Decomposing Representations for Deterministic Uncertainty Estimation

Haiwen Huang **Joost van Amersfoort** **Yarin Gal**
OATML Group
Department of Computer Science
University of Oxford
Oxford, United Kingdom
{haiwen.huang2, joost.van.amersfoort, yarin}@cs.ox.ac.uk

1 Introduction

Uncertainty estimation is a key component in any deployed machine learning system: when the system encounters unfamiliar or noisy input, it needs to refrain from making a decision and instead defer to an expert. There has been great progress in uncertainty estimation in recent years, particularly in a sub-field called deterministic uncertainty estimation, where only a single forward pass is necessary to accurately estimate uncertainty [1, 2, 3].

One way to evaluate uncertainty estimation is using "out-of-distribution" (OoD) detection, where it is expected that data points far away from the training distribution have higher uncertainty than points similar to the training distribution. A concrete way to evaluate this is by distinguishing between two data distributions, one used for training and one never seen during training, using uncertainty. Recently a challenging version of this benchmark was introduced: distinguishing CIFAR-10 from CIFAR-100 and vice versa. This was called "near" OoD detection to distinguish from "far" OoD detection [4], for example in CIFAR-10 vs SVHN. Near OoD detection is significantly more difficult as the near OoD data are more semantically similar to the training data. For example, CIFAR-10 and CIFAR-100 both come from the 80-million tiny images dataset, but contain no class overlap [5].

It was shown that in the near OoD setting, standard methods such as Mahalanobis distance [1] and DDU [2] do not perform well [6]. We show that the proposed fix in [6] does not work well in the far OoD setting, and propose a simple solution that does work. We identify two types of features: discriminative (class-relevant) and non-discriminative (class-irrelevant) features. Intuitively, one can consider the discriminative features sufficient for distinguishing near OoD data, while non-discriminative features are used for far OoD detection. In Table 1, we show that when using a representation that is a mix of these features, uncertainty estimators based on feature density that perform well on one OoD detection benchmark have poor performance on the other.

This is because when the OoD data mainly differ in one specific type of representations, the other type can still influence the uncertainty. To solve this, we propose to decompose the learned representations into two proposed types of representations and integrate the uncertainties estimated on them separately. Through experiments, we demonstrate that we can greatly improve the performance and the interpretability of the uncertainty estimation.

2 Motivating Observations

First, we use Mahalanobis distance for OoD detection [1] and its variants [2, 4, 6] to demonstrate the influence of class-relevance on uncertainty estimation. Since our analysis is focused on the feature maps, it can be extended to any other representation-based uncertainty estimation methods.

6th workshop on Bayesian Deep Learning (NeurIPS 2021), Sydney, Australia

Table 1: AUROC (%) comparison of variants of Mahalanobis distance calculated on features at penultimate layer (pre-logit).

Method	CIFAR10 vs SVHN	CIFAR10 vs CIFAR100
Maha [1]	**97.96**	75.75
Marginal Maha [9]	96.07	60.10
Relative Maha [6]	95.43	**90.97**

Given a pretrained feature extractor $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and dataset $\{x_i, y_i\}_{i=1}^N$ where input $x_i \in \mathbb{R}^n$, and label $y_i \in \{1, \cdots, C\}$, the Mahalanobis distance for OoD detection first models the distribution of the features as a class-conditional multivariate Gaussian distribution with a single covariance matrix shared across classes. That is, for each class c, it models the distribution of features as $p(z|y = c) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_c, \Sigma)$. It then calculates class-wise Mahalonobis distance (i.e. the same quantity as in the exponent of a multi-variate Gaussian distribution) of a feature representation $z = f(x)$ and uses the maximum across classes as the distance:

$$M(x) = \max_c -(f(x) - \hat{\boldsymbol{\mu}}_c)^T \hat{\Sigma}^{-1}(f(x) - \hat{\boldsymbol{\mu}}_c), \quad (1)$$

where $\hat{\Sigma} = \frac{1}{N} \sum_{c=1}^C \sum_{i:y_i=c} (f(\boldsymbol{x}_i) - \hat{\boldsymbol{\mu}}_c)(f(\boldsymbol{x}_i) - \hat{\boldsymbol{\mu}}_c)^T$, $\hat{\boldsymbol{\mu}}_c = \sum_{i:y_i=c} f(\boldsymbol{x}_i)$. Recently, many works [2, 4] model the feature distribution using a *class-dependent* covariance matrix and found it beneficial. In this case, the uncertainty metric is $M(x) = \max_c[-(f(x) - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} (f(x) - \hat{\boldsymbol{\mu}}_c) - \log((2\pi)^m \det \hat{\boldsymbol{\Sigma}}_c)]$ where $\hat{\boldsymbol{\Sigma}}_c = \frac{1}{|\{i:y_i=c\}|} \sum_{i:y_i=c} (f(\boldsymbol{x}_i) - \hat{\boldsymbol{\mu}}_c)(f(\boldsymbol{x}_i) - \hat{\boldsymbol{\mu}}_c)^T$.

Variants of Mahalanobis distance were proposed to improve on *some specific tasks*. Notably, marginal Mahalanobis distance (*Marginal Maha*) [7] calculates the mean and covariance without using class-relevant information. Specifically, in Equation (1), $\Sigma = \frac{1}{N} \sum_{i=1}^N (f(\boldsymbol{x}_i) - \boldsymbol{\mu})(f(\boldsymbol{x}_i) - \boldsymbol{\mu})^T$, $\boldsymbol{\mu}_c = \boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^N f(\boldsymbol{x}_i)$. Another variant is called Relative Mahalanobis distance (*Relative Maha*) [6] $M_{rel}(x) = M(x) - M_{marg}(x)$ where M_{rel} and M_{marg} refers to *Relative Maha* and *Marginal Maha*, respectively. It aims to mitigate the influence of the class-irrelevant information. As we can see from the definitions, the key difference among the three variants is how the class-relevant information is (not) used. In terms of the degree of class-relevance: *Relative Maha > Maha > Marginal Maha*.

To empirically compare these three methods, we use two OoD detection benchmarks: CIFAR10 [5] vs. SVHN [8] and CIFAR10 vs. CIFAR100. The OoD detection results can be found in Table 1. We can see that on CIFAR10 vs. SVHN, all the methods can perform relatively well with *Maha* being the top. While on CIFAR10 vs. CIFAR100, there is a sharp difference among the methods: *Relative Maha* shows significantly better performance than others while *Marginal Maha* almost completely mixes the two datasets (AUROC=50% means random guess). Note that although the compared methods are well-known, none of them has been tested *simultaneously* on both benchmarks. Our experiments demonstrate that *none of these methods can beat every other method on both benchmarks*.

3 Decomposing Representations

Considering the difference among the three compared methods in the last section, we hypothesize that the differences in OoD detection performances can be attributed to the different *degrees of class relevance*. Therefore, we propose decomposing the discriminative and non-discriminative information from the learned representations. This allows us to estimate uncertainty on each type of representations separately and provides a new perspective to interpret the uncertainty estimation.

Formally, an ideal decomposition of the features z into the discriminative part z_d and non-discriminative part z_n, i.e., $z = (z_d; z_n)$, should have the following properties:

- $I(z_d; z_n|y) = 0$ (the two decomposed parts should be conditionally independent given the labels y);
- $I(z_n; y) = 0$ (z_n is non-discriminative, i.e., independent from the labels y);
- $z_d = \arg\min_S I(z_d; z)$ where $S = \{z_d \mid I(z_d; y) = I(z; y)\}$ (z_d includes and only includes all the discriminative information in z).

Table 2: AUROC (%) comparison of different methods calculated on features at penultimate layer (pre-logit). D_{in} = CIFAR10.

Method	Datasets D_{out} = SVHN $d_{dis} = 3.75$ $d_{nondis} = 4.09$	 D_{out} =CIFAR100 $d_{dis} = 4.55$ $d_{nondis} = 1.15$	 D_{out} = SVHN ∪ CIFAR100 $d_{dis} = 2.08$ $d_{nondis} = 1.73$
Maha	97.96	75.75	86.86
Marginal Maha	96.07	60.10	78.08
Relative Maha	95.43	**90.97**	93.20
PCA top score	94.59	90.20	92.39
PCA bottom score	98.23	83.77	91.00
PCA top score + bottom score	97.95	89.90	93.92
Discriminative score	96.14	90.46	93.30
Non-discriminative score	98.08	76.14	87.11
Dis score + Non-dis score	**98.41**	89.95	**94.18**

To perform the decomposition, a simple way is to use Principal Component Analysis (PCA) to transform the features and then take the first d principal components (PCs) as z_d and the remaining PCs as z_n. However, since PCA is a linear method and does not use any label information, there is no guarantee that PCA can achieve the ideal decomposition. Alternatively, we can use the independent cross-entropy (iCE) loss proposed by [10]. Specifically, given a fixed pretrained feature extractor $f : \mathbb{R}^M \to \mathbb{R}^D$ which maps from inputs to representations, we first perform a transformation of the features $z = f(x)$ using an invertible function $F : \mathbb{R}^D \to \mathbb{R}^D$. The invertibility is to make sure we keep all the information of the features after the transformation. Our aim then is to extract discriminative features z_d into the first d dimensions of $F(z)$ and non-discriminative features z_n into the other dimensions. As a convenient choice, we set $d = C$, i.e., the number of classes of the training dataset. z_d then serves as logits for the classification. We then uses another linear map $D : \mathbb{R}^{D-d} \to \mathbb{R}^C$ to map z_n to a $C-$dimensional logits. In this way, we can now use the iCE loss to encourage the decomposition of discriminative and non-discriminative features. The iCE loss is defined as:

$$\min_\theta \max_\phi \mathcal{L}_{iCE}(\theta, \phi) = \sum_{i=1}^{C} -y_i \log[\text{softmax}[F_\theta(z)]]_i + \sum_{i=1}^{D-C} y_i \log[\text{softmax}[D_\phi\left(F_\theta(z)_{[C+1:D]}\right)]]_i. \tag{2}$$

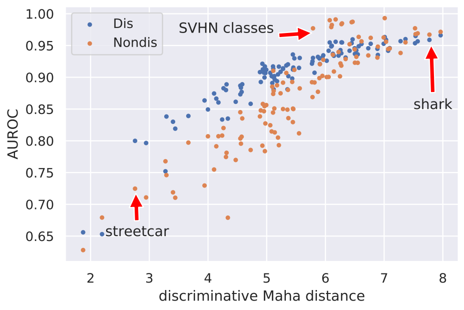
We point out that for the second loss term, the minimization is concerning a lower bound on $I(y; z_n)$, while the maximization aims to tighten the bound [10]. Therefore, the iCE loss aims to maximize $I(y; z_s)$ (first term) and minimize $I(y; z_n)$ (second term) at the same time to achieve the second and third properties in our definition of decomposition (note the minimization of $I(z; z_d)$ is also achieved during optimization [11]). The first independence property is not explicitly enforced in our implementation, but can still be achieved through the optimization. We discuss in the appendix B.

When the decomposition is finished, we can get a new uncertainty estimate based on the uncertainty calculated on the decomposed features. Because ideally, $\log p(z) = \log p(z_d; z_n) = \log p(z_d) + \log p(z_n)$ (since $I(z_d; z_n) = 0$), the new uncertainty estimate can simply be $M'(z) := 1/2M(z_d) + 1/2M(z_n)$, where M is an uncertainty estimator on the features, e.g., *Maha*. We can also design a *KL-based dataset distance metric* based on the decomposition: $d(D_{in}, D_{out}) = \mathcal{D}_{KL}[p_{in}(M(z))\|p_{out}(M(z))]$, where p_{in} and p_{out} are the distributions of the OoD scores (e.g. *Maha* distance) on $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ respectively. Depending on the features we use (i.e., z_d or z_n), we can define d_{dis} and d_{nondis} correspondingly. Details can be seen in the appendix B.3.

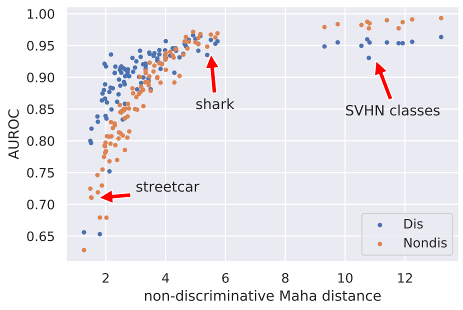
4 Experiments

Main Result

In Table 2, we show the comparison of different methods on the three OoD benchmarks. The scoring function we use is *Maha*. From Table 2, we can see that:



(a)



(b)

Figure 1: AUROCs of OoD detection using discriminative and non-discriminative features (y-axis) against the averaged Maha distances on the two features for each class (x-axis). Every two dots with the same x-axis coordinate belong to the same class in either SVHN or CIFAR100.

1. Our dataset distance metric is a good indicator of the OoD types and the AUROC differences. SVHN is far from CIFAR10 in both features but d_{nondis} is slightly larger. So while either type of features can achieve high AUROC for detecting SVHN, non-discriminative features are slightly better. For CIFAR100, d_{dis} is much larger than d_{nondis}. So when using discriminative features to detect CIFAR100, we can see considerable improvement.
2. Discriminative score performs best on discriminative features, and non-discriminative score performs best on non-discriminative features. Moreover, the sum of the two scores combines the best of both features, yielding close to best performances on both benchmarks and best performance on the union benchmark.
3. PCA is a strong baseline for the decomposition. Note that this conclusion can change when we are not using the feature extractor of a discriminatively trained neural network.

4.1 Interpreting Uncertainty

The dataset distance under different features also provides us with a tool to interpret the uncertainty estimates. In Figure 1, we conduct a class-wise analysis for a fine-grained understanding of the OoD classes. Specifically, we calculate the AUROC using discriminative and non-discriminative scores to detect the 110 classes of SVHN $\cup$ CIFAR100 from CIFAR10 (one class at a time). This gives us two AUROC scores for each class, *dis AUROC* and *nondis AUROC*. We then plot them against the dataset distance between CIFAR10 and each class. In this way, we can interpret the OoD detection performances (y-axis) by looking at dataset distances (x-axis).

We highlight three types of OoD: (1) Classes with much higher non-discriminative distances than discriminative ones, e.g., *SVHN classes*. For these classes, non-discriminative features are more suitable for OoD detection. (2) Classes with small discriminative distances, e.g., *streetcar*. The small discriminative distances are usually due to similar categories in CIFAR10, e.g., automobile and truck for the streetcar. These classes typically also have small non-discriminative distances. Usually, discriminative distances are still larger, so more suitable for OoD detection. (3) Classes with large discriminative distances, e.g., *shark*. These classes typically also have large non-discriminative distances. The OoD performances using the two kinds of distances are usually similar.

5 Discussion

In this work, we showed that current state-of-the-art uncertainty estimation methods could not consistently outperform other methods across different benchmarks. We solved this problem by decomposing the features. Specifically, we summed the OoD scores calculated separately on the discriminative and non-discriminative features and achieved consistently high performance across different types of benchmarks. Our decomposition can also help us interpret the uncertainty by looking at the uncertainty estimates on the decomposed features.

References

[1] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7167–7177, Red Hook, NY, USA, 2018. Curran Associates Inc.

[2] Jishnu Mukhoti, Andreas Kirsch, Joost van Amersfoort, Philip H. S. Torr, and Yarin Gal. Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty. *CoRR*, abs/2102.11582, 2021.

[3] Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. Improving deterministic uncertainty estimation in deep learning for classification and regression, 2021.

[4] Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R. Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, A. Taylan Cemgil, S. M. Ali Eslami, and Olaf Ronneberger. Contrastive training for improved out-of-distribution detection. *CoRR*, abs/2007.05566, 2020.

[5] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[6] Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. A Simple Fix to Mahalanobis Distance for Improving Near-OOD Detection. *arXiv:2106.09022 [cs]*, June 2021. arXiv: 2106.09022.

[7] Ryo Kamoi and Kei Kobayashi. Why is the Mahalanobis Distance Effective for Anomaly Detection? *arXiv:2003.00402 [cs, stat]*, April 2020. arXiv: 2003.00402.

[8] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[9] Ryo Kamoi and Kei Kobayashi. Why is the Mahalanobis Distance Effective for Anomaly Detection? *arXiv:2003.00402 [cs, stat]*, April 2020. arXiv: 2003.00402.

[10] Joern-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability. In *International Conference on Learning Representations*, 2019.

[11] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

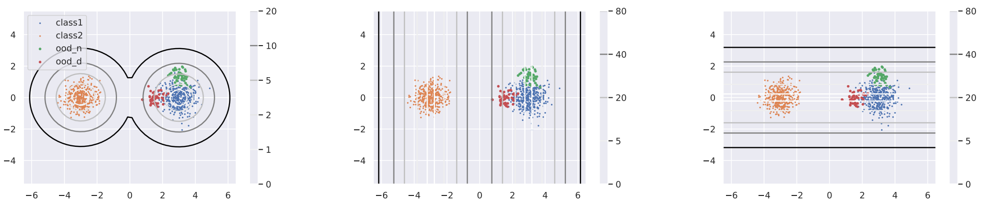
[12] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

[13] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[14] Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. Invertible Residual Networks. In *International Conference on Machine Learning*, pages 573–582. PMLR, May 2019. ISSN: 2640-3498.

[15] Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

[16] Qing Wang, Sanjeev R Kulkarni, and Sergio Verdú. Divergence estimation for multidimensional densities via k-nearest-neighbor distances. *IEEE Transactions on Information Theory*, 55(5):2392–2405, 2009.



(a) Mahalanobis distance on full features.
AUROC$_{red}$ = 0.94,
AUROC$_{green}$=0.94.

(b) Mahalanodis distance on discriminative features.
AUROC$_{red}$= 0.98,
AUROC$_{green}$ $= 0.36$.

(c) Mahalanodis distance on non-discriminative features.
AUROC$_{green}$ = 0.98,
AUROC$_{red}$ $= 0.34$.

Figure 2: A 2D example showing the effect of decomposition. The *orange* and *blue* points are two in-distribution classes, *green* points are OoD from non-discriminative directions and *red* points are OoD from discriminative directions. Contour lines are colored according to the Mahalanobis distance to the in-distribution points (darker means higher). We can see that *Maha* on the full feature weighs on both discriminative and non-discriminative directions simultaneously (and in our case, equally). This makes *Maha* on full features (**(a)**) suboptimal in detecting OoD data compared to *Maha* calculated on either direction only (**(b), (c)**).

A Additional Experiments

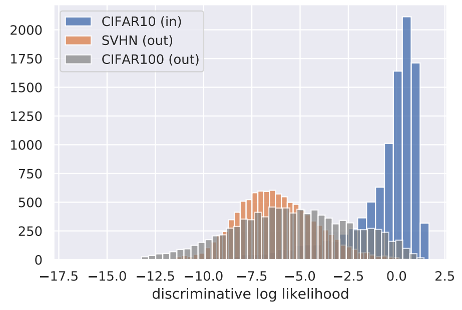
A.1 Toy experiments

We consider both low-dimensional (2D) simulation and high-dimensional (128D) simulation. For the low dimensional simulation, we generate two Gaussian distributions with mean (3,0) and (-3,0) and tied covariance matrix $\Sigma = 0.5I_2$. The two OoD distributions are also Gaussian distributions with mean (3,1.4) and (1.6, 0) and the same covariance matrix $\Sigma = 0.3I_2$. We then use Equation (1) to calculate the Mahalanobis distance on the chosen features (full, discriminative, non-discriminative). The results in Figure 2 show that *Maha* calculated on the full features can only reach sub-optimal performance (94% AUROC) compared to on the decomposed features for the corresponding type of OoD data (98% AUROC).

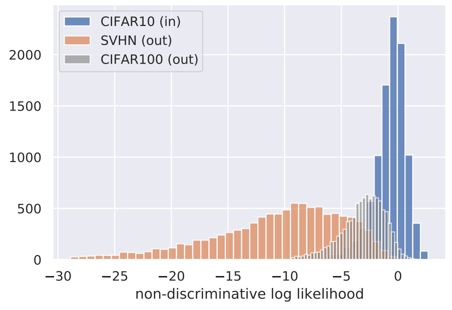
For the high-dimensional simulation, we generate ten 128D Gaussian distributions $\{\boldsymbol{z}_c \sim \mathcal{N}(\boldsymbol{\mu}_c, \Sigma)\}_{c=1}^{10}$ with mean $\boldsymbol{\mu}_c = 10\boldsymbol{e}_c$, diagonal covariance matrix $\Sigma = I_{128}$. Here $\boldsymbol{e}_c$ is the standard basis. We also consider two types of OoD: (1) OoD along discriminative directions: $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}_k, \Sigma)$ where $\boldsymbol{\mu}_k = 5\boldsymbol{e}_k$, $k \in \{1, 2, ..., 10\}$.; (2) OoD along non-discriminative directions: $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}_{k'}, \Sigma)$ where $\boldsymbol{\mu}_{k'} = 10\boldsymbol{e}_c + 5\boldsymbol{e}_l$, $c \in \{1, 2, ..., 10\}, 11 \leq l \leq 128$. The remaining process of calculating Mahalanobis distance is the same as the low-dimensional simulation. When using Mahalanobis distance on the full feature, the OoD detection performance is 83.95% AUROC. While we we use Mahalanobis distance on the decomposed features, we can achieve 99.13% AUROC on OoD data along discriminative directions and 84.41% along non-discriminative directions. This shows that the *uncertainty estimation is dominated by the non-discriminative representations*. This is similar to the CIFAR10 vs CIFAR100 benchmark (see Table 2).

A.2 Histogram comparison on image datasets

To further understand the dataset distances under two different features, in Figure 3, we show the histograms of the two scores (log likelihoods) calculated on the three datasets. From the histogram, we can see that SVHN is far from CIFAR10 in both histograms, but the absolute values of non-discriminative log likelihoods are larger, so they can be easier to detect. For CIFAR100, there is a sharp difference between two features: the discriminative log likelihoods are more flat and far from CIFAR100, while the non-discriminative log likelihoods are very close to CIFAR10. So discriminative scores are a better choice to detect CIFAR100. These observations are aligned with the AUROC results of OoD detection and the dataset distances in Table 2.



(a)



(b)

Figure 3: OoD detection using uncertainty estimated on different features. CIFAR10 is the in-distribution dataset, and SVHN and CIFAR100 are two different OoD datasets. Notably, when using non-discriminative features, CIFAR100 is much closer to the in-distribution than SVHN.

A.3 Setup of experiments on image datasets

For the experiments on image datasets, we use CIFAR10 [12] as the in-distribution datasets and SVHN [8], CIFAR100 [12], and the union of the two datasets as three different benchmarks. This is to demonstrate better how OoD datasets' differences influence the OoD detection performances using different methods.

We use WRN-28-10 [13] trained on CIFAR10 as our pretrained model to extract the penultimate features for different methods. For the decomposition using iCE loss, the invertible transformation is implemented as a 4-layer invertible residual networks [14] with linear residual function and 0.9 spectral coefficient. The decomposition takes 3000 iterations using SGD optimizer, and we get 96.12% test accuracy for discriminative logits and 11.2% test accuracy for non-discriminative logits. For each method, we repeat the experiments using five differently randomly initialized models and report the mean.

B Discussion of the Decomposition

B.1 Definition of the decomposition

In Section 3, we listed three properties that an ideal decomposition should possess. For demonstration purposes, we can write them in a slightly redundant way:

- $I(z_d; z_n|y) = 0$ (the two decomposed parts should be conditionally independent given the labels y);
- $I(z_n; y) = 0$ (z_n are non-discriminative, i.e., independent from labels y);
- $I(z_d; y) = I(z; y)$ (z_d includes all the discriminative information in z);
- $z_d = \arg\min_S I(z_d; z)$ where $S = \{z_d \mid I(z_d; y) = I(z; y)\}$ (z_d *only* includes all the discriminative information in z).

We first show that we can derive the third property from the first two. To start with, from the first two properties we have $I(z_d; z_n) = I(z_n; y) + I(z_n; z_d \mid y) - I(z_n; y \mid z_d) = -I(z_n; y \mid z_d) \leq 0$, thus $I(z_d; z_n) = 0$. Then since $I(z_d; z_n; y) = I(z_n; y) - I(z_n; y \mid z_d) = I(z_d; z_n) - I(z_d; z_n \mid y)$, we can derive that $I(z_n; y \mid z_d) = I(z_d; z_n; y) = 0$. Finally, combined with the chain rule of mutual information, $I(z; y) = I(z_d; y) + I(z_n; y \mid z_d)$, we can derive the third property $I(z; y) = I(z_d; y)$. Similarly, when the first and third properties are satisfied, we can also derive the second property.

The fourth property is to ensure that z_d does not include more information than discriminative part. For example, simply setting $z_d = z$ and $z_n = 0$ could satisfy the first three properties. Note that since $I(z_d; y) \leq I(z; y)$, to achieve the fourth property, we may use the following optimization scheme:

$$z_d = \arg\min I(z_d; z) - I(z_d; y).$$

We can achieve this by adding a cross entropy loss on the logits (z_d) based on the information bottleneck method [11].

We can also show that when the last three properties are satisfied, the first property will be satisfied as well. As a sketch proof, assume the last three properties are already satisfied for z_d, z_n, if $I(z_d; z_n|y) > 0$, we can then simply set $z'_d = z_d|z_n$ [1], then $I(z'_d; z) = I(z'_d; z_n) + I(z'_d; y) + H(z'_d|y, z_n) = I(z_d; y) + H(z_d|y, z_n) < I(z_d; z_n) + I(z_d; y) + H(z_d|y, z_n) = I(z_d; z)$, while $I(z_n; y) = 0$ and $I(z'_d; y) = I(z; y)$ still hold. This leads to a contradiction against the last property. Intuitively, when z_d *only* includes discriminative information and z_n *only* includes the rest information of z, z_d and z_n are conditionally independent given labels y.

B.2 Implementation of the decomposition

The relationship among the properties of the decomposition gives us two options of implementing the decomposition. One implementation is to maximize $I(z_d; y)$ (and minimize $I(z_d; z)$) and minimize $I(z_n; y)$ simultaneously using iCE loss, as detailed in Section 3 of the paper. From our discussion above, this will also lead to independence of z_d and z_n. Alternatively, we can also put cross entropy loss only on z_d and add an independence regularizer (e.g., HSIC [15]) on z_d and z_n. This can enforce the first and third properties in Section 3 (or first, third, and fourth in A.1), and as we demonstrated, the second property will be then satisfied automatically. This can be a future direction for exploration.

It is worth pointing out that throughout our work, we are focusing on the scenario where we are given a pre-trained feature extractor f, so $z = f(x)$ is fixed for each x. In practice, the properties of z also play a big role in the success of the decomposition. If the feature extractor f cannot extract semantically meaningful features, the decomposition will also fail. When the feature extractor is bi-Lipschitz, we will have $I(z; y) = I(x; y)$. In most cases, we can simply think $I(x; y)$ as an upper bound for $I(z_d; y)$ in the decomposition, i.e., we want z_d to extract all the discriminative information in the *input* x.

B.3 Dataset distance metric based on decomposition

We can design a new dataset distance metric by measuring the difference in the distributions of uncertainty estimates. The decomposition gives such divergence a clear meaning: dataset distances in either discriminative or non-discriminative directions.

To make the scores of the two features comparable, we first normalize the discriminative (dis) and non-discriminative (non-dis) scores of the test data using the mean and standard deviation of the dis and non-dis scores calculated on the training data. To reflect the difference between the two distributions of scores $p_{in}(M(z))$ and $p_{out}(M(z))$, we can compute a Kullback–Leibler (KL) divergence $D_{KL}[p_{in}\|p_{out}] = \sum_{k=1}^{K} p_{in,k} \log \frac{p_{in,k}}{p_{out,k}}$. In our case, since we only have the samples drawn from the two distributions p_{in}, p_{out}, we use an estimator of the KL-divergence based on k-nearst-neighbor distances [16]. When we use the decomposed discriminative and non-discriminative features, we can also define $d_{dis} = D_{KL}[p_{in}(M(z_d))\|p_{out}(M(z_d))]$ and $d_{nondis} = D_{KL}[p_{in}(M(z_n))\|p_{out}(M(z_n))]$. We show d_{dis} and d_{nondis} in Table 2.

[1]Note $I(z_d; z_n|y) = I(z_d; z_n)$ when the second and third properties hold. This is because $I(z_n; y|z_d) = I(z; y) - I(z_d; y) = 0$, thus $I(z_d; z_n; y) = I(z_n; y) - I(z_n; y|z_d) = 0$. And since we also have $I(z_d; z_n; y) = I(z_d; z_n) - I(z_d; z_n|y)$, we now get $I(z_d; z_n|y) = I(z_d; z_n)$.