This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Generative Classifiers as a Basis for Trustworthy Image Classification

Radek Mackowiak*
Visual Learning Lab, Heidelberg University

Lynton Ardizzone*
Visual Learning Lab, Heidelberg University

Ullrich Köthe
Visual Learning Lab, Heidelberg University

Carsten Rother
Visual Learning Lab, Heidelberg University

* equal contribution

Abstract

With the maturing of deep learning systems, trustworthiness is becoming increasingly important for model assessment. We understand trustworthiness as the combination of explainability and robustness. Generative classifiers (GCs) are a promising class of models that are said to naturally accomplish these qualities. However, this has mostly been demonstrated on simple datasets such as MNIST and CIFAR in the past. In this work, we firstly develop an architecture and training scheme that allows GCs to operate on a more relevant level of complexity for practical computer vision, namely the ImageNet challenge. Secondly, we demonstrate the immense potential of GCs for trustworthy image classification. Explainability and some aspects of robustness are vastly improved compared to feed-forward models, even when the GCs are just applied naively. While not all trustworthiness problems are solved completely, we observe that GCs are a highly promising basis for further algorithms and modifications. We release our trained model for download in the hope that it serves as a starting point for other generative classification tasks, in much the same way as pretrained ResNet architectures do for discriminative classification.
Code: `github.com/VLL-HD/trustworthy_GCs`

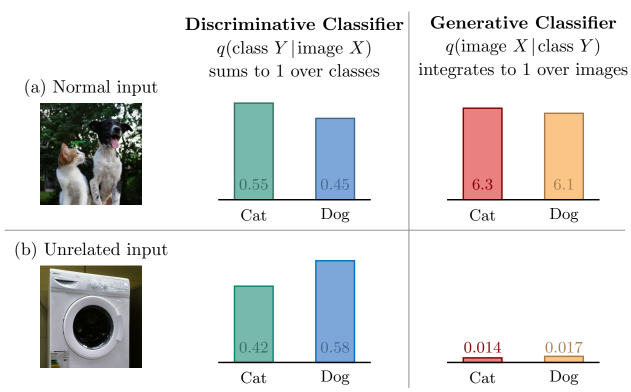

Figure 1: Example of one advantage of generative classifiers: The class posterior of a DC always sums up to 1, while the likelihoods of the GC do not have this restriction, constituting inherently more informative outputs. E.g. the GC can show if a prediction is uncertain because the input agrees with both classes, or with neither.

1. Introduction

Generative classifiers (GCs) and discriminative classifiers (DCs) represent two contrasting ways of solving classification tasks. In short, while standard DCs model the class probability given an input directly, $p(\text{class} \mid \text{image})$ (e.g. softmax classification), generative classifiers (GCs) take the opposite approach: They model the likelihood of the input image, conditioned on each class, $p(\text{image} \mid \text{class})$. The actual classification is then performed by finding the class under which the image has the highest likelihood.

The application of GCs has so far been limited to very simple datasets such as MNIST, SVHN and CIFAR-10/100. For any practical image classification tasks, DCs are used exclusively, due to their excellent discriminative performance. In principle, GCs are said to have various advantages over DCs, which align with the term *trustworthiness*. In general agreement with [24], we understand trustworthiness as the combination of explainability and robustness.

Explainability: DCs based on deep neural networks are notorious for being 'black boxes', prompting many developments in the field of explainable AI. In the taxonomy laid out in [18], most commonly used algorithms fall into categories I or II: post-hoc methods that visualize how a network processes information (I), or that show its internal representations (II). The explanations can vary depending on the chosen method, and there is no guarantee that the results faithfully reflect what the DC is doing internally.

In contrast, GCs bring to mind Feynman's mantra "What I cannot create, I do not understand". As GCs are able to model the input data itself, not just the class posteriors, they have fundamentally more informative outputs. For instance, GCs allow us to tell if a decision between two classes is uncertain because the input agrees well with *both* classes, or with *neither* (see Fig. 1). In addition, most GCs have interpretable latent spaces with meaningful features, allowing for the actual decision process to be directly visualized without post-hoc techniques. Therefore, it could be argued that GCs belong to category III of the explainability taxonomy [18], i.e. methods that intrinsically work in an explainable way, without relying on additional algorithms.

Robustness: A second large concern about the practical use of deep learning based classification systems is their robustness, which can have different meanings, depending on the context. In particular, GCs have been assumed to be superior to DCs in terms of generalization under dataset shifts [51, 39] and accurately calibrated posteriors [3]. In addition, a big advantage of GCs is their capability to explicitly identify abnormal inputs in a natural way, thus indicating when a decision should not be trusted. Furthermore, GCs were found to be more robust towards adversarial attacks [33] and allow for their explicit detection [17].

It is still unclear if GCs can also manifest these advantages in more complex tasks while remaining competitive to DCs in task performance. For example, the authors of [15] find while GCs can successfully detect adversarially attacked MNIST images, this already fails for the CIFAR-10 dataset. The authors of [34, 30] observe that detection of other forms of OoD data also fails in various ways for natural images. In [16], the authors cast doubt on whether GCs can be used for high-dimensional input data at all.

In light of this background, our work makes the following contributions: *(i)* We design and train a GC that performs at a level relevant to practical image classification, demonstrated on the ImageNet dataset. *(ii)* We show various native explainability techniques unique to GCs. *(iii)* We examine the model in terms of robustness.

Overall, we find our GC to work better than a comparable DC in terms of trustworthiness. However, we do observe that previous findings on superior generalization under dataset shift [51] and immunity to adversarial attacks [41] do not hold for the ImageNet dataset. For other aspects of robustness, our GC shows some great benefits, such as naturally detecting OoD inputs and adversarial attacks.

2. Related Work

Years before the deep learning revolution, works such as [37, 51, 39] already compared the properties of GCs vs DCs, theoretically and experimentally, with agreement that GCs are more robust and more explainable. Works like [6, 5, 54] presented models that combine the aspects of GCs and DCs, to reach a more favourable trade-off compared to each extreme. However, all these works consider simple problems, and with the unmatched task performance later delivered by deep-learning based DCs in the 2010s, GCs became rarely used.

As one example of more recent work, [16] investigates normalizing-flow based GCs trained on natural images. The authors find that naively trained GC models achieve very poor classification performance, and argue that this is due to some model properties that are not properly penalized by maximum likelihood training. Later, [3] propose that this problem can be avoided by training with the Information Bottleneck loss function instead. The authors of [32] modify the problem, and train a GC on features previously extracted from a standard feed-forward network. For all these works, the most complex dataset used is CIFAR-100, at a resolution of 32×32 pixels.

So-called hybrid models [38] have been more successful in practice. Here, a likelihood estimation method is involved, commonly for the marginal $p(\text{image})$, while the actual classification is still performed in a discriminative way, using shared features between the two tasks, the main motivation being semi-supervised learning. Notable examples are [29, 14, 11, 35, 20]. They have some fundamental differences to GCs, e.g. that the conditional likelihoods are not directly modeled and the latent space has no explicit class structure.

Concerning OoD detection with generative models, the authors of [34] and later [30] observed that likelihood models trained on natural images fail to detect certain OoD inputs, and may perform significantly worse than random. This problem is addressed e.g. by [36, 10, 43, 45, 55], where different OoD scores are introduced that correct for these shortcomings. These works only consider unconditional likelihood models for OoD detection, while a separate classifier is still needed to perform the actual task. GCs combine both these steps into a single model, simplifying the process and potentially improving OoD detection at the same time.

GCs have also been examined for adversarial defense recently [41, 17, 33]. While these works highlight the potential of GCs, they are limited to simple datasets such as MNIST and SVHN, and do not scale to problems with more than approx. 10 classes, or to natural images [15].

3. Methods

3.1. Invertible Neural Networks

While VAEs have been used as generative classifiers with some success [41, 17, 33], perhaps the most natural choice are normalizing flows, due to their exact likelihood estimation capabilities [13]. The networks used in normalizing flows are so-called invertible neural networks (INNs), a class of neural network architectures that meet the following conditions: (i) the network represents a diffeomorphism by construction (essentially, a smooth and invertible function), (ii) the inversion can be computed efficiently, and (iii) the network has a tractable Jacobian determinant. These conditions place some restrictions on the architecture, e.g. that the number of input and output dimensions have to be equal, and that non-invertible operations such as max-pooling can not be used. In recent years, various different invertible architectures have been developed that fulfill these conditions [12, 13, 4, 19]. In this work, we employ the affine coupling block architecture proposed in [13], with additional modifications, as described in Appendix B.1.

In any generative setting, there are training images X, that follow some unknown image distribution $p(X)$. The goal is then to approximate $p(X)$ as closely as possible with a distribution given by the network, which we denote as $q_\theta(X)$. In the case of normalizing flows, $q_\theta(X)$ is represented by transforming possible inputs X to a latent space Z using an INN f_θ ('flow'), with a prescribed standard normal latent distribution $p(Z) = \mathcal{N}(0, 1)$ ('normalizing'). Then, the change-of-variables formula can be used to compute $q_\theta(X)$ at any point x through

$$q_\theta(x) = p\Big(Z = f_\theta(x)\Big)\ |\det J(x)| \tag{1}$$

with $J \equiv \partial f_\theta / \partial X$ being the Jacobian. It can be shown that the network will learn the true distribution ($q_\theta(X) = p(X)$) by maximizing the expected log-likelihood $\log q_\theta(X)$, as given through Eq. 1 above [47]. After training is complete, the model can not only be used to estimate likelihoods $q_\theta(X)$, but also to generate new samples by inverting the network, in order to map sampled instances of Z back to image space.

In our case, this approach is not sufficient, as we want to use the INN as a generative classifier, meaning we need to model conditional likelihoods $q_\theta(X \mid Y)$. While different approaches for this exist [52, 2], we adopt the form introduced in [25]. Here, the latent distribution is a conditional density $p(Z \mid Y)$: The standard normal distribution $p(Z)$ is replaced with a Gaussian Mixture Model (GMM) containing a unit-variance mixture component per class

$$p(Z \mid Y) = \mathcal{N}(Z; \mu_Y, \mathbb{1}) \tag{2}$$

$$p(Z) = \sum_y p(y)\, p(Z \mid y) = \sum_y p(y) \mathcal{N}(Z; \mu_y, \mathbb{1}) \tag{3}$$

where μ_y is the mean of class y in latent space; and the mixture weights are the class priors $p(y)$, i.e. the frequency of occurrence of each class in the dataset. The conditional likelihood $q_\theta(X|Y)$ can be evaluated with the change-of-variables formula (Eq. 1) as before by replacing the full distribution $p(Z)$ with the appropriate mixture component:

$$q_\theta(X \mid Y) = p\Big(Z = f_\theta(X)\ \Big|\ Y\Big)\ |\det J|\,. \tag{4}$$

3.2. Training INNs with Information Bottleneck

An INN naively trained with a class-conditional log-likelihood loss will perform very poorly as GC, even on mildly challenging tasks [16]. Instead, we require a loss function where the focus on the generative and class-separating capabilities can be explicitly controlled. For this, we utilize the IB objective [49], the ideal loss function for robust classification from an information theoretic point of view. Given some features Z of a network, inputs X, and ground-truth outputs Y, the IB loss consists of two terms using the mutual information I (MI):

$$\mathcal{L}_{\mathrm{IB}} = I(X, Z) - \hat{\beta} I(Y, Z). \tag{5}$$

The MI quantifies the degree of shared information between variables and can be written as $I(V, W) = D_{\mathrm{KL}}(p(V, W) \| p(V)p(W))$. Minimizing the IB loss means maximizing the information about the desired output Y contained in the features, $I(Y, Z)$. Simultaneously, it minimizes the information about the original image contained in the features, $I(X, Z)$, resulting in robust and efficient representations Z. The trade-off between these two aspects is explicitly adjusted by choosing $\hat{\beta}$.

How to apply this objective to INNs is not immediately obvious, as INNs preserve information, and the loss becomes ill-defined. The authors of [3] show that this can be avoided by adding very low noise to the inputs. This is already an established practice in the context of normalizing flows for the purpose of dequantization. From this, the authors go on to derive two loss terms representing the IB objective, $\mathcal{L}_{\mathrm{IB}} = \mathcal{L}_X + \beta \mathcal{L}_Y$. In practice, the two terms amount to the following:

$$\mathcal{L}_X(x) = -\log|\det J_x| + \frac{1}{2} \operatorname*{logsumexp}_{y'} \left(v_{y'}^2 - 2 w_{y'}\right) \tag{6}$$

$$\mathcal{L}_Y(x, y) = \mathrm{onehot}(y) \cdot \operatorname*{log\,softmax}_{y'} \left(\frac{v_{y'}^2}{2} - w_{y'}\right) \tag{7}$$

Hereby, we use $v_y := f(x) - \mu_y$, and $w_y := \log p(y)$ ($\log(1/(\#\text{ classes}))$ for uniform class priors in our case). J_x is the Jacobian $\partial f(x)/\partial x$. y' denotes the summation over all classes in the logsumexp and log softmax operations. The difference between $\hat{\beta}$ in the original IB and β in the loss is a constant weighting factor for convenience [3], producing a sensible objective for manageable values of β in the rough range $(1, 100)$.

Intuitively, we find the following: The $\mathcal{L}_X$-loss forces the data to follow the GMM in latent space, making the INN a generative model. However, it has no effect on the class-conditional aspect, as the class y is summed out. This loss can be rearranged to look similar to the maximum-likelihood-loss used for normalizing flows, but with a GMM as a latent distribution. On the other hand, the $\mathcal{L}_Y$-loss bears resemblance to the categorical cross entropy loss, except that the usual logits are replaced by $\log p(z|y)p(y) = \log p(z, y)$. Therefore, $\mathcal{L}_Y$ is responsible for making the likelihood model conditional on the class, but otherwise ignores the generative performance.

x

3.3. Detecting OoD Inputs

For likelihood-based generative models, detecting OoD inputs is straight forward, by directly utilizing the estimated probability density q_θ: in principle, if an input is outside the support of the training data, and the model has learned the true distribution, the OoD sample should be assigned $\log q_\theta(x) = -\infty$. In practice, it is only required that OoD samples have lower likelihood scores than the training data. From here, any input with an inferred likelihood below a threshold can be treated as OoD. However, in [34], the authors identified various special cases where OoD inputs have an unnaturally high log-likelihood score. This prompted the development of a *typicality-test* in [36], that uses both an upper and a lower threshold. Even better performing extensions to this exist [10, 43, 45, 55], but we choose the typicality-test as the simplest option, to examine the natural capabilities of the model. We slightly modify the typicality-test to make it a traditional hypothesis test, with the null hypothesis being that the input is in-distribution, more details in Appendix A.1. The p-value for the hypothesis test is the fraction of training samples with scores in the OoD-zone, which also equals the false positive rate. To evaluate the OoD detection capabilities independent of the threshold, we vary the p-value of the test and produce a receiver operating characteristic (ROC) curve. The area under this curve (ROC-AUC), in percent, serves as a scalar measurement of the OoD detection capabilities, with ROC-AUC of 100% meaning that the OoD samples and in-distribution samples are perfectly separated, and a value at 50% or below indicates a random performance or worse.

4. Experiments

A detailed description of the network architecture is found in Appendix B.1, we summarize the main points in the following. We construct the invertible network (INN) from affine coupling blocks, as introduced in [13], with various modifications from other recent works [1, 2, 26, 28]. As invertible alternatives to 2×2 max-pooling and global mean-pooling, we use a Haar wavelet transform [2] and a DCT transform [26] respectively.

Because of the similarities between affine coupling blocks and residual blocks as used in a ResNet, we match the design of the INN to that of a standard ResNet-50 wherever possible. The overall layout is summarized in Table 1, c.f. [21, Table 1]. Some differences arise due to the constraint of invertibility: the number of feature channels and the available receptive field vary between the two networks. Regarding the effective rather than maximum receptive field, see Appendix B.2. The invertibility is also associated with an extra cost of parameters and computation, summarized in Appendix Table 5: Both in terms of network parameters, as well as FLOPs for one forward pass, the cost of the INN is about twice as high as a standard ResNet-50. We are optimistic that this overhead can be reduced in the future with more efficient INN architectures.

Layer	**Blocks**	**Im. size**	**Channels**		**R.F.**	
			INN	ResNet	INN	ResNet
Input		224	3	3		
Entry flow	1	112	12	64	8	6
Pool (Haar/max)		56	48	64	10	10
Conv_2_x	3	56	48	256	34	34
Conv_3_x	4	28	192	512	106	90
Conv_4_x	6	14	768	1024	314	266
Conv_5_x	3	7	3072	2048	538	426
Pool (DCT/avg.)		1	150 528	2048	∞	∞

Table 1: For each of the resolution levels in the INN and ResNet-50, the number of coupling/residual blocks and spatial size is given, along with the number of feature channels and the maximum possible receptive field (R.F.).

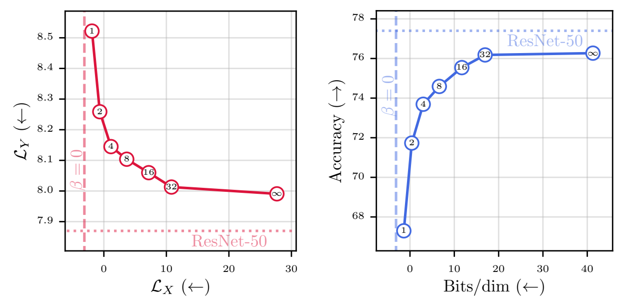

Figure 2: Trade-off between the two losses $\mathcal{L}_X$ and $\mathcal{L}_Y$ (left), and between generative modeling accuracy in bits/dim, and top-1 accuracy (right). Each point represents one model, trained with a different beta. A standard ResNet has no $\mathcal{L}_X$ loss and is shown as a horizontal line. The model with $\beta = 0$ (standard normalizing flow) is missing the $\mathcal{L}_Y$ loss and is shown as a vertical line. The small numbers inside the markers give the value of β of that particular model.

4.1. General Performance

We train several generative classifiers, with the following values for the hyperparameter $\beta \in \{1, 2, 4, 8, 16, 32, \infty\}$. Again, β controls how much the model focuses on the generative likelihood estimation aspect (low β), vs. prioritizing good classification performance (high β). In addition, we include a model trained with $\beta = 0$, i.e. no classification at all, analogous to a standard normalizing flow, as well as a standard feed-forward ResNet-50 [21], i.e. a pure DC.

The primary performance metrics used in Table 2 and Fig. 2 are firstly, the top-1 accuracy on the test set (in our case, the ILSVCR 2012 validation set [40]). We use 10-crop testing, which is most commonly used for performance evaluation in this setting. Secondly, for the generative likelihood estimation performance, we use the bits per dimension ('*bits/dim*') metric, as this is the prevalent evaluation metric for likelihood-based generative models such as normalizing flows. It quantitatively measures the accuracy of the density estimation (i.e. generative performance), explained e.g. in [48], where a lower bits/dim corresponds to a more accurate generative model.

In Table 2, we report the test losses and the two dis-

β	$\mathcal{L}_X^{(\text{test})}$ (↓)	$\mathcal{L}_Y^{(\text{test})}$ (↓)	Bits/dim (↓)	Acc. (%) (↑)	OCE (↓)
1	−1.90	8.52	4.34	67.30	3.87
2	−0.65	8.26	6.14	71.73	4.13
4	1.14	8.14	8.72	73.69	4.31
8	3.66	8.10	12.35	74.59	4.73
16	7.17	8.06	17.43	75.54	4.15
32	10.81	8.01	22.68	76.18	4.94
∞	27.68	7.99	47.01	76.27	5.12
0	−3.11	–	2.59	–	–
ResNet	–	7.87	–	77.40	6.75

Table 2: Test losses and metrics for models trained with different β. Bits/dimension quantifies the performance of density estimation models (see text, smaller is better, i.e. more accurate generative model). As with the original ResNet, the classification accuracy uses 10-crop testing. OCE is the overconfidence error, i.e. how often confident predictions are wrong (see text, smaller is better).

cussed performance metrics for the different models. Further shown in Fig. 2, changing β moves smoothly between the limit cases of a feed-forward network, and a pure density estimation model: the classification accuracy increases continuously with β, but a minor gap remains to the feed-forward ResNet-50, in line with works such as [27]. Simultaneously and as expected, the bits/dim get worse as we move away from a purely generative model ($\beta = 0$).

Lastly, we examine the uncertainty calibration, a quantitative measure of the quality of the predictive posteriors. The full analysis is provided in the Appendix Table 6. Here we only report the overconfidence error 'OCE', which measures the normalized classification error of predictions with a high confidence $C \geq C_{\text{crit}} = 99.7\%$. For instance, if the error rate in these cases is 1.1%, although it should only be 0.3% according to the confidence, this gives an OCE of $1.1/0.3 \approx 3.7$. Our findings are in line with previous works, in that the uncertainty calibration improves with lower β and better generative capabilities [3].

4.2. Explainability

In the following, we demonstrate several examples on how GCs can be used for native and intuitive explanations of the data and the prediction outputs. Certainly, algorithms and approaches exist that can generate similar results for DCs. The point of the following examples is to show that in GCs a range of explanations is available using only the structure of latent space and the learned likelihoods, without requiring additional modifications or algorithms applied in a post-hoc manner.

Visualizing decision-space: The properties of a classification decision are fully determined by the latent code of an input image in relation to the surrounding classes. The only difficulty consists in reducing the high-dimensional latent space to a 2D plot. Fig. 3 shows one possibility: latent codes are visualized in a plane through the centers of the two most probable classes, such that relative distances to the centers and to their connecting axis are preserved. A second approach is given in Appendix C.1, where the classification among a subset of classes can be fully visualized.

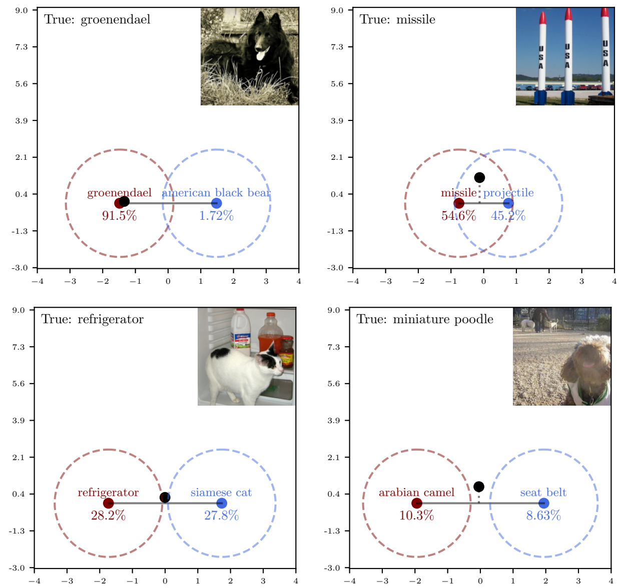

Figure 3: Latent space location of input images (*black point*) in the decision space spanned by the μ_y of the top 5 predicted classes. The horizontal axis of the plot is the axis connecting the top 2 predicted classes (*red and blue points*). The vertical axis of the plot shows the radial distance from the horizontal axis in the 5D space. The illustrative circles are chosen such that in both the vertical and horizontal directions, 90% of the mass of the Gaussian mixture component lies inside. Note that the axes in the plot are scaled differently to make it appear as a circle. Test examples from left to right: a confident in-distribution decision, an uncertain in-distribution decision due to ambiguous classes, an uncertain decision due to multiple plausible image interpretations, an uncertain out-of-distribution decision.

Class similarities: Building on Fig. 3, we see that different classes have various amounts of overlap, which represents the relationship between them. This is not possible for a feed-forward model, as there is no latent space where the input data is embedded in such a way. We observe that the locations μ_y of the Gaussian mixture components are close together for classes that are semantically similar, and far apart for classes that are dissimilar.

Importantly, this also has implications for predictions the model makes. For instance, in Fig. 3, top right, the classes overlap a lot. This means more points will lie in the overlap zone, and consequently more of these decisions will be uncertain, compared to e.g. bottom left, where most inputs will be in only one of two classes. More precisely, the closer two class centers are, the larger is the overlap, and the larger

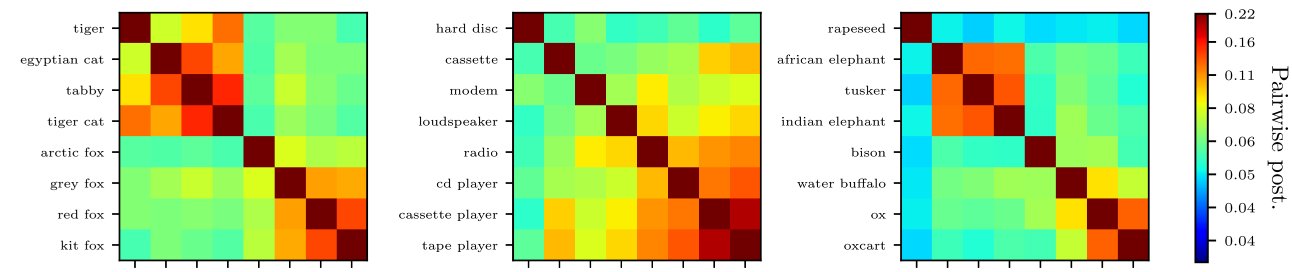

Figure 4: Latent similarity between different classes. The colormap indicates the pairwise distance of the μ_y as well as the expected pairwise posterior, meaning e.g. the binary decision between "tabby cat" and "tiger cat" is associated with 20% expected uncertainty, by construction (see text). The distance on the diagonal is 0 (outside colormap range).

Figure 5: Examples of the prediction heatmaps. Summing the bright areas directly gives the final class prediction. Top: bowtie and sunglasses are located, suit is distributed over a large area. Middle: The head of the bird causes it to be classified as a limpkin, whereas the feathers are more indicative of an eagle or vulture. Bottom: The heatmaps of both Rottweiler and Appenzeller classes are located in the same area (ambiguous classes), while the soccer ball is separate.

the proportion of split decisions between these classes. In fact, if a class A is the top prediction, the expected confidence for any other class B can be worked out explicitly from the distance between μ_A and μ_B in latent space, see Appendix C.2. Some examples are shown in Fig. 4, with the full similarity matrix in Appendix Fig. 16.

These considerations highlight an important fact: the latent mixture model contains a built-in uncertainty between classes. A decision between similar classes will always be uncertain, by the structure of the latent space alone. This may be one of the reasons explaining why the predictive uncertainties are better calibrated in such GCs.

Posterior Heatmaps: To increase the trust in a decision, it is often helpful to show which regions of the image were relevant. Examples are widespread where models e.g. base the decision on the background of the image, not the object in question, or focus only on a specific detail that identifies an object. Approaches such as CAM or GradCAM [56, 42] are used to generate coarse heatmaps showing regions that are influential for a particular decision. With the IB-INN, we can provide such heatmaps as a direct decomposition of the prediction output, meaning they can be understood simply as a different way of representing the model output, rather than a post-hoc explanation technique.

To produce a spatially structured output, we consider the following: Due to the invertibility of every part of the model, we can start from the output z, and transform it back through the DCT operation. Unlike standard mean-pooling, the DCT pooling does not lose any information in either direction. We define the following for short:

$$w^{(y)} = \mathrm{DCT}^{-1}\big(z - \mu_y\big). \tag{8}$$

Importantly, $w^{(y)}$ has the spatial structure of the final convolutional outputs, $w^{(y)}_{kl}$, with height- and width indexes k and l. Because the DCT is linear and orthogonal, it conserves distances, i.e. $\|z - \mu_y\| = \|w^{(y)}\|$, which allows us to write

$$q(z|y) \propto \exp\left(-\frac{\|w^{(y)}\|^2}{2}\right) = \exp\left(-\sum_{kl} \frac{\big(w^{(y)}_{kl}\big)^2}{2}\right) \tag{9}$$

This means the latent density is can be written as a sum over spatial coordinates inside the exponential. We can do the same kind of decomposition to the posterior with a few extra steps, noting $q(y|x) = q(z|y)p(y)/q(z)$. This leads to our heatmap $Q_{\text{Class}}(k,l,y)$, that sums to the class posterior over space in the same way as in Eq. 9:

$$q_\theta(y|x) = \exp\left(\sum_{kl} Q_{\text{Class}}(k,l,y)\right). \tag{10}$$

Q_{Class} has a single hyperparameter that adjusts the contrast of the heatmaps. The derivation is given in Appendix C.4. Examples are shown in Fig. 5. Similarly, we can compute a salience map $Q_{\text{Salience}}(k,l,y)$, that decomposes $q_\theta(x)$ spatially, showing which parts of the image contain the most information according to the model, explained and shown in Appendix C.3.

						Noise			Blur				Weather				Digital			
β	Clean Error	mCE	rel. mCE	Δ entrop.	OoD	Gauss.	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG
0	–	–	–	–	77.51	**94.9**	**94.3**	**98.0**	**95.7**	**89.8**	**88.3**	**89.5**	*38.1*	*43.1*	**94.8**	*44.7*	**96.7**	65.5	63.0	66.2
1	32.7	98.5	116	1.62	67.9	**95.3**	**95.2**	**98.6**	**92.9**	**87.1**	84.9	**87.4**	*33.0*	*45.4*	**96.5**	*43.5*	**97.0**	60.4	61.9	55.6
2	28.27	92.5	119	1.75	73.6	**94.8**	**95.2**	**98.5**	**87.8**	82.6	81.3	84.9	*30.9*	*43.2*	**96.5**	*44.1*	**95.2**	56.6	61.0	*51.2*
4	26.31	88.2	117	1.72	70.84	**92.7**	**93.8**	**97.4**	77.6	76.7	75.6	81.7	*31.0*	*43.2*	**95.5**	*44.5*	**89.2**	*54.1*	61.7	*48.0*
8	25.41	86.8	117	1.81	65.85	**89.3**	**91.2**	**94.6**	56.9	63.5	63.1	73.7	*37.6*	*46.6*	**87.8**	*45.1*	71.2	*53.1*	65.1	*49.1*
16	24.46	84.9	115	1.79	62.43	83.7	84.6	**88.0**	*46.7*	56.7	63.5	67.9	*43.2*	*52.0*	80.2	*45.6*	66.3	*53.3*	62.0	*42.7*
32	23.82	83.1	113	1.71	55.83	81.6	81.5	84.0	*39.8*	*51.6*	*50.1*	*54.8*	*43.9*	*44.3*	61.6	*44.6*	*53.9*	*52.4*	*52.5*	*41.1*
∞	23.73	83.4	114	1.58	44.24	*39.5*	*44.5*	*40.6*	*42.8*	*48.1*	*46.3*	*46.0*	*40.9*	*38.9*	*36.1*	*44.3*	*48.5*	*52.2*	*47.9*	*47.0*
ResNet	22.6	78.2	109	1.51	–	–														

Table 3: We report the error on the unperturbed images (*clean error*), the mean corruption error (*mCE*) and the relative mCE, describing the relative performance degradation caused the corruptions (Δ *entrop.*). Furthermore, we report the OoD ROC-AUC detection score (*OoD*) averaged over all corruptions as well as for the individual corruptions. Meaning of colors: **good detection** $\geq 85\%$; some detection $> 55\%$; *random or worse detection* $\leq 55\%$.

4.3. Robustness

Different Measures of Robustness: In current literature, there is no agreement upon a single measurement that clearly defines robustness in deep learning. In general, the question is how a model reacts to out-of-distribution (OoD) inputs, meaning inputs that do not come from the same distribution as the training data. We identify four different concepts of robustness, which are commonly used:

(1) Especially for dataset shifts that preserve the semantic information, a robust model is one that **retains good performance** for the OoD inputs.

(2) There are other cases where definition (1) is not applicable: There is no 'correct' prediction if the OoD input does not contain any of the classes which were trained for. The second idea of robustness is therefore that the model should at least make **uncertain predictions** for OoD inputs, measured by discrete entropy of the predictive outputs [44]. In reality, standard (non-robust) models make highly confident predictions on OoD data [44].

(3) A robust model can be one that is able to **explicitly detect** OoD inputs. In this case, along with the usual task output, the model has some auxiliary output that indicates whether an input is OoD. The model is robust by explicitly indicating that it's prediction may not be trusted in these cases. GCs are uniquely suited for this, as the estimated likelihood of the inputs can serve as a built-in OoD detection mechanism, but other approaches also exist [31, 23, 9]. To measure this, metrics such as the area under the receiver-operator curve can be used (AUC-ROC).

(4) In the context of adversarial attacks, robustness is commonly understood to be the **amplitude of adversarial perturbation** necessary to trick the model [53].

Handling Corrupted Images: We first consider the robustness test in the sense of (1) established by [22]. Here, the existing ImageNet validation images are corrupted with 5 severity levels in 15 different ways, examples are shown in Appendix D.1. The authors propose the mean corruption error (mCE) and the relative mean corruption error (rel. mCE) score to measure the robustness of a classifier. We also measure the increase in predictive entropy as in [44] for robustness in the sense of (2), and perform OoD detection (3).

As can be seen in Table 3 the GC does not show an improvement compared to the ResNet in terms of (rel.) mCE, regardless of β. However, it infers more uncertain predictions on corrupted data. For OoD detection, we observe overall better scores for smaller values for β. We find the GC trained with $\beta = 2$ to be the most robust classification model: It is able to detect a wide range of corruption types while being a reasonably good classifier (4.54 percentage point classification accuracy gap compared to the $\beta = \infty$ model and 5.67 gap compared to the ResNet).

Handling Adversarial Attacks: We are interested in finding out if generative classifiers are more robust to adversarial attacks in the sense of (4). We are not proposing a new, competitive method of adversarial attack defense, the goal is simply to examine whether GCs are naturally more robust to adversarial attacks on ImageNet, in the same way it was observed for e.g. MNIST previously [33, 41]. For this, we perform the well established 'Carlini-Wagner' white-box targeted attack introduced in [8], which optimizes the following objective:

$$\mathcal{L}_{\mathrm{CW}} = \|x - x_{\mathrm{adv}}\|^2 + c \cdot \mathcal{L}_{\mathrm{class}}^{(\kappa)}(y_{\mathrm{target}}), \tag{11}$$

i.e. the attacked image x_{adv} should be close to the original image x, while being classified as a target class y_{target}. κ is a hyperparameter that specifies how large the difference in logits should be between y_{target} and the next highest class, controlling how confident the classifier will be forced to be in its (wrong) decision. When facing a model such as a GC, which can detect attacks, it is also possible to add an extra loss term $\mathcal{L}_{\mathrm{detect}}$ in order to fool the detection mechanism as well, as proposed in [7]:

$$\mathcal{L}_{\mathrm{CWD}} = \|x - x_{\mathrm{adv}}\|^2 + c \cdot \mathcal{L}_{\mathrm{class}}^{(\kappa)}(y_{\mathrm{target}}) + d \cdot \mathcal{L}_{\mathrm{detect}} \tag{12}$$

The full formulation of the attack objectives is given in Appendix D.2.

For evaluation, we examine standard CW attacks and two detection-fooling attacks with $d = 66$ and $d = 1000$,

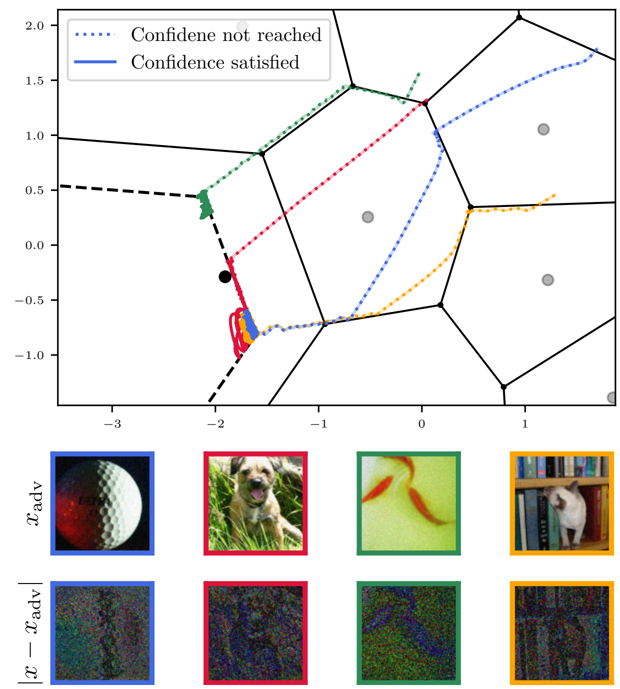

Figure 6: Trajectory of four adversarial attacks shown in latent space (*colored curves*), with $\kappa = 1$, $d = 0$ (standard CW). The large black dot indicates the position of μ_{target}, the target class being 'Harvestman (spider)'. The solid black lines are the decision boundaries to the surrounding classes. The dashed black lines are the boundaries of the region where the classifier is fooled with sufficiently high confidence corresponding to κ. In the dotted section of the colored trajectories, the classifier is not yet fooled with sufficiently high confidence. In the solid section, the classifier has been fooled, and the attack only tries to reduce the perturbation. Below, the four perturbed images are shown, along with the absolute perturbation. More examples and detailed explanation in Appendix D.3.

each for three values of $\kappa \in \{0.01, 1, \infty\}$. For these 9 attack settings, we measure the L2 perturbation of the images after the attack and the ROC-AUC of the attack detection.

The results are presented in Fig 7, from which we make several key observations. We conclude that the GC requires roughly $2\times$ larger perturbations for a standard adversarial compared to the ResNet, in line with [33]. We also observe the attack detection mechanism to be partially robust against attacks; even with $d = 1000$ it still works reasonably well for some cases. Furthermore, within this setting, the size of the perturbation is even more extensive compared to the standard attack setting. Fooling the classifier to predict the wrong class with greater confidence also increases the necessary perturbation as the detectability. An intuitive visualization of the adversarial attack in latent space is shown in Fig. 6.

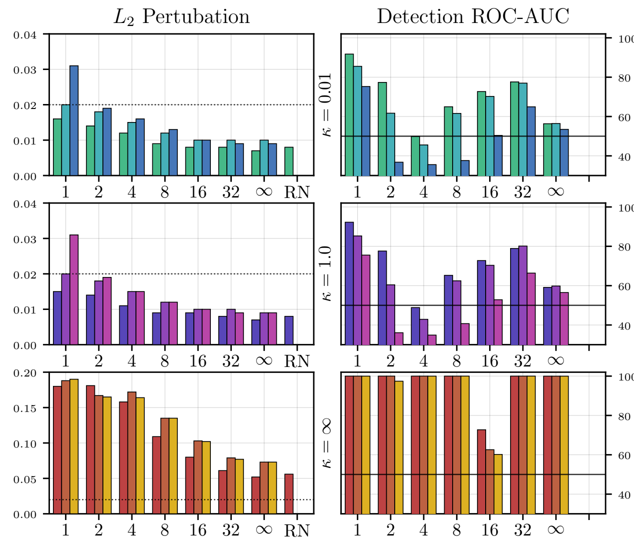

Figure 7: Behaviour of GCs under adversarial attacks. The first column of plots shows the mean perturbation, the second shows the detection ROC-AUC. The three rows of plots correspond to adversarial attacks with $\kappa = 0.01$ (any confidence for the target prediction is enough), $\kappa = 1$ (should have high confidence), and $\kappa = \infty$ (should be as confident as possible). The labels on the x-axis give the values of β, 'RN' is a ResNet-50. The three bars for each β correspond to: standard adversarial attack ($d = 0$), $d = 66$, and $d = 1000$, i.e. the detection mechanism is fooled at the same time as the prediction. The dotted line in the perturbation plots roughly indicates the level at which attacks are visible by eye. Note that this is subjective and only a rough indication. The line in the detection plots indicates random performance, i.e. the OoD detection does nothing useful.

5. Conclusion

In this work we have addressed the question of trustworthiness for image classification. In the past, many properties linked with trustworthiness have been ascribed to generative classifier (GCs), such as increased robustness and explainability. Our GC performs nearly on-par with a standard discriminative classifier (DC), here ResNet, when tuned for discriminative performance. We observe that our GC offers significant improvements over standard DCs in terms of explainability and native out-of-distribution detection capability, but does not automatically solve all aspects of trustworthiness: Contrary to common belief, it does not generalize better under image corruptions than a DC, and it does not fully prevent adversarial attacks. In the future, we expect that robustness can be increased with further modifications or additional post-processing algorithms, as already exist for DCs. Finally, we contribute downloadable GC models pre-trained on ImageNet.

References

[1] Lynton Ardizzone, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. In *Intl. Conf. on Learning Representations*, 2019.

[2] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019.

[3] Lynton Ardizzone, Radek Mackowiak, Carsten Rother, and Ullrich Köthe. Training normalizing flows with the information bottleneck for competitive generative classification. *Advances in Neural Information Processing Systems*, 33, 2020.

[4] Jens Behrmann, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. *arXiv:1811.00995*, 2018.

[5] Christopher M. Bishop and Julia Lasserre. Generative or discriminative? getting the best of both worlds. *Bayesian statistics*, 8(3):3–24, 2007.

[6] Guillaume Bouchard and Bill Triggs. The tradeoff between generative and discriminative classifiers. In *16th IASC International Symposium on Computational Statistics (COMPSTAT'04)*, pages 721–728, 2004.

[7] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[8] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57. IEEE Computer Society, 2017.

[9] Jiefeng Chen, Yixuan Li, Xi Wu, Yingyu Liang, and Somesh Jha. Robust out-of-distribution detection in neural networks. *CoRR*, abs/2003.09711, 2020.

[10] Hyunsun Choi, Eric Jang, and Alexander A Alemi. Waic, but why? generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

[11] LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *Advances in neural information processing systems*, pages 4088–4098, 2017.

[12] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. *arXiv:1410.8516*, 2014.

[13] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *arXiv:1605.08803*, 2016.

[14] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *Intl. Conf. on Learning Representations*, 2017.

[15] Ethan Fetaya, Jörn-Henrik Jacobsen, Will Grathwohl, and Richard S. Zemel. Understanding the limitations of conditional generative models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[16] Ethan Fetaya, Jörn-Henrik Jacobsen, and Richard S. Zemel. Conditional generative models are not robust. *CoRR*, abs/1906.01171, 2019.

[17] Partha Ghosh, Arpan Losalka, and Michael J. Black. Resisting adversarial attacks using gaussian mixture variational autoencoders. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 541–548. AAAI Press, 2019.

[18] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

[19] Will Grathwohl, Ricky TQ Chen, Jesse Betterncourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[20] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[23] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized ODIN: detecting out-of-distribution image without learning from out-of-distribution data. *CoRR*, abs/2002.11297, 2020.

[24] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. A survey of safety and trustworthiness of deep neural networks. *arXiv preprint arXiv:1812.08342*, 2018.

[25] Pavel Izmailov, Polina Kirichenko, Marc Finzi, and Andrew Gordon Wilson. Semi-supervised learning with normalizing flows. *arXiv preprint arXiv:1912.13025*, 2019.

[26] Jörn-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability. *arXiv preprint arXiv:1811.00401*, 2018.

[27] Jörn-Henrik Jacobsen, Arnold W.M. Smeulders, and Edouard Oyallon. i-RevNet: deep invertible networks. In *International Conference on Learning Representations*, 2018.

[28] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv:1807.03039*, 2018.

[29] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[30] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Why normalizing flows fail to detect out-of-distribution

data. *Advances in neural information processing systems*, 33, 2020.

[31] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 7167–7177, 2018.

[32] Kimin Lee, Sukmin Yun, Kibok Lee, Honglak Lee, Bo Li, and Jinwoo Shin. Robust inference via generative classifiers for handling noisy labels. *arXiv preprint arXiv:1901.11300*, 2019.

[33] Yingzhen Li, John Bradshaw, and Yash Sharma. Are generative classifiers more robust to adversarial attacks? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3804–3814. PMLR, 2019.

[34] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Görür, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[35] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Görür, and Balaji Lakshminarayanan. Hybrid models with deep and invertible features. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 4723–4732, 2019.

[36] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *CoRR*, abs/1906.02994, 2019.

[37] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 841–848, 2001.

[38] Rajat Raina, Yirong Shen, Andrew Mccallum, and Andrew Y Ng. Classification with hybrid generative/discriminative models. In *Advances in neural information processing systems*, pages 545–552, 2004.

[39] Christian Raymond and Giuseppe Riccardi. Generative and discriminative algorithms for spoken language understanding. In *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[41] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[42] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[43] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. *arXiv preprint arXiv:1909.11480*, 2019.

[44] Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D Sculley, Joshua Dillon, Jie Ren, and Zachary Nado. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13969–13980, 2019.

[45] Jiaming Song, Yang Song, and Stefano Ermon. Unsupervised out-of-distribution detection with batch normalization. *arXiv preprint arXiv:1910.09115*, 2019.

[46] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[47] Esteban G Tabak and Cristina V Turner. A family of non-parametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[48] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[49] Naftali Tishby, Fernando C. N. Pereira, and William Bialek. The information bottleneck method. *CoRR*, physics/0004057, 2000.

[50] Jakub M Tomczak and Max Welling. Improving variational auto-encoders using householder flow. *arXiv preprint arXiv:1611.09630*, 2016.

[51] Ilkay Ulusoy and Christopher M. Bishop. Comparison of generative and discriminative techniques for object detection and classification. In *Toward Category-Level Object Recognition*, pages 173–195, 2006.

[52] Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019.

[53] Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil Jain. Adversarial attacks and defenses in images, graphs and text: A review. *arXiv preprint arXiv:1909.08072*, 2019.

[54] Jing-Hao Xue and D. M. Titterington. On the generative-discriminative tradeoff approach: Interpretation, asymptotic efficiency and classification performance. *Computational Statistics & Data Analysis*, 54(2):438–451, 2010.

[55] Yufeng Zhang, Wanwei Liu, Zhenbang Chen, Ji Wang, Zhiming Liu, Kenli Li, Hongmei Wei, and Zuoning Chen.

Out-of-distribution detection with distance guarantee in deep generative models. *arXiv preprint arXiv:2002.03328*, 2020.

[56] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.