Out-of-Distribution Detection for Automotive Perception

Julia Nitsch[1,2], Masha Itkina[3], Ransalu Senanayake[3], Juan Nieto[2], Max Schmidt[1], Roland Siegwart[2], Mykel J. Kochenderfer[3], and Cesar Cadena[2]

***Abstract*— Neural networks (NNs) are widely used for object recognition tasks in autonomous driving. However, NNs can fail on input data not well represented by the training dataset, known as out-of-distribution (OOD) data. A mechanism to detect OOD samples is important in safety-critical applications, such as automotive perception, in order to trigger a safe fallback mode. NNs often rely on softmax normalization for confidence estimation, which can lead to high confidences being assigned to OOD samples, thus hindering the detection of failures. This paper presents a simple but effective method for determining whether inputs are OOD. We propose an OOD detection approach that combines auxiliary training techniques with post hoc statistics. Unlike other approaches, our proposed method does not require OOD data during training, and it does not increase the computational cost during inference. The latter property is especially important in automotive applications with limited computational resources and real-time constraints. Our proposed method outperforms state-of-the-art methods on real world automotive datasets.**

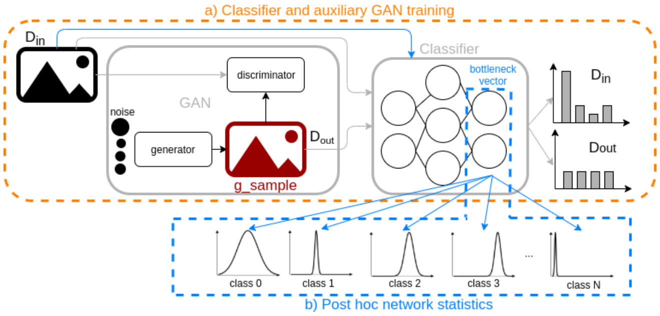


Fig. 1: a) depicts the processing of D_{in} and the generation of D_{out} on the decision boundary of the classifier. b) depicts the post hoc computation of class-conditioned Gaussian distributions on D_{in}. During inference, a threshold classifies D_{out} based on the distance to these distributions.

I. Introduction

Autonomous cars must rely on a robust perception system to ensure safe driving maneuvers in various driving scenarios. Neural networks (NNs) have resulted in perception systems that achieve state-of-the-art object recognition performance [1]–[4]. However, NNs tend to fail on out-of-distribution (OOD) data. The ability to accurately detect these failures on OOD samples is especially helpful in safety-critical applications like automotive perception, where we can use that information to trigger a safe fallback mode. NNs typically rely on softmax normalization for uncertainty computation [5], which can naively be used as an indicator of OOD data. Although softmax tends to perform well on data that follows the training distribution (D_{in}), it assigns overconfident values to OOD samples (D_{out}) due to the fast-growing exponential function [6]. For example, if very specific weather conditions or corner cases are not present in the training dataset, the NN could classify them incorrectly with high confidence [6]–[10].

These incorrectly assigned high confidences from the softmax function are misleading and indistinguishable from accurate classifications for the overall perception system. Thus, these confidences pose multiple risks such as decreasing overall perception system accuracy, hiding failures, or lacking system reliability and forcing autonomous systems to take more conservative actions. In particular, for safety-critical systems, it is important to detect these perception errors in order to take appropriate measures to ensure safe execution [5]. Safety-critical systems often include redundant perception systems that are either realized through multiple processing paths using different algorithms or through a multi-modal sensor setup [11]–[14]. If OOD samples are detected, then a monitoring system can be triggered to switch to a fallback solution of the redundant system.

The literature discusses various approaches for identifying D_{out}. One category of approaches is based on Bayesian NN inference, which includes dropout-based variational inference, Markov Chain Monte Carlo (MCMC), and Monte Carlo dropout [15]–[17]. Monte Carlo methods require multiple forward passes of the network, which is often infeasible in automotive applications due to limited computational resources and real-time constraints. Another category of approaches is dedicated OOD detection methods, which includes (1) training techniques such as auxiliary losses or modifications to the NN architecture [6], [18], [19] and (2) post hoc statistics [20], [21]. These OOD detection techniques often rely on D_{out} during development. However, for autonomous driving, it is challenging to collect an appropriate D_{out} dataset due to the difficulty of predicting the space of possible weather and interactive driving scenario combinations. Thus, we follow the research direction that proposes uncertainty-based OOD detection without a pre-recorded D_{out} dataset [16], [22]–[24].

This paper presents an uncertainty-based OOD detection approach that combines auxiliary training techniques with post hoc statistics. Our approach does not require D_{out}

[1] Ibeo Automotive Systems GmbH, {julia.nitsch, max.schmidt}@ibeo-as.com
[2] Autonomous Systems Lab, ETH Zurich, {jnieto, rsiegwart, cesarc}@ethz.ch
[3] Stanford Intelligent Systems Laboratory, Stanford University, {mitkina, ransalu, mykel}@stanford.edu

during training and comes without additional computational costs. The latter attribute, in particular, is important for automotive applications that run on limited resources with real-time constraints. During training, our proposed method uses an auxiliary Generative Adversarial Network (GAN) to produce D_{out} to encourage the object classifier to assign low confidences to samples on and outside the decision boundary. Post hoc, the parameters of class-conditioned Gaussian distributions over the bottleneck layer of the NN are computed from the training data. A sample is classified as D_{out} based on its distance to the previously computed Gaussian distributions during inference. The overview of the approach is illustrated in Fig. 1. Our proposed method is generic, but we focus on an object classification task using real-world automotive datasets, specifically KITTI [25] and nuScenes [26].

The contributions of this paper are as follows. We present an OOD detection approach without a prerequisite for an externally collected D_{out} dataset and no additional computation costs during inference. Our OOD detection method is based on the combination of an auxiliary training technique with the computation of post hoc statistics. We empirically validate our OOD detection method on real world data, demonstrating superior performance to baseline techniques.

II. Related Work

This section discusses related work on both implicit and dedicated OOD detection techniques.

A. Implicit OOD Detection Techniques

Bayesian Neural Networks (BNNs) can identify D_{out} by design unlike Maximum Likelihood Estimation (MLE) approaches [15], [27]. BNNs model epistemic uncertainty by learning a parameterized distribution over the neural network weights. NN ensembles also approximate the distribution over paramters, thus, generate epistemic uncertainties Jospin *et al.* [27] and Wehenkel *et al.* [28] which can be used for uncertainty-based OOD detection. Although BNNs and NN ensembles are promising research directions, current state-of-the-art automotive perception usually relies on deep NNs with MLE [1]–[4] due to lower computational costs. We are following this line of research in this paper.

Monte Carlo (MC) Dropout [29] has gained popularity for uncertainty-based OOD detection [16], [17]. MC Dropout is a simple, effective method for epistemic uncertainty estimation that does not require an explicit D_{out} dataset to be available during training. However, it has been shown to cause a performance drop in accuracy on real world data for semantic segmentation tasks [30]. Furthermore, MC dropout requires multiple forward passes during inference, which is unsuitable for applications that run on limited resources and real-time constraints.

Normalizing flows have also been used for threshold-based OOD detection using the feature space in semantic segmentation tasks [31]. However, the memory footprint of normalizing flows is unsuitable for larger images and, thus, Wellhausen *et al.* [24] apply normalizing flows only on the bottleneck vector of a NN. We follow the latter approach in modelling distributions on the bottleneck vector but instead of normalizing flows we fit class-conditioned Gaussian distributions [20].

B. Dedicated OOD Detection Techniques

Dedicated OOD detection approaches consider auxiliary losses during training, architecture design, or post hoc network statistics. An OOD detector uses the output of a classification module and the difference between the real input to the reconstructed input from an auxiliary decoder to distinguish D_{in} and D_{out} [6]. DeVries *et al.* [19] propose a threshold-based OOD detection method using an additional output neuron of a NN dedicated to D_{out} identification. Hendrycks *et al.* [18] train a network to assign uniform output distribution to D_{out} by adding an auxiliary loss function to the classification loss. Post hoc approaches compute network statistics by fitting, for example, Gaussian distributions to network weights of an already trained network [20]. Then, Mahalanobis distances from D_{in} and D_{out} may be used by an OOD detection network to differentiate samples.

Combining auxiliary training methods from Hendrycks *et al.* [18] with pot hoc statistics from Lee *et al.* [20] has been proposed by Papadopoulos *et al.* [21]. These approaches require a carefully designed D_{out} dataset, with the assumption that its distribution can also be expected during inference. However, for autonomous driving applications, it is infeasible to design and record such an accurate and extensive D_{out} dataset. To avoid the design of D_{out}, Lee *et al.* [23] propose an auxiliary GAN to generate D_{out} during training and an additional loss function to assign uniform distribution to D_{out}. Sensoy *et al.* [22] use variational autoencoders (VAEs) to generate D_{out} during training. We also use a generative NN to produce D_{out} samples during training. Following Papadopoulos *et al.* [21], we combine this auxiliary training procedure with post hoc statistics. With this combination, we circumvent the need for D_{out} dataset collection and further improve the performance of Lee *et al.* [23].

III. Method

We propose an uncertainty-based OOD detection approach that combines auxiliary training techniques with post hoc statistics. We demonstrate our method on an object classification task. An overview of the proposed method is shown in Fig. 1. Similar to Lee *et al.* [23], we use an auxiliary GAN to produce D_{out} during training to encourage the object classifier to assign low confidence to samples on and outside the decision boundary. We propose a different architecture, however, for the discriminator (see Fig. 2) and generator (see Fig. 3) due to stability issues during training on automotive datasets. The main differences between our discriminator architecture and that of Lee *et al.* [23] are as follows. We use a smaller number of convolutional layers with larger 5×5 kernels. We also replace batch normalization with dropout regularization. Within the generator, we insert a fully connected layer and again use kernels of size 5×5. Like Lee *et al.* [23], we use the VGG-13 architecture for the

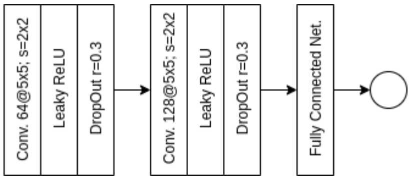


Fig. 2: The discriminator consists of two convolutional layers with 5×5 kernels, strides of 2×2, and leaky ReLU activations followed by a dropout layer with a dropout rate of 0.3. The first layer produces 64 feature maps and the second layer 128 feature maps. The convolutions are followed by a fully connected layer with a single output.

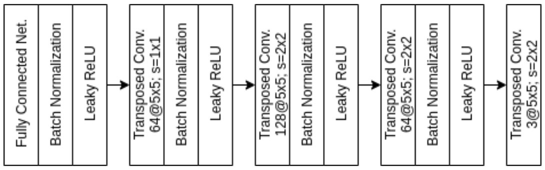


Fig. 3: The first block of the generator consists of a fully connected layer with batch normalization and a leaky ReLU activation. Then, the network contains three transposed convolution blocks with 5×5 kernels, batch normalization, and leaky ReLU activations. The last block consists of a transposed convolution with 5×5 kernel only.

classification network [32]. We extend the approach of Lee *et al.* [23] by proposing an additional post hoc component to the OOD detection method. We compute the parameters of class-conditioned Gaussian distributions over the weights of the bottleneck layer of the network based on the training data. Class-conditioned Gaussian distributions are a reasonable choice for modelling distributions on pre-trained features according to the analysis by Lee *et al.* [20]. During inference, a simple distance computation to the Gaussian distributions facilitates the identification of D_{out} through an empirically determined threshold.

A. Training

The GAN and the object classification network are jointly trained with sequentially updated losses (see Algorithm 1). We use a 100-dimensional standard Gaussian noise vector as input to the generator network in order to produce D_{out}. The discriminator loss is computed to distinguish samples from D_{in} and D_{out}. The discriminator loss is thus defined as the sum of the binary cross entropy (BCE) losses from D_{in} and from D_{out}:

$$\begin{aligned} Loss_d =& \text{BCE}\left(\text{disc}(D_{in}), \mathbf{1}\right) + \\ & \text{BCE}\left(\text{disc}(\text{gen}(noise)), \mathbf{0}\right) \end{aligned} \tag{1}$$

The generator is not only trained to fool the discriminator, but also to encourage the object classifier to assign low confidences to samples on and outside the decision boundary. This is achieved through the additive Kullback-Leibler (KL) divergence term between the classifier output and the uniform distribution. The generator loss is:

$$\begin{aligned} Loss_g =& \text{BCE}\left(\text{disc}(\text{gen}(noise)), \mathbf{1}\right) + \\ & \text{KL}\left[\text{cls}\left(\text{gen}(noise)\right) \middle\| \frac{\mathbf{1}}{c}\right] \end{aligned} \tag{2}$$

where c is the number of classes, $\frac{\mathbf{1}}{c}$ is the uniform distribution over classes, and cls is the output of the classification network with softmax normalization. Similarly, the classifier weights are updated to classify D_{in} correctly using the cross entropy loss (CE) and to assign uniform distribution to D_{out} with the same KL term as in $Loss_g$ (see Eq. (2)). The classifier loss is defined as:

$$\begin{aligned} Loss_{cls} =& \text{CE}\left(\text{cls}(D_{in}), labels\right) + \\ & \text{KL}\left[\text{cls}\left(\text{gen}(noise)\right) \middle\| \frac{\mathbf{1}}{\mathbf{c}}\right] \end{aligned} \tag{3}$$

Algorithm 1 Joint Loss Training

```
for epoch in epochs do
    $batches\_din \leftarrow split\ D_{in}\ in\ batches\ of\ batch\_size$
    for (batch_data, batch_label) in batches_din do
        $noise \leftarrow$ CREATE_NOISE_VECTOR ()
        $generated\_samples \leftarrow$ GENERATOR $(noise)$
        # 1) update discriminator
        $logits_{D_{in}} \leftarrow$ DISCRIMINATOR $(batch\_data)$
        $logits_{gen} \leftarrow$ DISCRIMINATOR $(generated\_samples)$

        $loss_d \leftarrow$ BCE $(logits_{D_{in}}, 1)$ + BCE $(logits_{gen}, 0)$
        $optimizer_{disc}.update($DISCRIMINATOR, $loss_d)$
        # 2) update generator
        $logits_{gen} \leftarrow$ DISCRIMINATOR $(generated\_samples)$

        $logits_{cls\_gen} \leftarrow$ CLS $(generated\_samples)$
        $loss_g \leftarrow$ BCE $(logits_{gen}, 1)$ + KL $\left(logits_{cls\_gen}, \frac{1.0}{|amount\_classes|}\right)$
        $optimizer_{gen}.update($GENERATOR, $loss_g)$
        # 3) update classifier
        $logits_{cls\_D_{in}} \leftarrow$ CLS $(batch\_data)$
        $loss_{cls} \leftarrow$ CE $(logits_{cls\_D_{in}}, batch\_label)$ + KL $\left(logits_{cls\_gen}, \frac{1.0}{|amount\_classes|}\right)$
    end for
end for
```

The sequential learning procedure is demonstrated in detail in Algorithm 1. All weights are initialized with the Xavier initialization [33], biases are initialized with zero, and the network is trained with the Adam optimizer [34]. The initial learning rate of 2×10^{-4} is exponentially decreased with a decay rate of 0.5 and a step rate of 30000. The network is trained for 100 epochs with a batch size of 128.

B. Post Hoc Network Statistics

Post hoc, we compute the parameters, $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$, of class-conditioned Gaussian distributions over the logits of the classification network. During inference, we compute the distance to the Gaussian distribution deemed most likely by the softmax distribution in order to identify OOD samples. We first consider the cosine similarity measure, which is a known metric for comparing high dimensional feature vectors and thus, an effective means for OOD detection:

$$CosSim\left(\mathbf{x}, \boldsymbol{\mu}_c\right) = \frac{\mathbf{x} \cdot \boldsymbol{\mu}_c}{|\mathbf{x}| \ |\boldsymbol{\mu}_c|} \tag{4}$$

where $\mathbf{x}$ is the current logit vector generated from a test input and c is the most likely class according to the softmax distribution.

Since the cosine similarity does not take into account the dispersion of the features, we also consider computing the Mahalanobis distance:

$$d_{M_c} = \sqrt{\left(\mathbf{x} - \boldsymbol{\mu}_c\right)^T \boldsymbol{\Sigma}_c^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_c\right)} \tag{5}$$

where $\boldsymbol{\Sigma}_c$ is the covariance matrix and c is the most likely class according to softmax. Since d_{M_c} is not bounded, which is required for the common interpretation of confidence measures, we compute the confidence based on the chi-square (χ^2) distribution. d_M^2 follows the χ^2 distribution where the degrees of freedom correspond to the feature dimensions. This probability is then used for outlier detection [35], [36].

IV. Experiments

We report one set of experiments where the classifier is trained using the standard cross entropy loss (CE loss) and another set of experiments where the classifier is trained using the sequential losses in Eq. (1) - Eq. (3) (joint loss). For both sets of experiments, the predicted class label is the most likely class according to the softmax layer. For the predicted class, the following confidence measures are computed to detect D_{out}: largest softmax probability mass amongst all classes (softmax), mutual information (MI) from MC dropout [29], cumulative density function (CDF) of the χ^2 distribution (ours), and cosine similarity (ours). Our proposed method with cosine similarity is tested on noised input as an ablation study. The joint loss training with softmax probabilities, inspired by [23], serves as a baseline to our proposed combination with post hoc statistics (χ^2 and CosSim). We compare our proposed method against MC dropout with MI using $T = 100$ inference steps as it is a popular method for OOD detection [16], [17].

A. Datasets

First, we report OOD detection results with the softmax normalization on the CIFAR10 dataset [37] as D_{in} and with Tiny ImageNet[1] and LSUN [38] as D_{out} to demonstrate the newly proposed GAN architecture is in line with the results of Lee *et al.* [23]. Next, we evaluate the proposed OOD detection method on two automotive datasets: KITTI [25] and nuScenes [26]. We extract *Car*, *Truck*, *Cyclist*, and *Pedestrian* object classes from the KITTI dataset represented as 192×256×3 patches following the train/test split as suggested by Nitsch *et al.* [39]. We also report OOD detection results on the nuScenes dataset [26]. Objects from the nuScenes dataset include ten different classes (*Barrier*, *Bicycle*, *Bus*, *Car*, *Construction Vehicle*, *Bike*, *Officer*, *Cone*, *Trailer*, *Truck*) of patch size of 128×128×3 and following the proposed train/validation split of Caesar *et al.* [26]. Within our experiments their validation set serves as test set, since the test labels have not been released yet. For the KITTI and nuScenes experiments, the ImageNet test dataset is chosen as D_{out}. D_{in} datasets are used for training whereas D_{out} are purely used for OOD detection evaluation.

B. Quantitative Experiments

We report the *Detection Accuracy* and the *Area Under Precision Recall (AUPR) curve* [20], [23]. The *Detection Accuracy* is defined as:

$$Detection\ Accuracy = 1 - e \tag{6}$$

The *Minimum Error Probability* e expresses the minimum OOD classification error over a discrete range of possible thresholds τ based on the computed distances $q(\mathbf{x})$ of D_{in} and D_{out} samples [23]:

$$\begin{aligned} e = 1 - \min_{\tau} \big\{ & P\left(q(\mathbf{x}_{in}) \leq \tau\right) P\left(\mathbf{x}_{in}\right) \\ & + P\left(q(\mathbf{x}_{out}) > \tau\right) P\left(\mathbf{x}_{out}\right) \big\} \end{aligned} \tag{7}$$

where $\mathbf{x}_{in}$ and $\mathbf{x}_{out}$ belongs to D_{in} and D_{out}, respectively, and P is the sample probability. Since the *Detection Accuracy* reflects the best possible classification result only, it should be considered in combination with the *AUPR-in* (D_{in} positive class) and *AUPR-out* (D_{out} positive class).

Table I and Table II demonstrate that the joint loss outperforms the CE loss in the experiments on CIFAR10 (D_{in}) and TinyImageNet (D_{out}), LSUN (D_{out}), as expected. These results are in line with the results of Lee *et al.* [23], which validates our implementation and confirms our choice of the GAN architecture.

Table III and Table IV show the results on the automotive datasets, KITTI (D_{in}) and nuScenes (D_{in}). The best result is highlighted in **bold** and the second best in blue. In all experiments, the cosine similarity is amongst the best results with a clear advantage on the *AUPR-in* metric. These experiments show that the cosine similarity is a simple yet effective measure for OOD detection. When considering the post hoc approaches to OOD detection on their own, without auxiliary training techniques, we found the cosine similarity to still be the superior choice for standard CE loss trained classifiers. In Table III, the cosine similarity achieves a $+5\%$ better *Detection Accuracy* than softmax as well as a significant difference in the *AUPR-in* metric. Only in *AUPR-out* it has a lower score compared to the softmax. Within Table IV, a similar detection accuracy is reported for the softmax and cosine similarity. However, in the *AUPR-in* measure, the cosine similarity clearly outperforms softmax with $+15\%$ whereas sacrificing only 3% on the *AUPR-out*. Furthermore, the cosine similarity outperforms the MI on the *Detection Accuracy* and *AUPR-in* metrics on both datasets. Within the experiment in Table III it performs similar to the MI on the *AUPR-out* metric. Moreover, our approach is real-time capable, unlike the MI, which requires multiple NN

[1] http://cs231n.stanford.edu/tiny-imagenet-200.zip

TABLE I: D_{in} CIFAR10 - D_{out} Tiny ImageNet

	Acc.%	Output	Detection acc. %	AUPR in	AUPR out
CE loss	80.72	softmax	64.53	60.16	69.35
joint loss	**81.18**	softmax	**73.34**	**76.74**	**77.19**

TABLE II: D_{in} CIFAR10 - D_{out} LSUN

	Acc.%	Output	Detection acc. %	AUPR in	AUPR out
CE loss	80.72	softmax	67.10	62.46	71.26
joint loss	**81.18**	softmax	**72.68**	**75.49**	**75.42**

forward passes for computation. χ^2-based detection achieves comparable results but is still outperformed by the cosine similarity measure. Although χ^2 has been used successfully for outlier identification [35], [36], our experiments show that this measure is not suitable for uncertainty-based OOD detection in object classification tasks. Further investigation into covariance estimation may improve OOD detection with the χ^2 based uncertainty.

We visualize the accumulated cosine similarity values in Fig. 4 and the accumulated softmax confidences in Fig. 5. This plot shows a normalized histogram of samples having larger values than the threshold τ. Due to the imbalanced number of test samples from nuScenes (D_{in}) and test samples from ImageNet (D_{out}), we normalize the histograms. For the cosine similarity, Fig. 4 demonstrates that D_{out} has lower distance values than those of D_{in} at higher τ, and, therefore, a τ can be found to distinguish between D_{in} and D_{out}. For the softmax-based uncertainties, this distinction is not as clear since many samples from D_{out} are assigned very high confidence, which is also reflected in the *AUPR-in* metric. The *Detection Accuracy* reflects the best τ choice in Eq. (7).

TABLE III: D_{in} KITTI - D_{out} ImageNet

	Acc.%	Output	Detection acc. %	AUPR in	AUPR out
CE loss	94.09	softmax	51.98	62.14	86.42
		MI	84.61	54.67	82.51
		χ^2 (ours)	86.49	76.12	65.20
		CosSim (ours)	86.15	81.34	80.08
joint loss	93.30	softmax	84.61	78.95	**94.67**
		MI	84.61	60.47	90.56
		χ^2 (ours)	88.16	76.40	60.71
		CosSim (ours)	**89.61**	**89.86**	90.25

TABLE IV: D_{in} nuScenes - D_{out} ImageNet

	Acc.%	Output	Detection acc. %	AUPR in	AUPR out
CE loss	88.56	softmax	83.85	64.52	81.67
		MI	83.76	55.11	78.87
		χ^2 (ours)	83.80	65.92	64.40
		CosSim (ours)	84.98	88.82	82.84
joint loss	88.85	softmax	**87.40**	74.97	**90.64**
		MI	83.79	56.88	82.12
		χ^2 (ours)	83.83	66.97	63.74
		CosSim (ours)	87.13	**91.51**	87.90

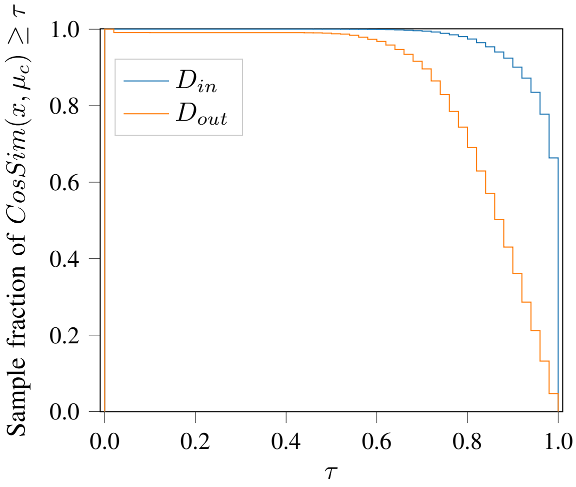


Fig. 4: Accumulated sample fraction of cosine similarity values $CosSim(\mathbf{x}, \boldsymbol{\mu}_c)$ on the nuScenes dataset of a classifier trained with joint loss.

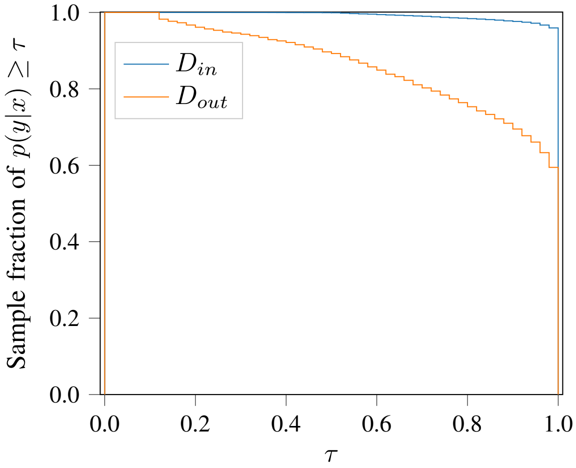


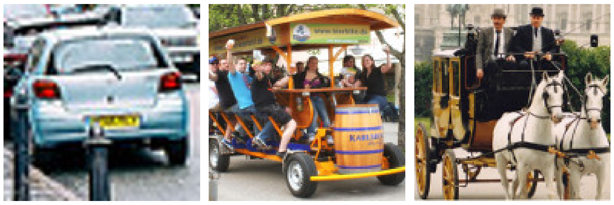
Fig. 5: Accumulated sample fraction of softmax uncertainties $p(\mathbf{y}|\mathbf{x})$ on the nuScenes dataset of a classifier trained with joint loss.

C. Ablation Study

For the ablation study, we consider noised versions of the KITTI test set as D_{out} and the original KITTI images as D_{in}. We investigate the effects of the level of salt and pepper noise in D_{out} on OOD detection performance. We report the performance of the proposed cosine similarity and the softmax uncertainty on a network trained with sequential losses Eq. (1)–Eq. (3) (joint loss) in Table V. We do not include the MI and χ^2 results in Table V for clarity since they underperform both cosine similarity and softmax uncertainty in these experiments. The joint loss network is sensitive to this noise pattern and immediately achieves worse classification accuracy than on the original test set. However, this novel noise pattern is also detected as OOD with our proposed cosine similarity metric. It significantly outperforms softmax on the *Detection Accuracy* and *AUPR-*

TABLE V: D_{in} KITTI - D_{out} noised KITTI

Noise level %	Acc.%	Output	Detection acc. %	AUPR in	AUPR out
0	93.30	softmax	n/a	n/a	n/a
		CosSim	n/a	n/a	n/a
1	29.01	softmax	76.60	69.90	**92.35**
		CosSim	**85.00**	**91.40**	90.55
5	25.50	softmax	79.10	72.00	**92.28**
		CosSim	**86.15**	**93.23**	88.21
35	24.20	softmax	81.05	74.42	**89.11**
		CosSim	**90.15**	**95.43**	85.44



(a) Baseline: D_{in} Ours: D_{in}

(b) Baseline: D_{out} Ours: D_{out}

(c) Baseline: D_{in} Ours: D_{out}

Fig. 6: The visualized objects are not present in the nuScenes dataset. Nevertheless, the car in Fig. 6a follows the distribution of nuScenes car objects and should be considered as D_{in}, whereas the other objects are true OOD samples (D_{out}). Shown are the uncertainty-based OOD detection results from the softmax (baseline) and the cosine similarity (ours) of a network trained with the joint loss on the nuScenes dataset. Our approach classifies all objects correctly, whereas the baseline fails on the horse-drawn carriage (see Fig. 6c).

in measures, while sacrificing only minor performance on the *AUPR-out* measure. This ablation study shows that the proposed combination of auxiliary training and post hoc statistics generalizes to novel noise patterns and is able to accurately classify noised data as OOD.

D. *Qualitative Experiments*

Qualitative OOD detection results are shown on a network trained with sequential losses Eq. (1)–Eq. (3) (joint loss) on the nuScenes dataset (D_{in}). Figure 6, shows examples of road users that are not present in the nuScenes dataset alongside OOD detection results from the softmax (baseline) and cosine similarity (ours) detectors. The threshold used to classify a sample as D_{in} or D_{out} is the optimal τ from the previously stated quantitative experiments. The car in Fig. 6a is from the ImageNet dataset (D_{out}) but is correctly recognized as D_{in}. Although it is not present in the nuScenes dataset (D_{in}), it visually belongs to the same distribution as the cars in the dataset. The example in Fig. 6b is a *beer bike* which can be seen in select German cities. This road user is not in the distribution of the nuScenes training data and both the softmax and the cosine similarity accurately classify it as D_{out}. Fig. 6c shows a *Fiaker* which is a special horse-drawn carriage unique to Vienna and thus, not present in the nuScenes dataset. The cosine similarity metric correctly detects it as D_{out} whereas the softmax metric incorrectly classifies it as D_{in}. Although the baseline network was trained with auxiliary training techniques, the softmax still assigns high probability to the horse-drawn carriage in Fig. 6c, classifying it as D_{in}. This example highlights that the addition of post hoc statistics adds value to auxiliary training methods in OOD detection.

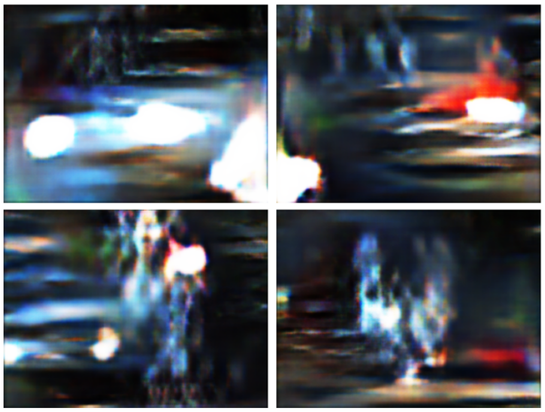


Fig. 7: Generated D_{out} images by the GAN trained on KITTI objects. The upper images show similarities to car objects whereas the lower images show a combination of pedestrians or cyclists with cars.

Fig. 7 shows examples of generated objects by the GAN network trained on KITTI data as D_{in}. The generated images often depict known road users like cars or pedestrians but are blurred or appear as combinations of several objects in the dataset. These image class combinations validate the effectiveness of the GAN to generate objects on the decision boundary of the classifier. Furthermore, light dots appear to be a common pattern in the generated images. We hypothesize these dots are learned from the recorded road users' car lights, which seem to become a very prominent feature for the GAN.

V. Conclusion and Future Work

In this paper, we introduced an approach that effectively combines auxiliary training techniques and post hoc statistics to perform OOD detection. We proposed using the simple yet effective cosine similarity metric as a measure during inference to identify OOD samples. Our method achieves the best *AUPR-in* performance on real-world datasets over the MC Dropout [29] and softmax [23] baseline approaches for OOD detection. We consider our work a step in the right direction for OOD detection in NNs. In particular, the combination of generative networks during training and the post hoc computed class-conditioned Gaussian distributions achieves promising performance. We highlight that our proposed approach does not require the definition and recording of an explicit D_{out} dataset and only requires a single inference step of the NN.

Another important finding is that networks trained with auxiliary GANs and evaluated with the cosine similarity during inference are sensitive to novel noise patterns which are not present in D_{in}, immediately assigning low confidence to these samples. We believe this could be an interesting direction for future work to investigate these networks for anomaly detection applications. Automated anomaly detection or simply detecting scenarios which are not yet present in D_{in} is an important use case for automotive perception, when thousands of hours of recordings for qualification and scenario coverage are required.

Acknowledgement

This work was funded in part by the Ford-Stanford Alliance and supported by the National Center of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation.

References

[1] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 918–927.

[2] P. Bhattacharyya and K. Czarnecki, "Deformable PV-RCNN: Improving 3D object detection with learned deformations," *arXiv preprint arXiv:2008.08766*, 2020.

[3] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3D object detection," *arXiv preprint arXiv:1908.09492*, 2019.

[4] X. Zhu, Y. Ma, T. Wang, Y. Xu, J. Shi, and D. Lin, "SSN: Shape signature networks for multi-class object detection from point clouds," *arXiv preprint arXiv:2004.02774*, 2020.

[5] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, *et al.*, "The limits and potentials of deep learning for robotics," *International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 405–420, 2018.

[6] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *International Conference on Learning Representations (ICLR)*, 2017.

[7] K. Nguyen and B. O'Connor, "Posterior calibration and exploratory analysis for natural language processing models," *Conference on Empirical Methods in Natural Language Processing*, 2015.

[8] D. Yu, J. Li, and L. Deng, "Calibration of confidence measures in speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2461–2473, 2011.

[9] J. Provost Foster, F. Tom, and K. Ron, "The case against accuracy estimation for comparing induction algorithms," in *International Conference on Machine Learning (ICML)*, 1998, pp. 445–453.

[10] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 427–436.

[11] J. H. Lala and R. E. Harper, "Architectural principles for safety-critical real-time applications," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 25–40, 1994.

[12] N. Gosala, A. Bühler, M. Prajapat, C. Ehmke, M. Gupta, R. Sivanesan, A. Gawel, M. Pfeiffer, M. Bürki, I. Sa, R. Dubé, and R. Siegwart, "Redundant perception and state estimation for reliable autonomous racing," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6561–6567.

[13] T. Peynot, J. Underwood, and S. Scheding, "Towards reliable perception for unmanned ground vehicles in challenging conditions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 1170–1176.

[14] J. Nitsch, J. Nieto, R. Siegwart, M. Schmidt, and C. Cadena, "Object classification based on unsupervised learned multi-modal features for overcoming sensor failures," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 4369–4375.

[15] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 13 991–14 002.

[16] A. Sedlmeier, T. Gabor, T. Phan, L. Belzner, and C. Linnhoff-Popien, "Uncertainty-based out-of-distribution detection in deep reinforcement learning," *International Symposium on Applied Artificial Intelligence (ISAAI)*, 2019.

[17] A. Loquercio, M. Segù, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters (RA-L)*, 2020.

[18] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," *International Conference on Learning Representations (ICLR)*, 2019.

[19] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," *arXiv preprint arXiv:1802.04865*, 2018.

[20] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 7167–7177.

[21] A.-A. Papadopoulos, M. R. Rajati, N. Shaikh, and J. Wang, "Outlier exposure with confidence control for out-of-distribution detection," *arXiv preprint arXiv:1906.03509*, 2019.

[22] M. Sensoy, L. Kaplan, F. Cerutti, and M. Saleki, "Uncertainty-aware deep classifiers using generative models," *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[23] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," *International Conference on Learning Representations (ICLR)*, 2018.

[24] L. Wellhausen, R. Ranftl, and M. Hutter, "Safe robot navigation via multi-modal anomaly detection," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1326–1333, 2020.

[25] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[26] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," *arXiv preprint arXiv:1903.11027*, 2019.

[27] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, "Hands-on Bayesian neural networks–a tutorial for deep learning users," *arXiv preprint arXiv:2007.06823*, 2020.

[28] A. Wehenkel and G. Louppe, "You say normalizing flows i see Bayesian networks," *arXiv preprint arXiv:2006.00866*, 2020.

[29] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on Machine Learning (ICML)*, 2016, pp. 1050–1059.

[30] H. Blum, P.-E. Sarlin, J. Nieto, R. Siegwart, and C. Cadena, "Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving," in *IEEE International Conference on Computer Vision Workshops*, 2019.

[31] N. Marchal, C. Moraldo, H. Blum, R. Siegwart, C. Cadena, and A. Gawel, "Learning densities in feature space for reliable segmentation of indoor scenes," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1032–1038, 2020.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations (ICLR)*, 2015.

[33] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010, pp. 249–256.

[34] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, Dec. 2014.

[35] P. J. Rousseeuw and B. C. Van Zomeren, "Unmasking multivariate outliers and leverage points," *Journal of the American Statistical Association*, vol. 85, no. 411, pp. 633–639, 1990.

[36] P. J. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[37] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[38] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015.

[39] J. Nitsch, J. Nieto, R. Siegwart, M. Schmidt, and C. Cadena, "Learning common and transferable feature representations for multi-modal data," *IEEE Intelligent Vehicles Symposium (IV)*, 2020.