Exact Information Bottleneck with Invertible Neural Networks: Getting the Best of Discriminative and Generative Modeling

Lynton Ardizzone∗1, Radek Mackowiak∗2, Carsten Rother1, Ulrich Köthe1

1 Visual Learning Lab, Heidelberg University, Heidelberg, Germany, 2 Robert Bosch Corporate Research - Computer Vision, Hildesheim, Germany

∗ These authors contributed equally to this work

Abstract

Generative models are more informative about underlying phenomena than discriminative ones and offer superior uncertainty quantification and out-of-distribution robustness. However, these advantages often come at the expense of reduced classification accuracy. The Information Bottleneck objective (IB) formulates this trade-off in a clean information-theoretic way, but its practical application is hampered by a lack of accurate high-dimensional estimators of mutual information (MI), its main constituent. To overcome this limitation, we develop the theory and methodology of IB-INNs, which optimize the IB objective by means of Invertible Neural Networks (INNs), without the need for approximations of MI. Our experiments show that IB-INNs allow for a precise adjustment of the generative/discriminative trade-off: They learn accurate models of the class conditional likelihoods, generalize well to unseen data and reliably detect out-of-distribution examples, while at the same time exhibiting classification accuracy close to purely discriminative feed-forward networks.

Code available at github.com/VLL-HD/FrEIA

1 INTRODUCTION

The distinction between discriminative and generative classifiers (DCs vs. GCs) is fundamental to machine learning. DCs directly predict posterior class probabilities \( p(Y|X) \), where \( X \) and \( Y \) denote input and output variables respectively. GCs instead model the joint probability \( p(X,Y) \), usually as a product of class priors \( p(Y) \) and conditional data likelihoods \( p(X|Y) \).

They are able to generate synthetic data, and posterior class probabilities can simply be inferred by Bayes’ rule:

\[
p(Y|X) = \frac{p(X|Y)p(Y)/\mathbb{E}_{p(Y)}[p(X|Y)]}{p(Y)}.
\]

DCs optimize prediction performance directly and therefore achieve better results in this respect. However, their models for \( p(Y|X) \) tend to be most accurate near decision boundaries (where it matters), but deteriorate away from them (where deviations incur no noticeable loss). Consequently, they are poorly calibrated (Guo et al., 2017) and out-of-distribution data can not be recognized at test time (Ovadia et al., 2019). In contrast, GCs model full likelihoods \( p(X|Y) \) and thus implicitly full posteriors \( p(Y|X) \), which leads to the opposite behavior – better predictive uncertainty at the price of reduced accuracy.

In practice, models trained in a purely generative way (in particular with maximum likelihood), achieve highly unsatisfactory accuracy, so that some recent work has called into question the overall effectiveness of GCs (Fetaya et al., 2019; Nalisnick et al., 2019b). Others have introduced different architectures and augmented losses (Jacobsen et al., 2019; Nalisnick et al., 2019a), that lead to an improvement of the discriminative capabilities. In-depth studies of the ideal cases (Bishop & Lasserre, 2007; Bishop, 2007) revealed the existence of a trade-off, controlling the balance between discriminative and generative performance. The existence of a trade-off is not self evident, as in principle, a more accurate generative model should also lead to better classification.

We propose the Information Bottleneck (IB) objective (Tishby et al., 2000) as an alternative viewpoint of this phenomenon, when applied to generative classification. IB formulates the discriminative/generative trade-off in a very general information-theoretic form: It postulates existence of a latent space \( Z \), where all information flow between \( X \) and \( Y \) is channeled through (hence the method’s name). In order to optimize pre-
Figure 1: Our generative classifier as the Information Bottleneck Invertible Neural Network (IB-INN).

predictive performance, IB attempts to maximize the mutual information \( I(Y, Z) \) between \( Y \) and \( Z \). Jointly, it strives to minimize the mutual information \( I(X, Z) \) between \( X \) and \( Z \), forcing the model to ignore irrelevant aspects of \( X \), which do not contribute to classification performance and only increase the potential for overfitting. The objective can therefore be expressed as

\[
L_{IB} = I(X, Z) - \beta I(Y, Z) .
\]

Unfortunately, practical application of IB as a loss function is difficult because existing estimators for mutual information (MI) are not sufficiently reliable in high dimensions. The Variational Information Bottleneck (VIB, Alemi et al., 2017; Kolchinsky et al., 2017) provides a feasible approximation in form of an upper bound for IB, which however does not work as well as the asymptotically exact solution presented in this work.

Using Invertible Neural Networks (INNs), we can, for the first time, train generative classifiers directly with the IB objective, cf. Fig. 1. This major advance arises from two critical properties of this network type: (i) the transformation between \( X \) and \( Z \) has a tractable Jacobian determinant, and (ii) the latent space \( Z \) can be shaped as a Gaussian Mixture Model (GMM). As a result, the IB objective is analytically expressible in terms of the change-of-variables formula, allowing for standard gradient descent training without additional approximations. Moreover, properties (i) and (ii) facilitate latent space exploration which enables out-of-distribution detection and the analysis of class similarities. The trade-off parameter \( \beta \) occurring in the IB loss (1) allows us to explicitly alter the trade-off between our model’s classification performance and it’s generative modeling capabilities. When \( \beta \) is adjusted properly, our experiments reveal that IB-INNs simultaneously exhibit high predictive accuracy, well calibrated uncertainties and allow to reliably detect out-of-distribution examples.

To summarize, we combine two concepts – the Information Bottleneck (IB) objective and Invertible Neural Networks (INNs) – into a new generative classifier type called IB-INN. Our contributions are as follows:

- We derive an asymptotically exact formulation of the IB loss for a special GC type by utilizing INNs.
- We show experimentally that our models outperform existing GCs on CIFAR10/100 in terms of classification error, and incur at worst minor degradation relative to feed-forward DCs.
- We demonstrate good uncertainty quantification in terms of accurate posterior calibration and reliable outlier detection.

2 METHOD

In the following, upper case letters denote random variables (RVs) (e.g. \( X \)) and lower case letters their instances (e.g. \( x \)). The probability density function of an RV is written as \( p(X) \), the evaluated density as \( p(x) \) (or \( p(X = x) \) when ambiguous), and all RVs are understood as vector quantities. We distinguish true distributions from modeled ones by the letters \( p \) and \( q \) respectively. The distributions \( q \) always depend on model parameters, but we do not make this explicit to avoid notation clutter. The proofs to all Propositions are provided in the appendix.

Our models have two kinds of learnable parameters:

1. An invertible neural network with parameters \( \theta \) bijectively maps inputs \( X \) to latent variables \( Z \)

\[
\text{INN: } Z = g_\theta(X) \leftrightarrow X = g_\theta^{-1}(Z) \quad (2)
\]

2. A Gaussian mixture model with means \( \mu_y \) and unit covariance matrices is used as a reference distribution for the latents \( Z \)

\[
q(Z \mid Y) = \mathcal{N}(\mu_y, \mathbb{I}) \quad (3)
\]

\[
q(Z) = \sum_y p(y) \mathcal{N}(\mu_y, \mathbb{I}) \quad (4)
\]

where \( y \) are the class labels. For simplicity, we assume that the label distribution is known, i.e. \( q(Y) = p(Y) \).
Our derivation rests on a quantity we call mutual cross-information in analogy to the well-known cross-entropy

\[ CI(U, V) = E_{u,v \sim p(U,V)} \left[ \log \frac{q(u,v)}{p(u)p(v)} \right] \]  

(5)

Note that the expectation is taken over the true distribution, whereas the logarithm involves model distributions. The following proposition (proof in Appendix) clarifies the relationship between mutual information \( I \) and \( CI \):

**Proposition 1.** Assume that \( q(.) \) can be chosen from a sufficiently rich model family (e.g. a universal density estimator). Then for every \( \eta > 0 \) there is a model such that

\[ |I(U,V) - CI(U,V)| < \eta \]  

(6)

and \( I(U,V) = CI(U,V) \) if \( p(u,v) = q(u,v) \).

### 2.1 Estimating Mutual Information with INNs

Estimation of the mutual cross-information \( CI(X, Z) \) between inputs and latents is problematic for deterministic mappings from \( X \) to \( Z \) (Amjad & Geiger, 2018), and specifically for invertible networks, which are bijective by construction. In this case, the joint distributions \( q(X, Z) \) and \( p(X, Z) \) are not valid Radon-Nikodym densities and both \( CI \) and \( I \) are therefore undefined. We resolve this degeneracy by means of deterministic augmentation: Instead of inputting \( X \) to the network, we feed it with a noisy version \( X' = X + \mathcal{E} \), where \( \mathcal{E} \sim \mathcal{N}(0, \sigma^2\mathbb{I}) \) is Gaussian with mean zero and covariance \( \sigma^2\mathbb{I} \). This is motivated by practical experience: For normalizing flows and similar models, such noise augmentation is required anyway to dequantize \( X \) (i.e. to turn discrete pixel values into real numbers).

In other words, the INN actually learns the mapping \( Z_\theta = g_\theta(X + \mathcal{E}) \), which guarantees that \( CI(X, Z_\theta) \) is well-defined. Minimizing \( CI(X, Z_\theta) \) according to the IB principle means that \( g_\theta(X + \mathcal{E}) \) is encouraged to amplify the noise \( \mathcal{E} \) and ignore the data \( X \), see Fig. 3. If the global minimum can be reached, \( I \) and \( CI \) will coincide, as \( CI(X, Z_\theta) \) is an upper bound (cf. Prop. 1):

**Proposition 2.** For the specific case that \( Z_\theta = g_\theta(X + \mathcal{E}) \), it holds that \( I(X, Z_\theta) \leq CI(X, Z_\theta) \).

We now derive the training procedure for flow-type networks with the noise-augmented \( CI(X, Z_\theta) \) in the limit of small noise \( \sigma \to 0 \). Full details found in appendix. We decompose the mutual cross-information into two terms

\[ CI(X, Z_\theta) = E_{x,\varepsilon \sim p(X),p(\mathcal{E})} \left[ -\log q(Z_\theta = g_\theta(x + \varepsilon)) \right] + E_{x,\varepsilon \sim p(X),p(\mathcal{E})} \left[ \log q(Z_\theta = g_\theta(x + \varepsilon) \mid X = x) \right] \]

\[
\begin{align*}
&:= C \\
&+ E_{x,\varepsilon \sim p(X),p(\mathcal{E})} \left[ \log q(Z_\theta = g_\theta(x + \varepsilon) \mid X = x) \right]
\end{align*}
\]

Figure 3: The more the noise is amplified in relation to the noise-free input, the lower the mutual cross-information between noisy latent vector \( Z_\varepsilon \) and noise-free input \( X \).

The first expectation can be approximated by the empirical mean over a finite dataset, because the Gaussian mixture distribution \( q(Z_\varepsilon) \) is known analytically. To approximate the second term, we first note that the condition \( X = x \) can be replaced with \( Z = g_\theta(x) \), because \( g_\theta \) is bijective and both conditions convey the same information

\[ C = E_{p(X),p(\mathcal{E})} \left[ \log q(Z_\theta = g_\theta(x + \varepsilon) \mid Z = g_\theta(x)) \right] \]

We now linearize \( g_\theta \) by its first order Taylor expansion

\[ g_\theta(x + \varepsilon) = g_\theta(x) + J_x \varepsilon + O(\varepsilon^2) \]

where \( J_x = \frac{\partial g_\theta(x)}{\partial x} \) denotes the Jacobian at \( X = x \). Inserting this into \( C \), the \( O(\varepsilon^2) \) can be moved out of the expectation due to the dominated convergence theorem (DCT):

\[ C = E_{p(X),p(\mathcal{E})} \left[ \log q(g_\theta(x) + J_x \varepsilon + O(\varepsilon^2)) \right] + O(\varepsilon^2) \]

Since \( \varepsilon \) is Gaussian with mean zero and covariance \( \sigma^2\mathbb{I} \), the conditional distribution is Gaussian with mean \( g_\theta(x) \) and covariance \( \sigma^2J_xJ_x^T \). The expectation with respect to \( p(\mathcal{E}) \) is thus the negative entropy of a multivariate Gaussian and can be computed analytically as well

\[ C = E_{p(X)} \left[ -\frac{1}{2} \log \left( \det(2\pi e\sigma^2 J_xJ_x^T) \right) \right] + O(\sigma^2) \]

\[ = E_{p(X)} \left[ -\log |\det(J_x)| \right] - d \log(\sigma) - \frac{d}{2} \log(2\pi e) + O(\sigma^2) \]

with \( d \) the dimension of \( X \). To avoid running the model twice (for \( x \) and \( x + \varepsilon \)), we approximate the expectation of the Jacobian determinant by \( 0^{th}\)-order Taylor expansion as

\[ E_{p(X)}[\log |\det(J_x)|] = E_{p(X),p(\mathcal{E})}[\log |\det(J_x)|] + O(\sigma) \]
where \( J_x \) is the Jacobian evaluated at \( x+\varepsilon \) instead of \( x \). The residual can be moved outside of the expectation because of DCT, and because \( J_x \) is always bounded in our networks.

Putting everything together, we drop terms from \( CI(X,Z) \) that are independent of the model or vanish with rate at least \( O(\sigma) \) as \( \sigma \to 0 \). The resulting loss \( \mathcal{L}_x \) becomes

\[
\mathcal{L}_x = \mathbb{E}_{p(x),p(\varepsilon)} \left[ - \log q(g_\theta(x+\varepsilon)) - \log |\det(J_x)| \right]
\]

(7)

Since the network's generative distribution is defined by the change of variables formula as

\[
q_X(x) = q(Z = g_\theta(x)) \left| \det(J_x) \right|
\]

we recognize \( \mathcal{L}_x \) as the negative log-likelihood of the perturbed data under \( q_X \)

\[
\mathcal{L}_x = \mathbb{E}_{p(x),p(\varepsilon)} \left[ - \log q_X(x+\varepsilon) \right]
\]

(9)

The crucial difference between \( CI(X,Z) \) and \( \mathcal{L}_x \) is the elimination of the term \(-d \log(\sigma)\). It is huge for small \( \sigma \) and would dominate the model-dependent terms, making minimization of \( CI(X,Z) \) very hard. Intuitively, \( CI(X,Z) \) diverging for \( \sigma \to 0 \) highlights why \( CI(X,Z) \) is undefined. In practice, we estimate \( \mathcal{L}_x \) by the empirical mean \( \mathcal{L}^{(N)}_x \) on a training set \( \{x_i,\varepsilon_i\}_{i=1}^N \) of size \( N \):

\[
\mathcal{L}^{(N)}_x = \frac{1}{N} \sum_{i=1}^N \left[ - \log q(g_\theta(x_i+\varepsilon_i)) - \log |\det(J_i)| \right]
\]

(10)

where \( J_i \) is the Jacobian of \( g_\theta \) evaluated at \( x_i+\varepsilon_i \).

It remains to be shown that replacing \( I(X,Z) \) with \( \mathcal{L}^{(N)}_x \) in the IB loss Eq. 1 does not fundamentally change the solution of the learning problem in the limit of large \( N \), small \( \sigma \) and sufficient model power. Sufficient model power here means that the family of generative distributions realizable by \( g_\theta \) should be a universal density estimator. This is the case if \( g_\theta \) can represent increasing triangular maps (Bogachev et al., 2005), which has been proven for certain network architectures explicitly (e.g., Jaini et al., 2019; Huang et al., 2018). Propositions 1 & 2 then tell us that we may optimize \( CI(X,Z) \) as an estimator of \( I(X,Z) \).

The above derivation of the loss can be strengthened into

**Proposition 3.** For any \( \epsilon, \eta > 0 \) and \( 0 < \delta < 1 \) there are \( \sigma_0 > 0 \) and \( N_0 \in \mathbb{N} \), such that \( \forall N \geq N_0 \) and \( \forall \sigma < \sigma_0 \),

\[
\Pr \left( \left| CI(X,Z) + d \log \sqrt{2\pi e\sigma^2} - \mathcal{L}^{(N)}_x \right| < \epsilon \right) > 1 - \delta
\]

and

\[
\Pr \left( \left\| \frac{\partial}{\partial \theta} CI(X,Z) - \frac{\partial}{\partial \theta} \mathcal{L}^{(N)}_x \right\| < \eta \right) > 1 - \delta
\]

holds uniformly for all model parameters \( \theta \).

The first statement proves consistency of \( \mathcal{L}^{(N)}_x \), and the second justifies gradient-descent optimization on the basis of \( \mathcal{L}^{(N)}_x \). Proofs can be found in the appendix.

### 2.2 The IB-INN

Similarly to the first term in the IB-loss in Eq. 1, we also replace the mutual information \( I(Y,Z) \) with

\[
CI(Y,Z) = E_{x,y \sim p(X,Y)} \left[ -\log p(y) \right]
\]

(11)

\[
+ E_{x,y\sim p(X,Y),\varepsilon\sim p(\varepsilon)} \left[ \log \frac{q(Z = g_\theta(x+\varepsilon), y)}{\sum_{y'} q(g_\theta(x+\varepsilon), y')} p(y) \right]
\]

Inserting the likelihood \( q(z \mid y) = \mathcal{N}(z; \mu_y, I) \) of our latent Gaussian mixture model and recalling that \( q(Y) = p(Y) \), this can be decomposed into

\[
CI(Y,Z) = E_{y \sim p(Y)} \left[ -\log p(y) \right]
\]

\[
+ E_{x,y\sim p(X,Y),\varepsilon\sim p(\varepsilon)} \left[ \log \frac{q(g_\theta(x+\varepsilon) \mid y)}{\sum_{y'} q(g_\theta(x+\varepsilon) \mid y')} p(y') \right]
\]

The first expectation is independent of the model and can be dropped, whereas the second is the expectation of the GMM's log-posterior \( \log q(y \mid z) \). Since all mixture components have unit covariance, the elements of \( Z \) are conditionally independent and the likelihood factorizes as \( q(z \mid y) = \prod_i q(z_i \mid y) \), so that \( q(y \mid z) \) can be interpreted as a naive Bayes classifier. In contrast to a naive Bayes classifier in data space, which typically performs badly because raw features are not conditionally independent, our training process pushes latent features towards conditional independence and results in very accurate classification.

Defining the loss \( \mathcal{L}^{(N)}_Y \) as the empirical mean of the log-posterior in a training set \( \{x_i, y_i, \varepsilon_i\}_{i=1}^N \) of size \( N \), we get

\[
\mathcal{L}^{(N)}_Y = \frac{1}{N} \sum_{i=1}^N \log \mathcal{N}(g_\theta(x_i+\varepsilon_i; \mu_{y_i}, I)) p(y_i)
\]

(12)

and our model parameters \( \theta \) and \( \{\mu_1, \ldots, \mu_K\} \) are trained by gradient descent of the IB-INN loss

\[
\mathcal{L}^{(N)}_{\text{IB-INN}} = \mathcal{L}^{(N)}_X - \beta \mathcal{L}^{(N)}_Y
\]

(13)
cancels out. This means that the L

rect classes. For an unregularized model trained with

but there is no term repulsing them from the incor-

tion: points are only drawn towards the correct class,

which show that indeed the class-NLL loss causes a

class discriminatory nature, as Fig. 4 illustrates. This

is explained in more detail by Fetaya et al. (2019),

to low loss. The green and orange arrows indicate attractive and repulsive interactions with the cluster centers.

2.3 Advantages of the IB-Loss

In this section we will interpret and discuss the nature of the loss function in Eq. 13. We also form an intuitive understanding of why it is more suitable than the class-conditional negative-log-likelihood (‘class-NLL’) traditionally used for generative classifiers of this type. The findings are represented graphically in Fig. 4.

\textbf{L}_X-term: As demonstrated in Eq. 9, the term is the (unconditional) negative-log-likelihood loss used for normalizing flows, with the difference that \( q(Z) \) is a GMM rather than a unimodal Gaussian. We conclude that this loss term encourages the INN to become an accurate likelihood model under the marginalized latent distribution and completely ignores any class content.

\textbf{L}_Y-term: Examining Eq. 12, we see that for any pair \( g(x + \varepsilon), y \), the cluster centers \( (\mu_Y \neq y) \) of the other classes are repulsed, while \( g_\theta(x + \varepsilon) \) and the correct cluster center \( \mu_y \) are drawn together. Note that the class-NLL loss only captures the second aspect, and therefore has a much weaker training signal. We can also view this in a different way: by substituting \( q(x|y) \left| \det (J_y(x)) \right|^{-1} \) for \( q(z|y) \) the second summand of Eq. 11 simplifies to \( \log q(y|x) \), since the Jacobian cancels out. This means that the \( L_Y \) loss directly maximizes the correct class probability, while ignoring the data likelihood. Again, this improves the training signal, as Fetaya et al. (2019) showed that the data likelihood dominates the class-NLL loss, so that the discriminative aspect is not properly learned.

\textbf{Classical class-NLL loss:} For \( \beta = 1 \), the IB-INN loss reduces to the class-NLL loss, because the first summand in Eq. 10 cancels with the denominator in Eq. 12. The INN is then no longer penalized for overlapping mixture components, and the GMM looses its class discriminatory nature, as Fig. 4 illustrates. This is explained in more detail by Fetaya et al. (2019), who show that indeed the class-NLL loss causes a vanishingly small training signal for the class separation: points are only drawn towards the correct class, but there is no term repulsing them from the incorrect classes. For an unregularized model trained with class-NLL, this causes all cluster centers to collapse together, leading the INN to effectively just model the marginal data likelihood.

2.4 Practical Implementation

We learn \( \mu_Y \) as a free parameter jointly with the remaining model parameters in an end-to-end fashion using the loss in Eq. 13. The entire training is performed in log-space for numerical stability, as the likelihoods become both too large and too small otherwise (see Appendix Sec. 2 for details). We apply two additional techniques while learning the model, label smoothing and loss rebalancing:

\textbf{Label smoothing} We observed that the individual class means \( \mu_Y \) drift apart during training, because training with hard labels enforces the Gaussian mixture components to become perfectly separated. This can cause problems during training, as there is a high loss barrier between the clusters due to \( L_X \), preventing points from moving smoothly from one class to the other during training. To avoid this effect, we simply apply a small amount of label smoothing (Szegedy et al., 2016), where the one-hot training vectors are softened with \( \alpha = 0.05 \) in our case. We do the same for all comparison models.

\textbf{Loss rebalancing} To avoid the laborious process of having to adjust hyperparameters for vastly different loss magnitudes, we employ the following rebalancing scheme: Firstly, we divide the loss \( L_X \) by the number of dimensions of \( X \). This ensures that \( L_X \) remains in a similar range when changing e.g. the input image size, as it scales linearly with the number of input dimensions. Secondly, we define a matching \( \tilde{\beta} \equiv \beta / \text{dim}(X) \). Lastly, we reweight the entire loss by a factor \( 2/(1 + \tilde{\beta}) \). This ensures that the loss keeps the same magnitude when changing \( \beta \) over wide ranges. Thanks to this rebalancing scheme, we can use the same learning rate and hyperparameters for all experiments:

\[
L_{IB}^{(N)} = \frac{2}{1 + \tilde{\beta}} \left( L_X^{(N)} \frac{\text{dim}(X)}{\text{dim}(X)} - \beta L_Y^{(N)} \right) \quad (14)
\]
3 EXPERIMENTS

We construct our IB-INN by combining the design efforts of various previous works on INNs and normalizing flows. In brief, we use a Real-NVP design consisting of affine coupling blocks (Dinh et al., 2017), with added improvements from recent works, such as Kingma & Dhariwal (2018); Jacobsen et al. (2019, 2018); Ardizzone et al. (2019). A more detailed description of the architecture is provided in the appendix.

3.1 Comparison of Methods

In addition to the IB-INN, we train several alternative methods. For each, we use exactly the same INN model, or an equivalent feed-forward ResNet model. Every method has the exact same hyperparameters and training procedure, the only difference being the loss function, and the lack of invertibility for pure feed-forward models.

**Feed-forward** As a baseline, we train a feed-forward ResNet (He et al., 2016) with softmax cross entropy loss. Each affine coupling is simply replaced by a ResNet block. Except for the invertibility, the architectures are identical.

**i-RevNet** (Jacobsen et al., 2018): To rule out any differences stemming from the constraint of invertibility, we additionally train the INN as a standard softmax classifier, by projecting the outputs to class logits. While the architecture is invertible, the model it is not generative and trained as a standard feed-forward classifier.

**Variational Information Bottleneck (VIB):** We train the VIB, as presented by Alemi et al. (2017), using a feed-forward ResNet. Note that the authors define their $\beta$ in the opposite way, by weighting $I(X, Z)$. For consistency we convert this to our definition of $\beta$.

**Class-NLL:** As a standard generative classifier, we firstly train an INN with a GMM in latent space completely naively as a conditional generative model, using the class-conditional maximum likelihood loss $\mathcal{L}_{\text{class-NLL}} = -E \log (q_0(x|y))$. Secondly, we also train a regularized version, to increase the classification accuracy. The regularization consists of constraining the class centroids $\mu_Y$ to points on a sphere with a fixed radius. As the radius becomes comparable to the typical intra-class distances, the training signal for the classification is amplified. We therefore choose $\sqrt{\dim(Z)}$ as radius.

3.2 Quantitative measurements

In the following, we describe the scores reported in Tab. 1.

**Calibration error:** In general, the calibration curve measures whether the confidence of a model agrees with its actual performance. All prediction outputs are binned according to their predicted probability $P$ (‘confidence’), and it is recorded which fraction of predictions in each bin was correct, $Q$. For a perfectly calibrated model, we have $P = Q$, e.g. predictions with 30% confidence are correct 30% of the time. There are various metrics to measure deviation from this perfect behaviour, and we largely adhere to those used by Guo et al. (2017). Exact descriptions found in Appendix. Specifically, we consider the expected calibration error (ECE), the maximum calibration error (MCE), and a quantity termed the overconfidence (OVC). The overconfidence measures the normalized fraction of highly confident but wrong predictions. It should be $\in [0, 1]$ for a well calibrated model, and is $\gg 1$ for an overconfident one. Because we find the metrics to be partly inconsistent, we also include the geometric mean of all three (i.e. $\sqrt[3]{\text{ECE} \cdot \text{MCE} \cdot \max(1, \text{OVC})}$). The geometric mean is used because it properly accounts for the different magnitudes of the metrics. Note, that we only penalize the OVC if it is above the upper bound for a well-calibrated model.

**Out-of-distribution (OoD) prediction entropy:** For data that is OoD, we expect from a model that it returns uncertain class predictions, as it has not been trained on such data. In the ideal case, each class is assigned the same probability of $1/(\text{nr. classes})$. The performance measure commonly used for this, e.g. by Ovadia et al. (2019), is to measure this through the discrete entropy of the class prediction outputs $H(Y|X_{\text{Ood}})$, which should be as high as possible.

**OoD detection score:** For OoD detection, we use the (unconditional) negative log-likelihood (NLL) predicted by each model as an outlier score. This is a basic approach, but also the most common in the generative classifier literature. For the VIB, the situation is different: as opposed to a VAE, it does not estimate $p(X)$ due to the missing reconstruction loss. As a substitute, we use the ELBO loss (also termed ‘info-loss’ in the VIB literature), that stems from the $I(X, Z)$-term.

To quantify the OoD detection capabilities of each model, we record the degree of separability between in-distribution data and OoD data: for a random inlier and a random outlier, what is the probability that the outlier has the higher outlier score. The detection score would be 1.0 for perfectly separated in- and outliers, and 0.5 if each point is assigned a random score. Note, this definition is exactly equal to the widely used ROC-AUC metric.

**OoD datasets:** The inlier dataset consist of CIFAR10/100 images, i.e. $32 \times 32$ colour images showing 10/100 object classes. Additionally we created four different OoD datasets, that cover different aspects, see
Fig. 5. Firstly, we create a random 3D rotation matrix with an adjustable rotation angle $\alpha$, and apply it to the RGB color vectors of each pixel of CIFAR10 images. We set a fixed value of $\alpha = 0.3\pi$ for quantitative comparisons. Secondly, we add random uniform noise with a small amplitude to CIFAR10 images, as an alteration of the image statistics. Thirdly, we use the QuickDraw dataset of hand drawn objects (Ha & Eck, 2018), and filter only the categories corresponding to CIFAR10 classes and color each grayscale line drawing randomly. Therefore the semantic content is the same, but the image modality is different. Lastly, we downscale the ImageNet validation set to $32 \times 32$ pixels. In this case, the semantic content is different, but the image statistics are very similar to CIFAR10. However, we find that none of the models can reliably detect the ImageNet data as OoD, so we use this primarily to measure the prediction entropy behaviour on OoD data.

### 3.3 Quantitative Model Comparison

A comparison of all models is performed in Table 1 for CIFAR10 and CIFAR100. To our best knowledge, in the literature so far, generative classifiers have never been used for CIFAR100, as this is generally very challenging.

In summary, we find that the IB-INN has slightly worse classification performance than a standard DC, which is in line with the performance drop from feed-forward to invertible architecture (i-RevNet). However, the uncertainties in the form of calibration error and entropy on OoD data, are far superior. For ablation where only $\mathcal{L}_Y$ is used ($\tilde{\beta} \to \infty$), the performance is more comparable to a standard DC, demonstrating that it is really the IB objective giving the improvements. In addition, the OoD detection is on par or better than for the other GCs. Furthermore, the GCs naively trained with class-NLL make no meaningful predictions on either dataset. While the regularized version achieves better accuracy, and maintains the OoD detection capabilities, the calibration and uncertainty is very poor. Lastly, we find that the VIB brings some improvement over a standard feed-forward model in terms of uncertainties, especially for CIFAR100, but still inferior to IB-INN.

### 3.4 Effect of Beta

To study the trade-off between classification performance and modeling capabilities parameterized by $\tilde{\beta}$, we train 24 IB-INN models for different $\tilde{\beta}$ ranging from 0.02 to 50. For comparison, we also train corresponding VIB models, and summarize our findings in Fig. 8. As expected, the classification accuracy improves as we increase $\tilde{\beta}$, while the uncertainty estimates become worse. The trend of OoD detection depends on the dataset: It is almost constant for the RGB rotated data, improves for the hand-drawn data, and degrades for the noisy data. This is due to whether the focus on class information is helpful in detecting OoD data, or whether simply modeling natural images suffices. The classification accuracy of the VIB stays almost constant over the whole range, and the uncertainties also show little variations, indicating that the used loss function is indeed only a rough bound on the true IB.

### 3.5 Latent Space Exploration

To better understand what the IB-INN learns, we analyze the latent space in different ways. Firstly, Fig. 7 shows the layout of the latent space GMM through a linear projection. We find that the clusters of ambiguous classes, e.g., truck and car, are connected in latent space, to account for uncertainty. Secondly, Fig. 6 shows interpolations in latent space between two test set images, using models trained with different values of $\tilde{\beta}$. We observe that for low $\tilde{\beta}$, the IB-INN has a very well structured latent space, resulting in good generative capabilities and plausible interpolations. For larger $\tilde{\beta}$, the interpolation quality continually degrades.
Figure 7: GMM Latent space behaviour by increasing $\tilde{\beta}$. The class separation increases with larger $\tilde{\beta}$. Note however, how ambiguous classes (truck and car) stay connected, to account for uncertainty.

![Diagram](image)

Figure 8: Effect of changing the parameter $\tilde{\beta}$ (logarithmic x-axis) on the different performance measures (y-axis). The VIB results are the dotted lines for comparison, except for OoD detection. The inlier dataset is CIFAR10. The arrows indicate if a larger or smaller score is better. While classification accuracy improves with $\tilde{\beta}$, the uncertainty measures grow worse. The trend for the OoD detection depends on the OoD data used, and whether the focus on class information is important for the detection. The special case $\tilde{\beta} = 1$ (class-NLL) translates to $\tilde{\beta} \approx 3 \cdot 10^{-4}$, where the classification accuracy is unusable (cf. Table 1). All curves shown separately in the appendix.

Table 1: Results on the CIFAR10 and CIFAR100 datasets. All models have the same number of parameters and were trained with the same hyperparameters. All values except entropy and overconfidence are given in percent. The arrows indicate whether a higher or lower value is better. The prediction entropy of the model trained with $\tilde{\beta}$ is shown separately in the appendix.

<table>
<thead>
<tr>
<th>CIFAR10</th>
<th>Classif. err. (%)</th>
<th>Calibration error (%)</th>
<th>OoD prediction entropy (%)</th>
<th>OoD detection score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IB-INN (ours, $\beta = 1$)</td>
<td>9.98</td>
<td>1.069</td>
<td>0.93</td>
<td>8.20</td>
</tr>
<tr>
<td>IB-INN-only $L_Y (\tilde{\beta} \to \infty)$</td>
<td>8.29</td>
<td>6.690</td>
<td>1.22</td>
<td>23.81</td>
</tr>
<tr>
<td>IB-INN-only $L_Y (\tilde{\beta} = 0)$</td>
<td>8.29</td>
<td>6.690</td>
<td>1.22</td>
<td>23.81</td>
</tr>
<tr>
<td>Class-NLL</td>
<td>20.57</td>
<td>24.176</td>
<td>3.93</td>
<td>20.33</td>
</tr>
<tr>
<td>Class-NLL (regularized)</td>
<td>19.36</td>
<td>24.019</td>
<td>3.93</td>
<td>20.33</td>
</tr>
<tr>
<td>VIB ($\beta = 1$)</td>
<td>7.29</td>
<td>4.466</td>
<td>0.68</td>
<td>27.83</td>
</tr>
<tr>
<td>Standard ResNet</td>
<td>6.76</td>
<td>5.656</td>
<td>0.73</td>
<td>40.61</td>
</tr>
<tr>
<td>i-RevNet</td>
<td>9.09</td>
<td>2.224</td>
<td>0.54</td>
<td>14.78</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CIFAR100</th>
<th>Classif. err. (%)</th>
<th>Calibration error (%)</th>
<th>OoD prediction entropy (%)</th>
<th>OoD detection score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IB-INN (ours, $\beta = 1$)</td>
<td>36.59</td>
<td>1.157</td>
<td>0.15</td>
<td>6.67</td>
</tr>
<tr>
<td>IB-INN only $L_Y (\tilde{\beta} \to \infty)$</td>
<td>33.86</td>
<td>1.171</td>
<td>0.15</td>
<td>4.68</td>
</tr>
<tr>
<td>IB-INN only $L_Y (\tilde{\beta} = 0)$</td>
<td>36.59</td>
<td>1.157</td>
<td>0.15</td>
<td>6.67</td>
</tr>
<tr>
<td>Class-NLL</td>
<td>98.59</td>
<td>34.622</td>
<td>1.47</td>
<td>92.86</td>
</tr>
<tr>
<td>Class-NLL (regularized)</td>
<td>79.08</td>
<td>32.392</td>
<td>1.40</td>
<td>94.58</td>
</tr>
<tr>
<td>VIB ($\beta = 1$)</td>
<td>42.92</td>
<td>1.259</td>
<td>0.18</td>
<td>8.63</td>
</tr>
<tr>
<td>Standard ResNet</td>
<td>29.14</td>
<td>4.863</td>
<td>0.23</td>
<td>17.26</td>
</tr>
<tr>
<td>i-RevNet</td>
<td>34.27</td>
<td>2.889</td>
<td>0.24</td>
<td>17.76</td>
</tr>
</tbody>
</table>
4 RELATED WORK

Generative Classification: An in-depth analysis of the trade-offs between discriminative and generative models was first performed by Ng & Jordan (2001) and was later extended by Bouchard & Triggs (2004); Bishop & Lasserre (2007); Xue & Titterington (2010), who investigated the possibility of balancing the strengths of both methods via a hyperparameter. Promising applications of these ideas to natural language processing, based on neural networks, were recently presented by Shen et al. (2017); Yogatama et al. (2017); Wang & Wu (2018). Li et al. (2019) showed that generative classifiers may be more robust against adversarial attacks, and Hwang et al. (2019) demonstrated their robustness against missing data. On the other hand, Fetaya et al. (2019) found that conditional normalizing flows have poor discriminative performance, making them unsuitable for classification tasks.

Information Bottleneck: The IB was introduced by Tishby et al. (2000) as a tool for information-theoretic optimization of compression methods. This idea was expanded on by Chechik et al. (2005); Gilad-Bachrach et al. (2003); Shamir et al. (2010) and Friedman et al. (2013). A relationship between IB and deep learning was first proposed by Tishby & Zaslavsky (2015), and later experimentally examined by Shwartz-Ziv & Tishby (2017), who use IB for the understanding of neural network behavior and training dynamics. A close relation of IB to dropout, disentanglement, and variational autoencoding was discovered by Achille & Soatto (2018), which led them to introduce Information Dropout as a way to take advantage of IB in discriminative models. The approximation of IB in a variational setting was proposed independently by Kolchinsky et al. (2017) and Alemi et al. (2017), who especially demonstrate improved robustness against adversarial attacks.

5 CONCLUSIONS

We addressed the application of the Information Bottleneck (IB) to Invertible Neural Networks (INNs) as a loss function. We find that we can formulate an asymptotically exact version of the IB, which results in an INN that is a generative classifier. From our experiments, we conclude that the IB-INN provides higher quality uncertainties and out-of-distribution detection, while reaching almost the same classification accuracy as standard feed-forward methods, and generally outperforms other supervised generative classifiers on CIFAR10 and CIFAR100.

References


Nalisnick, E. T., Matsukawa, A., Teh, Y. W., Gökür, D., and Lakshminarayanan, B. 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, 2019b. URL https://openreview.net/forum?id=H1xwNhCcYm.


Exact Information Bottleneck with Invertible Neural Networks:
Getting the Best of Discriminative and Generative Modeling
– Appendix –

6 PROOFS AND DERIVATIONS
6.1 Mutual Cross-Information as Estimator for MI

In our case, we only require $CI(X,Z_{\xi})$ and $CI(Y,Z_{\xi})$, but we show the correspondence for two unspecified random variables $U,V$, as it may be of general interest. However, note that our estimator will likely not be particularly useful outside of our specific use-case, and other methods should be preferred (e.g. Belghazi et al., 2018).

For the joint input space $\Omega = U \times V$, we assume that $U$ is a compact domain in $\mathbb{R}^d$, and $V$ is either also a compact domain in $\mathbb{R}^d$ (Case 1), or discrete, i.e. a finite subset of $\mathbb{N}$ (Case 2). In Case 1, we assume that $p(U,V)$ is absolutely continuous with respect to the Lebesgue measure, and in Case 2, $p(U,v)$ is absolutely continuous for all values of $v \in V$.

In Case 1, $q(U), q(V), q(U,V)$, the densities can all be modeled separately, by three flow networks $g^{(U)}_{\theta}(u), g^{(V)}_{\theta}(v), g^{(UV)}_{\theta}(u,v)$. Although in our formulation, we are later able to approximate the latter two through the first.

In Case 2, we only model $q(U|V)$, and assume that $q(V)$ is either known beforehand and set to $p(V)$ (e.g. label distribution), or the probabilities are parametrized directly. Either way, $q(U,V) = q(U|V)q(V)$ and $q(U) = \sum_{v \in V} q(U,v)$.

**Proposition 1.** Assume that the $q(.)$ densities can be chosen from a sufficiently rich model family (e.g. a universal density estimator). Then for every $\eta > 0$ there is a model such that

$$\left| I(U,V) - CI(U,V) \right| < \eta \quad (15)$$

and $I(U,V) = CI(U,V)$ if $p(U,V) = q(U,V)$.

**Proof.** Writing out the definitions explicitly, and rearranging, we find

$$CI(U,V) = I(U,V) + D_{KL}(p(U,V)||q(U,V))$$

$$- D_{KL}(p(U)||q(U)) - D_{KL}(p(V)||q(V)) \quad (16)$$

Shortening the KL terms to $D_1, D_2$ and $D_3$ for convenience:

$$|I_{\theta^*}(U,V) - I(U,V)| = |D_1 - D_2 - D_3| \quad (17)$$

$$\leq D_1 + D_2 + D_3 \quad (18)$$

$$\leq 3 \max(D_1, D_2, D_3) \quad (19)$$

At this point, we can simply apply results from measure transport: if the $g_{\theta}$ are from a family of universal density estimators, we can choose $\theta^*$ to make $\max(D_1, D_2, D_3)$ arbitrarily small by matching $p$ and $q$. This was shown in general for increasing triangular maps, e.g. in Hyvärinen & Pajunen (1999), Theorem 1 for an accessible proof, or Bogachev et al. (2005) for a more in-depth approach (specifically Corollary 4.2).

Generality was also proven for several concrete architectures, e.g. Jaini et al. (2019), Huang et al. (2018).

For the second part of the Proposition, we note the following: if $p(U,V) = q(U,V)$, we have $D_1 = D_2 = D_3 = 0$, and therefore $CI(U,V) = I(U,V)$. \hfill $\square$

6.2 Loss Function $L_X$

In the following, we use the subscript-notation for the cross entropy:

$$h_q(U) = E_{u \sim p(U)} \left[-\log q(u)\right], \quad (20)$$

to avoid confusion with the joint entropy that arises with the usual notation ($h(p(U), q(U))$).

We use an INN $g_{\theta}$ representing a homeomorphic transform, where the network parameter space $\Theta$ is a compact subdomain of $\mathbb{R}^n$. We assume that $g_{\theta}(u)$ and $J_\theta$ are uniformly bounded, and furthermore the absolute Jacobian determinant $|\det J_\theta|$ is also uniformly bounded from below by a constant $> 0$. We also assume $J_\theta$ is continuous and differentiable in both $X$ and $\theta$. All these assumptions are fulfilled for most architectures used in practice, and certainly for the coupling block design. As $J_\theta$ is bounded, this also implies that $g_{\theta}$ is Lipschitz-continuous. The $X$ input space $\mathcal{X}$ is a compact subdomain of $\mathbb{R}^d$.

**Proposition 2.** For the case given in the paper, that $Z_{\xi} = g_{\theta}(X + \xi)$, it holds that $I(X,Z_{\xi}) \leq CI(X,Z_{\xi})$.

**Proof.** In the following, we first use the invariance of the (cross-)information to homeomorphic transforms.
Then, we use $p(X+E|X) = q(X+E|X) = p(E)$ (known exactly) and write out all the terms, most of which cancel. Finally, we use the inequality that the cross entropy is larger than the entropy, $h_q(U) \geq h(U)$ regardless of $q$. The equality holds iff the two distributions are the same.

$$CI(X, Z_E) - I(X, Z_E) = CI(X, X+E) - I(X, X+E)$$

$$= h_q(X) - h(X) + 0$$

With equality iff $p(X) = q(X)$. 

We now want to show that the network optimization procedure that arises from the empirical loss, in particular the gradients w.r.t. network parameters $\theta$, are consistent with those of $CI(X, Z_E)$:

**Proposition 3.** The defined loss is a consistent estimator for $CI(X, Z_E)$ up to a known constant, and a consistent estimator for the gradients. Specifically, for any $\epsilon_1, \epsilon_2 > 0$ and $0 < \delta < 1$ there are $\sigma_0 > 0$ and $N_0 \in \mathbb{N}$, such that for all $N \geq N_0$ and $\forall \sigma < \sigma_0$,

$$\text{Pr}\left(\left\lvert CI(X, Z_E) + d \log \sqrt{2 \pi \sigma^2} - L_X^{(N)}\right\rvert < \epsilon_1\right) > 1-\delta$$

and

$$\text{Pr}\left(\left\lvert \frac{\partial}{\partial \theta} CI(X, Z_E) - \frac{\partial}{\partial \theta} L_X^{(N)}\right\rvert < \epsilon_2\right) > 1-\delta$$

holds uniformly for all model parameters $\theta$.

The loss function is as defined in the paper:

$$L_X = h_q(Z_E) - \mathbb{E}_{x \sim p(x+E)} \left[\log \left\lvert \det J_\theta(x)\right\rvert\right]$$

as well as its empirical estimate using $N$ samples, $L_X^{(N)}$.

We split the proof into two Lemmas, which we will later combine.

**Lemma 1.** For any $\eta_1, \eta_2 > 0$ and $\delta > 0$ there is an $N_0 \in \mathbb{N}$ so that

$$\text{Pr}\left(\left\lvert L_X^{(N)} - L_X\right\rvert < \eta_1\right) > 1-\delta$$

$$\text{Pr}\left(\left\lvert \frac{\partial}{\partial \theta} L_X^{(N)} - \frac{\partial}{\partial \theta} L_X\right\rvert < \eta_2\right) > 1-\delta$$

$\forall N \geq N_0$

**Proof.** For the first part (Eq. 25), we simply have to show that the uniform law of large numbers applies, specifically that all expressions in the expectations are bounded and change continuously with $\theta$. For the Jacobian term in the loss, this is the case by definition. For the $h_q(Z_E)$-term, we can show the boundedness of $\log q$ occurring in the expectation by inserting the GMM explicitly. We find

$$- \log(q(z)) \leq \max_y [(z - \mu_y)^2/2] + \text{const.}$$

while we know that $z = g_\theta(x)$ is bounded. Therefore, the uniform law of large numbers (Newey & McFadden, 1994, Lemma 2.4) guarantees existence of an $N_1$ to satisfy the condition for all $\theta \in \Theta$.

For the second part (Eq. 26), we will show that the gradient w.r.t. $\theta$ and the expectation can be exchanged, as the gradient is also bounded by the same arguments as before. We find that the conditions for exchanging expectation and gradient are trivially satisfied, again due to the bounded gradients (see L’Ecuyer (1995), assumption A1, with $\Gamma$ set to the upper bound). This results in an $N_2 \in \mathbb{N}$ for which Eq. 26 is satisfied. As a last step, we simply define $N_0 := \max(N_1, N_2)$.

**Lemma 2.** For any $\eta_1, \eta_2 > 0$ there is an $\sigma_0 > 0$, so that

$$\left\lvert CI_\theta(X, Z_E) + d \log \sqrt{2 \pi \sigma^2} - L_X\right\rvert < \eta_2$$

$$\left\lvert \frac{\partial}{\partial \theta} CI_\theta(X, Z_E) - \frac{\partial}{\partial \theta} L_X\right\rvert < \eta_2$$

$\forall \sigma < \sigma_0$

**Proof.** In the following proof, we make use of the $O(\cdot)$ notation, see e.g. De Bruijn (1981):

We write $f(\sigma) = O(g(\sigma))$ ($\sigma \to 0$) iff there exists a $\sigma_0$ and an $M \in \mathbb{R}$, $M > 0$ so that

$$\|f(\sigma)\| < Mg(\sigma) \quad \forall \sigma \leq \sigma_0.$$

Furthermore, to discuss the limit case, it is necessary we reparametrize the noise variable $E$ in terms of noise $S$ with a fixed standard normal distribution:

$$E = \sigma S \quad \text{with} \quad p(S) = N(0,1)$$

To begin with, we use the invariance of $CI$ under the homeomorphic transform $g_\theta$. This can be easily verified by inserting the change-of-variables formula into the definition. See e.g. Cover & Thomas (2012) Sec. 8.6. This results in

$$CI(X, Z_E) = CI(Z, Z_E) = h_q(Z_E) - h_q(Z_E|Z)$$

Next, we series expand $Z_E$ around $\sigma = 0$. We can use Taylor’s theorem to write

$$Z_E = Z + J_\theta(Z)E + O(\sigma^2)$$

We have written the Jacobian dependent on $Z$, but note that it is still $\partial g_\theta/\partial X$, and we simply substituted
We can now combine the two Lemmas 1 and 2, to show Proposition 3.

**Proposition 3 - Proof.**

*Proof.* The Proposition follows directly from Lemmas 1 and 2: for a given \( \epsilon_1, \epsilon_2 \) and \( \delta \), we choose each \( \eta_i = \epsilon_i/2 \), and apply the triangle inequality, meaning there exists \( N_0 \) and \( \sigma_0 \) so that

\[
CI(X, Z_{\epsilon}) + d \log \sqrt{2\pi e} \sigma^2 - \mathcal{L}_X^{(N)} \leq CI(X, Z_{\epsilon}) + d \log \sqrt{2\pi e} \sigma^2 - \mathcal{L}_X + \mathcal{L}_X - \mathcal{L}_X^{(N)} < \frac{\epsilon_1}{2} + \frac{\epsilon_1}{2}
\]

And therefore \( \Pr(\ldots) > 1 - \delta \). Equivalently for the gradients. \( \square \)

## 7 NETWORK ARCHITECTURE

As in previous works, our INN architecture consists of so-called coupling blocks. In our case, each block consists of one affine coupling (Dinh et al., 2017), illustrated in Fig. 9, followed by random and fixed soft permutation of channels (Arzìzone et al., 2019), and a fixed scaling by a constant, similar to ActNorm layers introduced by Kingma & Dhariwal (2018). For the coupling coefficients, each subnetwork predicts multiplicative and additive components jointly, as done by Kingma & Dhariwal (2018). Furthermore, we adopt the soft clamping of multiplication coefficients used by Dinh et al. (2017).

For downsampling blocks, we introduce a new scheme, whereby we apply the i-RevNet downsampling (Jacobsen et al., 2018) only to the inputs to the affine transformation (\( u_2 \) branch in Fig. 9), while the affine coefficients are predicted from a higher resolution \( u_1 \) by using a strided convolution in the corresponding subnetwork. After this, i-RevNet downsampling is applied to the other half of the channels \( u_1 \) to produce \( v_1 \), before concatenation and the soft permutation. We adopt this scheme as it more closely resembles the standard ResNet downsampling blocks, and makes the downsampling operation at least partly learnable.

We then stack sets of these blocks, with downsampling blocks in between, in the manner of [8, down, 25, down, 25]. Note, we use fewer blocks for the first resolution level, as the data only has three channels, limiting the expressive power of the blocks at this level. Finally, we apply a discrete cosine transform to replace the global average pooling in ResNets, as introduced by Jacobsen et al. (2019), followed by two blocks with fully connected subnetworks.

We perform training with SGD, learning rate 0.1, momentum 0.9, and batch size 128, as in the original...
Forward computation (left to right):
\[ v_1 = u_1, \quad v_2 = T(u_2; nn(v_1)) \]

Inverse computation (right to left):
\[ u_1 = v_1, \quad u_2 = T^{-1}(v_2; nn(u_1)) \]

Figure 9: Illustration of a coupling block. \( T \) represents some invertible transformation, in our case an affine transformation. The transformation coefficients are predicted by a subnetwork (\( nn \)), which contains fully-connected or convolutional layers, nonlinear activations, batch normalization layers, etc., similar to the residual subnetwork in a ResNet (He et al., 2016). Note that the subnetwork does not have to be inverted itself.

ResNet publication by (He et al., 2016). We train for 450 epochs, decaying the learning rate by a factor of 10 after 150, 250, and 350 epochs.

8 ADDITIONAL EXPERIMENTS

In Figure 11 we show the trajectory of a sample in latent space, when gradually increasing the RGB-rotation OoD augmentation used in the paper. It travels from in-distribution to out-of-distribution. Note that such images were never seen during training.

Figure 10 provides all the performance metrics discussed in the paper over the range of \( \tilde{\beta} \).
Figure 10: Effect of changing the parameter $\tilde{\beta}$ (x-axis) on the different performance measures (y-axis). The VIB results are added as dotted lines for comparison. The arrows indicate if a larger or smaller score is better. Details are explained in the paper.

Figure 11: The scatter plot shows the location of test set data in latent space. A single sample is augmented by rotating the RGB color vector as described in the paper. The small images show the successive steps of augmentation, while the black arrow shows the position of each of these steps in latent space. We observe how the points in latent space travel further from the cluster center with increasing augmentation, causing them to be detected as OoD.