Classification Uncertainty of Deep Neural Networks Based on Gradient Information

Philipp Oberdiek[1(✉)], Matthias Rottmann[2], and Hanno Gottschalk[2]

[1] Technische Universität Dortmund, 44227 Dortmund, Germany
philipp.oberdiek@udo.edu
[2] Bergische Universität Wuppertal, 42119 Wuppertal, Germany
{hanno.gottschalk,rottmann}@uni-wuppertal.de

Abstract. We study the quantification of uncertainty of Convolutional Neural Networks (CNNs) based on gradient metrics. Unlike the classical softmax entropy, such metrics gather information from all layers of the CNN. We show for the EMNIST digits data set that for several such metrics we achieve the same meta classification accuracy – i.e. the task of classifying predictions as correct or incorrect without knowing the actual label – as for entropy thresholding. We apply meta classification to unknown concepts (out-of-distribution samples) – EMNIST/Omniglot letters, CIFAR10 and noise – and demonstrate that meta classification rates for unknown concepts can be increased when using entropy together with several gradient based metrics as input quantities for a meta classifier. Meta classifiers only trained on the uncertainty metrics of known concepts, i.e. EMNIST digits, usually do not perform equally well for all unknown concepts. If we however allow the meta classifier to be trained on uncertainty metrics for some out-of-distribution samples, meta classification for concepts remote from EMNIST digits (then termed known unknowns) can be improved considerably.

Keywords: Deep learning · Uncertainty quantification
Meta classification

1 Introduction

In recent years deep learning has outperformed other classes of predictive models in many applications. In some of these, e.g. autonomous driving or diagnostics in medicine, the reliability of a prediction is of highest interest. In classification tasks, the thresholding on the highest softmax probability or thresholding on the entropy of the classification distributions (softmax output) are commonly used metrics to quantify classification uncertainty of neural networks, see e.g. [11]. However, misclassification is oftentimes not detected by these metrics and it is also well known that these metrics can be fooled easily. Many works demonstrated how an input can be designed to fool a neural network such that it incorrectly classifies the input with high confidence (termed adversarial examples, see e.g. [9,13,18,19]). This underlines the need for measures of uncertainty.

L. Pancioni et al. (Eds.): ANNPR 2018, LNAI 11081, pp. 113–125, 2018.
https://doi.org/10.1007/978-3-319-99978-4_9

A basic statistical study of the performance of softmax probability thresholding on several datasets was developed in [11]. This work also assigns proper out-of-distribution candidate datasets to many common datasets. For instance a network trained on MNIST is applied to images of handwritten letters, scaled gray scale images from CIFAR10, and different types of noise. This represents a baseline for comparisons.

Using classical approaches from uncertainty quantification for modeling input uncertainty and/or model uncertainty, the detection rate of misclassifications can be improved. Using the baseline in [11], an approach named ODIN, which is based on input uncertainty, was published in [15]. This approach shows improved results compared to pure softmax probability thresholding. Uncertainty in the weights of a neural network can be modeled using Bayesian neural networks. A practically feasible approximation to Bayesian neural networks was introduced in [8], known as Monte-Carlo dropout, which also improves over classical softmax probability thresholding.

Since the softmax removes one dimension from its input by normalization, some works also perform outlier detection on the softmax input (the penultimate layer) and outperform softmax probability thresholding as well, see [2].

In this work we propose a different approach to measure uncertainty of a neural network based on gradient information. Technically, we compute the gradient of the negative log-likelihood of a single sample during inference where the class argument in the log-likelihood is the predicted class. We then extract compressed representations of the gradients, e.g., the norm of a gradient for a chosen layer. E.g., a large norm of the gradient is interpreted as a sign that, if the prediction would be true, major re-learning would be necessary for the CNN. We interpret this 're-learning-stress' as uncertainty and study the performance of different gradient metrics used in two meta classification tasks: separating correct and incorrect predictions and detecting in- and out-of-distribution samples.

The closest approaches to ours are probably [2,11] as they also establish a self evaluation procedure for neural networks. However they only incorporate (non-gradient) metrics for particular layers close to the networks output while we consider gradient metrics extracted from all the layers. Just as [2,11] our approach does not make use of input or model uncertainty. However these approaches, as well as our approach, are somewhat orthogonal to classical uncertainty quantification and should be potentially combinable with input uncertainty and model uncertainty, as used in [8,15], respectively.

The remainder of this work is structured as follows: First, in Sect. 2 we introduce (gradient) metrics, the concept of meta classification and threshold independent performance measures for meta classification, AUROC and AUPR, that are used in the experiments. In Sect. 3 we introduce the network architecture and the experiment setup containing the choice of data sets. We use EMNIST [6] digits as a known concept on which the CNN is trained and EMNIST letters, CIFAR10 images as well as different types of noise as unknown/unlearned concepts. Then we statistically investigate the separation performance of our metrics for correct vs. incorrect classifications provided by CNNs. This is followed by a

performance study for the detection of in- and out-of-distribution samples (detection of unlearned concepts) in Sect. 4. Therefore we also combine available metrics for training and comparing different meta classifiers. In this section meta classifiers are trained only using known concepts, i.e., EMNIST digits. Afterwards, in Sect. 5, we insert unlearned concepts (which therefore become known unknowns) into the training of the meta classifiers. While the softmax baseline achieves an AUROC value of 95.83% our approach gains 0.81% in terms of AUROC and even more in terms of AUPR values.

2 Entropy, Softmax Baseline and Gradient Metrics

Given an input $x \in \mathbb{R}^n$, weights $w \in \mathbb{R}^p$ and class labels $y \in \mathcal{C} = \{1, \ldots, q\}$, we denote the output of a neural network by $f(y|x, w) \in [0, 1]$. The entropy of the estimated class distribution conditioned on the input (also called Shannon information, [16])

$$E(x, w) = -\frac{1}{\log(q)} \sum_{y \in \mathcal{C}} f(y|x, w) \log(f(y|x, w)), \tag{1}$$

is a well known dispersion measure and widely used for quantifying classification uncertainty of neural networks. In the following we will use the term entropy in the sense explained above. Note that this should not be confused with the entropy underlying the (not estimated and joint) statistical distribution of inputs and labels. The softmax baseline proposed by [11] is calculated as

$$S(x, w) = \max_{y \in \mathcal{C}} f(y|x, w). \tag{2}$$

Using the maximum a posteriori principle (MAP), the predicted class is defined by

$$\hat{y}(x, w) := \arg\max_{y \in \mathcal{C}} f(y|x, w) \tag{3}$$

according to the Bayes decision rule [3], or as one hot encoded label $\hat{g}(x, w) \in \{0, 1\}^q$ with

$$\hat{g}_k(x, w) = \begin{cases} 1, & \hat{y}(x, w) = k \\ 0, & \text{else} \end{cases} \tag{4}$$

for $k = 1, \ldots, q$. Given an input sample x^i with one hot label y^i, predicted class label $\hat{g}^i$ (from Eq. (4)) and a loss function $L = L(f(y|x^i, w), y^i)$, we can calculate the gradient of the loss function with respect to the weights $\nabla_w L = \nabla_w L(f(y|x^i, w), \hat{g}^i)$. In our experiments we use the gradient of the negative log-likelihood at the predicted class label, which means

$$L = L(f(y|x^i, w), \hat{g}^i) = -\sum_{y \in \mathcal{C}} \hat{g}^i_y \log\left(f(y|x^i, w)\right) = -\log\left(f(\hat{y}|x^i, w)\right). \tag{5}$$

We apply the following metrics to this gradient:

- Absolute norm ($\|\nabla_w L\|_1$)
- Euclidean norm ($\|\nabla_w L\|_2$)
- Minimum ($\min(\nabla_w L)$)
- Maximum ($\max(\nabla_w L)$)
- Mean ($\text{mean}(\nabla_w L)$)
- Skewness ($\text{skew}(\nabla_w L)$)
- Kurtosis ($\text{kurt}(\nabla_w L)$)

These metrics can either be applied layerwise by restricting the gradient to those weights belonging to a single layer in the neural network or to the whole gradient on all layers.

The metrics can be sampled over the input X and conditioned to the event of either correct or incorrect classification. Let $T(w)$ and $F(w)$ denote the subset of correctly and incorrectly classified samples for the network $f(y|x, w)$, respectively. Given a metric M (e.g. the entropy E or any gradient based one), the two conditioned distributions $M(X, w)|_{T(w)}$ and $M(X, w)|_{F(w)}$ are further investigated. For a threshold t, we measure $P(M(X, w) < t \,|\, T(w))$ and $P(M(X, w) \geq t \,|\, F(w))$ by sampling X. If both probabilities are high, t gives a good separation between correctly and incorrectly classified samples. This concept can be transfered to the detection of out-of-distribution samples by defining these as incorrectly classified. We term this procedure (classifying $M(X, w) < t$ vs. $M(X, w) \geq t$) *meta classification*.

Since there are many possible ways to compute thresholds t, we compute our results threshold independent by using **A**rea **U**nder the **R**eceiver **O**perating **C**urve (AUROC) and **A**rea **U**nder the **P**recision **R**ecall curve (AUPR). For any chosen threshold t we define

$$\begin{aligned} TP &= \#\{\text{correctly predicted positive cases}\}, \\ TN &= \#\{\text{correctly predicted negative cases}\}, \\ FP &= \#\{\text{incorrectly predicted positive cases}\}, \\ FN &= \#\{\text{incorrectly predicted negative cases}\}. \end{aligned}$$

and can compute the quantities

$$\begin{aligned} R = TPR &= \frac{TP}{TP + FN} && \text{(True positive rate or Recall)}, \\ FPR &= \frac{FP}{FP + TN} && \text{(False positive rate)}, \\ P &= \frac{TP}{TP + FP} && \text{(Precision)}. \end{aligned}$$

When dealing with threshold dependent classification techniques, one calculates TPR (R), FPR and P for many different thresholds in the value range of the variable. The AUROC is the area under the receiver operating curve, which has

the FPR as ordinate and the TPR as abscissa. The AUPR is the area under the precision recall curve, which has the recall as the ordinate and the precision as abscissa. For more information on these performance measures see [7].

The AUPR is in general more informative for datasets with a strong imbalance in positive and negative cases and is sensitive to which class is defined as the positive case. Because of that we are computing the AUPR-In and AUPR-Out, for which the definition of a positive case is reversed. In addition the values of one variable are multiplied by -1 to switch between AUPR-In and AUPR-Out as in [11].

3 Meta Classification – A Benchmark Between Maximum Softmax Probability and Gradient Metrics

We perform all our statistical experiments on the EMNIST data set [6], which contains 28×28 gray scale images of 280 000 handwritten digits (0–9) and 411 302 handwritten letters (a–z, A–Z). We train the CNNs only on the digits, in order to test their behavior on untrained concepts. We split the EMNIST data set (after a random permutation) as follows:

- 60,000 digits (0–9) for training
- 20,000 digits (0–9) for validation
- 200,000 digits (0–9) for testing
- 20,000 letters (a–z, A–Z) as untrained concepts

Additionally we included the CIFAR10 library [12], shrinked and converted to gray scale, as well as 20,000 images generated from random uniform noise. All concepts can be seen in Fig. 1.

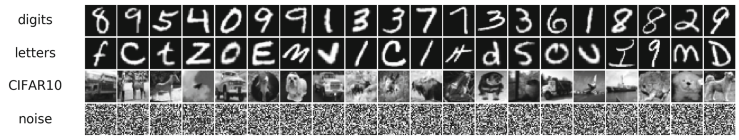

Fig. 1. Different concepts used for our statistical experiments

The architecture of the CNNs consists of three convolutional (conv) layers with 16 filters of size 3×3 each, with a stride of 1, as well as a dense layer with a 10-way softmax output. Each of the first two conv layers are equipped with leaky ReLU activations

$$LeakyReLU(x) = \begin{cases} x, & x > 0 \\ 0.1x, & x < 0 \end{cases} \tag{6}$$

and followed by 2×2 max pooling. We employ L^2 regularization with a regularization parameter of 10^{-3}. Additionally, dropout [17] is applied after the first and third conv layer. The dropout rate is 33%.

The models are trained using stochastic gradient descent with a batch size of 256, momentum of 0.9 and categorical cross entropy as cost function. The initial learning rate is 0.1 and is reduced by a factor of 10 every time the average validation accuracy stagnates, until a lower limit for the learning rate of 0.001 is reached. All models were trained and evaluated using Keras [5] with Tensorflow backend [1]. Note, that the parameters where chosen from experience and not tuned to any extent. The goal is not to achieve a high accuracy, but to detect the uncertainty of a neural network reliably.

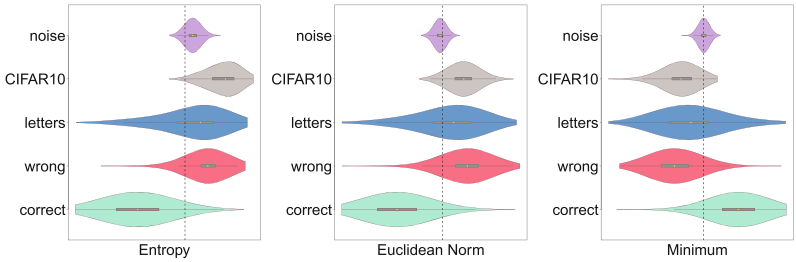

Fig. 2. Empirical distribution for entropy, euclidean norm and minimum applied to correctly predicted and incorrectly predicted digits from the test data (green and red) of one CNN. Further distributions are generated from EMNIST samples with unlearned letters (blue), CIFAR10 images (gray) and uniform noise images (purple). (Color figure online)

In this section, we study the performance of gradient metrics, the softmax baseline and the entropy in terms of AUROC and AUPR for EMNIST test data, thus considering the *error and success prediction* problem, formulated in [11]. First of all we demonstrate that gradient metrics are indeed able to provide good separations. Results for the entropy, euclidean norm and minimum are shown in Fig. 2 (green and red). Note that we have left out the mean, skewness and kurtosis metric, as their violin plots showed, that they are not suitable for a threshold meta classifier.

In what follows we define EMNISTc as the set containing all correctly classified samples of the EMNIST test set and EMNISTw as the set containing all incorrectly classified ones. From now on we resample the data splitting and use ensembles of CNNs. More precisely, the random splitting of the 280,000 digit images in training, validation and test data is repeated 10 times and we train one CNN for each splitting. In this way we train 10 CNNs that differ with respect to initial weights, training, validation and test data. We then repeat the above meta classification for each of the CNNs. With this non parametric bootstrap, we try to get as close as possible to a true sampling of the statistical law underlying the EMNIST ensemble of data and obtain results with statistic validity.

Table 1. AUROC, AUPR-In (EMNISTc as positive case) and AUPR-Out (EMNISTc as negative case) values for the threshold classification on the softmax baseline, entropy as well as selected gradient metrics. All values are in percentage and averaged over 10 differently initialized CNNs with distinct splittings of the training data. Values in brackets are the standard deviation of the mean in percentage. To get the standard deviation within the sample, multiply by $\sqrt{10}$.

Metric	EMNISTc/ EMNISTw	EMNISTc/ EMNIST letters	EMNISTc/ CIFAR10	EMNISTc/ uniform noise
AUROC				
Softmax baseline	**97.82** (0.03)	87.62 (0.20)	99.13 (0.11)	92.95 (1.86)
Entropy	97.74 (0.04)	**88.44** (0.21)	**99.24** (0.03)	**93.52** (1.83)
Absolute norm	97.77 (0.03)	87.22 (0.16)	98.19 (0.03)	90.66 (1.96)
Euclidean norm	97.78 (0.03)	87.27 (0.17)	98.38 (0.02)	91.05 (1.92)
Minimum	97.78 (0.03)	87.30 (0.20)	98.40 (0.03)	90.50 (2.16)
Maximum	97.70 (0.03)	86.92 (0.20)	98.31 (0.04)	87.05 (2.70)
Standard deviation	97.78 (0.03)	87.26 (0.17)	98.38 (0.02)	90.98 (1.93)
AUPR-In				
Softmax baseline	**99.97** (0.00)	98.39 (0.03)	**99.98** (0.00)	99.31 (0.19)
Entropy	99.97 (0.00)	98.38 (0.04)	99.95 (0.00)	**99.36** (0.19)
Absolute norm	99.97 (0.00)	**98.42** (0.02)	99.89 (0.00)	99.07 (0.21)
Euclidean norm	99.97 (0.00)	98.42 (0.02)	99.90 (0.00)	99.11 (0.21)
Minimum	99.97 (0.00)	95.20 (5.40)	99.90 (0.00)	99.05 (0.23)
Maximum	99.97 (0.00)	95.03 (5.65)	99.89 (0.00)	98.67 (0.31)
Standard deviation	99.97 (0.00)	95.04 (5.70)	99.90 (0.00)	99.11 (0.21)
AUPR-Out				
Softmax baseline	39.96 (0.57)	59.04 (0.37)	77.10 (1.90)	40.10 (4.87)
Entropy	**95.56** (0.05)	**60.36** (0.42)	**91.27** (0.39)	**42.46** (5.38)
Absolute norm	95.28 (0.06)	58.39 (0.37)	66.62 (0.35)	33.08 (3.58)
Euclidean norm	95.30 (0.06)	58.27 (0.38)	70.81 (0.52)	34.03 (3.68)
Minimum	95.36 (0.05)	58.76 (0.42)	72.72 (0.36)	33.00 (3.83)
Maximum	95.32 (0.06)	55.01 (0.41)	74.59 (0.71)	26.84 (3.02)
Standard deviation	95.30 (0.06)	58.26 (0.38)	70.75 (0.52)	33.88 (3.66)

Table 1 shows that the softmax baseline as well as some selected gradient metrics exhibit comparable performance on the test set in the error and success prediction task. Column one corresponds to the empirical distributions depicted in Fig. 2 for 200,000 test images.

In a next step we aggregate entropy and all gradient based metrics (evaluated on the gradient of each layer in the CNN) in a more sophisticated

classification technique. Therefore we choose a variety of regularized and unregularized logistic regression techniques, namely a **G**eneralized **L**inear **M**odel (GLM) equipped with the logit link function, the **L**east **A**bsolute **S**hrinkage and **S**election **O**perator (LASSO) with a L^1 regularization term and a regularization parameter $\lambda_1 = 1$, the ridge regression with a L^2 regularization term and a regularization parameter of $\lambda_2 = 1$ and finally the Elastic net with one half L^1 and one half L^2 regularization, which means $\lambda_1 = \lambda_2 = 0.5$. For details about these methods, cf. [10].

To include a non linear classifier we train a feed forward NN with one hidden layer containing 15 rectified linear units (ReLUs) with L^2 weight decay of 10^{-3} and 2-way softmax output. The neural network is trained in the same fashion as the CNNs with stochastic gradient descent. Both groups of classifiers are trained on the EMNIST validation set. Results for the logistic regression techniques can be seen in Table 2 (column one) and those for the neural network in Table 3 (first row of each evaluation metric). For comparison we also include the entropy and softmax baseline in each table. The regression techniques perform equally well or better compared to the softmax baseline. This is however not true for the NN. For the logistic regression types including more features from early layers did not improve the performance, the neural network however showed improved results. This means the additional information in those layers can only be utilized by a non linear classifier.

4 Recognition of Unlearned Concepts

A (C)NN, being a statistical classifier, classifies inside the prescribed label space. In this section, we empirically test the hypothesis that test samples out of the label space will be all misclassified, however at a statistically different level of entropy or gradient metric, respectively. We test this hypothesis for three cases: First we feed the CNN with images from the EMNIST letter set and determine the entropy as well as the values for all gradient metrics for each of it. Secondly we follow the same procedure, however the inputs are gray scale CIFAR10 images coarsened to 28×28 pixels. Finally, we use uncorrelated noise that is uniformly distributed in the gray scales with the same resolution. Roughly speaking, we test empirical distributions for unlearned data that is close to the learned concept as in the case of EMNIST letters, data that represents a somewhat remote concept as in the case of CIFAR10 or, as in the case of noise, do not represent any concept at all.

We are classifying the output of a CNN on such input as incorrect label, this way we solve the *in- and out-of-distribution detection problem* from [11], but are still detecting misclassifications in the prescribed label space. The empirical distributions of unlearned concepts can be seen in Fig. 2. As we can observe, the distributions for incorrectly classified samples are in a statistical sense significantly different from those for correctly classified ones. The gradient metrics however are not able to separate the noise samples very well, but also resulting in an overall good separation of the other concepts, as for the entropy. The

threshold classification evaluation metrics can be seen in Table 1. For the logistic regression results in Table 2 one can see that the GLM is inferior to the other methods. Regression techniques with a regularization term like LASSO, Ridge and Elastic net are performing best. We get similar AUROC values as for the threshold classification with single metrics, but can improve between 5% and 14.08% over the softmax baseline in terms of AUPR-Out values for unknown concepts, showing a better generalization.

Table 2. Average AUROC, AUPR-In and AUPR-Out values for different regression types trained on the validation set and all metric features including the entropy but excluding the softmax baseline. The values are averaged over 10 CNNs and displayed in percentage. The values in brackets are the standard deviations of the mean in percentage. To get the standard deviation within the sample, multiply by $\sqrt{10}$.

Metric/Regression technique	EMNISTc/ EMNISTw	EMNISTc/ EMNIST letters	EMNISTc/ CIFAR10 noise	EMNISTc/ uniform
AUROC				
Softmax baseline	**97.82** (0.03)	87.62 (0.20)	99.13 (0.11)	92.95 (1.86)
Entropy	97.74 (0.04)	88.44 (0.21)	**99.42** (0.10)	93.52 (1.83)
GLM	94.76 (0.70)	85.94 (0.46)	80.26 (5.46)	89.41 (2.90)
LASSO	97.75 (0.03)	**89.34** (0.17)	99.23 (0.03)	93.86 (1.04)
Ridge	97.59 (0.03)	88.63 (0.11)	98.93 (0.02)	**94.08** (0.67)
Elastic net	97.79 (0.06)	89.27 (0.24)	98.82 (0.06)	93.47 (0.67)
AUPR-In				
Softmax baseline	**99.97** (0.00)	**98.39** (0.03)	99.98 (0.00)	99.31 (0.19)
Entropy	99.97 (0.00)	98.38 (0.04)	**99.99** (0.00)	**99.36** (0.19)
GLM	99.81 (0.05)	96.80 (0.21)	95.51 (1.15)	97.81 (0.84)
LASSO	99.97 (0.00)	98.30 (0.06)	99.95 (0.00)	99.33 (0.12)
Ridge	99.97 (0.00)	97.86 (0.04)	99.93 (0.00)	99.36 (0.08)
Elastic net	99.97 (0.00)	98.26 (0.09)	99.92 (0.00)	99.29 (0.08)
AUPR-Out				
Softmax baseline	39.96 (0.57)	59.04 (0.37)	77.10 (1.90)	40.10 (4.87)
Entropy	**95.56** (0.05)	60.36 (0.42)	86.07 (1.76)	42.46 (5.38)
GLM	31.27 (0.79)	57.72 (0.74)	62.90 (6.77)	46.43 (5.24)
LASSO	36.27 (0.32)	**64.04** (0.26)	**91.18** (0.44)	**48.38** (3.12)
Ridge	38.17 (0.34)	61.92 (0.18)	82.95 (0.61)	47.30 (2.20)
Elastic net	38.71 (0.65)	63.43 (0.62)	79.56 (1.76)	45.03 (1.92)

5 Meta Classification with Known Unknowns

In the previous section we trained the meta classifier on the training or validation data only. This means it has no knowledge of entropy or metric distributions for unlearned concepts, hence we followed a puristic approach treating out of distribution cases as unknown unknowns. The classification accuracy could be improved, by extending the training set of the meta classifier with the entropy and gradient metric values of a few unlearned concepts and labeling them as false, i.e., incorrectly predicted. As in the previous sections we then train meta classifiers on the metrics. For this we use the same data sets as [11], namely the omniglot handwritten characters set [14], the notMNIST dataset [4] consisting of letters from different fonts, the CIFAR10 dataset [12] coarsened and converted to gray scale as well as normal and uniform noise. In order to investigate the influence of unknown concepts in the training set of the meta classifier, we used the LASSO regression and the NN introduced in Sect. 3 and supplied them with different training sets, consisting of

- EMNIST validation set
- EMNIST validation set and 200 uniform noise images
- EMNIST validation set, 200 uniform noise images and 200 CIFAR10 images
- EMNIST validation set, 200 uniform noise images, 200 CIFAR10 images and 200 omniglot images

We are omitting the results for the LASSO here, since they are inferior to those of the NN. Including known unknowns into the training set, the NN has far better performance on the unknown concepts, even though the amount of additional training data is small. Noteworthily the validation set together with only 200 uniform noise images increases the results on the AUPR-Out values for all unknown concepts already significantly by 13.74%, even comparable to using all concepts. Together with the fact, that noise is virtually available at no cost, it is a very promising candidate for improving the generalization of the meta classifier without the need of generating labels for more datasets. The in-distribution detection rate of correct and wrong predictions is also increased when using additional training concepts, making it only beneficial to include noise into the training set of the meta classifier. Our experiments show however that normal noise does not have such a high influence on the performance as uniform noise and is even decreasing the in-distribution meta classification performance. All in all we reach a 3.48% higher performance on the out of distribution examples compared to the softmax baseline in AUPR-Out and 0.81% in AUROC, whereas the increase in AUPR-In is marginal (0.12%).

Table 3. AUROC, AUPR-In (EMNISTc is positive case) and AUPR-Out (EMNISTc is negative case) values for a NN meta classifier. "All" contains omniglot, notMNIST, CIFAR10, normal noise and uniform noise. We used 200 samples of each concept that was additionally included into the training set. The supplied features are all gradient based metrics as well as the entropy. The displayed values are averages over 5 differently initialized NN meta classifiers for each of the 10 CNNs trained on the EMNIST dataset. All values are in percentage and the values in brackets are the standard deviations of the mean in percentage. To get the mean within the sample multiply by $\sqrt{10}$.

Wrong datasets	Entropy	Softmax baseline [11]	Training set for the neural network meta classifier			
			EMNIST validation	EMNIST validation + uniform noise	EMNIST validation + uniform noise + CIFAR10	EMNIST validation + uniform noise + CIFAR10 + omniglot
AUROC						
EMNISTw	97.74 (0.02)	**97.84** (0.02)	94.59 (0.17)	96.51 (0.08)	96.69 (0.08)	96.68 (0.07)
Omniglot	**98.05** (0.03)	97.84 (0.03)	94.38 (0.15)	97.29 (0.12)	97.44 (0.10)	97.84 (0.06)
notMNIST	**95.41** (0.15)	95.24 (0.15)	85.90 (0.49)	93.22 (0.46)	94.49 (0.28)	94.86 (0.22)
CIFAR10	**99.24** (0.03)	99.03 (0.04)	81.19 (1.40)	96.27 (0.63)	99.12 (0.03)	99.09 (0.03)
Normal noise	94.36 (0.54)	94.49 (0.50)	56.09 (1.56)	**98.37** (0.08)	98.34 (0.09)	98.17 (0.10)
Uniform noise	94.31 (0.84)	93.87 (0.85)	86.77 (1.16)	94.22 (0.54)	93.87 (0.71)	**94.42** (0.70)
All	96.04 (0.19)	95.83 (0.19)	80.55 (0.49)	95.49 (0.29)	96.36 (0.19)	**96.64** (0.16)
AUPR-In						
EMNISTw	99.97 (0.02)	**99.97** (0.02)	99.89 (0.17)	99.95 (0.08)	99.96 (0.08)	99.96 (0.07)
Omniglot	**99.84** (0.03)	99.82 (0.03)	99.04 (0.15)	99.73 (0.12)	99.75 (0.10)	99.80 (0.06)
notMNIST	**99.45** (0.15)	99.43 (0.15)	95.86 (0.49)	98.83 (0.46)	99.19 (0.28)	99.29 (0.22)
CIFAR10	**99.95** (0.03)	99.94 (0.04)	95.47 (1.40)	99.41 (0.63)	99.94 (0.03)	99.93 (0.03)
Normal noise	99.59 (0.54)	99.60 (0.50)	92.72 (1.56)	99.89 (0.08)	**99.89** (0.09)	99.88 (0.10)
Uniform noise	**99.65** (0.84)	99.62 (0.85)	98.05 (1.16)	99.56 (0.54)	99.53 (0.71)	99.57 (0.70)
All	98.66 (0.19)	98.59 (0.19)	84.98 (0.49)	97.72 (0.29)	98.53 (0.19)	**98.71** (0.16)
AUPR-Out						
EMNISTw	35.83 (0.30)	**39.94** (0.32)	32.95 (0.28)	36.02 (0.39)	35.98 (0.42)	35.33 (0.40)
Omniglot	**83.48** (0.21)	80.45 (0.22)	74.17 (0.38)	80.36 (0.71)	81.46 (0.60)	83.40 (0.39)
notMNIST	74.86 (0.38)	14.59 (0.06)	64.57 (0.42)	71.53 (0.79)	74.91 (0.61)	**75.13** (0.49)
CIFAR10	**91.27** (0.39)	87.38 (0.46)	54.45 (1.17)	73.93 (2.39)	90.84 (0.55)	89.82 (0.64)
Normal noise	54.98 (2.16)	57.32 (1.79)	18.57 (0.76)	**68.73** (1.73)	67.89 (1.77)	65.12 (1.74)
Uniform noise	37.97 (2.50)	36.63 (2.23)	56.66 (1.90)	58.56 (2.70)	56.59 (2.82)	**59.53** (2.94)
All	89.17 (0.40)	88.07 (0.39)	75.64 (0.48)	89.38 (0.47)	91.23 (0.35)	**91.55** (0.32)

6 Conclusion and Outlook

We introduced a new set of metrics that measures the uncertainty of deep CNNs. These metrics have a comparable performance with the widely used entropy and maximum softmax probability to meta-classify whether a certain classification proposed by the underlying CNN is presumably correct or incorrect. Here the performance is measured by AUROC, AUPR-In and AUPR-Out. Entropy and softmax probability perform equally well or slightly better than any single member of the new gradient based metrics for the detection of unknown concepts like EMNIST letters, gray scale converted CIFAR10 images and uniform noise where simple thresholding criteria are applied. But still, our new metrics allow

contributions of different layers and weights to the total uncertainty. Combining the gradient metrics together with entropy in a more complex meta classifier increases the ability to identify out-of-distribution examples, so that in some cases these meta classifiers outperform the baseline. Additional calibration by including a few samples of unknown concepts increases the performance significantly. Uniform noise proved to raise the overall performance, without the need of more labels. Overall the results for the classification of correct or incorrect predictions increased when the meta classifier was supplied with more distinct concepts in the training set. It seems that the higher number of uncertainty metrics helps to better hedge the correctly classified samples from the variety of out of sample classes, which would be difficult, if only one metric is available. Note that this increase in meta classification is particularly valuable, if one does not want to deteriorate the classification performance of the underlying classifier by additional classes for the known unknowns.

As future work we want to evaluate the performance and robustness of such gradient metrics on different tasks in pattern recognition. Further features could be generated by applying the metrics to activations rather than gradients. One could also investigate the possibility of generating artificial samples, labeled as incorrect, for the training set of the meta classifier in order to further improve the results.

Acknowledgement. We thank Fabian Hüger and Peter Schlicht from Volkswagen Group Research for discussion and remarks on this work. We also thank the referees for their comments and criticism helping us to improve the paper.

References

1. Abadi, M., Agarwal, A., Barham, P., et al.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). http://tensorflow.org/
2. Bendale, A., Boult, T.E.: Towards open set deep networks. CoRR abs/1511.06233 (2015). http://arxiv.org/abs/1511.06233
3. Berger, J.O.: Statistical Decision Theory and Bayesian Analysis. Springer, New York (1980). https://doi.org/10.1007/978-1-4757-4286-2
4. Bulatov, Y.: NotMNIST dataset (2011). http://yaroslavvb.blogspot.de/2011/09/notmnist-dataset.html
5. Chollet, F., et al.: Keras (2015). https://github.com/fchollet/keras
6. Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: EMNIST: an extension of MNIST to handwritten letters (2017). http://arxiv.org/abs/1702.05373
7. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, 25–29 June 2006, pp. 233–240 (2006). https://doi.org/10.1145/1143844.1143874
8. Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, ICML 2016, vol. 48, pp. 1050–1059. JMLR.org (2016). http://dl.acm.org/citation.cfm?id=3045390.3045502

9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint (2014). http://arxiv.org/abs/1412.6572
10. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics. Springer, New York (2001). https://doi.org/10.1007/978-0-387-84858-7
11. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. CoRR abs/1610.02136 (2016). http://arxiv.org/abs/1610.02136
12. Krizhevsky, A.: CIFAR-10 (2009). https://www.cs.toronto.edu/~kriz/cifar.html
13. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. CoRR abs/1607.02533 (2016). http://arxiv.org/abs/1607.02533
14. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction (2015)
15. Liang, S., Li, Y., Srikant, R.: Principled detection of out-of-distribution examples in neural networks. CoRR abs/1706.02690 (2017). http://arxiv.org/abs/1706.02690
16. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**, 379–423, 623–656 (1948). http://math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf
17. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014). http://jmlr.org/papers/v15/srivastava14a.html
18. Szegedy, C., et al.: Intriguing properties of neural networks. CoRR abs/1312.6199 (2013). http://arxiv.org/abs/1312.6199
19. Yuan, X., He, P., Zhu, Q., Bhat, R.R., Li, X.: Adversarial examples: attacks and defenses for deep learning. CoRR abs/1712.07107 (2017)