A Novel Data Augmentation Technique for Out-of-Distribution Sample Detection using Compounded Corruptions

Ramya Hebbalaguppe[1,2] Soumya Suvra Ghosal[1] Jatin Prakash[1]
Harshad Khadilkar[2] Chetan Arora[1]
[1]IIT Delhi, India [2]TCS Research, India
https://github.com/cnc-ood

Abstract. Modern deep neural network models are known to erroneously classify out-of-distribution (OOD) test data into one of the in-distribution (ID) training classes with high confidence. This can have disastrous consequences for safety-critical applications. A popular mitigation strategy is to train a separate classifier that can detect such OOD samples at test time. In most practical settings OOD examples are not known at train time, and hence a key question is: *how to augment the ID data with synthetic OOD samples for training such an OOD detector?* In this paper, we propose a novel **C**ompou**n**ded **C**orruption (CnC) technique for the OOD data augmentation. One of the major advantages of CnC is that it does not require any hold-out data apart from training set. Further, unlike current state-of-the-art (SOTA) techniques, CnC does not require backpropagation or ensembling at the test time, making our method much faster at inference. Our extensive comparison with 20 methods from the major conferences in last 4 years show that a model trained using CnC based data augmentation, significantly outperforms SOTA, both in terms of OOD detection accuracy as well as inference time. We include a detailed post-hoc analysis to investigate the reasons for the success of our method and identify higher relative entropy and diversity of CnC samples as probable causes. Theoretical insights via a piece-wise decomposition analysis on a two-dimensional dataset to reveal (visually and quantitatively) that our approach leads to a tighter boundary around ID classes, leading to better detection of OOD samples.

Keywords: OOD detection · Open Set recognition · Data augmentation

1 Introduction

Deep neural network (DNN) models generalize well when the test data is independent and identically distributed (IID) with respect to training data [42]. However, the condition is difficult to enforce in the real world due to distributional drifts, covariate shift, and/or adversarial perturbations. A *reliable* system based on a DNN model must be able to detect an OOD sample, and either abstain from making any decision on such samples, or flag them for human intervention. We assume that the in-distribution (ID) samples belong to one of the K known classes, and club all OOD samples into a new class called a *reject*/OOD class. We do not attempt to identify which specific class (unseen label) the unknown sample belongs to. Our goal is to build a classifier to accurately detect OOD samples as the $(K+1)^{\mathrm{th}}$ OOD class, with an objective to reject samples belonging to any novel class.

Most techniques for OOD detection assume the availability of validation samples from the OOD set for tuning model hyper-parameters [33,31,2,19]. Based on the samples, the techniques either update the model weights so as to predict lower scores for the OOD samples, or try to learn correlation between activations and the output score vector [31]. Such approaches have limited utility as in most practical scenarios, either the OOD samples are not available, or cover a tiny fraction of OOD sample space. Yet, other class of techniques learn the threshold on the uncertainty of the output score using deep ensembling [28] or MC dropout [9]. Understandably, OOD detection capability of these techniques suffer when the samples from a different OOD domain are presented.

The other popular class of OOD detectors do not use representative samples from OOD domain, but generate them synthetically [17,36,37]. The synthetic samples can be used to train any of the earlier mentioned SOTA models in lieu of the real OOD samples. This obviates the need for any domain specific OOD validation set. Such methods typically use natural corruptions (e.g. blur, noise, and geometric transformations etc.) or adversarial perturbations to generate samples near decision boundary of a classifier. This class also have limited accuracy on real OOD datasets, as the synthetic images generated in such a way are visually similar/semantically similar to the ID samples, and the behavior of a DNN when shown natural OOD images much farther (in terms of ℓ_2 distance in RGB space) from the ID samples still remains unknown.

Recent theoretical works towards estimating or minimizing open set loss recommend training with OOD samples covering as much of the probable input space as possible. For example, [24] show that a piece-wise DNN model shatters the input space into a polyhedral complex, and prove that empirical risk of a DNN model in a region of input space scales inversely with the density of training samples lying inside the polytope corresponding to the region. Similarly, [8] show that under an unknown OOD distribution, the best way to minimize the open set loss is by choosing OOD samples uniformly from the support set in the input space. Encouraged by such theoretical results, we propose a data augmentation technique which does not focus on generating samples visually similar to the ID samples but synthesizing OOD samples in two key regions of the input space: (i) finely distributed at the boundary of ID classes, and (ii) coarsely distributed in the inter-ID sample space (See Sec. 3.3 for details). We list the key contributions:

1. We propose a novel data augmentation strategy, **C**ompou**n**ded **C**orruptions (CnC) for OOD detection. Unlike contemporary techniques [12,19,31,33] the proposed approach does not need a separate OOD train or validation dataset.
2. Unlike SOTA techniques which detect OOD samples by lowering the confidence of ID classes [1,18,31,35], we classify OOD samples into a separate reject class. We show empirically that our approach leads to clearer separation between ID and OOD samples in the embedding space (Fig. 4).

3. Our method does not require any input pre-processing at the test time, or a second forward pass with perturbation/noise. This makes it significantly faster in inference as compared to the other SOTA methods [22,33].
4. Visualization and analysis of our results indicate that finer granularity of the polyhedral complex around the ID regions learnt by a model is a good indicator of performance of a OOD data augmentation technique. Based on our analysis, we also recommend higher entropy and diversity of generated OOD samples as good predictors for OOD detection performance.

2 Related Work

Our approach is a hyper-parameter-free OOD detection technique, which does not need access to a validation OOD dataset. We review contemporary works below.

Hyper-parameter tuning using OOD data This class comprises of OOD detection methods that fine-tune hyper-parameters on a validation set. ODIN [33] utilizes temperature scaling with input perturbations using the OOD validation dataset to tune hyper-parameters for calibrating the neural networks. However, hyper-parameters tuned with one OOD dataset may not generalize to other datasets. Lee et al.[31] propose training a logistic regression detector on the Mahalanobis distance vectors calculated between test images' feature representations and class conditional Gaussian distribution at each layer.

Retraining a model using OOD data G-ODIN [22] decompose confidence score along with modified input pre-processing for detecting OOD, whereas ATOM [2] essentially makes a model robust to the small perturbations, and hard negative mining for OOD samples. MOOD [34] introduce multi-level OOD detection based on the complexity of input data, and exploit simpler classifier for faster OOD inference.

Using a pre-trained model's score for OOD detection Hendrycks and Gimpel [18] use maximum confidence scores from a softmax output to detect OOD. Liu et al.[35] use energy as a scoring function for OOD detection without tuning hyper-parameters. Shastry and Oore [41] leverage p^{th}-order Gram matrices to identify anomalies between activity patterns and the predicted class. Blundell et al.[1] focus on a closed world assumption which forces a DNN to choose from one of the ID classes, even for the OOD data. *OpenMax* estimates the probability of an input being from an unknown class using a Weibull distribution. G-OpenMax[10] explicitly model OOD samples and report findings on small datasets like MNIST.

OOD detection using uncertainty estimation OOD samples can be rejected by thresholding on the uncertainty measure. Graves et al.[11], Wen et al.[46] propose anomaly detection based on stochastic Bayesian inference. Gal et al.[9] propose MC-dropout to measure uncertainty of a model using multiple inferences. Deep Ensembles [28] use multiple networks trained independently to improve uncertainty estimation.

Data augmentation for `OOD` detection This line of research augments the training set to improve `OOD` detection. Data augmentations like flipping and cropping generate samples that can be easily classified by a pre-trained classifier. Generative techniques based on VAEs, and GANs try to synthesize data samples near the decision boundary [7,30,32,39,47,45,40]. Other data augmentation strategies do not directly target `OOD` detection, but domain generalization: SaliencyMix [44], CutOut[6], GridMask[3], AugMix [20], RandomErase [52], PuzzleMix [26], RandAugment [4], SuperMix [5]. Mixup [51] generates new data through convex combination of training samples and labels to improve DNN generalization. CutMix [48] which generates samples by replacing an image region with a patch from another training image. The approach is not directly suitable for `OOD` detection, as the generated samples lie on the line joining the training samples, and may not cover the large input space[24,8].

3 Proposed Approach

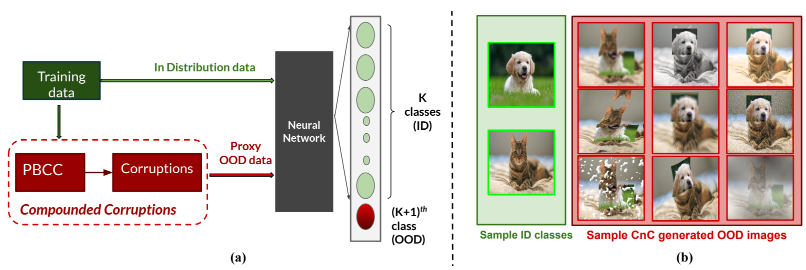


Fig. 1. Creating augmented data samples using Compounded Corruptions (CnC). Pane (a) shows block diagram of the training procedure: first we take a patch based convex combination (PBCC) of patches chosen from image pair belonging to $\binom{K}{2}$ labels; second, we apply corruptions on the data points obtained using PBCC. This proxy `OOD` data is then used to train a $(K+1)$ way classifier, where, first K classes correspond to the `ID` classes and $(K+1)^{th}$ class contains synthesized `OOD` samples corresponding to reject/`OOD` class. Pane (b) shows CnC synthesized sample images from `cat` and `dog` classes. Intuitively, CnC gives two knobs for generating `OOD` samples: a coarse exploration ability through linear combination of two `ID` classes achieved through PBCC operation, and a finer warping capability through corruption of these images. The order of the two operations (PBCC before corruption) is important, as we show later.

3.1 Problem Formulation

We consider a training set, $\mathcal{D}_{\text{in}}^{\text{train}}$, consisting of N training samples: $(x_n, y_n)_{n=1}^{N}$, where samples are drawn independently from a probability distribution: $\mathcal{P}_{X,Y}$. Here, $X \in \mathcal{X}$ is a random variable defined in the image space, and $Y \in \mathcal{Y} =$

$\{1, \dots, K\}$ represents its label. Traditionally, a classifier $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ is trained on in-distribution samples drawn from a marginal distribution $\mathcal{P}_X$ of X derived from the joint distribution $\mathcal{P}_{X,Y}$. Let θ refers to model parameters and $\mathcal{Q}_X$ be another distinct data distribution defined on the image space $\mathcal{X}$. During testing phase, input images are drawn from a conditional mixture distribution $\mathcal{M}_{X|Z}$ where $Z \in \{0, 1\}$, such that $\mathcal{M}_{X|Z=0} = \mathcal{P}_X$, and $\mathcal{M}_{X|Z=1} = \mathcal{Q}_X$. We define all $\mathcal{Q}_X \not\approx \mathcal{P}_X$ as OOD distributions, and Z is a latent (binary) variable to denote ID if $Z = 0$ and OOD if $Z = 1$.

One possible approach to detecting an OOD sample is if confidence of f_θ for a given input is low for all elements of $\mathcal{Y}$. However, we use an alternative approach where we learn to map OOD samples generated using our technique to an additional label $(K + 1)$. Given any two ID samples $x_1, x_2 \sim \mathcal{P}_X$, we generate the synthetic data using the CnC operation $C(x_1, x_2) : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{X}$. We then define an extended label set $\mathcal{Y}^+ = \{1, \dots, K + 1\}$, and train a classifier f_θ^+ over $\mathcal{Y}^+$. The goal is to train f_θ^+ to implicitly build an estimate $\hat{Z}$ of Z, such that the output of f_θ^+ is $(K + 1)$ if $\hat{Z} = 1$, and one of the elements of $\mathcal{Y}$ if $\hat{Z} = 0$.

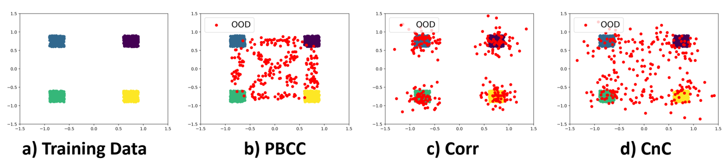


Fig. 2. Intuition with an illustrative plot of OOD synthesis on a toy dataset with four ID classes. Each sample is in $\mathbb{R}^2$. Consider $\mathbf{p}_1 = (x_1, y_1)$, and $\mathbf{p}_2 = (x_2, y_2)$ to be the two input samples belonging to distinct classes 1 and 2, then $\mathbf{p}_3 = (x_3, y_3)$ is the geometric convex combination of $\mathbf{p}_1$ and $\mathbf{p}_2$ such that: $\mathbf{p}_3 = \lambda \mathbf{p}_1 + (1 - \lambda)\mathbf{p}_2$, $0 \leq \lambda \leq 1$. (a) training data corresponding to 4 distinct classes; Synthesised OOD points are in red; (b) PBCC generates OOD points through a convex combination of ID points from different classes in $\binom{4}{2}$ ways, whereas corruptions depicted in (c) can generate OOD points around each cluster. Observe that points generated by CnC spans wider OOD space including inter-ID-cluster area and outside the convex hull of ID points.

3.2 Synthetic OOD Data Generation

Our synthetic sample generation strategy consists of following two steps.

Step 1: Patch Based Convex Combination (PBCC) We generate synthetic samples by convex combination of two input images. Let $x \in \mathbb{R}^{W \times H \times C}$, and y denote a training image and its label respectively. Here, W, H, C denote width, height, channels of the image respectively. A new sample, $\tilde{x}$, is generated by a convex combination of two training samples (x_A, y_A), and (x_B, y_B):

$$\tilde{x} = \mathbf{M} \odot x_A + (\mathbf{1} - \mathbf{M}) \odot x_B. \quad (1)$$

Here, x_A and x_B do not belong to a same class ($y_A \neq y_B$), and $\mathbf{M} \in \{0,1\}^{W \times H}$ denotes a rectangular binary mask that indicates which region to drop, or use from the two images. $\mathbf{1}$ is a binary mask filled with ones, and $\odot$ is element-wise multiplication. To sample $\mathbf{M}$, we first sample the bounding box coordinates $\mathbf{B} = (r_x, r_y, r_w, r_h)$, indicating the top-left coordinates, and width, and height of the box. The region $\mathbf{B}$ in x_A is cut-out and filled in with the patch cropped from $\mathbf{B}$ of x_B. The coordinates of $\mathbf{B}$ is uniformly sampled according to: $r_x \sim \mathrm{U}(0, W), r_w = W\sqrt{1-\lambda}$ and similarly, $r_y \sim \mathrm{U}(0, H), r_h = H\sqrt{1-\lambda}$. Here, $\lambda \in [0,1]$ denotes the crop area ratio, and is fixed at different values for generating random samples. The cropping mask $\mathbf{M}$ is generated by filling zeros within the bounding box $\mathbf{B}$ and ones outside. We generate the samples by choosing each pair of labels in $\binom{K}{2}$ ways, and then randomly selecting input images corresponding to the chosen labels. This generates OOD samples spread across various inter-class regions in the embedding space. For ablation on range of λ to ensure that a large number of OOD samples are generated outside the ID clusters see supplementary. We label all generated samples as that of the $(K+1)^{\text{th}}$ reject class.

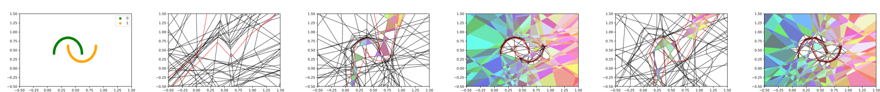
PBCC and CutMix [48]: Note that PBCC and CutMix[48] both rely on the same basic operation **convex combination of images**, but for two very different objectives. Whereas, CutMix uses the combination step to guide a model to attend on less discriminative parts of objects e.g. leg as opposed to head of a person letting the network generalize better on object detection. On the other hand, we use PBCC as a first step for OOD data generation, where the operation generates samples in a large OOD space between a pair of classes in $\binom{K}{2}$ ways.

PBCC Shortcomings: Note that PBCC performs a convex combination of the two ID images belonging to two distinct classes. Hence, unlike adversarial perturbations, it is able to generate sample points far from the ID points in the RGB space. However, still it can generate samples from only within the convex hull of the ID points corresponding to all classes.Thus, as we show in our ablation studies, sample generated using this step alone are insufficient to train a good OOD detector. Below we show how to improve upon the shortcoming of PBCC.

Step 2: Compounded Corruptions We aim to address the above shortcomings by using corruptions on top of PBCC generated samples, thus increasing the sample density in inter-class regions as well as generating samples outside the convex hull. We reason that such compounded corruptions increase the spread of the augmented data to a much wider region. Thus, a reasoning based on "per sample" generalisation error bound from [24]:[Fig. 1, Equation 11] could be utilized for our problem. [24] constructs an input-dependent generalization error bound by analysing the subfunction membership of each input, and show that generalisation error bound improves with smoother training sample density (as defined by number of samples in each region). Intuitively, corruptions over PBCC produces a smoother approximation of ID classes with a finer fit at the ID class boundary. A detailed analysis is given in Fig. 3.3. To give an intuitive understanding, Fig 2 shows visualizations of the generated OOD samples in red using a 4 class toy dataset in two dimensions.

Hendrycks et al. [17] benchmark robustness of a DNN using 15 algorithmically generated image corruptions that mimic natural corruptions. Each corruption severity ranges from 1 to 5 based on the intensity of corruption, where 5 is most severe. The corruptions can be seen as perturbing a sample point in its local neighborhood, while remaining in the support space of the probability distribution of valid images. We apply these corruptions on the samples generated using PBCC step described earlier. Together, PBCC, and corruptions, allow us to generate a synthetic sample far from, and outside the convex hull of ID samples. At the same time, unlike pure random noise images, the process maintains plausibility of the generated samples. Specifically we apply following corruptions: Gaussian noise, Snow, Fog, Contrast, Shot noise/Poisson noise, Elastic transform, JPEG compression, and blur such as Defocus, Motion etc.

Fig. 1 gives a pictorial overview of the overall proposed scheme with a few OOD image samples generated by our approach. CnC formulates the problem as $(K+1)$ class classification which improves the model representation of underlying distribution, and at the same time improves DNN calibration as seen in Sec. 5.2. Please see Suppl. for the precise steps of our algorithm.



(a) Training Data (b) Vanilla (c) PBCC (d) Corruption (e) r-CnC (f) CnC

Fig. 3. Visualization of trained classifiers as a result of OOD augmentation. A ReLU type DNN is trained on the two-dimensional half-moon data set shown in (a). The shattered neural networks [16] show that CnC has the tightest fit around the ID regions, as measured by the area of the (white colored) polytopes in which no training ID point is observed but a network predicts a point in that region as ID. The measured areas for such polytopes are (b)**Vanilla training without data augmentation**:5.65, (c)**PBCC**: 8.20, (d) **Corruption**:0.40, (e)**r-CnC**: 5.66, (f)**CnC**: 0.37. Note: [24] state that the more densely supported a polytope is by the training set, the more reliable the network is in that region. Hence, the samples declared ID in the regions where no ID sample is observed may actually be OOD with high probability. We observe that PBCC/r-CnC/Vanilla, all predict ID in many such polytopes. Note: r-CnC we reverse the order of PBCC and corruptions **Best viewed at 200%**

3.3 CnC Analysis via Polyhedral Decomposition of Input Space

While we validate the improved performance of CnC in Sec. 5, in this section we seek to provide a plausible explanation for the CnC's performance. We draw inspiration from theoretical support provided in recent work by [24] who formally derive and empirically test prediction unreliability for ReLU based neural networks.

Consider a ReLU network with n inputs and m neurons in total. [24] show that parameters of a trained model partition the input space into a polyhedral complex (PC) consisting of individual *convex* polytopes (also called *activation regions* in [24]). See Fig. 3 for an example with a 2D input space. Each possible input corresponds to a unique state (active or inactive) of each of the m ReLU neurons, and the interior of each polytope corresponds to a unique combination of states of all m neurons. Thus a trained network behaves linearly in the interior of corresponding polytopes. Each edge in the PC corresponds to the state flip of a single neuron (active to inactive, or vice versa).

For the purpose of classification based on the final layer activation, a key corollary from [24] is that *the decision boundary between two classes must be a straight line within a polytope, and can only turn at the vertices.* This is an immediate consequence of the observation that the decision boundary is the locus along which the two highest activations (most probable labels) in the output layer remain equal to each other. This implies that smaller polytopes near the decision boundary are needed for finer control over the boundary between training samples from different classes. Note also that the authors in [24]:[equation (11)] infer that (paraphrased) "the more a subfunction (polytope) is surrounded by samples with accurate predictions, the lower its empirical error and bound on generalization gap, and thus the lower its expected error bound".

The key question from OOD detection perspective is, how do we force a network to create tighter polytopes at the ID class decision boundaries? We believe the answer is to distribute a large number of the augmented samples (over which we have control) with contrasting OOD and ID labels all around each ID region, forcing the decision boundary to form a tight bounding surface. At the same time, we must also retain a good fraction of the augmented samples in the open space between ID classes, which can be covered by relatively large polytopes (recall that the maximum number of polytopes is bounded by the number of neurons, and thus small polytopes in one region may need to be traded off by larger polytopes in another region). Neglecting the inter-ID space entirely would run the risk of creating very large polytopes in this region, which increases the empirical error bound ([24]:[equation (5) and (11), large subfunctions have low probability mass and hence higher error bound. Refer Supplementary for further details.]. CnC lets us achieve this dual objective by using compounding to sample the space between ID classes, and corruption to pepper the immediate neighborhoods around ID classes (especially for λ values near 0 and 1).

In Fig. 3, we show polyhedral complex corresponding to the DNN models trained on two-dimensional half-moon dataset [16,25], and OOD samples generated using various techniques. The first plot shows the input space with training samples from two ID classes (green and yellow semicircles). The learnt polytope structure for vanilla uses a neural network of size $[2, 32, 32, 2]$, while the remaining three plots use $[2, 32, 32, 3]$ (with an additional *reject*/OOD class).

Recall from Fig. 2 that PBCC produces samples sparsely between the ID classes, but not around the ID class boundaries. Pure corruptions produce samples only near and on ID classes, but not in the inter-ID space. On the other hand, CnC produces samples both near the ID boundaries as well as in the inter-ID space. In Fig. 3, we define any polytope that is fully or partially (decision boundary crosses through it) classified as ID, as an "ID classified polytope" and mark it in white color. *Visually, we can see that the white polytopes occupy a smaller total area when we compare Vanilla to CnC, with the actual values noted in the caption. This indicates that the CnC produces the tightest approximation of ID classes in our example, which in turn leads to better OOD detection.* Though we show for two-dimensional data, we posit that the same generalizes to higher dimensional input data as well, and is the reason for success of CnC based OOD detection.

CnC and Robustness to Adversarial Attacks: Note that, small polytopes in the input space partitioned by a DNN may also provide better safety against black box adversarial attacks as suggested by [16,25]. This is because the black box adversarial attacks extrapolate the gradients based upon a particular test sample. Since the linearity of the output, and thus the gradients is only valid inside a polytope, smaller polytopes near the ID or in the OOD region makes it difficult for an adversary to extrapolate an output to a large region. However, since adversarial robustness is not the focus of this paper, we do not further explore this direction.

3.4 Training Procedure

We train a $(K+1)$ class classifier network f_θ^+, where first K classes correspond to the multi-classification ID classes, and the $(K+1)^{th}$ class label indicates the OOD class. Our training objective takes the form:

$$\begin{aligned}\mathcal{L} = \underset{\theta}{\text{minimize}} \quad & \mathbb{E}_{(x,y)\sim D_{\text{in}}^{\text{train}}}[\mathcal{L}_{\text{CE}}(x, y; f_\theta^+(x))] \\ & +\alpha \cdot \mathbb{E}_{(x,y)\sim D_{pbcc}^{corr}}[\mathcal{L}_{\text{CE}}(x, K+1; f_\theta^+(x))], \end{aligned} \tag{2}$$

where $\mathcal{L}_{\text{CE}}$ is the cross entropy loss, $f_\theta^+(x)$ denotes the softmax output of neural network for an input sample x. We use $\alpha = 1$ in our experiments based on the ablation study reported in the supplementary material. For above experiments setup we set the ratio of IID:OOD training points as $1:1$.

3.5 Inference

After training, we obtain a trained model F^+. We use $F^+(x)[K+1]$ as the OOD score of x during testing, and define an OOD detector $D(x)$ as:

$$D(x) = \begin{cases} 0, & \text{if } F^+(x)[K+1] > \delta \\ 1, & \text{if } F^+(x)[K+1] \leq \delta \end{cases} \tag{3}$$

where, $D(x) = 0$ indicates an OOD prediction, and $D(x) = 1$ implies an ID sample prediction. δ is a threshold such that TPR, i.e., fraction of ID images correctly classified as ID is 95%. For images which are characterized as ID by $D(x)$, the labels are given as:.

$$\hat{y} = \underset{i \in 1,\ldots,K}{\arg\max}\, F^+(x)_i \tag{4}$$

4 Dataset and Evaluation Methodology

In-Distribution Datasets: For ID samples, we use SVHN (10 classes) [38], CIFAR-10 (10 classes), CIFAR-100 (100 classes)[27] containing images of size 32×32. We also use TinyImageNet (200 classes) [29] containing images of resolution 64×64 images. Out-of-Distribution Datasets: For comparison, we use the following OOD datasets: TinyImageNet-crop (TINc), TinyImageNet-resize (TINr), LSUN-crop (LSUNc), LSUN-resize (LSUNr), iSUN, SVHN. Evaluation Metrics: We compare the performance of various approaches using TNR@TPR95, AUROC and Detection Error. See Suppl. for description on evaluation metrics.

$\mathcal{D}_{\text{in}}^{\text{train}}$	**Method**	**TNR@TPR95** ↑	**AUROC** ↑	**DetErr** ↓	**ID Acc.** ↑
CIFAR-10 DenseNet-BC	MSP (ICLR'17) [18]	56.1	93.5	12.3	95.3
	ODIN (ICLR'18)[33]	92.4	98.4	5.8	95.3
	Maha(NeurIPS'18)[31]	83.9	93.5	10.2	95.3
	Gen-ODIN (CVPR'20)[22]	94.0	98.8	5.4	94.1
	Gram Matrices(ICML'20)[41]	96.4	99.3	3.6	95.3
	ATOM(ECML'21) [2]	98.3	99.2	1.2	94.5
	CnC(Proposed)	**98.4** ± **0.8**	**99.5** ± **1.2**	**2.7** ± **0.2**	94.7
CIFAR-100 DenseNet-BC	MSP (ICLR'17) [18]	21.7	75.2	31.4	77.8
	ODIN (ICLR'18)[33]	61.7	90.6	16.7	77.8
	Gen-ODIN (CVPR'20)[22]	86.5	97.4	8.0	74.6
	Maha (NeurIPS'18)[31]	68.3	92.8	13.4	77.8
	Gram Matrices(ICML'20)[41]	88.8	97.3	7.3	77.8
	ATOM(ECML'21)[2]	67.7	93	5.6	75.9
	CnC(Proposed)	**97.1** ± **1.4**	**98.5** ± **0.4**	**4.6** ± **0.6**	76.8
TIN RN50	MSP (ICLR'17) [18]	53.15	85.3	22.1	57.0
	ODIN (ICLR'18)[33]	68.5	93.7	12.3	57.0
	CnC(Proposed)	**97.8** ± **0.8**	**99.6** ± **0.2**	**2.1** ± **0.2**	60.5
C-10 WRN	OE (ICLR'19) [19]	93.23	98.64	5.32	94.8
	EBO (NeurIPS'20)[35]	**96.7**	99.0	**3.83**	95.2
	CnC(Proposed)	96.2 ± 1.5	**99.02** ± **0.1**	4.5 ± 0.8	94.3
C-100 WRN	OE (ICLR'19) [19]	47.35	86.02	21.24	75.6
	EBO (NeurIPS'20)[35]	54.0	86.65	19.7	75.7
	CnC(Proposed)	**97.6** ± **0.9**	**99.5** ± **0.1**	**2.2** ± **0.3**	75.1

Table 1. Comparison of competing OOD detectors. TIN: TinyImageNet, and RN50: ResNet50, WRN : WideResNet-40-2 Values are averaged over all OOD benchmark datasets. We give individual dataset-wise results in the supplementary. Note that ATOM[2], and OE [19] require large image datasets like 80-Million Tiny Images [43] as representative of OOD samples. However, CnC synthesises its own OOD dataset using the ID training data. CnC models were trained using the same configuration as defined by OE [19] and EBO [35] paper, with the exception that CnC did not use any external auxiliary OOD dataset like [43] in training. CnC reasults are averaged on 3 evaluation runs.

Data Augmentation Methods	**TNR (95% TPR)** ↑	**AUROC** ↑	**Detection Err** ↓
Mixup (ICLR'18) [51]	60.6	90.9	15.5
CutOut (arXiV'17) [6]	80.8	94.8	10
CutMix (ICCV'19) [48]	83.2	92.7	8.6
GridMask (arXiV'20) [3]	50.3	79.1	23.6
SaliencyMix (ICLR'21) [44]	85.3	95.7	8.0
AugMix (ICLR'20) [20]	81.3	94.6	11.2
RandomErase (AAAI'20) [52]	41.9	68.1	24.2
Corruptions (ICLR'19) [17]	98.0	99.4	2.8
PuzzleMix (ICML'20) [26]	66.8	84.1	15.2
RandAugment (NeurIPS'20) [4]	89.5	97.9	4.7
Fmix (ICLR'21) [13]	73	90.3	12.6
Standard Gaussian Noise	71.5	93.2	11.7
CnC(Proposed)	**98.4 ± 0.8**	**99.5 ± 1.2**	**2.7 ± 0.2**

Table 2. Comparison with other synthetic data generation methods. We consider CIFAR10 as ID. The values are averaged over all OOD benchmarks. We have used DenseNet[23] as the architecture for all methods trained for $(K+1)$ class classification. Samples obtained through the listed data augmentation schemes were assumed to be of $(K+1)^{\text{th}}$ class. Observe that CnC has superior OOD detection performance. We report average and standard deviation of CnC trained models computed over 3 runs.

5 Experiments and Results

To show that our data augmentation is effective across different feature extractors, we train using both DenseNet-BC [23] and ResNet-34 [14]. DenseNet has 100 layers with growth rate of 12. WideResNet [49] models have the same training configuration as [35].

5.1 Comparison with State-of-the-art

OOD Detection Performance: Tab. 1 shows comparison of CnC with recent state-of-the-art. The numbers indicate averaged OOD detection performance on 6 datasets as mentioned in Sec. 4 (TinyImagenet, TinyImageNet-crop (TINc), TinyImageNet-resize (TINr), LSUN-crop (LSUNc), LSUN-resize (LSUNr), iSUN, SVHN) with more details included in the supplementary. We would like to emphasize that CnC does not need any validation OOD data for fine-tuning. But ODIN [33] and Mahalanobis [31] require OOD data for fine-tuning the hyper-parameters; the hyper-parameters for ODIN and Mahalanobis methods [33,31] are set by validating on 1K images randomly sampled from the test set $\mathcal{D}_{\text{in}}^{\text{test}}$. Tab. 1 clearly shows that CnC outperforms the existing methods.

Comparison with Other Data Generation Methods : Tab. 2 shows how CnC fairs against recent OOD data generation methods. In each case we train a $(K+1)$ way classier where first K classes correspond to ID and $(K+1)^{\text{th}}$ class comprised of OOD data generated by corresponding method. As seen from the table, CnC outperforms the recent data augmentation schemes.

5.2 Other Benefits of CnC

Method	TNR@0.95TPR	AUROC	DetErr
MSP (ICLR'17) [18]	24.4	80.1	26.5
ODIN (ICLR"18) [33]	46.0	88.6	18.9
Gen-ODIN (CVPR'20) [22]	45.0	88.7	18.8
Mahalanobis (NeurIPS'18) [31]	14.0	56.2	41.6
Gram Matrices (ICML'20) [41]	35.0	81.5	25.8
CnC (Proposed)	**60.0**	**91.6**	**15.7**

Table 3. Detecting domain shift using CnC. A model trained with CnC data on CIFAR-100 as the ID using DenseNet-BC [23] feature extractor can successfully detect the domain shift when observing ImageNet-R at the test time.

Detecting Domain Shift as OOD: We analyze if a model trained with CnC augmented data can detect non-semantic domain shift, i.e. images with the same label but different distribution. For the experiments we use a model trained using CIFAR-100 as ID, and ImageNet-O/ImageNet-R/Corrupted-ImageNet [21] as the OOD. While testing, we downsample the images from ImageNet-O, ImageNet-R and TinyImageNet-C to a size of 32×32. Tab. 3 shows results on ImageNet-R OOD dataset. We outperform the next best technique by 14% on TNR@0.95TPR, 2.9% in AUROC, 3.1% in detection error. See supplementary for results on ImageNet-O and Corrupted ImageNet.

Model Calibration Another benefit of training with CnC is model calibration on ID data as well. A classifier is said to be calibrated if the confidence probabilities matches the empirical frequency of correctness [12,15], hence a crucial to measure of trust in classification models. Tables in the supplementary show the calibration error for a model trained on CIFAR-10, and CIFAR-100 as the ID data, with CnC samples as the $(K+1)^{\text{th}}$ class. Note that the calibration error is measured only for the ID test samples. We compare the error for a similar model, trained using only ID train data, and calibrated using temperature scaling (TS) [12].

Time Efficiency For applications demanding real-time performance, it is crucial to have low latency in systems using DNN for inference. Supplementary reports the competative performance of our method.

5.3 Ablation Studies

Rationale for Design choice of K vs. (K+1) Classifier We empirically verify having a separate class helps in better optimization/learning during training a

Method	TNR@ 0.95TPR ↑	AUROC ↑	DetErr ↓	Mean Diversity ↑	Mean Entropy ↑
PBCC	93.7	98.6	6.2	2.30	0.33
Corruptions	95.5	97.4	3.5	2.68	0.38
CnC	**98.3**	**99.6**	**2.6**	**3.40**	**0.80**

Table 4. Using entropy/diversity of synthesized data to predict quality of OOD detection. Please refer to text for more details.

model using CnC augmentation. Fig. 4 shows the advantages of using a $(K+1)$ way classifier as compared to standard K class training with better ID-OOD separation. Supplementary material details the advantage of CnC with ACET [16] (CVPR'19) for uncertainty quantification on a half-moon dataset.

Recommendation for a Good OOD detector We performed detailed comparison of various configurations of our technique to understand the quantitative scores which can predict the quality of an OOD detector. For the experiment we keep the input images used same across configs, PBCC and corruptions applied are also fixed to remove any kind of randomness. We use ResNet34 as feature extractor for all methods. CIFAR-10 is used as ID dataset and TinyImageNet-crop as OOD dataset. We observe that the quality of OOD detection improves as the diversity, and entropy of the synthesized data increases (Tab 4). Here, entropy is computed as the average entropy of the predicted probability vectors by the K class model for the synthesized data. We adapt data diversity from Zhang et al.[50] to measure diversity of OOD data. Refer supplementary for Algorithm for diversity computation.

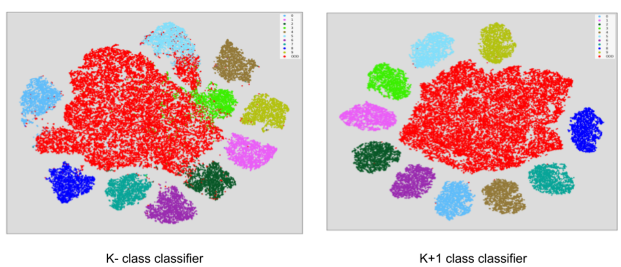


Fig. 4. We show sample t-SNE plots for K Vs. $(K+1)$ classifiers, where CIFAR-10 is used as ID and SVHN is used as OOD (marked in red). The K-class classifier uses temperature scaling (TS) [12], where T is tuned on SVHN test set. On the other hand, the $(K+1)$ class classifier uses SVHN data for $(K+1)^{\text{th}}$ class during training. The visualization shows that the OOD data (marked in red) is better separated in a $(K+1)$-class classifier as compared to a K-class classifier

Limitations of CnC data augmentation : Introduction of additional synthetic data indeed increases training time. For eg., training a model with CnC data on TinyImageNet dataset takes 10 mins. 23 secs./epoch, whereas without CnC data it takes 5 mins 30 secs./epoch on the same Nvidia V100 GPU. Performance gain the overhead of training time can be discounted as inference time remains same. We assume the absence of adversarial intentions in this approach, Our method fails when tested against L_∞ norm bounded perturbed image. In future we intend to look at `OOD` detection using CnC variants for non-visual domains.

6 Conclusions

We have introduced **C**ompou**n**ded **C**orruptions(CnC), a novel data augmentation technique for `OOD` detection in image classifiers. CnC outperforms all the `SOTA` `OOD` detectors on standard benchmark datasets tested upon. The major benefit of CnC over `SOTA` is absence of `OOD` exposure requirement for training or validation. We also show additional results for robustness to distributional drift, and calibration for CnC trained models. CnC requires just one inference pass at the test time, and thus has much faster inference time compared to `SOTA`. Finally, we also recommend high diversity and entropy of the synthesized data as good measures to predict quality of `OOD` detection using it.

7 Acknowledgements

Thanks to Subhashis Banerjee, Rahul Narain, Lokender Tiwari, Sandeep Subramanian for insightful comments.

References

1. Bendale, A., Boult, T.E.: Towards open set deep networks. In: Proceedings of the IEEE CVPR. pp. 1563–1572 (2016)
2. Chen, J., Li, Y., Wu, X., Liang, Y., Jha, S.: Atom: Robustifying out-of-distribution detection using outlier mining. In Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) (2021)
3. Chen, P., Liu, S., Zhao, H., Jia, J.: Gridmask data augmentation (2020)
4. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.: Randaugment: Practical automated data augmentation with a reduced search space. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in NeurIPS (2020)
5. Dabouei, A., Soleymani, S., Taherkhani, F., Nasrabadi, N.M.: Supermix: Supervising the mixing data augmentation. In: Proceedings of the IEEE/CVF CVPR. pp. 13794–13803 (2021)
6. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout (2017)
7. Du, X., Wang, Z., Cai, M., Li, Y.: VOS: learning what you don't know by virtual outlier synthesis. ICLR (2022)

8. Fang, Z., Lu, J., Liu, A., Liu, F., Zhang, G.: Learning bounds for open-set learning. In: International Conference on Machine Learning. pp. 3122–3132. PMLR (2021)
9. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: ICML (2016)
10. Ge, Z., Demyanov, S., Chen, Z., Garnavi, R.: Generative openmax for multi-class open set classification. In: BMVC (2017)
11. Graves, A.: Practical variational inference for neural networks. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 24, pp. 2348–2356 (2011)
12. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.Q.: On calibration of modern neural networks. In: ICML. pp. 1321–1330 (2017)
13. Harris, E., Marcu, A., Painter, M., Niranjan, M., Prügel-Bennett, A., Hare, J.: Fmix: Enhancing mixed sample data augmentation (2021)
14. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks (2016)
15. Hebbalaguppe, R., Prakash, J., Madan, N., Arora, C.: A stitch in time saves nine: A train-time regularizing loss for improved neural network calibration. In: Proceedings of the IEEE/CVF CVPR. pp. 16081–16090 (2022)
16. Hein, M., Andriushchenko, M., Bitterwolf, J.: Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In: Proceedings of the IEEE/CVF CVPR. pp. 41–50 (2019)
17. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: ICLR (2018)
18. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. ICLR (2017)
19. Hendrycks, D., Mazeika, M., Dietterich, T.: Deep anomaly detection with outlier exposure. International Conference on Learning Representations (ICLR) (2019)
20. Hendrycks*, D., Mu*, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: Augmix: A simple method to improve robustness and uncertainty under data shift. In: International Conference on Learning Representations (2020)
21. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: Proceedings of the IEEE/CVF CVPR. pp. 15262–15271 (2021)
22. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In: Proceedings of the IEEE/CVF CVPR. pp. 10951–10960 (2020)
23. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
24. Ji, X., Pascanu, R., Hjelm, R.D., Vedaldi, A., Lakshminarayanan, B., Bengio, Y.: Predicting unreliable predictions by shattering a neural network. CoRR **abs/2106.08365** (2021), https://arxiv.org/abs/2106.08365
25. Jordan, M., Lewis, J., Dimakis, A.G.: Provable certificates for adversarial examples: Fitting a ball in the union of polytopes. 33rd Conference on Neural Information Processing Systems (NeurIPS) (2019)
26. Kim, J.H., Choo, W., Song, H.O.: Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In: ICML (2020)
27. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
28. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: Advances in NeurIPS. pp. 6402–6413 (2017)

29. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N p. 3 (2015)
30. Lee, K., Lee, H., Lee, K., Shin, J.: Training confidence-calibrated classifiers for detecting out-of-distribution samples. arXiv preprint arXiv:1711.09325 (2017)
31. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. Advances in NeurIPS **31** (2018)
32. Li, D., Chen, D., Goh, J., Ng, S.k.: Anomaly detection with generative adversarial networks for multivariate time series. ACM KDD
33. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks. In: ICLR (2018)
34. Lin, Z., Roy, S.D., Li, Y.: Mood: Multi-level out-of-distribution detection. In: Proceedings of the IEEE/CVF CVPR. pp. 15313–15323 (2021)
35. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based out-of-distribution detection. Advances in Neural Information Processing Systems (NeurIPS) (2020)
36. Mohseni, S., Pitale, M., Yadawa, J., Wang, Z.: Self-supervised learning for generalizable out-of-distribution detection. AAAI pp. 5216–5223 (Apr 2020)
37. Neal, L., Olson, M., Fern, X., Wong, W.K., Li, F.: Open set learning with counterfactual images. In: ECCV. pp. 613–628 (2018)
38. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NeurIPS'21 (2011)
39. Perera, P., Nallapati, R., Xiang, B.: Ocgan: One-class novelty detection using gans with constrained latent representations. In: IEEE/CVF CVPR (2019)
40. Ramírez Rivera, A., Khan, A., Bekkouch, I.E.I., Sheikh, T.S.: Anomaly detection based on zero-shot outlier synthesis and hierarchical feature distillation. IEEE Transactions on Neural Networks and Learning Systems **33**(1), 281–291 (2022)
41. Sastry, C.S., Oore, S.: Detecting out-of-distribution examples with gram matrices. In: International Conference on Machine Learning. pp. 8491–8501. PMLR (2020)
42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
43. Torralba, A., Fergus, R., Freeman, W.T.: 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Trans. on PAMI (2008)
44. Uddin, S., Monira, M.S., Shin, W., Chung, T., Bae, S.H.: Saliencymix: A saliency guided data augmentation strategy for better regularization. In: ICLR (2021)
45. Wang, W., Wang, A., Tamar, A., Chen, X., Abbeel, P.: Safer classification by synthesis. arXiv preprint arXiv:1711.08534 (2017)
46. Wen, Y., Vicol, P., Ba, J., Tran, D., Grosse, R.: Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In: ICLR (2018)
47. Xiao, Z., Yan, Q., Amit, Y.: Likelihood regret: An out-of-distribution detection score for variational auto-encoder. Advances in Neural Information Processing Systems (2020)
48. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF ICCV. pp. 6023–6032 (2019)
49. Zagoruyko, S., Komodakis, N.: Wide residual networks. BMVC (2016)
50. Zhang, C., Öztireli, C., Mandt, S., Salvi, G.: Active mini-batch sampling using repulsive point processes. In: AAAI. vol. 33, pp. 5741–5748 (2019)
51. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: ICLR (2018)
52. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. In: AAAI. vol. 34, pp. 13001–13008 (2020)