Detecting unusual input to neural networks

Jörg Martin¹ · Clemens Elster¹

Accepted: 2 September 2020 / Published online: 30 October 2020
© The Author(s) 2020, corrected publication 2021

Abstract

Evaluating a neural network on an input that differs markedly from the training data might cause erratic and flawed predictions. We study a method that judges the unusualness of an input by evaluating its informative content compared to the learned parameters. This technique can be used to judge whether a network is suitable for processing a certain input and to raise a red flag that unexpected behavior might lie ahead. We compare our approach to various methods for uncertainty evaluation from the literature for various datasets and scenarios. Specifically, we introduce a simple, effective method that allows to directly compare the output of such metrics for single input points even if these metrics live on different scales.

Keywords Deep learning · Trustworthiness · Fisher information · Uncertainty · Out-of-distribution

1 Introduction

The reliability and performance of a machine learning algorithm depends crucially on the data used for training it [2]. Having an incomplete training set or encountering input data that have unprecedented deviations might lead to unexpected, erroneous behavior [3–5]. In this work we consider the task of judging whether an input to a neural network, trained for classification, is *unusual* in the sense that it is different from the training data, which is also sometimes referred to as out-of-distribution or out-of-data in the literature [6–9]. For this purpose we consider a quantity based on the Fisher information matrix [10]. A similar quantity was shown by the authors to be useful for the related task of detecting adversarial examples [11]. Two kinds of scenarios are studied:

- The input data was *modified*, e.g. by being noisy or contorted.
- The training set is *incomplete*, by missing some structural component that will emerge in practice.

Our considerations are carried out along various datasets and in comparison with related methods from the literature. In particular, we present an easy and efficient method to

directly compare different metrics that judge the uncertainty behind a prediction.

We consider neural networks that are trained for classifying inputs into C classes $1, \dots, C$. After application of a softmax activation the output of a neural network for an input x can be read as a vector of probabilities $(p_\theta(y|x))_{y=1,\dots,C}$, where we wrote θ for the parameters of the neural network (usually weights and biases) [12]. The class $\hat{y}$ where this vector is maximal is then the prediction of the network. It is quite tempting to consider $p_\theta(\hat{y}|x)$ as a measure for the confidence behind this prediction. The leftmost column of Fig. 1 shows an image from the Intel Image Classification dataset [1] and the output of a neural network that was trained on it, details will follow in Section 3 below. The image is classified correctly as “sea” with $p(\hat{y}|x) = 98.8\%$. In this article we are more interested in the probability of *misclassification*

$$1 - p(\hat{y}|x) = 1.2\%. \quad (1)$$

One might expect that the higher this probability, the more unreliable the prediction of the network. The second column of Fig. 1 shows the same image but with an inverted green color channel. While the classification remains “sea”, the probability of error rises to 14.1% which appears to be a sensible behavior. Unfortunately, the behavior of the output probability is not always meaningful. The rightmost column of Fig. 1 shows the same picture with Gaussian noise added. The image is misclassified as showing a “forest” but with a quite small error probability $1 - p(\hat{y}|x)$ of around 2.1%. The fact that the softmax probability is not fully exhaustive or even misleading in judging the reliability of the prediction

✉ Jörg Martin
joerg.martin@ptb.de

¹ Physikalisch-Technische Bundesanstalt (PTB),
Braunschweig and Berlin, Germany

Fig. 1 An image showing the sea from the Intel Image Classification dataset [1] in its original form (left), after inverting its green color channel in RGB format (middle) and after adding Gaussian noise (right). Below we denote the prediction $\hat{y}$ of the neural network from Section 3.1 together with its *error probability* $1 - p(\hat{y}|x)$ and the

quantity $\langle \mathcal{F}_\theta(x) \rangle$ based on the *Fisher information* we introduce in Section 2. We observe in this example that the naive error probability can be misleading, while the Fisher information indicates an unusual input due to its value close to 100%

of the network led to various developments in the literature like a Bayesian judgement of uncertainty [13–19] and deep ensembles [7]. We here study the behavior of a quantity, called the *Fisher form* and denoted by $\mathcal{F}_\theta$, that evaluates if the input x differs from what the network “has learned”. $\mathcal{F}_\theta(x)$ was introduced by the authors for the first time in [11]. This article differs from that previous work in three different aspects: first, and most importantly, while in [11] the Fisher form was used for the purpose of adversarial detection [20, 21] we here study its behavior on unusual data points. Secondly, we use a slightly updated definition in (4) that makes use of the entropy. Finally, we introduce some normalization $\langle \dots \rangle$ that allows us to compare the Fisher form to quantities that live on different scales so that the actual considered quantity is $\langle \mathcal{F}_\theta(x) \rangle$. In Fig. 1 we see that this quantity $\langle \mathcal{F}_\theta(x) \rangle$ shows a quite natural behavior on the depicted images, where values close to 100% indicate an unusual input.

The main contribution of this article can be summarized as follows:

- We discuss various test cases for unusual input to neural networks.
- We show that the Fisher form can be used to detect such input.
- We introduce a normalization that allows to directly compare different measures of unusualness.

The article is structured as follows: In Section 2 we introduce, and motivate, the method used in this article and introduce the normalization that allows us to compare different metrics. Section 3 presents the results for various datasets and splits in two halves: in Subsection 3.1 we study the effect of modifying the input, while Subsection 3.2 considers the case where the training data lack some

structural aspects. Finally, we provide some conclusions and an outlook for future research.

2 Method

A slightly more complete summary of the output probabilities than (1) can be gained by considering the *entropy* instead [10, 16]:

$$\mathcal{H}_\theta(x) = - \sum_{y=1}^C p_\theta(y|x) \log p_\theta(y|x). \quad (2)$$

If the entropy is high the output probabilities $(p(y|x))_{y=1,\dots,C}$ are roughly equal in magnitude and the prediction is uncertain. $\mathcal{H}_\theta(x)$ suffers still from the same drawback than $p_\theta(\hat{y}|x)$: Specifically, it completely ignores the uncertainties of θ and uses their point estimates only. One way around this problem is to use another, more elaborate distribution such as an approximate to the posterior predictive or a mixed distribution from a deep ensemble.

In [11] the authors introduced a quantity that gives a deeper insight than (2) and is based on the Fisher information. The Fisher information matrix for a neural network with output $p_\theta(y|x)$ for an input x and a p -dimensional parameter θ equals $\mathbb{F}_\theta(x) = \sum_{y=1}^C \nabla_\theta p_\theta(y|x) \cdot \nabla_\theta \log p_\theta(y|x)^T$. Even a smaller network has around $p = 10^6$ parameters, which makes $\mathbb{F}_\theta$ a matrix with 10^{12} entries and is thus, in practice, infeasible. A way out, proposed in [11], is to consider the effect of this matrix in a specific direction v , i.e. to consider the quadratic form

$$\mathcal{F}_\theta(x) = v^T \mathbb{F}_\theta(x) v, \quad (3)$$

where v has the same dimensionality as θ . We will refer to (3) as the *Fisher form*. The quantity (3) measures *how much*

information is gained/lost in a (small) step in the direction v . More precisely, the Kullback-Leibler divergence between $p_\theta(y|x)$ and $p_{\theta+\varepsilon v}(y|x)$ can be written as $\frac{\varepsilon^2}{2} \mathcal{F}_\theta(x) + \mathcal{O}(\varepsilon^3)$ [22]. After the parameter θ is learned, this divergence will be large for x that are informative, that is *different from those used to infer the learned value of θ* . To use (3) we have to fix a suitable direction v . A natural choice, which we will use throughout this article, is the negative gradient of the entropy

$$v = -\partial_\theta \mathcal{H}_\theta(x), \quad (4)$$

where $\mathcal{H}_\theta(x)$ is as in (2). The motivation behind a choice like (4) is as follows: Once trained the network will produce an input with the highest probability for one class $\hat{y}$, its “prediction”, and lower probabilities for all other classes $y \neq \hat{y}$. The vector v then denotes the direction in parameter space that decreases the entropy, in other words that tends to increase the probability for $\hat{y}$ further and to decrease all other probabilities.

If the input is unusual and leads to a wrong classification $\hat{y}$, then a step in the direction of v will increase the information substantially, as the used x will usually be completely different from the training data that were used for inferring θ . For another choice of v compare [11].

Taking the expression $\mathcal{F}_\theta(x)$ as the useful consequence that we can rewrite the quadratic form in (3) as $\mathcal{F}_\theta(x) = \sum_{y=1}^C D_v p_\theta(y|x) D_v \log p_\theta(y|x)$, where $D_v f(\theta) = v^T \nabla_\theta f(\theta)$ denotes the directional derivative w.r.t θ in direction of v , which can be computed either directly or using a finite difference approximation that avoids the need for backpropagation [11].

Before analyzing how the quantity (3) performs in practice, there is one final point that needs some consideration. While quantities such as the Fisher form (3) or the entropy (2) can each be compared for two different data points there is no direct possibility for a comparison between the value of the entropy and the Fisher information for the same data point. Moreover, requiring someone applying these quantities to get an intuition first before they can judge whether a quantity is “high enough” to rise suspicion is rather unsatisfactory.

When faced with a binary decision problem, such as “will produce unexpected behavior or not”, we can use the *receiver operator characteristic* (ROC) and the associated *area under the ROC curve* (AUC) to compare different metrics, cf., for example [11]. However, a comparison based on the ROC/AUC still has the drawback that it is not applicable for *single data points* and, moreover, that not all problems can be seen as a binary decision. We here describe a rather easy, yet efficient approach that solves these issues.

Let us introduce the following *normalization* for a quantity $q(x)$ and a test set T

$$\langle q(x) \rangle = \frac{\#\{x' \text{ in } T \text{ s.t. } q(x') < q(x)\}}{\#\{x' \text{ in } T\}}, \quad (5)$$

where $\#$ counts the number of elements. In other words $\langle q(x) \rangle$ denotes fraction of test samples x' in T for which $q(x') < q(x)$. By construction, $\langle q(x) \rangle$ depends on the input x and on the test set T that is used. This normalization has a few nice properties: It is

- *bounded*, as it lies always between 0 and 1,
- *monotone*: from $q(x_1) < q(x_2)$ it follows $\langle q(x_1) \rangle < \langle q(x_2) \rangle$,
- *invariant* under strictly monotone transformations such as taking a logarithm or scaling.

This normalization allows us to compare different measures of reliability/ uncertainty, since relations such as $\langle \mathcal{F}_\theta(x) \rangle = 75\%$ and $\langle \mathcal{H}_\theta(x) \rangle = 75\%$ have a similar meaning: 75% of the test samples have a smaller value than the one for x . If we see the magnitude of a quantity such as $\mathcal{F}_\theta(x)$ as an indicator for the “unusualness” of a data point x , then a value of $\langle \mathcal{F}_\theta(x) \rangle$ close to 100% can be seen as a strong indication that the reliability for a prediction based on x will be quite questionable. In particular a relation such as $\langle \mathcal{F}_\theta(x) \rangle > \langle \mathcal{H}_\theta(x) \rangle$ for an x that causes a wrong prediction will indicate that $\mathcal{F}_\theta(x)$ detected the underlying uncertainty more clearly than $\mathcal{H}_\theta(x)$ (based on the test set), so that a normalization via $\langle \dots \rangle$ make measures of reliability directly comparable.

Figure 2 shows the distribution of $\langle 1 - p_\theta(\hat{y}|x) \rangle$ (left), the normalized version of (1), the normalized entropy $\langle \mathcal{H}_\theta(x) \rangle$ (middle) and Fisher form $\langle \mathcal{F}_\theta(x) \rangle$ (right) for the same neural network as in Fig. 1 for the original images (in green) and those with inverted green color channel (in magenta). The test set for the normalization was chosen equal to the test set from [1]. Note that only the Fisher information has a distinct peak at high values for the modified images.

3 Experiments

In this section we discuss the behavior of the Fisher form $\mathcal{F}_\theta$, the entropy $\mathcal{H}_\theta$, the entropy $\mathcal{H}_\theta^{\text{DE}}$ for the mixture distribution predicted by a deep ensemble [7] and the entropy $\mathcal{H}_\theta^{\text{GD/BD}}$ of an approximate to the posterior predictive. More precisely, $\mathcal{H}_\theta^{\text{DE}}$ refers to the average of entropies produced by each member of the ensemble, and $\mathcal{H}_\theta^{\text{GD/BD}}$ to the mean of entropies obtained by repeated predictions of the trained net using dropout. All deep ensembles were constructed using 5 networks, trained

Fig. 2 Using the normalization (5) we can directly compare of $1 - p(\hat{y}|x)$ from (1), the entropy $\mathcal{H}_\theta(x)$ from (2) and the Fisher form $\mathcal{F}_\theta(x)$ from (3) and ignore the fact that these quantities live on different scales. The histograms depict the behavior of these quantities for

images from the Intel Image Classification dataset [1] unmodified (in green) and after inverting the green color channel (in magenta) as in the middle of Fig. 1

independently of each other. To approximate the posterior predictive we used Gaussian dropout [14] for MNIST and Bernoulli dropout [13] (with rate 0.5) for all other examples, as the latter seemed to converge substantially faster. The entropy for the distribution produced by the Gaussian dropout will be denoted by $\mathcal{H}_\theta^{\text{GD}}$ and we will write $\mathcal{H}_\theta^{\text{BD}}$ for Bernoulli dropout. The architectures of all used networks are sketched in the Appendix of this article.

In Section 3.1 we will analyze the behavior of $\mathcal{F}_\theta$, $\mathcal{H}_\theta$, $\mathcal{H}_\theta^{\text{DE}}$ and $\mathcal{H}_\theta^{\text{GD/BD}}$ on data points that are modified, either by adding noise or by a transformation based on a variational autoencoder. Section 3.2 studies the effect of missing certain features while training.

3.1 Modified data

As a first example we will consider the MNIST dataset [23] for digit recognition. We trained a convolutional neural network (Fig. 9a) for 25 epochs until it reached an accuracy of around 99% on both, the test and training set. Finally we trained a *variational autoencoder* (VAE) [15], sketched in Fig 9b, for 100 epochs until it reached a mean squared error accuracy of around 1% of the pixel range. Using a VAE is a popular form for interpolating between data points [24, 25]. To this end, we split the VAE into an encoder E and a decoder D . For single input images $\tilde{x}_0$ and $\tilde{x}_1$ we then take $z_0 = E(\tilde{x}_0)$, $z_1 = E(\tilde{x}_1)$ and set for λ between 0 and 1:

$$x_\lambda = D(z_0 + \lambda(z_1 - z_0)). \quad (6)$$

Figure 3a shows the interpolation between an image showing a 4 and an image showing a 9 for different values of λ . The curves above the reconstructed images show the dependency of $\langle \mathcal{H}_\theta(x_\lambda) \rangle$ (in black), $\langle \mathcal{F}_\theta(x_\lambda) \rangle$ (in magenta), the Gaussian dropout entropy $\langle \mathcal{H}_\theta^{\text{GD}}(x_\lambda) \rangle$ (in blue) and the deep ensemble entropy $\langle \mathcal{H}_\theta^{\text{DE}}(x_\lambda) \rangle$ (in cyan) on λ . We used the MNIST test set as the set T for normalization. All methods show a similar behavior with a distinct peak in the transition phase.

As a next dataset we will consider the *Intel Image Classification* dataset [1] that contains images of 6 different classes (street, building, forest, mountain, glacier, sea). We touched this example already in the introduction of this article. We trained a convolutional neural network (Fig. 10b) on this dataset for 15 epochs until it reached an accuracy of around 80% on the test set of [1], which we also used for normalization. We here want to consider the effect of two modifications on the data points. The effect of inverting the green color channel was already shown in Figs. 1 and 2. This modification lets the accuracy of the network drop from around 80% to 30%. Figure 4 shows the ROC for $\mathcal{H}_\theta$, $\mathcal{F}_\theta$, $\mathcal{H}_\theta^{\text{DE}}$ and $\mathcal{H}_\theta^{\text{BD}}$ (now with Bernoulli dropout) for detecting such modified images, where we used once more the test set as reference. Note that a normalization as in (5) is not needed as the ROC is invariant under monotone transformations. The AUC for $\mathcal{H}_\theta$ and $\mathcal{H}_\theta^{\text{BD}}$ are 0.63 and 0.72, while $\mathcal{H}_\theta^{\text{DE}}$ and $\mathcal{F}_\theta$ both share an AUC of 0.76.

Next, we consider the effect of noise. The upper plot in Fig. 3b shows the evolution of $\langle \mathcal{H}_\theta(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{BD}}(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{DE}}(x_\lambda) \rangle$ and $\langle \mathcal{F}_\theta(x_\lambda) \rangle$ where x_λ is now given as

$$x_\lambda = (x + \lambda \cdot \varepsilon)_{0,1}, \quad (7)$$

with x being a single image from the Intel Image Classification dataset that shows a street, ε being an array of standard normal noise of the same shape as x and with $(\dots)_{0,1}$ denoting the pixelwise min-max-normalization to the range of the images (from 0 to 1). The lower row of Fig. 3b displays x_λ for various λ .

The crosses in the upper plot in Fig. 3b depict if the network classifies the image correctly as a street (in green) or whether it chooses the wrong class (in red). We observe that, while all quantities $\langle \mathcal{H}_\theta(x_\lambda) \rangle$, $\langle \mathcal{F}_\theta(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{DE}}(x_\lambda) \rangle$ and $\langle \mathcal{H}_\theta^{\text{BD}}(x_\lambda) \rangle$ share a similar trend, the Fisher form is well above all other quantities in the transition phase where the noise starts flipping the classification. Inspecting the maximal softmax output $p(\hat{y}|x_\lambda)$ (in dashed gray) shows that beyond $\lambda = 0.3$ this probability is almost identical

Fig. 3 **a** Behavior of the normalized entropies $\langle \mathcal{H}_\theta(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{GD}}(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{DE}}(x_\lambda) \rangle$ for the softmax output, Gaussian dropout and a deep ensemble (in black, blue and cyan) and the Fisher form $\langle \mathcal{F}_\theta(x_\lambda) \rangle$ (in purple) when interpolating two images from the MNIST dataset [23] using a variational autoencoder. x_λ and λ are as in (6). The small crosses indicate the classification as 4 (in green) and 9 (in

blue), **b** Behavior of the normalized entropies $\langle \mathcal{H}_\theta(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{GD}}(x_\lambda) \rangle$, $\langle \mathcal{H}_\theta^{\text{DE}}(x_\lambda) \rangle$ for the softmax output, Bernoulli dropout and a deep ensemble (in black, blue and cyan) and the Fisher form $\langle \mathcal{F}_\theta(x_\lambda) \rangle$ (in purple) when perturbing an image from the Intel Image Classification dataset by Gaussian noise. x_λ and λ are as in (7). The crosses indicate whether the street has been classified correctly (in green) or not (in red)

Fig. 4 The ROC for detecting whether an image from the Intel Image Classification dataset was modified by inverting its green color channel (as in the middle of Fig. 1) via the entropies $\mathcal{H}_\theta(x)$, $\mathcal{H}_\theta^{\text{BD}}(x)$, $\mathcal{H}_\theta^{\text{DE}}(x)$ (in black, blue and cyan) and the Fisher form $\mathcal{F}_\theta(x)$ (in purple)

to 1, which finally pushes all quantities down towards 0. To understand why this also finally pushes the Fisher form down to 0 recall that v as defined in (4) is pointing towards the direction in parameter space that makes the prediction $\hat{y}$ more likely. Once $\hat{y}$ is predicted with a probability essentially equal to 1 there is no distinct direction left. Interestingly, we can observe a similar phenomenon in the shape of a small dent around $\lambda = 0.25$ where $\hat{y}$ starts to change and where there is thus no clear prediction.

In Table 1 we summarize the AUCs for all scenarios considered in this article. As the interpolation of the MNIST images (Fig. 3a) and the noisification of images from the Intel Image Classification dataset (Fig. 3b) do not come with an obvious binary problem for which an AUC can be computed we considered for them the task to distinguish $x_{\lambda_1}, x_{\lambda_2}$ with $\lambda_1 = 0$ and $\lambda_2 = 0.5$ or 0.3 respectively, with x_{λ_i} as in (6) and (7) and randomly drawn x . For the MNIST interpolation we fixed the target image to be the same as Fig. 3a (showing a 9). On all datasets in this article, except for MNIST, we see that the Fisher form $\mathcal{F}_\theta$ and Deep Ensemble entropy $\mathcal{H}_\theta^{\text{DE}}$ outperform the simple entropy approach and those based on Dropout. The second example of Section 3 will present an example where the Fisher form is working while the Deep Ensemble is not (last column of Table 1).

Table 1 AUCs for the different scenarios, listed by dataset, and methods studied in Sections 3.1 and 3.2

	Section 3.1			Section 3.2	
	<i>MNIST</i>	<i>Intel Image Classification</i>		<i>Credit Card</i>	<i>DogsVsCats</i>
	(6): $x_{0.5} \leftrightarrow x_0$	(7): $x_{0.3} \leftrightarrow x_0$	inv. green	<i>Fraud Det.</i>	8 epochs
$\mathcal{H}_\theta$	0.83	0.56	0.63	0.63	0.18
$\mathcal{H}_\theta^{\text{BD}}$	/	0.55	0.72	0.63	0.18
$\mathcal{H}_\theta^{\text{GD}}$	0.80	/	/	/	/
$\mathcal{H}_\theta^{\text{DE}}$	0.86	0.70	0.76	0.70	0.17
$\mathcal{F}_\theta$	0.84	0.71	0.76	0.70	0.78

For the interpolation of the MNIST data and the noisification of the Intel Image Classification data in Section 3.1 we forged the considered problems into a binary form by asking to distinguish between x_{λ_1} with $\lambda_1 = 0$ and x_{λ_2} with $\lambda_2 = 0.5$ or 0.3 , where x_{λ_i} is as in (6) and (7) respectively.

3.2 Incomplete training data

While in Section 3.1 we looked at the behavior of $\mathcal{F}_\theta$, $\mathcal{H}_\theta$, $\mathcal{H}_\theta^{\text{BD}}$ and $\mathcal{H}_\theta^{\text{GD/DE}}$ for modified data, we will now analyze another scenario. What happens if the data we used in training was *incomplete*, in the sense that it lacked one or several important features that will occur in its application after training [3, 27, 28]?

As a first example we use the *Credit Card Fraud Detection* dataset [26, 29]. This, extremely unbalanced, dataset contains 284,807 transactions with 492 frauds which are indicated by a binary label. The data are provided anonymized by a principal component analysis. The distribution of the main component, dubbed ‘V1’, is shown in Fig. 5a. We split the data in two halves. A training set, where V1 is below a threshold of -3.0 (in green), and a test set where V1 is bigger than the threshold (in red). On the green part we trained, balancing the classes, a small fully connected neural network (Fig. 10b) for 20 epochs

until it reached an accuracy of 95% on a smaller subset of the “train” set that was excluded from the training process. On the “test” set from Figure 5a the accuracy is with 79% markedly lower.

Figure 5b shows the ROC for the detection whether a data point x is from the red part of Fig 5a based on $\mathcal{H}_\theta(x)$, $\mathcal{F}_\theta(x)$, $\mathcal{H}_\theta^{\text{DE}}(x)$ and $\mathcal{H}_\theta^{\text{BD}}(x)$. The AUC for $\mathcal{F}_\theta$ and $\mathcal{H}_\theta^{\text{DE}}$ is around 0.70, while $\mathcal{H}_\theta$ and $\mathcal{H}_\theta^{\text{BD}}$ arrive both at 0.63.

We will now consider a problem where the splitting is more subtle and, maybe, more realistic. For this purpose we will use the *DogsVsCats* dataset from [30]. We trained a variational autoencoder (Fig. 11b) [15] on the combined collection of dog images from [30, 31] for 30 epochs until it reached a mean squared error accuracy of around 2% of the pixel range. A reconstructed image can be seen in Fig. 6a. We then selected the bottleneck node, called En from now on, that exhibits the largest variance when evaluated on these dog images. Figure 6b shows the distribution of En for dog images from [30] together with median(En) =

Fig. 5 **a** Distribution of the main PCA component V1 from the Credit Card Fraud Detection dataset [26]. We split this dataset in two parts, where one is used for training a neural network (in green) and the other one for testing on unusualness (in red) using the metrics from Section 2, **b** ROC for detecting whether a data point is from the red or green area in Fig 5a using the entropy (in black), the uncertainty computed from a deep ensemble (in blue) and the Fisher form (in purple)

Fig. 6 **a** Output of the used VAE for a dog image from DogsVsCats dataset [30] where the variable in latent space was taken to be the mean of the distribution of the bottleneck node En . For this image the bottleneck node En is larger than the threshold 0.3, **b** Distribution of the bottleneck node En of the used VAE for dog images. En is the mean

bottleneck node with largest variance within the dog images of the DogsVsCats dataset, **c** The output for the same as input for Fig. 6a but with the sign of En flipped so that (8) is satisfied. Note that while the background in Fig. 6a was bright, it is now turned dark

0.3. What is the meaning of En ? Fig. 6c shows the same reconstructed image as in 6a but with flipped sign of En .

We can see that the background of the depicted image becomes darker. Looking at various data points from the dataset hardens this impression so that we will see this as the “meaning” of En , although there is, probably, no one-to-one correspondence with a human interpretation. We trained a convolutional neural network (Fig. 11a) on the data from [30], but by omitting those dog images with

$$En < \text{median}(En) = 0.3, \quad (8)$$

i.e. those on the left side of the median in Fig. 6b or, with the interpretation above, those images with dark background. We trained this network for 20 epochs until it reached an accuracy of around 80% on a subset of the images with $En \geq \text{median}(En)$ that we ignored while training. Evaluating now the network on dog images from [30, 31] with $En < \text{median}(En)$ lets the accuracy drop to 1%! In other words almost all dogs images with $En < 0.3$ are classified as cats.

Our task is now to detect whether an input is *unusual* in the sense that (8) holds. Figure 7 depicts the evolution of the AUC for the quantities $\mathcal{F}_\theta$, $\mathcal{H}_\theta$, $\mathcal{H}_\theta^{\text{DE}}$ and $\mathcal{H}_\theta^{\text{BD}}$ for each epoch and the accuracy of the trained network (in dashed gray). The difference between the Fisher form $\mathcal{F}_\theta$ and all other forms is striking. The quantities based on entropy seem to completely fail in detecting the “unusualness” of images with $En < 0.3$. In fact they yield AUCs below 0.5, thereby marking the actual training images as more suspicious than the modified images on which the classification actually fails. The Fisher form $\mathcal{F}_\theta$ however is yielding a value well above 0.5 for a wide range of the

training. After around 15 epochs the AUC for the Fisher form starts to drop down as well.

As this coincides roughly with the point where the accuracy starts stabilizing we suspect that this is a similar effect as we observed in Fig. 3b: Once the maximal probability of the softmax output becomes almost identical

Fig. 7 Evolution the AUC for detecting whether a dog image from the DogsVsCats dataset [30] is “unusual” in the sense of (8) while training on data points where (8) is wrong. As metrics for detecting we use the entropies $\mathcal{H}_\theta(x)$, $\mathcal{H}_\theta^{\text{BD}}(x)$, $\mathcal{H}_\theta^{\text{DE}}(x)$ (in black, blue and cyan) and the Fisher form $\mathcal{F}_\theta(x)$ (in purple). The dashed gray line shows the accuracy of the trained network

Fig. 8 The behavior of the entropy $\langle \mathcal{H}_\theta^{\text{DE}}(x) \rangle$ predicted by the deep ensemble and the Fisher form $\langle \mathcal{F}_\theta(x) \rangle$ for dog images from [30, 31]. The ordinate shows the value of the bottleneck node En which was used to formulate the criterion (8). Red marks those data points that were classified wrong (as a cat). The histograms show the distribution of the marginal, again split in correct and wrong classified data points

to one, this forces the Fisher information to decrease. In fact it turns out that after 20 epochs the maximal probabilities on the unusual data are in average around 99%. Compare this to the tiny accuracy of 1%! We will focus now on a point before this, apparently unreasonable, saturation happened, namely on the trained state after 8 epochs, marked by a gray cross.

The normalization from (5), for which we here used the training data, allows us to compare the behavior of $\mathcal{H}_\theta(x)$, $\mathcal{H}_\theta^{\text{DE}}(x)$, $\mathcal{H}_\theta^{\text{BD}}(x)$ and $\mathcal{F}_\theta(x)$ for single data points x . As all entropies show a quite a similar behavior for this example, we will only compare $\mathcal{H}_\theta^{\text{DE}}$ and $\mathcal{F}_\theta$. We drew 1200 images all showing dogs and depicted the value of En versus $\langle \mathcal{H}_\theta^{\text{DE}}(x) \rangle$ and $\langle \mathcal{F}_\theta(x) \rangle$ in Fig. 8. Wrong classifications as cats are marked in red. The median 0.3 of En is drawn as a blue dashed line, so that the “unusual data points” are below this line. The marginal distributions of the detection quantities and of En are shown, split according to classified correctly (in black) and wrongly (in red).

Note first, that indeed most of the images satisfying (8) are classified wrong, so that the network is in fact not suitable for classifying these data points. Moreover, the deep ensemble entropy is not able to capture that there is something off with these data. Even worse, the corresponding points are cumulated in the lower left corner which should actually indicate a high confidence in the prediction for these points. The Fisher information on the other hand leads to a cumulation in the lower right corner and thus correctly detects many of the “unusual” data points. In fact, even for “normal” data points we observe that most of those that were classified wrong have an encoder value En close to 0.3 and possess again a high $\mathcal{F}_\theta$

4 Conclusion and outlook

We studied the *Fisher form* to detect whether an input is unusual, in the sense that it differs from the data that were used to infer the learned parameters. Spotting such a disparity is important, as it can substantially decrease the reliability. Several examples for this have been presented in Section 3. We observed that the Fisher form performs equally or even better than competing methods in detecting unusual inputs. In particular, we introduced a normalization that allows to directly compare these metrics for single data points.

The last example treated in this article showed an effect, which could be a starting point for future research. As we observed in Fig. 7 the ability to detect unusual data might erode for a too long training. When only looking at the accuracy for an incomplete test set, as we did in Fig. 7, such a process might happen unnoticed. This somehow different flavor of overfitting could be worth studying. Measures to remedy this phenomenon might also be worth investigating. For the Fisher form we saw this effect arising as a consequence of $p(\hat{y}|x)$ being essentially identical to 1 which lets the direction v from (4) become ambiguous. Exploring alternative choices for v or applying probability calibration [32] might therefore allow to improve the performance of the Fisher form further.

Another point that would deserve a more detailed investigation is of a more conceptual nature. For each of the examples treated in this work we used a single network to evaluate its Fisher form. One might wonder whether the effect we observed when considering unusual data will differ for another network whose training was carried out

with, say, different initial conditions. As neural networks are bound to end up in local minima it is not obvious whether a data point will be unusual for two of these networks, even if they are trained on the same training set. However, this is not what we observed. In fact, the behavior of the Fisher form seems rather robust in this regard. Modifications of the network architecture also seem to have no substantial effect. It is hard to say, whether this indicates that neural networks tend to learn similar patterns when presented with the same data. Looking at the quadratic form (3) for more than only one direction v could help to shed a bit of light on this question.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix

Fig. 9 **a** Architecture for the convolutional neural network trained on the MNIST dataset that we use in Section 3.1. For convolutional layers, we marked incoming channels by the number next to the arrow, **b** Architecture for the variational autoencoder we train on the MNIST dataset and use in Section 3.1. Note that the bottleneck is random: From the 10 “means” and 10 “variances” in the bottleneck we construct 10 Gaussian distributions from which we draw 10 random numbers that are then fed through the decoder

Fig. 10 **a** Architecture for the convolutional neural network trained on the Intel Image classification dataset that we use in Section 3.1. For convolutional layers, we marked incoming channels by the number next to the arrow, **b** Architecture for the fully connected neural network trained on the Credit Card Fraud Detection dataset that we use in Section 3.2

Fig. 11 **a** Architecture for the convolutional neural network trained on the DogsVsCats dataset that we use in Section 3.2. For convolutional layers, we marked incoming channels by the number next to the arrow, **b** Architecture for the variational autoencoder we train on the dog images from [30, 31] and use in Section 3.2. Note that the bottleneck is random: From the 500 “means” and 500 “variances” in the bottleneck we construct 500 Gaussian distributions from which we draw 500 random numbers that are than fed through the decoder

References

- Bansal P (2019) Intel image classification. Available on <https://www.kaggle.com/puneet6060/intel-image-classification>, Online; accessed 20th April 2020
- Cortes C, Jackel LD, Chiang W-P (1995) Limits on learning machine accuracy imposed by data quality. In: Advances in neural information processing systems, NIPS, pp 239–246
- Amodei D, Olah C, Steinhardt J, Christiano P, Schulman J, Mané D (2016) Concrete problems in ai safety. arXiv preprint arXiv:1606.06565
- Zhang M (2015) Google Photos Tags Two African-Americans As Gorillas Through Facial Recognition Software. Forbes
- Marcus G (2018) Deep learning: A critical appraisal. arXiv preprint arXiv:1801.00631
- Hendrycks D, Gimpel K (2017) A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: 5th international conference on learning representations, ICLR
- Lakshminarayanan B, Pritzel A, Blundell C (2017) Simple and scalable predictive uncertainty estimation using deep ensembles. In: Advances in neural information processing systems, NIPS, pp 6402–6413
- Kendall A, Gal Y (2017) What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems, NIPS, pp 5574–5584
- Liang S, Li Y, Srikant R (2018) Enhancing the reliability of out-of-distribution image detection in neural networks. In: 6th international conference on learning representations, ICLR
- Cover TM, Thomas JA (2012) Elements of information theory. John Wiley & Sons
- Martin J, Elster C (2020) Inspecting adversarial examples using the fisher information. Neurocomputing 382:80–86
- Bishop CM (2006) Pattern recognition and machine learning. Springer, Berlin
- Gal Y, Ghahramani Z (2016) Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: International conference on machine learning, pp 1050–1059
- Kingma DP, Salimans T, Welling M (2015) Variational dropout and the local reparameterization trick. In: Advances in neural information processing systems, NIPS, pp 2575–2583
- Kingma DP, Welling M (2014) Auto-encoding variational bayes. In: Bengio Y, LeCun Y (eds) 2nd international conference on learning representations, ICLR
- Gal Y (2016) Uncertainty in deep learning. University of Cambridge 1:3
- Gal Y, Hron J, Kendall A (2017) Concrete dropout. In: Advances in neural information processing systems, NIPS, pp 3581–3590
- Blundell C, Cornebise J, Kavukcuoglu K, Wierstra D (2015) Weight uncertainty in neural networks. In: Proceedings of the 32nd international conference on machine learning, PMLR, pp 1613–1622
- Filos A, Farquhar S, Gomez AN, Rudner TimGJ, Kenton Z, Smith L, Alizadeh M, deKroon A, Gal Y (2019) A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. arXiv preprint arXiv:1912.10481

20. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow IJ, Fergus R (2014) Intriguing properties of neural networks. In: 2nd international conference on learning representations, ICLR
21. Goodfellow IJ, Shlens J, Szegedy C (2015) Explaining and harnessing adversarial examples. In: 3rd international conference on learning representations, ICLR
22. Kullback S (1997) Information theory and statistics. Courier Corporation
23. LeCun Y, Cortes C, Burges CJ (2010) Mnist handwritten digit database. AT&T Labs 2:18. [Online]. Available: <http://yann.lecun.com/exdb/mnist>
24. Berthelot D, Raffel C, Roy A, Goodfellow IJ (2019) Understanding and improving interpolation in autoencoders via an adversarial regularizer. In: 7th international conference on learning representations, ICLR
25. Smith L, Gal Y (2018) Understanding measures of uncertainty for adversarial example detection. In: Proceedings of the thirty-fourth conference on uncertainty in artificial intelligence, UAI, pp 560–569, AUAI Press
26. ULB MLG (2018) Credit card fraud detection. Version 3 - Available on <https://www.kaggle.com/mlg-ulb/creditcardfraud>, Online; accessed 20th April 2020
27. Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608
28. Snoek J, Oquab Y, Fertig E, Lakshminarayanan B, Nowozin S, Sculley D, Dillon J, Ren J, Nado Z (2019) Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In: Advances in neural information processing systems, NIPS, pp 13969–13980
29. DalPozzolo A, Caelen O, LeBorgne Y-A, Waterschoot S, Bontempi G (2014) Learned lessons in credit card fraud detection from a practitioner perspective. Expert systems with applications 41(10):4915–4928
30. kaggle (2014) Dogs vs. cats competition. Available on <https://www.kaggle.com/c/dogs-vs-cats>, Online; accessed 20th April 2020
31. SchubertSlySchubert (2018) Cat and dog. Version 1 - Online; accessed 20th April 2020
32. Guo C, Pleiss G, Sun Y, Weinberger KQ (2017) On calibration of modern neural networks. In: Proceedings of the 34th international conference on machine learning, ICML, 70 of Proceedings of Machine Learning Research, pp 1321–1330, PMLR

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Jörg Martin received his Ph.D. in mathematics at Humboldt University Berlin in 2018 and is currently working in the data analysis group of the Physikalisch-Technische Bundesanstalt (PTB). His main research interests are the robustness, explainability and uncertainty evaluation for neural networks.

Clemens Elster received his Ph.D. on designing methods for experimental design and the optimization of noisy functions in 1993 from Freiburg University, Germany. He is currently leading PTBs Working Group Data Analysis and Measurement Uncertainty. His research interests are mainly in the field of statistical data analysis with focus on Bayesian methods.