A MEASURE FOR UNCERTAINTY QUANTIFICATION IN NEURAL NETWORKS

Jalil Taghia*& Fredrik Lindsten† and Thomas B. Schön*

* Department of Information Technology, Uppsala University, Sweden
† Division of Statistics and Machine Learning, Linköping University, Sweden

ABSTRACT

In safety-critical applications, we wish to have a measure of uncertainty quantification—a measure of confidence calibration that reflects the underlying uncertainty. For neural network based classifiers, the output probabilities on the target classes are used typically as a way of uncertainty quantification. However, it has been observed that the standard training of neural networks, in particular when trained for the best accuracy, may produce uncalibrated probabilities—probability estimates that are not representative of the true likelihood. Building on our previous work on the matrix multilayer perceptron (Taghia et al., 2019), here, we propose a measure for uncertainty quantification that appears to be more reliable (calibrated) in comparison to the standard output probabilities from the network.

1 PROBLEM FORMULATION

Consider a classification problem of $m > 1$ categories and let $\mathbb{M} = \{1, \ldots, m\}$. We are given a training dataset consists of the inputs and the corresponding labels,

$$\mathcal{D}_{\text{train}} = \{\underline{\mathbf{x}}, \underline{\mathsf{y}}\}, \quad \underline{\mathbf{x}} = \{\mathbf{x}_i \in \mathbb{R}^p\}_{i=1}^n, \quad \underline{\mathsf{y}} = \{\mathsf{y}_i \in \mathbb{M}\}_{i=1}^n. \tag{1}$$

Given enough data samples and a viable choice of loss function, the network learns to predict the label for an unseen test data sample $\mathbf{x}_* \in \mathcal{D}_{\text{test}}$. From a probabilistic classifier, it is expected to not only correctly identify the labels but also assign reliable probabilities to them. For a well calibrated classifier, these probabilities can then be directly interpreted as a measure of confidence level. Refer to Guo et al. (2017) for discussion on the calibration of the neural networks and to Vaicenavićius et al. (2019) for general discussion on evaluation of the model calibration in classification.

It has been observed that the standard training of neural networks, for example using MLP, gives biased probabilities. This is particular true when the network is trained for the best accuracy (Guo et al., 2017). Here, we propose an approach for deriving a set of "modified probabilities" that show better calibration characteristics than the standard output probabilities from the network.

2 METHOD

This section summarizes the steps for derivation of our new measure of uncertainty quantification.

2.1 CONSTRUCTING TRAINING SET

Recall our problem scenario in Section 1. We now instead assume that associated to a given input $\mathbf{x}_i$, there are two corresponding labels y_i and $\mathbf{Y}_i$, where $\mathbf{Y}_i$ is a square matrix in the space of symmetric positive definite (SPD) matrices of trace one, shown as $\mathbb{P}_1$. Our new training set is constructed as

$$\mathcal{D}_{\text{train}} = \{\underline{\mathbf{x}}, (\underline{\mathsf{y}}, \underline{\mathbf{Y}})\}, \quad \underline{\mathbf{Y}} = \{\mathbf{Y}_i \in \mathbb{P}_1^{d \times d}\}_{i=1}^n, \tag{2}$$

where $\underline{\mathbf{x}}$ and $\underline{\mathsf{y}}$ are defined as in (1).

In typical classification problems, we do not have access to $\underline{\mathbf{Y}}$. However, they can be constructed from the inputs $\underline{\mathbf{x}}$ and labels $\underline{\mathsf{y}}$. Let $\underline{\tilde{\mathbf{x}}} = \{\tilde{\mathbf{x}}_i \in \mathbb{R}^d\}_{i=1}^n$ be a set of features extracted from the input

set $\underline{\mathbf{x}}$ such that $\tilde{\mathbf{x}}_i$ is a feature vector associated to the input $\mathbf{x}_i$. Note that, in the simplest scenario, we can simply set $\tilde{\mathbf{x}}_i$ to be identical to $\mathbf{x}_i$. However, in general, $\tilde{\mathbf{x}}_i$ can be a sub-vector of $\mathbf{x}_i \in \mathbb{R}^p$ in particular when the input dimensionality p is large. An an example, one can apply principal component analysis to the inputs $\underline{\mathbf{x}}$ and use the resulting principal components to form $\underline{\tilde{\mathbf{x}}}$.

Let $\underline{\tilde{\mathbf{x}}}^{(j)}$ indicate the set of features associated with the category $j \in \mathbb{M}$. The trace-one normalized sample covariance matrix of the jth category is computed using:

$$\mathbf{\Sigma}_1^{(j)} = \frac{\mathbf{\Sigma}^{(j)}}{\mathrm{tr}(\mathbf{\Sigma}^{(j)})}, \quad \mathbf{\Sigma}^{(j)} = \frac{1}{n^{(j)}-1} \sum_{i:\mathsf{y}_i=j} (\tilde{\mathbf{x}}_i - \bar{\tilde{\mathbf{x}}})(\tilde{\mathbf{x}}_i - \bar{\tilde{\mathbf{x}}})^\top, \quad \bar{\tilde{\mathbf{x}}} = \frac{1}{n^{(j)}} \sum_{i:\mathsf{y}_i=j} \tilde{\mathbf{x}}_i, \tag{3}$$

where $n^{(j)}$ is the number of instances in the category j. Let $\underline{\mathbf{\Sigma}} = \{\mathbf{\Sigma}^{(i)}\}_{i=1}^m$, then our training set is constructed as

$$\mathcal{D}_{\text{train}} = \{\underline{\mathbf{x}}, (\underline{\mathsf{y}}, \underline{\mathbf{Y}})\}, \quad \underline{\mathbf{Y}} = \{\mathbf{Y}_i \in \underline{\mathbf{\Sigma}}\}_{i=1}^n. \tag{4}$$

In construction of $\underline{\mathbf{Y}}$, we used the sample covariance of the input features $\underline{\tilde{\mathbf{x}}}$. However, notice that, the framework is general and it only requires $\mathbf{Y}_i$ to be a SPD matrix of trace one.

2.2 Classification using matrix multilayer perceptron

Given our training set $\mathcal{D}_{\text{train}} = \{\underline{\mathbf{x}}, (\underline{\mathsf{y}}, \underline{\mathbf{Y}})\}$ as constructed in (4), we use the general form of the mMLP neural network (Taghia et al., 2019, Section 4.2, Eq. 7) for the classification purpose using the following loss function:

$$\ell(\widehat{\mathbf{Y}}, \widehat{\mathsf{y}}, \mathbf{Y}, \mathsf{y}) = (1-\beta)\ell_1 + \beta\ell_2, \tag{5}$$

where $\ell_1 = \Delta_{\text{sQRE}}(\widehat{\mathbf{Y}}, \mathbf{Y})$ is the symmetrized quantum relative entropy between the true SPD labels $\mathbf{Y}$ and its predictions $\widehat{\mathbf{Y}}$ as defined in (Taghia et al., 2019, Eq. 6), and $\ell_2 = \Delta_{\text{CE}}(\widehat{\mathsf{y}}, \mathsf{y})$ is the cross-entropy loss between the true labels y and its predictions $\widehat{\mathsf{y}}$. The free parameter β would determine the importance of each loss, and can be treated as a hyperparameter, $\beta \in (0, 1)$. In this work, we set $\beta = 0.5$.

2.3 New measure for uncertainty quantification

Let $\text{mMLP}_\theta : \mathbf{x}_* \rightarrow (\widehat{\mathsf{y}}_*, \widehat{\mathbf{Y}}_*)$ indicate the trained mMLP network where θ is a set which contains the neural network parameters. The trained network takes as input $\mathbf{x}_* \in \mathcal{D}_{\text{test}}$ and outputs $\widehat{\mathsf{y}}_*$ which is computed from the output probability vector $p(\mathbf{x}_* \,|\, \mathcal{D}_{\text{train}}, \theta)$, and $\widehat{\mathbf{Y}}_*$ which is a SPD matrix of trace one. We then define our measure of the uncertainty quantification as $\mathbf{q} = (\mathsf{q}_1, \ldots, \mathsf{q}_m)$ where $\sum_{i=1}^m \mathsf{q}_i = 1$ and $0 \leq \mathsf{q}_i \leq 1$ computed using:

$$\mathsf{q}_i = \frac{1 - \Delta_{\text{sQRE}}(\widehat{\mathbf{Y}}_*, \mathbf{\Sigma}_1^{(i)})}{\sum_{l=1}^m 1 - \Delta_{\text{sQRE}}(\widehat{\mathbf{Y}}_*, \mathbf{\Sigma}_1^{(l)})}, \quad \forall i = \{1, \ldots, m\}, \tag{6}$$

where $\mathbf{\Sigma}_1^{(\text{i})} \in \underline{\mathbf{\Sigma}}_1$ given by (3).

We interpret $\mathbf{q}$ as our "modified probability", and hypothesize that it is better calibrated in comparison to the standard probability $\mathbf{p}$.

3 Experimental Results

Dataset. We consider the classification task of medical reports. The reports are written in text files as the summary of the medical examinations[1], containing two types of diseases: Lung emboli and Aorta dissektion. Each disease has its own examination procedure which means the report contents from the examinations can vary largely. Refer to (Nelsson, 2018, Chapter 4) for additional details on the data.

We consider a binary classification problem where the target categories are:

[1]The dataset comes from the Vinnova project: ALFA—Autonomous Large-scale Findings Analysis (contract number: 2017-01545).

Table 1: Performance evaluation of the various classifiers. For the mMLP model, the calibration quality measures are computed both using the standard probabilities and the modified probabilities. For Brier score, ECE, and MCE, the lower values are preferred.

Small Training Set					
Classifier	Description	Accuracy (%)	Brier score	ECE	MCE
Logistic	output proba.	89.2	0.052	0.14	0.28
MLP	output proba.	94.8	0.065	0.20	0.60
mMLP	output proba.	95.1	0.054	0.14	0.35
mMLP	modified proba. (6)	95.1	0.041	0.12	0.29
Large Training Set					
Classifier	Description	Accuracy(%)	Brier score	ECE	MCE
Logistic	output proba.	91.0	0.043	0.18	0.33
MLP	output proba.	96.2	0.063	0.22	0. 55
mMLP	output proba.	96.1	0.046	0.20	0. 42
mMLP	modified proba., (6)	96.1	0.031	0.17	0.39

- **Ingen**: the disease was not found in the examination.
- **Funnen**: the disease was found in the examination.

The two class categories are different in sizes: the class category 'Ingen' accounts for about 83% (16890 samples) of the total samples and the class category 'Funnen' accounts for 17% (3512 samples). The dataset is divided into a training set and a test set. We consider two scenarios of large training set (80%) and small training set (20%). In both cases, 20% of the samples in the training set are used for the validation set.

Classifiers. Three different classifiers are compared against each other:

- the logistic classifier as an example of a well-calibrated classifier;
- the standard MLP with 3 hidden layers and 100 units per layer;
- the mMLP with 3 hidden layers and 100 units per layer and 30 units at the SPD layers.

Both MLP and mMLP use the ReLU activation function in their hidden layers. The mMLP uses the Mercer Sigmoid activation matrix function at the hidden layers. Classifiers are trained for the best accuracy score on the validation set.

Feature selection and dataset construction. To construct $\underline{\mathbf{x}}$, we first need to turn the text contents into numerical feature vectors. For this purpose, we extract "Term Frequency times Inverse Document Frequency", commonly known as tf-idf[2]. The number of features is set to 1000. The set $\underline{\mathrm{y}}$ includes the labels. For construction of $\underline{\mathbf{Y}}$, we carried out the following steps:

- Applied principal component analysis to the inputs $\underline{\mathbf{x}}$ and used the resulting 20 first principal components to form $\underline{\tilde{\mathbf{x}}}$.
- For the construction of $\underline{\mathbf{Y}}$, we used the sample covariance of the input features $\underline{\tilde{\mathbf{x}}}$ computed according to Section (3).

The training dataset $\mathcal{D}_{\mathrm{train}}$ for the mMLP model is constructed according to (4) while for the logistic classifier and the MLP, it is constructed according to (1).

Results. The objective of the experiment is not focused on the identification of the best classifier but rather on the evaluation of the calibration quality of the output probabilities by each model. That being said, in terms of the accuracy, both models, the MLP and the mMLP, performed equally well with no clear advantage (Table 1).

The calibration quality is measured in terms of the following measures:

[2]We used "TfidfTransformer" method provided by "sklearn" package (https://scikit-learn.org/).

- Brier score. The metric is a combination of calibration loss, defined as the mean squared deviation from empirical probabilities derived from the slope of ROC segments, and refinement loss, the expected optimal loss as measured by the area under the optimal cost curve (Brier, 1950).
- The expected calibration error (ECE) and the maximum calibration error (MCE). In computing these measures, the predictions are sorted and partitioned into k fixed number of bins ($k = 10$ in our experiments). The predicted value of each test instance falls into one of the bins. The ECE calculates the expected calibration error over the bins, and MCE calculates the maximum calibration error among the bins reflecting the worst case scenario (Naeini et al., 2015).

In computation of the calibration quality measures, for the MLP and logistic classifier, we use the standard output probabilities, while for the mMLP model, we use both the standard probabilities and the modified probabilities computed according to (6). The results are summarized in Table 1.

4 Discussion

We proposed a new measure for the purpose of uncertainty quantification in neural networks. Our preliminarily results seem encouraging. In our analysis on real data, we observed that the newly introduced measure has potentially better calibration characteristics than the standard probabilities from the network, when the network is trained for the goal of achieving the best accuracy. Additional analysis is needed to support the generality of this observation along with research on the theoretical justifications and guarantees.

Acknowledgements

This research is financially supported by Vinnova project: ALFA-Autonomous Large-scale Findings Analysis (J. Taghia and F. Lindsten, Ref: 2017-01545), by The Knut and Alice Wallenberg Foundation (J. Taghia, Ref: KAW2014.0392), by the project Learning flexible models for nonlinear dynamics (T. B. Schön, Ref: 2017-03807) funded by the Swedish Research Council, by the Swedish Foundation for Strategic Research (SSF) via the project ASSEMBLE (T. B. Schön, Ref: RIT15-0012), by the project Learning of Large-Scale Probabilistic Dynamical Models (F. Lindsten, Ref: 2016-04278) funded by the Swedish Research Council, by the Swedish Foundation for Strategic Research via the project Probabilistic Modeling and Inference for Machine Learning (F. Lindsten, Ref: ICA16-0015), and by Wallenberg AI, Autonomous Systems and Software Program (WASP).

References

G.W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(5):1–3, 1950.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *ICML*, 2017.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using Bayesian binning. In *AAAI*, 2015.

Mikael Nelsson. Deep learning for medical report texts, 2018. ISSN 1650–8319.

Jalil Taghia, Maria Bånkestad, Fredrik Lindsten, and Thomas B. Schön. Constructing the matrix multilayer perceptron and its application to the vae. *arXiv*, abs/1902.01182, 2019.

Juozas Vaicenavićius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas B. Schön. Evaluating model calibration in classification. In *AISTATS*, 2019.