibration graph presented in Fig. 1.

The findings from this study here have the potential to make a significant impact on the field of autonomous monitoring of SAS systems in a production setting. By providing an uncertainty estimate, it becomes possible to have a more granular analysis of the model performance, which in turn leads to an increase in the interpretability and trustworthiness of the model predictions.

6. ACKNOWLEDGEMENTS

This research was supported by grants from the Office of Naval Research (N-00014-22-WX-01861). The authors thank FFI's HUGIN-HUS operator group for collecting the data used in this study.

REFERENCES

[1] B. Beckler, A. Pfau, M. Orescanin, S. Atchley, N. Villemez, J. E. Joseph, C. W. Miller, and T. Margolina, "Multilabel classification of heterogeneous underwater soundscapes with bayesian deep learning," *IEEE Journal of Oceanic Engineering*, vol. 47, no. 4, pp.

1143–1154, 2022.

[2] D. P. Kingma, T. Salimans, and M. Welling, “Variational dropout and the local reparameterization trick,” in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015.

[3] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059. [Online]. Available: https://proceedings.mlr.press/v48/gal16.html

[4] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. B. Grosse, “Flipout: Efficient pseudo-independent weight perturbations on mini-batches,” in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rJNpifWAb

[5] H. Ritter, A. Botev, and D. Barber, “A scalable laplace approximation for neural networks,” in *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, vol. 6. International Conference on Representation Learning, 2018.

[6] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[7] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, “Flipout: Efficient pseudo-independent weight perturbations on mini-batches,” 2018. [Online]. Available: https://arxiv.org/abs/1803.04386

[8] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[9] J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, and R. A. Saurous, “Tensorflow distributions,” *arXiv preprint arXiv:1711.10604*, 2017.

[10] T. O. Sæbø, B. Langli, H. J. Callow, E. O. Hammerstad, and R. E. Hansen, “Bathymetric Capabilities of the HISAS Interferometric Synthetic Aperture Sonar,” in *Proceedings of Oceans 2007 MTS/IEEE*, Vancouver, Canada, October 2007.

[11] R. E. Hansen, H. J. Callow, T. O. Sæbø, and S. A. V. Synnes, “Challenges in Seafloor Imaging and Mapping with Synthetic Aperture Sonar,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 10, p. 3677–3687, October 2011.

[12] D. A. Cook and D. C. Brown, “Analysis of phase error effects on stripmap SAS,” *IEEE Journal of Oceanic Engineering*, vol. 34, no. 3, pp. 250–261, July 2009.