Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Uncertainty quantification in neural network classifiers—A local linear approach[☆]

Magnus Malmström [a,c,*], Isaac Skog [b], Daniel Axehill [a], Fredrik Gustafsson [a]

[a] Linköping University, Linköping, Sweden
[b] Uppsala University, Uppsala, Sweden
[c] Swedish Defence Research Agency (FOI), Linköping, Sweden

ARTICLE INFO

Article history:
Received 6 March 2023
Received in revised form 4 September 2023
Accepted 19 December 2023
Available online xxxx

Keywords:
Neural networks
Uncertainty descriptions
Information and sensor fusion
Identification and model reduction

ABSTRACT

Classifiers based on neural networks (NN) often lack a measure of uncertainty in the predicted class. We propose a method to estimate the probability mass function (PMF) of the different classes, as well as the covariance of the estimated PMF. First, a local linear approach is used during the training phase to recursively compute the covariance of the parameters in the NN. Secondly, in the classification phase, another local linear approach is used to propagate the covariance of the learned NN parameters to the uncertainty in the output of the last layer of the NN. This allows for an efficient Monte Carlo (MC) approach for; (i) estimating the PMF; (ii) calculating the covariance of the estimated PMF; and (iii) proper risk assessment and fusion of multiple classifiers. Two classical image classification tasks, i.e., MNIST, and CFAR10, are used to demonstrate the efficiency of the proposed method.

1. Introduction

In this paper, the problem of quantifying the uncertainty in the predictions from a neural network (NN) is studied. The uncertainty in the prediction stems from three different sources: errors caused by the optimization algorithm that is used to train the NN, errors in the data (aleatoric uncertainty), and errors in the model (epistemic uncertainty). In this paper, the focus is on uncertainty from the two latter sources.

In numerous applications, e.g., image recognition (Krizhevsky, Sutskever, & Hinton, 2012), and various control tasks (Karlsson & Hendeby, 2021; Li et al., 2017), NNs have shown high performance. Despite their high performance, the use of NNs in safety-critical applications is limited (Grigorescu, Trasnea, Cocias, & Macesanu, 2020; Paleyes, Urma, & Lawrence, 2020). It is partly a consequence of the fact that their predictions usually do not come with any measure of certainty of the prediction, which is crucial to have in a decision-making process to know to what degree the prediction can be trusted. In particular, this need was highlighted in the fatal Uber accident in 2018 where the lack of reliable classifications of surrounding objects played a role in the development of events that eventually led to the accident (NTSB, 2018). Moreover, the quantified measure of uncertainty can be used to detect and remove outliers in the data, and without knowledge about the uncertainty it is not possible to fuse the prediction from the NN with information from other sensors.

The problem to quantify the uncertainty in the prediction of NNs has lately gained increasing attention, and numerous methods to calculate the uncertainty have been suggested (D'Amour et al., 2020; Ghahramani, 2015; Lin, Clark, Trigoni, & Roberts, 2022; Ovadia et al., 2019; Patel & Waslander, 2022). For a survey of methods see Gawlikowski et al. (2023). The methods suggested in the literature can broadly be divided into one out of two categories. One category is based on creating an ensemble of predictions from which the uncertainty in the prediction is computed (Ayhan & Berens, 2018; Gal & Ghahramani, 2016; Ilg et al., 2018; Lakshminarayanan, Pritzel, & Blundell, 2017; Maddox, Izmailov, Garipov, Vetrov, & Wilson, 2019; Osawa et al., 2019; Teye, Azizpour, & Smith, 2018). In the other category, the NN structure is extended and the NN is trained to learn its own uncertainty (Blundell, Cornebise, Kavukcuoglu, & Wierstra, 2015; Charpentier, Zügner, & Günnemann, 2020; Gustafsson, Danelljan, Bhat, & Schön, 2020; Izmailov, Nicholson, Lotfi, & Wilson, 2021; Kendall & Gal, 2017). Concerning the first category, it has for example been suggested to create an ensemble by training multiple NNs, from whose predictions the uncertainty

[☆] This work is supported by Sweden's innovation agency, Vinnova, through project iQDeep (project number 2018-02700). Funding was also provided from the Swedish Research Council for the project Scalable Kalman Filters. The material in this paper was presented at The 25t IEEE International Conference on Information Fusion (FUSION), July 4–7, 2022, Linköping, Sweden. This paper was recommended for publication in revised form by Associate Editor Dario Piga under the direction of Editor Alessandro Chiuso.

* Corresponding author.
E-mail addresses: magnus.malmstrom@liu.se (M. Malmström), isaac.skog@angstrom.uu.se (I. Skog), daniel.axehill@liu.se (D. Axehill), fredrik.gustafsson@liu.se (F. Gustafsson).

https://doi.org/10.1016/j.automatica.2024.111563

is computed by Lakshminarayanan et al. (2017). Since training a single NN is often computationally expensive, this method has high computational complexity. In practice, it is only feasible from a computational perspective to create small ensembles. To decrease the computational complexity, it was in Gal and Ghahramani (2016) and Teye et al. (2018) suggested to use already existing regularization techniques to sample values of the parameters of the NN to create these ensembles. Another method to create ensembles is by sampling values of the parameters during the last part of the training phase (Maddox et al., 2019; Osawa et al., 2019). So-called test-time data augmentation methods have also been suggested to do perturbation on the test data to create an ensemble of predictions (Ayhan & Berens, 2018). Even though the methods in Ayhan and Berens (2018), Gal and Ghahramani (2016), Maddox et al. (2019), Osawa et al. (2019) and Teye et al. (2018) do not need multiple models to be trained they require multiple forward passes. Furthermore, they require specially tailored training algorithms and carefully constructed structures of the NN.

Another limitation of methods relying on creating ensembles is that they have trouble representing the uncertainty caused by the bias in the prediction from a model mismatch. The bias can be caused by an insufficiently flexible model, which could be a result of too high regularization or too low model order. The problem can be solved by NNs from the second category, i.e., where the structure of the NN is extended such that it learns its own uncertainty in the prediction. However, this requires a more intricate NN structure with tailored loss functions (Blundell et al., 2015; Charpentier et al., 2020; Gustafsson et al., 2020; Kendall & Gal, 2017). As a consequence, the training becomes more complex and computationally expensive. It also makes the methods sensitive to errors caused by the training algorithm, which are not possible to learn. Furthermore, there is also a need for more data to train complex model structures.

In this paper, we address the two limitations of the aforementioned methods using classical local approximations from the area of system identification (Ljung, 1999) based on a linearization of the model. Where the system identification methodology works well for static systems which NNs are an example of Avant and Morgansen (2023) and Shen, Varshney, and Zhu (2014). We will refer to this as the *delta method* (Liero & Zwanzig, 2011; Malmström, 2021; Malmström, Skog, Axehill, & Gustafsson, 2022). For regression tasks, the delta method has previously been used to quantify the uncertainty in the prediction of NNs, see e.g., Deng, Zhou, and Zhu (2022), Hwang and Ding (1997), Immer, Korzepa, and Bauer (2021), Liero and Zwanzig (2011) and Malmström (2021), and extended to classification tasks in Malmström et al. (2022).

2. Problem formulation and contributions

Consider the problem of learning a classifier from the training data set

$$\mathcal{T} \triangleq \{y_n, x_n\}_{n=1}^{N} \tag{1}$$

Here $y_n \in \{1, \dots, M\}$ is the class labels and $x_n \in \mathbb{R}^{n_x}$ is the input data of size n_x, e.g., pixels in an image. From a statistical point of view, the learning of the classifier can be seen as a system identification problem where a model $f(x;\theta)$ that predicts the conditional probability mass function (PMF) $p(y|x)$ of a categorical distribution, are to be identified. That is, the probability for $y = m$ given the input x is modeled as

$$p(y = m|x;\theta) = f_m(x;\theta), \quad m = 1, \dots, M. \tag{2}$$

Here $\theta \in \mathbb{R}^{n_\theta}$ denotes the n_θ-dimensional parameter vector that parameterizes the model. Further, the subscript m denotes the m'th element of the vector-valued output of the function. To ensure that the model $f(x;\theta)$ fulfills the properties associated with a PMF, i.e., $f_m(x;\theta) \geq 0 \ \forall m$ and $\sum_m f_m(x;\theta) = 1$, it is typically structured as

$$f(x;\theta) = \text{softmax}\left(g(x;\theta)\right), \tag{3a}$$

$$\text{softmax}(z) \triangleq \frac{1}{\sum_{m=1}^{M} e^{z_m}} \begin{bmatrix} e^{z_1} & \dots & e^{z_M} \end{bmatrix}^\top \tag{3b}$$

and $g(x;\theta)$ describes the underlying model of the classifier. For example, if $g(x;\theta)$ is given by an NN the parameters θ consist of all the weights and biases in the NN. In the case $g_m(x;\theta) = \theta^\top \phi_m(x)$, where $\phi_m(x)$ denotes, a possible nonlinear, transformation of the input x, then the model in (3a) becomes a standard multinomial logistic regression model (Baggio, Carè, Scampicchio, & Pillonetto, 2022; Lindholm, Wahlström, Lindsten, & Schön, 2022). Furthermore, if the transformation $\phi_m(x)$ is chosen randomly, the model becomes similar to the one used in extreme learning machine classifiers (Huang, Zhu, & Siew, 2006).

2.1. Parameter estimation

For most NN the number of model parameters $n_\theta > N$ and the model parameters θ cannot be uniquely identified from the training data $\mathcal{T}$ without some regularization or prior information regarding the parameters. Let $p(\theta)$ denote the prior for the model parameters. The maximum a posteriori estimate of the model parameters is then given by

$$\hat{\theta}_N = \arg\max_{\theta} p(\theta|\mathcal{T}) = \arg\max_{\theta} L_N(\theta) + \ln p(\theta), \tag{4}$$

where $p(\theta|\mathcal{T})$ denotes the a posteriori distribution of the parameters and

$$L_N(\theta) = \sum_{n=1}^{N} \ln f_{y_n}(x_n;\theta) \tag{5}$$

denotes the cross-entropy likelihood function (Lindholm et al., 2022). Here y_n is used as an index operator for the subscript m of $f_m(x;\theta)$.

2.2. Prediction and classification

Once the classifier has been learned, i.e., a parameter estimate $\hat{\theta}_N$ has been computed, then for a new input data point $x^\star$ the PMF can be predicted as

$$\hat{p}(y^\star = m|x^\star;\hat{\theta}_N) = f_m(x^\star;\hat{\theta}_N), \quad m = 1, \dots, M \tag{6}$$

and the most likely class can be found as $\hat{y}^\star = \arg\max_m f_m(x^\star;\hat{\theta}_N)$. Note that, the full PMF estimate $f(x;\hat{\theta}_N)$ is needed both for temporal fusion using several inputs from the same class and fusion over different classifiers. Furthermore, even small probabilities can pose a large risk, e.g., there might be a pedestrian in front of a car even if another harmless object is more likely according to the classifier. Hence, it is important that the prediction $\hat{p}(y^\star = m|x^\star;\hat{\theta}_N)$ is accurate. However, it is well known that due to, among other things, uncertainties in the parameter estimates $\hat{\theta}_N$ the disagreement between true and estimated PMF may be significant. Therefore, methods to calibrate the prediction $\hat{p}(y^\star|x^\star;\hat{\theta}_N)$ such that it better matches $p(y^\star|x^\star)$ has been developed.

2.3. Temperature scaling

One of the most commonly used methods to calibrate the predicted PMF is called temperature scaling (Guo, Pleiss, Sun, & Weinberger, 2017). In temperature scaling, $g(x;\theta)$ is scaled by a scalar quantity T before the normalization by the softmax operator. With a slight abuse of notation, introduce

$$f(x^\star; \hat{\theta}_N, T) = \text{softmax}\left(g(x^\star; \hat{\theta}_N)/T\right). \tag{7}$$

Using the temperature scaling parameter T, the variations between the components (classes) in the predicted PMF can be enhanced or reduced. When $T \to 0$, then $f(x^\star; \hat{\theta}_N, T) \to \vec{e}_i$, where $\vec{e}_i$ denotes the i'th standard basis vector, thereby indicating that input $x^\star_n$ with total certainty belongs to class i. Similarly, when $T \to \infty$, then $f_m(x^\star; \hat{\theta}_N, T) \to 1/M \ \forall m$, thereby indicating that input $x^\star_n$ is equally probable to belong to any of the classes. Noteworthy is that the temperature scaling is typically done after the parameters θ have been estimated, where it is estimated using validation data. For notational brevity, the dependency on the temperature scaling parameter T will only be explicitly stated when temperature scaling is considered.

2.4. Marginalization of parameter uncertainties

A more theoretically sound approach to take the uncertainties in the parameter estimate into account is via marginalization of the PMF with respect to the parameter distribution. That is, an estimate of the PMF and its covariance are calculated as

$$f(x^\star|\mathcal{T}) \triangleq \int_\theta f(x^\star; \theta) p\big(\theta|\mathcal{T}\big) d\theta \tag{8a}$$

$$P^f \triangleq \int_\theta \big(f(x^\star; \theta) - f(x^\star|\mathcal{T})\big)\big(\cdot\big)^\top p\big(\theta|\mathcal{T}\big) d\theta \tag{8b}$$

From hereon, $(x)(\cdot)^\top$ is used as a shorthand notation for $xx^\top$. The integral in (8a) is generally intractable, but can be approximated by Monte Carlo (MC) sampling as

$$\theta^{(k)} \sim p\big(\theta|\mathcal{T}\big), \quad k = 1, 2, \ldots, K, \tag{9a}$$

$$\hat{f}(x^\star|\mathcal{T}) = \frac{1}{K}\sum_{k=1}^{K} f(x^\star; \theta^{(k)}) \tag{9b}$$

$$\hat{P}^f = \frac{1}{K}\sum_{k=1}^{K}\big(f(x^\star; \theta^{(k)}) - \hat{f}(x^\star|\mathcal{T})\big)\big(\cdot\big)^\top. \tag{9c}$$

Here K denotes the number of samples used in the MC sampling.

2.5. Challenges and contributions

To realize the MC scheme in (9) the posterior parameter distribution $p\big(\theta|\mathcal{T}\big)$ must be computed and samples drawn from this high-dimensional distribution. Our contributions are: (i) a local linearization approach that leads to a recursive algorithm of low complexity to compute an approximation of $p\big(\theta|\mathcal{T}\big)$ during the training phase; (ii) a second local linearization approach to reduce the sampling space from n_θ to M-dimensional space in the prediction phase; and as a by-product (iii) a low-complexity method for risk assessment and information fusion.

3. Posterior parameter distribution

Next, a local linear approach to compute an approximation of $p\big(\theta|\mathcal{T}\big)$ during the training phase is presented.

3.1. Laplace approximation

Assume the prior distribution for the model parameters to be normal distributed as $p(\theta) = \mathcal{N}(\theta; 0, P_0)$, i.e., l^2 regularization is used. Then a Laplacian approximation of the posterior distribution $p(\theta|\mathcal{T})$ yields that (Bishop, 2006)

$$p(\theta|\mathcal{T}) \approx \mathcal{N}(\theta; \hat{\theta}_N, P^\theta_N), \tag{10a}$$

$$P^\theta_N = \left(-\frac{\partial^2 L_N(\theta)}{\partial \theta^2}\bigg|_{\theta=\hat{\theta}_N} + P_0^{-1}\right)^{-1}. \tag{10b}$$

That is, the prior distribution is approximated by a normal distribution with a mean located at the maximum a posteriori estimate and a covariance dependent upon the shape of the likelihood function in the vicinity of the estimate. The accuracy of the approximation will depend upon the amount of information in the training data $\mathcal{T}$. A weakness with the Laplacian approximation is that it is local and assumes $\hat{\theta}_N$ to be a local minimum and can hence miss global trend in the posterior distribution (Bishop, 2006).

3.2. Asymptotic distribution

According to Bernstein–von Mises theorem (Johnstone, 2010), if the true model belongs to the considered model set, the maximum a posteriori estimate $\hat{\theta}$ converges in distribution to

$$\hat{\theta}_N \xrightarrow{d} \mathcal{N}(\hat{\theta}_N; \theta_*, \mathcal{I}_\theta^{-1}), \tag{11}$$

when the information in the training data $\mathcal{T}$ tends to infinity. Here, θ_* denotes the true parameters and

$$\mathcal{I}_\theta \triangleq -\mathrm{E}\left\{\frac{\partial^2 L_N(\theta)}{\partial \theta^2}\right\}, \tag{12}$$

is the Fisher information matrix. Given the likelihood function in (5) the Fisher matrix becomes

$$\mathcal{I}_\theta \simeq \sum_{n=1}^{N}\sum_{m=1}^{M} \eta_{m,n}\frac{\partial g_m(x_n; \theta)}{\partial \theta}\left(\frac{\partial g_m(x_n; \theta)}{\partial \theta}\right)^\top, \tag{13a}$$

$$\eta_{m,n} \triangleq f_m(x_n; \theta)(1 - f_m(x_n; \theta)). \tag{13b}$$

See derivations in the Appendix.

3.3. Recursive computation of covariance

To compute the parameter covariance P^θ_N defined by (10b), the Hessian matrix of the log-likelihood (LL) must be calculated and then inverted. This has a complexity of $\mathcal{O}(NMn_\theta^2 + n_\theta^3)$, which for large n_θ in combination with large N can become intractable. However, by approximating the Hessian matrix of the LL with the Fisher information matrix using (13), the computation can be done recursively and with a complexity of $\mathcal{O}\big(NMn_\theta^2 + NM^3\big)$ (Malmström et al., 2022). Note that $\mathcal{I}_\theta$ in (13) can be written in a quadratic form. As a result the recursive update can be given as

$$K_n = P^\theta_n U_n\big(I_M + U_n^\top P^\theta_n U_n\big)^{-1} \tag{14a}$$

$$P^\theta_{n+1} = P^\theta_n - K_n U_n^\top P^\theta_n, \tag{14b}$$

$$U_n = \begin{bmatrix} u_{1,n} & \ldots & u_{M,n}\end{bmatrix} \in \mathbb{R}^{n_\theta \times M}, \tag{14c}$$

$$u_{m,n} \triangleq \sqrt{\eta_{m,n}}\frac{\partial g_m(x_n; \theta)}{\partial \theta}\bigg|_{\theta=\hat{\theta}_N}. \tag{14d}$$

where I_r denotes the identity matrix of size r. Here P^θ_n is the parameter covariance for n measurements, which is initialized as $P^\theta_0 = P_0$. To compute $u_{mn} \in \mathbb{R}^{n_\theta}$ only the gradient of the LL in (5) is required, which is nevertheless needed for the estimation of θ.

3.4. Approximating the covariance

An NN often has millions of parameters which might result in the amount of data needed to store P^θ_N being larger than the available memory capacity. A common approach to handle this is to approximate P_N^θ as a block-diagonal matrix (Martens & Grosse, 2015). Another common approach is to use the approximation

$$P_N^\theta \approx \begin{bmatrix} P_N^{\theta_r} & 0 \\ 0 & 0 \end{bmatrix}, \tag{15}$$

where $P_N^{\theta_r}$ denotes the covariance of the estimated parameters θ_r corresponding to the weights and biases of the r last layers in the NN (Kristiadi, Hein, & Hennig, 2020; Malmström et al., 2022). Depending on the number of included layers, this approximation might be more or less accurate. To compensate for the approximation error when doing the marginalization in (8), a scaling of $P_N^{\theta_r}$ with factor $T_c \geq 1$ can be introduced. The scaling can be estimated from validation data in a similar manner to the temperature scaling T in Section 2.3.

4. Efficient MC sampling

With access to the parameter covariance, one can propagate the uncertainty in the parameters to uncertainty in the prediction with the delta method using the principle of marginalization. Plugging in the approximate Gaussian distribution (11) into (8a). Then an MC approximation can be performed by changing (9a) to

$$\theta^{(k)} \sim \mathcal{N}\big(\theta; \hat{\theta}_N, P_N^\theta\big), \quad k = 1, 2, \ldots, K. \tag{16}$$

This is a feasible solution to the problem, but it comes with a high computational cost since it requires drawing MC samples from a high-dimensional Gaussian distribution and evaluating the whole network.

4.1. Marginalization using the delta method

The delta method, see e.g., Liero and Zwanzig (2011) and Malmström (2021), relies on linearization of the nonlinear model $g(x, \theta)$ and provides a remedy to the problem of sampling from the high-dimensional Gaussian distribution. The idea is to project the uncertainty in the parameters to uncertainty in the prediction before the softmax normalization (3b), thereby drastically reducing the dimension of the distribution that must be sampled. Using the delta method, the uncertainty in the parameters can be propagated to the prediction before the softmax normalization as

$$p(g(x^\star;\theta)|\mathcal{T}) \approx \mathcal{N}\left(g(x^\star;\theta); \hat{g}_N, P_N^g\right), \tag{17a}$$

$$\hat{g}_N = \mathrm{E}\{g(x^\star;\theta)\} \simeq g(x^\star;\hat{\theta}_N), \tag{17b}$$

$$P_N^g = \mathrm{Cov}\{g(x^\star;\theta)\} \simeq \left(\frac{\partial}{\partial\theta} g(x^\star;\theta)\right)^\top P_N^\theta \frac{\partial}{\partial\theta} g(x^\star;\theta). \tag{17c}$$

Using this Gaussian approximation of the parameter distribution, the MC approximation of the marginalization integral becomes

$$g^{(k)}(x^\star) \sim \mathcal{N}\left(g(x^\star,\theta); \hat{g}_N, P_N^g\right), \quad k = 1, 2, \ldots K \tag{18a}$$

$$f^{(k)}(x^\star) = \mathrm{softmax}\big(g^{(k)}(x^\star)\big), \tag{18b}$$

$$\hat{f}(x^\star|\mathcal{T}) = \frac{1}{K}\sum_{k=1}^{K} f^{(k)}(x^\star), \tag{18c}$$

$$\hat{P}^f = \frac{1}{K}\sum_{k=1}^{K}\big(f^{(k)}(x^\star) - \hat{f}(x^\star|\mathcal{T})\big)(\cdot)^\top. \tag{18d}$$

To summarize, the main idea of the delta method is linearization performed in two steps. First, the parameter uncertainty is computed using (11), and second, the uncertainty is propagated to the output of the model by (17). Hence, the delta method is a local linear approach that gives a linear approximation of a nonlinear model.

4.2. Fusion

Suppose there is a set of independent classifiers, each one providing a marginal distribution $\mathcal{N}\left(g_{N,c}; \hat{g}_{N,c}, P_{N,c}^g\right)$, $c = 1, \ldots, C$. Then the predictions (before the softmax normalization) from these classifiers can be fused as follows (Gustafsson, 2018)

$$P_N^g = \left(\sum_{c=1}^{C} \big(P_{N,c}^g\big)^{-1}\right)^{-1}, \tag{19a}$$

$$\hat{g}_N = P_N^g \sum_{c=1}^{C} \big(P_{N,c}^g\big)^{-1} \hat{g}_{N,c}. \tag{19b}$$

If a single classifier is used to classify a set of inputs $x_c^\star$, $c = 1, \ldots, C$, known to belong to the same class $y^\star$, then these predictions can be fused as follows

$$P_N^g = (H^\top R^{-1} H)^{-1}, \tag{20a}$$

$$\hat{g}_N = P_N^g H^\top R^{-1} z, \tag{20b}$$

$$z = \begin{bmatrix}\hat{g}_{N,1} & \ldots & \hat{g}_{N,C}\end{bmatrix}^\top, \quad H = \begin{bmatrix} I_{n_\theta} & \ldots & I_{n_\theta}\end{bmatrix}^\top, \tag{20c}$$

$$[R]_{i,j} = \frac{\partial}{\partial\theta} g(x_i^\star;\theta)^\top\big|_{\theta=\hat{\theta}_N} P_N^\theta \frac{\partial}{\partial\theta} g(x_j^\star;\theta)\big|_{\theta=\hat{\theta}_N}, \tag{20d}$$

where $z \in \mathbb{R}^{Cn_\theta}$, $H \in \mathbb{R}^{Cn_\theta, n_\theta}$ and the block $[R]_{i,j} \in \mathbb{R}^{n_\theta, n_\theta}$, $i, j = 1, \ldots, C$.

After fusion, the MC sampling in (18) can be applied as before to compute the PMF estimate.

4.3. Risk assessment

Risk assessment can be defined as the probability r_m that $p(y_n^\star = m|x_n^\star) > \gamma_m$. The probability r_m can be estimated from the identified model $f_m(x_n^\star|\mathcal{T})$ as follows

$$\hat{r}_m = \Pr\{f_m(x_n^\star|\mathcal{T}) > \gamma_m\} \simeq \frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\big(f_m^{(k)}(x_n^\star) > \gamma_m\big). \tag{21}$$

Here $\mathbb{1}(a > b)$ denotes the indicator function which is one if $a > b$ and zero otherwise. Note that the parameter γ_m is free to choose and corresponds to the accepted failure rate.

5. Validation

Suppose now we have a validation data set $\mathcal{V} = \{y_n^\circ, x_n^\circ\}_{n=1}^{N_\circ}$. In this section we will investigate metrics how to validate the estimated PMF $\hat{f}(x_n^\circ|\mathcal{T})$ obtained from (18). The inherent difficulty is that the validation data, just as the training data, consists of inputs and class labels, not the actual PMF. Indeed, there is a lack of unified qualitative evaluation metrics (Gawlikowski et al., 2023). That being said, some of the most commonly used metrics are classification accuracy, LL, Brier score, and expected calibration error (ECE) (Wójcik, Grela, Śmieja, Misztal, & Tabor, 2022). Both the negative LL and the Brier score are proper scoring rules, meaning that they emphasize careful and honest assessment of the uncertainty, and are minimized for the true probability vector (Gneiting & Raftery, 2007). However, neither of them is a measure of the calibration, i.e., reliability of the estimated PMF. Out of these metrics, only ECE considers the calibration. Hence, here ECE is the most important metric when evaluating a method used to measure the uncertainty (Guo et al., 2017; Vaicenavicius et al., 2019). To evaluate a method that quantifies uncertainty in the prediction, one can also measure its capability to distinguish between in- and out-of-distribution (ID and OOD) inputs, i.e., find outliers.

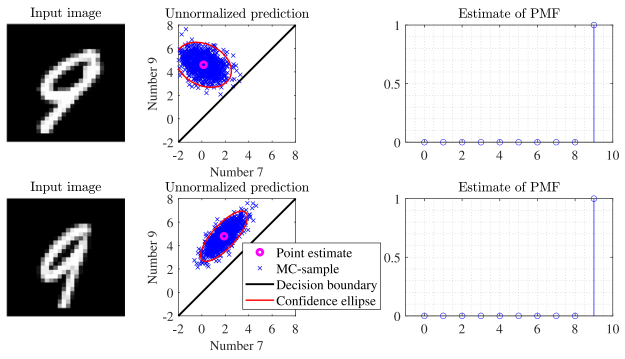


Fig. 1. Example of classification using (18). Left: inputs x_n°. Middle: Ellipse representation of P_N^g, MC samples $g^{(k)}(x_n^\circ)$ and decision line between the classes representing 7 and 9. Right: Estimated PMF $\hat{f}(x^\circ|\mathcal{T})$.

5.1. Brier score, accuracy, and reliability diagram

The Brier score (Gneiting & Raftery, 2007) corresponds to the least squares fit

$$\frac{1}{N_\circ}\sum_{n=1}^{N_\circ}\sum_{m=1}^{M}\left(\delta_{m,y_n} - \hat{p}(y_n^\circ = m|x_n^\circ)\right)^2, \tag{22}$$

where $\delta_{i,j}$ denotes the Kronecker delta function. Furthermore, $\hat{p}(y_n^\circ = m|x_n^\circ)$ denotes a generic PMF estimate.

Accuracy and reliability diagrams are calculated as follows. Calculate the J bin histogram defined as

$$B_j = \left\{n : \frac{j-1}{J} \le \max_m \hat{p}(y_n^\circ = m|x_n^\circ) < \frac{j}{J}\right\} \tag{23}$$

from the validation data. For a perfect classifier $B_j = \emptyset$ for $j < J$. For a classifier that is just guessing, all sets are of equal size, i.e., $|B_j| = |B_i|\ \forall i, j$. Note that $\max_m \hat{p}(y_n^\circ = m|x_n^\circ) \ge 1/M$, so the first bins will be empty if $J > M$. The accuracy of the classifier is calculated by comparing the size of each set with the actual classification performance within the set. That is,

$$\mathrm{acc}(B_j) = \frac{1}{|B_j|}\sum_{n\in B_j}\mathbb{1}(\hat{y}_n^\circ = y_n^\circ) \tag{24}$$

where $\hat{y}_n^\circ = \arg\max_m \hat{p}(y_n^\circ = m|x_n^\circ)$. A reliability diagram is a plot of the accuracy versus the confidence, i.e., the predicted probability frequency. A classifier is said to be calibrated if the slope of the bins is close to one, i.e., when $\mathrm{acc}(B_j) = (j-0.5)/J$ (Guo et al., 2017).

5.2. Confidence and expected calibration error

Instead of certainty, from hereon the standard, and equivalent, notion of confidence will be used (Guo et al., 2017; Vaicenavicius et al., 2019). The mean confidence in a set is denoted $\mathrm{conf}(B_j)$ and is defined as

$$\mathrm{conf}(B_j) = \frac{1}{|B_j|}\sum_{n\in B_j}\max_m \hat{p}(y_n^\circ = m|x_n^\circ), \tag{25}$$

This is a measure of how much the classifier trusts its estimated class labels. In contrast to the accuracy it does not depend on the annotated class labels y_n. Comparing accuracy to confidence gives the ECE, defined as

$$\mathrm{ECE} = \sum_{j}^{J}\frac{1}{|B_j|}|\mathrm{acc}(B_j) - \mathrm{conf}(B_j)|. \tag{26}$$

A small value indicates that the weight is a good measure of the actual performance.

5.3. Out-of-distribution detection

To evaluate the capability of a method to find OOD samples, the so-called area under the receiver operating characteristic (AUROC) curve and area under the precision–recall curve (AUPR) is often used (Goodfellow, Bengio, & Courville, 2016). Both the AUROC and the AUPR are between zero and one, and for a perfect detector, the value is one. Intuitively, AUROC is a measure of the proportion of detections, while AUPR measures the proportions of detections that are correct. Another approach to evaluate the capability of OOD detection is to study the difference in entropy between ID and OOD inputs $\Delta_s = S_{id} - S_{ood}$ where the entropy S is defined as

$$S = -\sum_{m=1}^{M}\hat{p}(y_n^\circ = m|x_n^\circ)\ln\left(\hat{p}(y_n^\circ = m|x_n^\circ)\right). \tag{27}$$

6. Experiment study

To illustrate the application of the proposed method to quantify uncertainty in the prediction, two datasets were used. First, an NN was trained using the MNIST dataset (LeCun, Cortes, & Burges, 1998) to classify images of handwritten digits. Second, an NN was trained on the CFAR10 dataset (Krizhevsky, 2009) to classify images of ten different objects including e.g., cars, cats, and aircraft. For OOD detection MNIST was used as ID and Fashion MNIST (FMNIST) (Xiao, Rasul, & Vollgraf, 2017) was used as OOD.

6.1. Classification setup

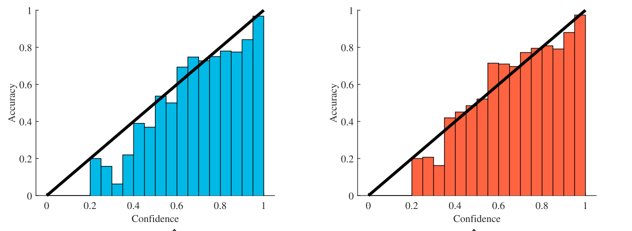
For the MNIST dataset, a five-layer NN with fully-connected nodes was used. For the CFAR10 dataset, a LeNet5-inspired structure was used with six convolutional layers followed by four fully connected layers. However, for both datasets, the three last layers were chosen to have the same structure, fully connected with 100, 40, and 10 hidden nodes. To decrease the size of the parameter covariance matrix used by the delta method, as described in Section 3.4, the first part of the NN was assumed fixed and used to create high-level features. Since the structure of the later layers was chosen identically, both models had $n_\theta = 4450$ parameters. To estimate the model parameters θ of the NN, the ADAM optimizer (Kingma & Ba, 2015) was used with standard settings. Three and ten epochs were used to train the MNIST dataset and CFAR10 dataset, respectively. Here, l^2 regularization of 10^{-4}, were used, hence the prior $P_0 = 10^{-4}I_{n_\theta}$. When computing P_N^θ, the choice of P_0 can be neglected since it is dominated by the Fisher information matrix $\mathcal{I}_\theta$.

6.2. Illustration of the uncertainties in the predictions

The low-dimensional space of the output from $g(x_n^\circ;\hat{\theta}_N)$ is particularly interesting to study when trying to understand how the uncertainty in the parameter estimate $\hat{\theta}_N$ affects the classification. Even if the parameter covariance P_N^θ is constant and only depends on the training data, the covariance P_N^g depends on the input x_n. Fig. 1 illustrates this via an example where we concentrate our study on the decision between just a subset of the number of classes in the MNIST dataset, even though the final decision is over all classes. More generally, for an input x_n° that is located in a dense region in the space of the training data, the covariance P_N^g is small, but for an input x_n° that is very far from the training data in some norm, the covariance P_N^g can be quite large. This indicates that the parameter estimate is quite sensitive in some

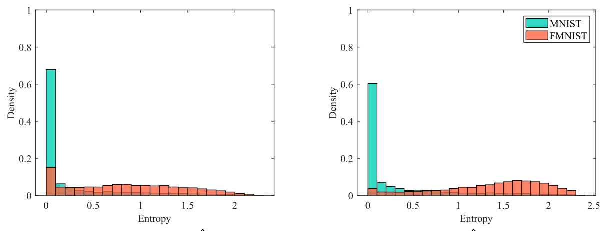
Table 1
Computed performance measure for the two datasets. The arrows indicate whether a high or low value is preferable.

Method, MNIST	acc. ↑	LL (10^3) ↑	Brier ↓	ECE ↓
Standard $f(x_n^\circ; \hat{\theta}_N)$	91%	**7.886**	0.134	1.078
Temp. sc. $f(x_n^\circ; \hat{\theta}_N, T)$ (Guo et al., 2017)	91%	7.818	0.133	0.951
Deep ensemble (Lakshminarayanan et al., 2017)	**96%**	7.856	**0.080**	2.868
MC-dropout (Gal & Ghahramani, 2016)	93%	7.424	0.123	2.424
Prop. met. $\hat{f}(x_n^\circ \| \mathcal{T})$	91%	7.845	0.151	1.242
Prop. met. $\hat{f}(x_n^\circ \| \mathcal{T}, T_c)$	91%	7.763	0.151	**0.821**
Method, CFAR10	**acc. ↑**	**LL (10^3) ↑**	**Brier ↓**	**ECE ↓**
Standard $f(x_n^\circ; \hat{\theta}_N)$	83%	7.904	0.291	1.328
Temp. sc. $f(x_n^\circ; \hat{\theta}_N, T)$ (Guo et al., 2017)	83%	7.740	0.269	0.612
Deep ensemble (Lakshminarayanan et al., 2017)	**87%**	7.834	**0.191**	1.479
MC-dropout (Gal & Ghahramani, 2016)	81%	**9.935**	0.301	2.829
Prop. met. $\hat{f}(x_n^\circ \| \mathcal{T})$	83%	8.176	0.243	2.140
Prop. met. $\hat{f}(x_n^\circ \| \mathcal{T}, T_c)$	82%	7.545	0.239	**0.540**



(a) Prop. met. $\hat{f}(x_n^\circ|\mathcal{T})$, (v) (b) Prop. met. $\hat{f}(x_n^\circ|\mathcal{T}, T_c)$, (vi)

Fig. 2. Reliability diagrams for prediction on the MNIST dataset using the proposed method in (18) with and without the scaling parameter T_c. In black is a calibration line.



(a) Temp. sc. $f(x_n^\circ; \hat{\theta}_N, T)$, (ii) (b) Prop. met. $\hat{f}(x_n^\circ|\mathcal{T}, T_c)$, (vi)

Fig. 3. Distribution for the predicted entropy (27) for ID (MNIST) and OOD (FMNIST) samples for the two calibrated methods.

directions. Even though the estimate of the PMF looks similar, by studying the unnormalized prediction $g(x_n^\circ; \hat{\theta}_N)$ it is clear that the prediction in the top has a higher uncertainty compared to the bottom one.

6.3. Results on quantifying the uncertainty

Six different methods to quantify the uncertainty in the classification, i.e., to estimate $p(y_n^\circ = m|x_n^\circ)$, were evaluated. These are: (i) Standard method, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ) = f_m(x_n^\circ; \hat{\theta}_N)$; (ii) Temp. scaling, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ) = f_m(x_n^\circ; \hat{\theta}_N, T)$, Guo et al. (2017); (iii) Deep ensemble, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ)$ is estimated using the ensemble method in Lakshminarayanan et al. (2017); number of trained NNs are 50 for MNIST and 10 for CFAR10; (iv) MC-dropout, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ)$ is estimated using the ensemble method in Gal and Ghahramani (2016); 50 samples of the parameters are used to create the ensemble; (v) Proposed method, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ) = \hat{f}_m(x_n^\circ|\mathcal{T})$; (vi) Proposed method with scaled covariance, i.e., $\hat{p}(y_n^\circ = m|x_n^\circ) = \hat{f}_m(x_n^\circ|\mathcal{T}, T_c)$, but with the covariance P_N^g in (17) scaled with a factor T_c.

The parameters T and T_c are estimated such that ECE is minimized on validation data. In Table 1, the accuracy, LL, Brier score, and ECE are shown for six different methods evaluated both using the MNIST and CFAR10 datasets. Table 1 shows that the proposed method attains the lowest ECE for both datasets, while still having reasonably good performance in terms of accuracy, LL, and Brier score. Neither computing the uncertainty in the prediction using the softmax (i), deep ensembles (iii), MC-dropout (iv), nor the proposed method without scaled covariance (v) gives calibrated estimates of the uncertainty. To get well-calibrated estimates of the uncertainty either the proposed method with scaled covariance (vi) or temperature scaling (ii) should be used. However, increasing the scaling factor decreases the LL. Hence, there is a trade-off between high LL and low ECE. In Fig. 2, the reliability diagram for the calibrated and non-calibrated versions of the proposed method is shown. Here one can see that the parameter T_c is necessary in order to obtain a calibrated uncertainty. For OOD detection, in Table 2 and Fig. 3, one can see that the proposed method is superior compared to temperature scaling (which is the only other calibrated method), while having similar performance to a deep ensemble whose performance is the best. However, recall that deep ensemble requires training of 50 models, hence, having significantly higher computational complexity compared to the proposed method.

On a standard laptop (Intel core i7), computing (17c) takes 930 s where the sampling in (18) is in comparison negligible (100 samples in 5 s). This is compared to sampling the parameters and then performing a forward pass per sample (such as in MC-dropout and deep ensemble) where only the forward pass takes 50 s. This is without considering that methods such as deep ensemble require training multiple NN.

7. Summary and conclusion

A method to estimate the uncertainty in classification performed by an NN has been proposed. The method also enables information fusion in applications where predictions from NNs are used as well as statistical risk assessment. The proposed method is based on a local linear approach and consists of two steps. In the first step, an approximation of the posterior distribution of the estimated NN parameters is calculated. This is done using a Laplacian approximation where the covariance of the parameters is calculated recursively using the structure of the Fisher information matrix. In the second step, an estimate of the PMF is calculated where the effect of the uncertainty in the estimated parameters is considered using marginalization over the posterior distribution of the parameter estimate. This is done

Table 2
Performance measure for OOD detection. The arrows indicate whether a high or low value is preferable.

Method, MNIST (ID) FMNIST (OOD)	Δ_s, (10^3) ↓	AUROC ↑	AUPR ↑
Temp. sc. $f(x_n^\circ; \hat{\theta}_N, T)$ (Guo et al., 2017)	−6.36	0.82	0.70
Deep ensemble (Lakshminarayanan et al., 2017)	**−13.12**	**0.94**	**0.91**
MC-dropout (Gal & Ghahramani, 2016)	−9.64	0.89	0.59
Prop. met. $\hat{f}(x_n^\circ \mid \mathcal{T}, T_c)$	−11.00	0.90	0.90

by propagating the uncertainty in the estimated parameters to the uncertainty in the output of the last layer in the NN using a second local linear approach. The uncertainty in the output of the last layer is approximated as a Gaussian distribution of the same dimension as the number of classes. The PMF and its covariance are then calculated via MC sampling, where samples are drawn from this low-dimensional distribution.

The proposed method has been evaluated on two classical classification datasets; MNIST and CFAR10. NNs with standard architectures were used. To handle a large number of parameters in these NN architectures, only the parameters of the last layers were considered in the uncertainty computations. The results, in terms of ECE, show that the proposed method in its standard form yielded a similar performance as standard methods which do not take the uncertainty in the estimated parameters into account. However, when using a rescaled parameter covariance matrix, used to compensate for the fact that only the uncertainty from the parameters in the last layers was considered, a significant reduction in the ECE was observed. This illustrates that the proposed method works well. However, that more advanced low-rank methods to approximate the parameter covariance are needed. This is a direction for future research. The result, in terms of detecting OOD detection, for the proposed method with a rescaled covariance is also shown to be superior compared to other calibrated standard methods.

Appendix. Derivation of Fisher information matrix

To calculate the Fisher information matrix in (13), it is necessary to compute the Hessian of the LL with respect to θ. To do so, note that $\frac{\partial f_j(x_n;\theta)}{\partial g_i(x_n;\theta)} = f_j(x_n,\theta)(\delta_{i,j} - f_i(x_n,\theta))$. Hence, it holds that $\frac{\partial \ln f_{y_n}(x_n;\theta)}{\partial g(x_n;\theta)} = \vec{e}_{y_n} - f(x_n;\theta)$. Using the chain rule the first derivative of the LL (5) can be computed as

$$\frac{\partial L_N(\theta)}{\partial \theta} = \sum_{n=1}^{N}\sum_{m=1}^{M}\big(\delta_{m,y_n} - f_m(x_n,\theta)\big)\frac{\partial g_m(x_n;\theta)}{\partial \theta}. \tag{A.1}$$

Differentiation of (A.1) with respect to θ gives

$$\frac{\partial^2 L_N(\theta)}{\partial \theta^2} = \sum_{n=1}^{N}\sum_{m=1}^{M}\big(\delta_{m,y_n} - f_m(x_n,\theta)\big)\frac{\partial^2 g_m(x_n;\theta)}{\partial \theta^2} - \eta_{m,n}\frac{\partial g_m(x_n;\theta)}{\partial \theta}\left(\frac{\partial g_m(x_n;\theta)}{\partial \theta}\right)^{\top}. \tag{A.2a}$$

with $\eta_{m,n}$ defined in (13b). And the Fisher information matrix in (12) then becomes

$$\mathcal{I}_\theta = -\sum_{n=1}^{N}\sum_{m=1}^{M}\big(\mathrm{E}\{\delta_{m,y_n}\} - f_m(x_n,\theta)\big)\frac{\partial^2 g_m(x_n;\theta)}{\partial \theta^2} + \eta_{m,n}\frac{\partial g_m(x_n;\theta)}{\partial \theta}\left(\frac{\partial g_m(x_n;\theta)}{\partial \theta}\right)^{\top} \tag{A.3a}$$

$$\simeq \sum_{n=1}^{N}\sum_{m=1}^{M}\eta_{m,n}\frac{\partial g_m(x_n;\theta)}{\partial \theta}\left(\frac{\partial g_m(x_n;\theta)}{\partial \theta}\right)^{\top}. \tag{A.3b}$$

The last approximative equality follows from that $\mathrm{E}\{\delta_{m,y_n}\} = p(y_n = m|x_n)$ and that $f_m(x_n,\theta)$ is an unbiased estimate of $p(y_n = m|x_n)$ when the information in the training data tends to infinity.

References

Avant, T., & Morgansen, K. A. (2023). On the sensitivity of pose estimation neural networks: rotation parameterizations, Lipschitz constants, and provable bounds. *Automatica*, *155*, Article 111112.

Ayhan, M. S., & Berens, P. (2018). Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. In *1st conf. on medical imaging with deep learn.*. Amsterdam, The Netherlands: 4–6 Jul.

Baggio, G., Carè, A., Scampicchio, A., & Pillonetto, G. (2022). Bayesian frequentist bounds for machine learning and system identification. *Automatica*, *146*, Article 110599.

Bishop, C. M. (2006). *Pattern recognition and machine learning* (pp. 213–220,315–319). New York, NY, USA: Springer.

Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015). Weight uncertainty in neural networks. In *Proc. of the 32nd int. conf. on mach. learn. (ICML).* (pp. 1613–1622). Lille, France: 6–11 Jul.

Charpentier, B., Zügner, D., & Günnemann, S. (2020). Posterior network: Uncertainty estimation without OOD samples via density-based pseudo-counts. *Vol. 33*, In *Adv. in neural inf. process. syst. (NIPS) 34*. Virtual.

D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., et al. (2020). Underspecification presents challenges for credibility in modern machine learning. arxiv preprint arXiv:2011.03395.

Deng, Z., Zhou, F., & Zhu, J. (2022). Accelerated linearized Laplace approximation for Bayesian deep learning. arXiv preprint arXiv:2210.12642.

Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proc. of the 33td int. conf. on mach. learn. (ICML).* (pp. 1050–1059). New York, NY, USA: 20–22 Jun.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., et al. (2023). A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 1–77.

Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence.. *Nature*, *521*(7553), 452–459.

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*(477), 359–378.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. London, England: MIT Press.

Grigorescu, S., Trasnea, B., Cocias, T., & Macesanu, G. (2020). A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, *37*(3), 362–386.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proc. of the 34th int. conf. on mach. learn. (ICML).* (pp. 1321–1330). Sydney, Australia: PMLR, 06–11 Aug..

Gustafsson, F. (2018). *Statistical sensor fusion*. Studentlitteratur: Lund Sweden.

Gustafsson, F. K., Danelljan, M., Bhat, G., & Schön, T. B. (2020). Energy-based models for deep probabilistic regression. In *Proc. of 16th European conf. on comput. vision* (pp. 325–343). Glasgow, UK/Online: 23-28 Aug.

Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: theory and applications. *Neurocomputing*, *70*(1–3), 489–501.

Hwang, J. T. G., & Ding, A. A. (1997). Prediction intervals for artificial neural networks. *Journal of the American Statistical Association*, *92*, 748–757.

Ilg, E., Cicek, O., Galesso, S., Klein, A., Makansi, O., Hutter, F., et al. (2018). Uncertainty estimates and multi-hypotheses networks for optical flow. In *Proc. of 15th European conf. on comput. vision* (pp. 652–667). Munich, Germany: 8-14 Sep.

Immer, A., Korzepa, M., & Bauer, M. (2021). Improving predictions of Bayesian neural nets via local linearization. In *Proc. of 24nd int. conf. on artificial intell. and statistics.* (pp. 703–711). San Diego, CA, USA: PMLR, 13-15 Apr.

Izmailov, P., Nicholson, P., Lotfi, S., & Wilson, A. G. (2021). Dangers of Bayesian model averaging under covariate shift. *Vol. 34*, In *Adv. in neural inf. process. syst. (NIPS) 35*. New Orleans, LA, USA:

Johnstone, I. (2010). High dimensional Bernstein-von mises: simple examples. *Institute of Mathematical Statistics collections*, *6*, 87–98.

Karlsson, R., & Hendeby, G. (2021). Speed estimation from vibrations using a deep learning CNN approach. *IEEE Sensors Letters*, *5*, 1–4.

Kendall, A., & Gal, Y. (2017). What uncertainties do we need in Bayesian deep learning for computer vision? In *Adv. in neural inf. process. syst. (NIPS) 31* (pp. 5574–5584). Long Beach, CA, USA: Curran Associates, Inc., 4–9 Dec.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization.. In *Proc. of 3rd int. conf. for learn. representations*. San Diego, CA, USA.

Kristiadi, A., Hein, M., & Hennig, P. (2020). Being Bayesian, even just a bit, fixes overconfidence in relu networks. In *Proc. of the 37th int. conf. on mach. learn. (ICML).*. Online: 13-18 July.

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images* M.Sc. thesis, University of Toronto, Canada.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Adv. in neural inf. process. syst. (NIPS) 25* (pp. 1097–1105). Lake Tahoe, NV USA: 3-8 Dec.

Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *Adv. in neural inf. process. syst. (NIPS) 31*. Long Beach, CA, USA: Curran Associates, Inc., 4–9 Dec.

LeCun, Y., Cortes, C., & Burges, C. J. (1998). The MNIST database of handwritten digits. URL http://yann.lecun.com/exdb/mnist/.

Li, Q., Qian, J., Zhu, Z., Bao, X., Helwa, M. K., & Schoellig, A. P. (2017). Deep neural networks for improved, impromptu trajectory tracking of quadrotors. In *Proc. of IEEE int. conf. on robot. and autom.* (pp. 5183–5189). Singapore, Singapore: IEEE, 19 May–3 June.

Liero, H., & Zwanzig, S. (2011). *Introduction to the theory of statistical inference*. Chapman and Hall CRC Texts in Statistical Science, Boca Raton, FL, USA.

Lin, S., Clark, R., Trigoni, N., & Roberts, S. (2022). Uncertainty estimation with a VAE-classifier hybrid model. In *Proc. of IEEE int. conf. on acoust., speech and signal processing* (pp. 3548–3552). Singapore, Singapore,: IEEE, 22–17 May.

Lindholm, A., Wahlström, N., Lindsten, F., & Schön, T. B. (2022). *Machine learning: A first course for engineers and scientists*. Cambridge University Press.

Ljung, L. (1999). *System identification: theory for the user (2nd edition)*. PTR Prentice Hall: Upper Saddle River, NJ, USA.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., & Wilson, A. G. (2019). A simple baseline for Bayesian uncertainty in deep learning. In *Adv. in neural inf. process. syst. (NIPS) 33*. Vancouver, Canada: 8–14 Dec.

Malmström, M. (2021). *Uncertainties in neural networks a system identification approach* (Ph.D. thesis), Dept. Elect. Eng., Linköping University, Linköping, Sweden.

Malmström, M., Skog, I., Axehill, D., & Gustafsson, F. (2022). Detection of outliers in classification by using quantified uncertainty in neural networks. In *Proc. of IEEE 25th int. conf. on inf. fusion*. Linköping, Sweden: IEEE, Jul 4-7.

Martens, J., & Grosse, R. (2015). Optimizing neural networks with kronecker-factored approximate curvature. In *Proc. of the 32nd int. conf. on mach. learn. (ICML).*. Lille, France: 6–11 Jul.

NTSB (2018). *Highway accident report HAR-19/03 HWY18MH010: Technical Specification (TS)*, National Transportation Safety Board (NTSB).

Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., et al. (2019). Practical deep learning with Bayesian principles. In *Adv. in neural inf. process. syst. (NIPS) 33*. Vancouver, Canada: 8–14 Dec.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., et al. (2019). Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *Adv. in neural inf. process. syst. (NIPS) 33*. Vancouver, Canada: 8–14 Dec.

Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2020). Challenges in deploying machine learning: a survey of case studies. *Vol. 33*, In *Adv. in neural inf. process. syst. (NIPS) 34 workshop: ML retrospectives, surveys & meta-analyses (ML-RSA)*. Virtual.

Patel, K., & Waslander, S. (2022). Accurate prediction and uncertainty estimation using decoupled prediction interval networks. arxiv preprint arXiv:2202.09664.

Shen, X., Varshney, P. K., & Zhu, Y. (2014). Robust distributed maximum likelihood estimation with dependent quantized data. *Automatica*, *50*(1), 169–174.

Teye, M., Azizpour, H., & Smith, K. (2018). Bayesian uncertainty estimation for batch normalized deep networks. In *Proc. of the 35th int. conf. on mach. learn. (ICML).* (pp. 4907–4916). Stockholm, Sweden: 6–11 Jul.

Vaicenavicius, J., Widmann, D., Andersson, C., Lindsten, F., Roll, J., & Schön, T. (2019). Evaluating model calibration in classification. In *Proc. of 22nd int. conf. on artificial intell. and statistics.* (pp. 3459–3467). Naha, Okinawa, Japan: PMLR, 16-18 Apr..

Wójcik, B., Grela, J., Śmieja, M., Misztal, K., & Tabor, J. (2022). SLOVA: Uncertainty estimation using single label one-vs-all classifier. *Applied Soft Computing*, *126*, Article 109219.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.

Magnus Malmström received his B.Sc. and M.Sc. in Applied Physics and Electrical Engineering from Linköping University, Sweden, in 2016 and 2018, respectively. In 2021, he received his degree of Lic.Eng. in Automatic Control from Linköping University, Sweden. Furthermore, in 2023, he received a Ph.D. in Electrical Engineering with a Specialization in Automatic Control from Linköping University, Sweden, with a thesis titled Approximate Uncertainty in Neural Network Predictions. Currently, he is a researcher at the Swedish Defence Research Agency (FOI) in Sweden.

Isaac Skog received his B.Sc. and M.Sc. degrees in Electrical Engineering from KTH Royal Institute of Technology, Sweden in 2003 and 2005, respectively. In 2010, he received a Ph.D. degree in Signal Processing with a thesis on low-cost navigation systems. In 2009, he spent 5 months with the Mobile Multi-Sensor System research team, University of Calgary, Canada, as a visiting scholar, and in 2011 he spent 4 months at the Indian Institute of Science (IISc), Bangalore, India, as a visiting scholar. Between 2010 and 2017 he was a researcher at KTH Royal Institute of Technology, Stockholm, Sweden. Between 2017 and 2022 he was an associate professor at Linköping University, Sweden. Currently, he is an associate professor at Uppsala University, Sweden, and a senior researcher at the Swedish Defence Research Agency (FOI), Sweden. He is the author and co-author of more than 60 international journal and conference publications.

Daniel Axehill received his M.Sc. degree in Applied Physics and Electrical Engineering in 2003. Furthermore, he received the degree of Lic.Eng. in Automatic Control in 2005 and the Ph.D. degree in Automatic Control in 2008. All three degrees are from Linköping University, Linköping, in Sweden. In year 2006 he spent three months at UCLA in Los Angeles. From January 2009 and until November 2010 he was a post-doc at the Automatic Control Laboratory at ETH Zurich. He is currently employed as a Senior Associate Professor at the Division of Automatic Control at Linköping University. He is an IEEE senior member.

Fredrik Gustafsson is professor in Sensor Informatics at Department of Electrical Engineering, Linköping University, since 2005. He received the M.Sc. degree in electrical engineering 1988 and the Ph.D. degree in Automatic Control, 1992, both from Linköping University.

He is the author of five books, more than 600 scientific papers, papers and some 30 patents. His current h-index is 71 (Google Scholar). He has supervised 25 Ph.D. and more than 200 master theses.

He was an associate editor for IEEE Transactions of Signal Processing 2000–2006, IEEE Transactions on Aerospace and Electronic Systems 2010–2012, and EURASIP Journal on Applied Signal Processing 2007–2012. He was awarded the Arnberg prize by the Royal Swedish Academy of Science (KVA) 2004, elected member of the Royal Academy of Engineering Sciences (IVA) 2007, and elevated to IEEE Fellow 2011. In 2014, he was awarded a Distinguished Professor grant from the Swedish Research Council. He was an adjunct entrepreneurial professor at Twente University 2012–2013.

He was awarded the Harry Rowe Mimno Award 2011 for his tutorial "Particle Filter Theory and Practice with Positioning Applications", which was published in the AESS Magazine in July 2010. This paper was ranked 4 out of 13077 articles published in the category IEEE Engineering Aerospace between 2010 and 2014 based on Web of Science citation data. He was a co-author of "Smoothed state estimates under abrupt changes using sum-of-norms regularization" that received the Automatica paper prize in 2014.

He is a co-founder of the companies NIRA Dynamics (automotive safety, including tire pressure monitoring systems found in about 100 million cars today), Softube (plug-ins used in tens of thousands music studios), and Senion (indoor navigation for smartphones deployed in more than 30 countries in all six continents), now part of Verizon.