Normalizing Flow based Feature Synthesis for Outlier-Aware Object Detection

Nishant Kumar[1], Siniša Šegvić[2], Abouzar Eslami[3], and Stefan Gumhold[1]

[1]TU Dresden, [2]University of Zagreb - FER, [3]Carl Zeiss Meditec AG

Abstract

Real-world deployment of reliable object detectors is crucial for applications such as autonomous driving. However, general-purpose object detectors like Faster R-CNN are prone to providing overconfident predictions for outlier objects. Recent outlier-aware object detection approaches estimate the density of instance-wide features with class-conditional Gaussians and train on synthesized outlier features from their low-likelihood regions. However, this strategy does not guarantee that the synthesized outlier features will have a low likelihood according to the other class-conditional Gaussians. We propose a novel outlier-aware object detection framework that distinguishes outliers from inlier objects by learning the joint data distribution of all inlier classes with an invertible normalizing flow. The appropriate sampling of the flow model ensures that the synthesized outliers have a lower likelihood than inliers of all object classes, thereby modeling a better decision boundary between inlier and outlier objects. Our approach significantly outperforms the state-of-the-art for outlier-aware object detection on both image and video datasets.

1. Introduction

General purpose object detectors such as Faster R-CNN [42] and Mask R-CNN [17] deliver high performance for inlier images. However, in many real-world scenarios, such as autonomous driving [3], plenty of unknown outliers (OD) naturally occur in an image or a video scene. Due to the co-existence of OD with the labeled inlier (ID) objects in the scene, object detectors confuse outliers with inliers. Therefore, reliable object-detection deployments require detecting such anomalies without degrading the performance of inlier object detection.

Many outlier detection approaches focus on multi-class image classification task by either performing outlier detection during inference [19, 24, 30, 31, 35, 44] or training on real outlier data [1, 20, 39, 53]. However, such OD inputs are unaware of the decision boundary between inliers and outliers, resulting in an inaccurate model regularization. A popular work [29] proposed a novel training scheme to
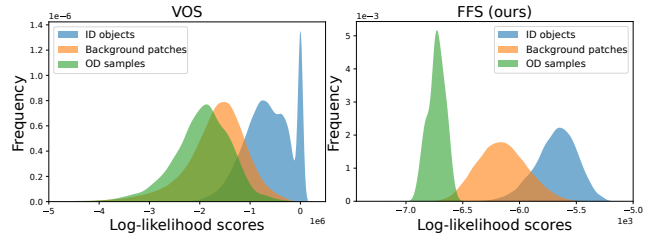

Figure 1. We compare distributions of log-likelihoods for inliers (ID), background patches, and synthetic outlier (OD) samples by applying a pre-trained model on Pascal-VOC [11] validation set. The left plot shows the likelihoods of class-conditional Gaussians from the official approach of VOS [10]. The right plot shows the likelihoods recovered with our normalizing flow. We observe that our flow separates the three distributions much better than VOS.

generate synthetic outlier samples in the high-dimensional pixel space using Generative Adversarial Networks for outlier aware training of an image classification model. However, GANs are challenging to optimize and are likely to deliver insufficient coverage [36]. Moreover, the previous work [29] generates an entire image as an outlier sample, whereas in an object detection problem, both inliers and outliers can coexist in the same image space.

Recently, [10] proposed to model the inlier features as class-conditional Gaussians and then synthesize OD features from the low likelihood region of the modeled distribution. Even though density is more easily estimated in the feature space than in pixel space, the assumption of class-conditional Gaussian may not provide an accurate decision boundary between inliers and outliers. Furthermore, synthesizing a low-likelihood OD sample from the Gaussian for class A does not guarantee that the sample is also of the low likelihood for the Gaussian of class B, as indicated in the left plot of Figure 1. Recent work for object detection in video [9] extracts the background patches for uncertainty regularization by thresholding the dissimilarity with the reference inlier features. However, the extracted background patches may lie far from the inlier-outlier decision boundary, leading to sub-optimal regularization of the model.

We propose a novel approach for open-set object detection which we call Flow Feature Synthesis (FFS). Our approach trains an invertible normalizing flow to map the data distribution of ID features of all object classes to a latent

representation that conforms to a multivariate Gaussian distribution with zero mean and diagonal unit covariance. As a result, it ensures principled estimation of the complex distribution of inlier features from all classes. We synthesize outlier features from low-likelihood regions of the learned distribution. This enables outlier features to be near the ID-OD decision boundary, leading to more robust uncertainty regularization of the object detector. In contrast, using generative adversarial models for the same task would be cumbersome due to its inability to infer density of the generated samples, Furthermore, variational autoencoders can only infer the lower bound of the sample density [27].

Specifically, FFS optimizes the normalizing flow model by maximum likelihood training on ID features. This training scheme enables the model to estimate the actual data distribution of the available ID features. Next, FFS utilizes the invertibility of the flow model to randomly sample from its latent space and generate synthetic features in the reverse direction of the model. The normalizing flow allows efficient and exact inference of the distribution density in the generated samples. Consequently, our approach can deliver suitable synthetic outlier data in fewer iterations than VOS [10]. It also requires fewer synthetically generated samples than VOS [10] to obtain OD features from the low-likelihood region of the modeled distribution by the flow. Furthermore, we developed our end-to-end trainable FFS framework to be effective on both image and video datasets. In contrast, previous works [9, 10] proposed standalone strategies for each task. Our main contributions are:

- We present a new outlier-aware object detection framework that utilizes Normalizing Flows to model the joint data distribution of inlier features. Invertibility of the flow allows efficient generation of synthetic outliers for effective uncertainty regularization.
- By mapping the data distribution of inlier features from all object classes to a multivariate Normal distribution in the flow's latent space, FFS ensures that an outlier sampled using the flow model is OD with reference to all ID classes.
- FFS achieves better OD detection performance while training faster than VOS [10] and STUD [9], due to having to generate fewer synthetic samples.
- We show that our method achieves state-of-the-art performance in OD object detection while preserving the baseline ID detection performance for image dataset PASCAL-VOC [11] and video datasets such as Youtube-VIS [51] and BDD100K [52].

2. Related work

The research in outlier-aware object detection is in the early stages compared to works in vanilla object detection [13, 17, 42] or outlier-aware semantic segmentation [14]. This section overviews prior approaches to outlier detection for image classification and proceeds to outlier-aware object detection for images and videos.

Outlier-aware image classification. Past approaches trained a classification model without knowledge about outliers and conducted OD detection during inference. For example, [19] used simple softmax probability scores during inference to detect OD samples, while other works used Mahalanobis distance [30], rectified activations [46], KL divergence [24] and Gram Matrices [44] instead of softmax scores to detect such samples. On the other hand, ODIN [31] and Generalized ODIN [23] performed perturbations to the test examples to enhance the performance of softmax function for OD detection. Recently, energy function [33, 35, 48] has performed better for distinguishing ID and OD samples than softmax scores.

Several other methods [20–22, 39, 47] proposed to regularize image classifiers by exposing them with outlier data during self-supervised training while a GAN based approach [29] synthesizes outliers in pixel space and uses it for outlier-aware image classification. However, such synthetic outliers are sampled from imprecise decision boundaries. Additionally, these approaches may not be applied to an object detection task since outliers defined by such methods are in the entire pixel space of the input image. In contrast, an image may contain both inlier and outlier objects simultaneously, and a trustworthy object detection model should detect such scenarios.

Outlier-aware object detection for images. One of the early open-set object detection works [5] highlighted the importance of adding an extra background class for unknown objects, though object detection inherently rejects unknown object proposals. Several works improve the background detection in open-set conditions [4, 15, 37, 38] by using Bayesian techniques, such as Monte Carlo-Dropout [12]. However, such methods have high latency during inference which could be sub-optimal for real-world applications. Others [16, 43] presented uncertainty regularization for localization regression but did not examine outlier-aware object detection with a focus on classification head. Works like [26] used the background patches as OD samples for model regularization, but the performance might be low as the samples may be far from the decision boundary. VOS [10] applies energy-based regularization [35] to synthetic OD features generated by class-conditional Gaussians. Instead, our method map features from all ID classes to a Gaussian prior with a normalizing flow model, and thus achieves a tighter decision boundary and improved performance.

Outlier-aware object detection for videos. Several works aim to identify anomalous events on the object and frame level. An object-level approach [8] uses the k-nearest neighbors algorithm to detect anomalous objects while [25] use k-means to cluster inlier features and train k binary clas-
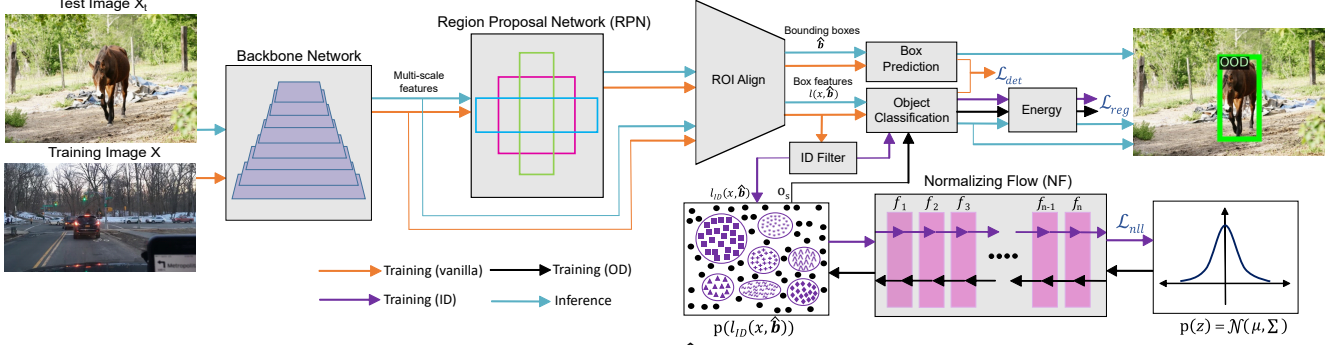
Figure 2. **Overview of the** FFS **framework.** The ID features $l_{ID}(x,\hat{b})$ are selected for training the normalizing flow with the negative log-likelihood loss $\mathcal{L}_{nll}$. We recover outlier features o_s by iterative sampling in the latent space of the normalizing flow. The classification logits of the ID features $l_{ID}(x,\hat{b})$ and the outlier features o_s are used to calculate the energy scores. The uncertainty loss $\mathcal{L}_{reg}$ encourages separation between the energy scores of inlier and outlier features. More details are given in Section 3.

sifiers so that each cluster can view other clusters as an outlier. A test object is labeled an outlier if the highest classification score is negative. Frame-level approaches [34, 41] predicted future frames that preserve the events of the previous frames by ensuring the optical flow of the predicted frame remains consistent with ground-truth. In contrast, events with a significant difference between prediction and ground-truth are labeled outliers. However, no previous work synthesizes outliers from video frames for model regularization. Recent work [49] introduced self-supervised learning to find the most distinct object in the next video frame given an inlier object in the previous frame, leading to more knowledge about unseen data domains. STUD [9] uses the inliers in the first frame to select the outliers in the following frames for model regularization. However, this may deliver outlier patches that are very far away from the decision boundary and thus lead to inferior model regularization. To summarize, no single outlier-aware object detection framework exists for images and video with an effective regularization that models a precise inlier manifold.

3. Method

Outlier-aware object detection problem is much more complex than detecting outliers for an image classification task since a real-world image may consist of both inlier and outlier objects. Let us define an input $x \in \mathcal{X}$ with $\mathcal{X} := (x_1, x_2, ..., x_N)$ being the training dataset consisting of N images or video frames, ground-truth class labels for each ID object as $y \in \mathcal{Y}$ with $\mathcal{Y} := \{1, 2, ..., K\}$ for K classes and the coordinates of ground-truth bounding boxes $\boldsymbol{b} \in \mathcal{B}$. Then, an end-to-end object detector with parameters θ treated as a conditional probability distribution estimates bounding boxes as $p_\theta(\boldsymbol{b}|x)$ and the object class as $p_\theta(y|x, \boldsymbol{b})$. Figure 2 shows the training and inference scheme of the FFS framework. The backbone network extracts the feature maps at different resolutions, and the Region Proposal Network (RPN) converts these

multi-resolution feature maps into object proposals. The ROI align module classifies object proposals and provides a vector of bounding boxes, $\hat{\boldsymbol{b}}$, and fixed-size box features, $l(x,\hat{\boldsymbol{b}})$, of both ID (inlier) and background patches. Note that $l(x,\hat{\boldsymbol{b}})$ is in a lower dimensional space than x. The box features $l(x,\hat{\boldsymbol{b}})$ and bounding boxes $\hat{\boldsymbol{b}}$ are used to train the object classification and box prediction heads according to the object detection loss $\mathcal{L}_{det}$, given the ground-truth ID classes and box coordinates, respectively. The box features from $l(x,\hat{\boldsymbol{b}})$ with the ground-truth class as background are filtered out to keep only ID features $l_{ID}(x,\hat{\boldsymbol{b}})$.

3.1. Overview

Our framework FFS is split into three stages: the training procedure, the threshold estimation for OD detection, and the model inference as summarized in Algorithm 1. During the training stage, we compute the standard object detection loss $\mathcal{L}_{det}$ via the chosen general-purpose object detector. We define maximum-likelihood training objective $\mathcal{L}_{nll}$ to train our normalizing flow module on inlier features $l_{ID}(x,\hat{\boldsymbol{b}})$. Subsequently, we sample synthetic outliers o_s from the flow model. The details related to the normalizing flow and the procedure to synthesize outlier features are given in Section 3.2. We then compute energy scores for inliers and outliers for the model regularization using $\mathcal{L}_{reg}$. Section 3.3 describes the details about model regularization, and Section 3.4 discusses the overall training objective of the FFS framework. We use the trained FFS model for threshold estimation to perform OD detection. We fix the energy-based threshold ξ such that the model correctly detects 95% of inlier objects in the validation set. During inference, given an object x_t in a test image, the trained FFS model provides a bounding box for the object as b_t. The decision of whether the object is an inlier or an outlier lies on the energy $E(h(l(x_t, b_t); \theta))$. Given an energy threshold ξ, we assign x_t as an inlier if $E(h(l(x_t, b_t); \theta)) < \xi$ and OD if $E(h(l(x_t, b_t); \theta)) \geq \xi$. We obtain object class labels for predicted inliers with the softmax confidence score

3

and bounding box. We label the bounding box for objects detected as outliers as OOD.

Algorithm 1 `FFS`: Normalizing Flow based Feature Synthesis for Outlier-Aware Object Detection

---

Input: Training data $\{(x_i, \mathbf{b}_i, y_i)\}_{i=1}^{N}$, vanilla object detector, normalizing flow f and binary classifier Φ with parameters θ, γ and ψ respectively, the loss weights α and β.
Output: Object detector with trained parameter θ.
while *train* **do**
    1. Input training data to `FFS` model and compute $\mathcal{L}_{det}$.
    2. Input the ID features $l_{ID}(x, \hat{\boldsymbol{b}})$ to the flow model f and obtain $\mathcal{L}_{nll}$ using Eq 2.
    if `i` $\leq$ *iterations* **then**
        3. Update i and the parameters θ and γ by backpropagation of the total loss in Eq 5 with $\alpha = 0$.
    else
        4. Generate synthetic features g_k by sampling z_k from flow's latent space with $g_k = f^{-1}(z_k)$.
        5. Select o_s from $p_\gamma(g_k)$'s low-likelihood region.
        6. Calculate energy scores $E(h(l_{ID}(x, \hat{\boldsymbol{b}}); \theta))$ and $E(h(o_s; \gamma))$ using the classification head h.
        7. Compute regularization loss $\mathcal{L}_{reg}$ using Eq 4.
        8. Update i and the parameters θ, γ and ψ by backpropagation of the total loss in Eq 5.
    end
end
Fix threshold ξ when 95% of inliers are correctly detected.
while *eval* **do**
    1. Calculate the energy score of a test object.
    2. Label it as an outlier or an inlier based on ξ.
end

3.2. Normalizing Flow for feature synthesis

For simplicity, we will denote the ID features $l_{ID}(x, \hat{\boldsymbol{b}})$ as $l \in \mathcal{L}$. In our framework, the normalizing flow f with parameters γ is a sequence of M invertible bijective mappings implemented as affine coupling layers. The flow $f : \mathcal{L} \to \mathcal{Z}$ transforms the complex data distribution of the features l to a multivariate Gaussian in its latent space $z \in \mathcal{Z}$ with $p(z) = \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where d is the number of feature vector components, and $\boldsymbol{\mu}$ is zero mean with $\boldsymbol{\Sigma}$ as the unit variance. The bijective mapping ensures that the input and the latent space share the same dimensionality d such that $f : \mathbb{R}^d \to \mathbb{R}^d$. Each coupling layer transforms the feature vector components by scaling s and translation t, which are learnable neural networks [27]. Such a transformation is expressive and, at the same time, easily invertible with high efficiency in computing Jacobian determinants.

3.2.1 Maximizing the likelihood of ID features

Given the above formulation of our normalizing flow f, which transforms input ID features l into z such that $z =$

$f(l)$, we aim to maximize the log-likelihood of recovering features l with respect to f. To achieve this, we have to compute the Jacobian matrix of $f(l)$ with respect to the feature vector components [27]. Let $f_i(l)$ be the components of f in latent space and l_j the components of feature space. The entries of the Jacobian matrix are defined as $J_{ij}^{f,l} = \frac{\partial f_i(l)}{\partial l_j}$ where $i, j \in 1, ..., d$. According to the change of variables formula, the posterior likelihood $p_\gamma(l)$ can be described as:

$$p_\gamma(l) = p(f(l)) * \left| \det J^{f,l} \right| \qquad (1)$$

To obtain the log-likelihood of the posteriors, we take a logarithm on both sides of the Eq 1. As the training of a neural network requires minimization of a loss function, we construct f by finding γ that maximizes $\log(p_\gamma(l))$ by minimizing the negative log-likelihood $\mathcal{L}_{nll}$ objective:

$$\mathcal{L}_{nll}(l; \gamma) = \frac{1}{N} \sum_{i=1}^{N} -\log(p_\gamma(l(x_i, b(x_i)))) \qquad (2)$$

Normalizing flows are known for stable and exact training according to the maximum likelihood objective in Eq 2.

3.2.2 Random sampling from the latent space

We train the flow model f so that its parameters γ learn the distribution of inlier features for a fixed number of training iterations. Next, we randomly sample k samples z_k from the latent distribution $p(z)$ and propagate it in the reverse direction of f to generate synthetic samples g_k where $g_k = f^{-1}(z_k)$ during active training. As the flow model learned the data distribution $p_\gamma(l)$ via the maximum likelihood training using Eq 2, the samples g_k are synthesized from $p_\gamma(l)$. Such generation procedure discourages mode collapse, common in generative adversarial models. Hence, we retrieve the exact data distribution of the features l in synthesized samples g_k. After that, we compute the log-likelihood scores of each of the k samples g_k using Eq 1. Reasonably, the synthesized g_k contains samples with a high likelihood of being obtained from $p_\gamma(l)$ and should lie well inside the boundary of the inlier distribution $p_\gamma(l)$. Nevertheless, there should also be samples in g_k with lower likelihood scores that lie near or away from the boundary of the distribution $p_\gamma(l)$.

3.2.3 Rejection Sampling based Outlier Synthesis

It is pivotal to sample useful synthetic outliers for an effective model regularization for OD detection. To achieve this, our first approach involves rejection sampling. We select s samples as outliers o_s from the low-likelihood region of the data distribution $p_\gamma(g_k)$, where $o_s \subset g_k$ and s being much smaller than k. The motivation of such an approach is to obtain synthetic outliers from near the decision boundary and outwards in the exterior outlier space. Figure 3 (a)-(b) shows that generating more synthetic samples

g_k and selecting a single outlier o_s with the least likelihood (where $s = 1$) improves the OD detection. This is because generating more samples g_k leads to the selection of a low-likelihood synthetic outlier o_s that genuinely lies in the outlier space, resulting in better model regularization.
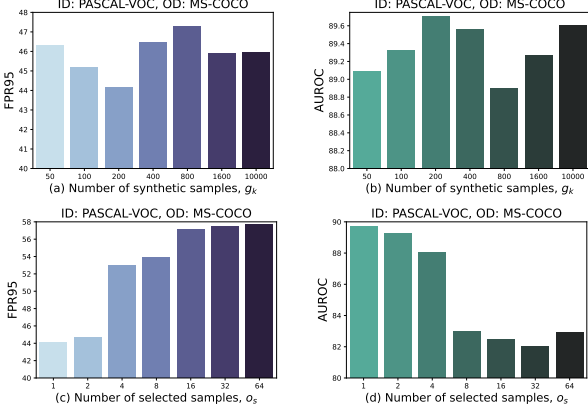

Figure 3. Validation of synthetic samples g_k and selected outliers o_s on OD detection. $s = 1$ in (a)-(b) and $k = 200$ in (c)-(d).

Eventually, however, the performance peaks as generating an even higher number of synthetic samples g_k may lead to the selection of an outlier o_s that is too far away in the outlier space, resulting in a model that is unaware of the decision boundary. Hence, by varying g_k, our rejection sampling approach can obtain outliers from the decision boundary and from the outlier region away from the boundary. Figure 3 (c)-(d) show the results when the number of selected outliers o_s is varied while keeping the number of the generated synthetic samples g_k fixed. The selection of more outliers o_s leads to a degradation in the OD detection performance as some selected outliers with high log-likelihood scores lie well within the inlier manifold leading to suboptimal model regularization.

3.2.4 Projection Sampling using Langevin Dynamics

We investigate whether our model can generalize the outlier space by using synthetic outliers sampled only close to the decision boundary while still being effective with OD detection. To accomplish this, we perform projection sampling based on Stochastic Gradient Langevin Dynamics (SGLD) to sample outliers from our flow model. Under the approach, we generate s samples as outliers o_s directly without needing rejection and require a likelihood threshold to define the boundary of the inlier feature distribution. We fix this threshold as the log-likelihood score δ of the inlier, which is least likely to be obtained from the inlier data distribution $p_\gamma(l)$. Next, we propagate the average log-likelihood score of the synthetic outliers $\log p_\gamma(o_s)$ in the inverse direction of the flow without updating its parameters and obtain the gradients $\frac{\partial \log p_\gamma(o_s)}{\partial o_s}$. Subsequently, the outliers o_s are updated as:

$$o_s := o_s - \tau \frac{\partial \log p_\gamma(o_s)}{\partial o_s} \qquad (3)$$

where τ is the step size of the gradient descent. We end this iterative process when the average log-likelihood of the updated o_s is equal to or lower than δ. Hence, the updated outliers o_s are precisely projected to the near decision boundary of the inlier manifold where the least-likelihood inlier is located. The results of our projection sampling approach are shown in Figure 5 (f) and discussed in Section 4.3.

3.3. Energy-based model regularization

We obtained the outliers o_s by sampling from the data distribution $p_\gamma(l)$, requiring no actual outliers during training. The classification head h now maps inlier features l and outliers o_s to $K + 1$ classification logits as $h(l; \theta)$ and $h(o_s; \gamma)$, respectively. Note that $h(l; \theta)$ is influenced by all θ parameters, whereas $h(o_s; \gamma)$ depends on the flow parameters γ and the parameters of the classification head. Recently, [10, 35] used energy function rather than softmax probabilities to differentiate inlier and outlier samples better. We follow [10] and apply the energy function on the classification logits $h(l; \theta)$ and $h(o_s; \gamma)$ using $E(h(.)) = -T \cdot \log \sum^K w_K \cdot e^{h_K(.)/T}$ where T is a temperature parameter of the energy function and weight w is learned to overcome class imbalances. We then pass the energy scores through a binary classifier Φ with parameters ψ and define the softmax output as p_l when $E(h(l; \theta))$ is the input and p_o when $E(h(o_s; \gamma))$ is the input. Finally, we use binary cross-entropy (BCE) based regularization loss $\mathcal{L}_{reg}$:

$$\mathcal{L}_{reg} = \frac{1}{N} \sum_{i=1}^{N} -(\log(p_l) + \log(1 - p_o)) \qquad (4)$$

The Eq 4 shapes contrasting inlier and outlier energy surfaces by assigning low energy for inliers and high energy for outliers.

3.4. Overall Training Objective

We describe the standard object detection loss $\mathcal{L}_{det}$ as a combination of losses for object classification given the ground-truth inlier class labels y and bounding box regression given the ground-truth box coordinates $\boldsymbol{b}$. Based on the formulation of the negative log-likelihood loss $\mathcal{L}_{nll}$ and the regularization loss $\mathcal{L}_{reg}$ from Eq 2 and 4 respectively, we merge all these loss terms in our overall training objective:

$$\min_{\theta, \gamma, \psi} \mathbb{E}_{(x, \mathbf{b}, y) \sim \mathcal{X}, \mathcal{B}, \mathcal{Y}} \ [\mathcal{L}_{det} + \beta \cdot \mathcal{L}_{nll} + \alpha \cdot \mathcal{L}_{reg}], \quad (5)$$

where α and β are the weights of $\mathcal{L}_{reg}$ and $\mathcal{L}_{nll}$ respectively. The $\mathcal{L}_{reg}$ loss acts as an adversary to $\mathcal{L}_{nll}$ loss as $\mathcal{L}_{nll}$ tries to maximize the likelihood of obtaining inlier features while $\mathcal{L}_{reg}$ forces the model to synthesize outliers away from inliers. Thus, such a training scheme encourages the generation of synthetic samples at the decision boundary resulting in outlier awareness of the object detector.

4. Experiments

This section evaluates our FFS framework for its effectiveness with outlier-aware object detection. Section 4.1 provides the details related to the image and video datasets, the neural network architecture, and the performance metrics. Section 4.2 shows the results obtained with FFS and compares it with other related approaches. Finally, Section 4.3 contains ablation studies to evaluate the performance of our approach in varied experimental settings.

4.1. Implementation details

Datasets. We train our FFS model on the publicly available image dataset PASCAL-VOC 2012 [11] containing 20 object categories as ID and use two OD datasets, namely MS-COCO [32] and OpenImages [28] to evaluate the performance. Additionally, we train FFS on two video datasets, namely Berkeley DeepDrive (BDD100K) [52] and Youtube-Video Instance Segmentation (Youtube-VIS) 2021 [51]. For the FFS trained on video datasets, we infer on two OD datasets, namely MS-COCO [32] and nuImages [2]. We use the pre-processed datasets by VOS [10] and STUD [9] so that the object categories in the OD dataset are independent of the object categories in the ID dataset.

Network architecture. We adopt the Faster R-CNN model [42] for object detection provided in the publicly available Detectron2 framework [50]. We use RegNetX-4.0GF [40] as the backbone architecture due to its superior inlier detection performance. We employ Glow [27] as our normalizing flow model after observing lower performance for other flow architectures (details in Section 4.3).

Metaparameters. We run our experiments using Python 3.8.6 and PyTorch 1.9.0 with each ID dataset trained on four NVIDIA A100-SXM4 GPUs. We start the training of the FFS framework by turning off uncertainty regularization loss $\mathcal{L}_{reg}$ for a fixed number of iterations. This enables the flow model to learn the data distribution of inlier objects from all object classes. Across datasets, we found the loss weightages $\beta = 10^{-4}$ and $\alpha = 0.1$ as reasonable values for stable end-to-end training of the FFS framework. In Section 4.3, we perform an exhaustive meta-parameter study.

Evaluation metrics. We compute the energy scores from K classification logits of the inlier and outlier objects in the validation set. We evaluate these scores according to the standard outlier detection metrics. These metrics are Area Under the Curve Receiver Operating Characteristic (AUROC ↑) and False Positive Rate at 95% True Positive Rate (FPR95 ↓), where True Positive is the correct detection of an inlier. For ID detection, we first compute Intersection over Union (IoU) between predicted bounding boxes and ground-truth boxes. Then, we compute the average precision (AP) for the standard range of IoU thresholds. Finally, the mean of the average precision over all known classes (mAP ↑) measures ID detection performance.

4.2. Comparison with the State-of-the-Art

Tables 1 and 2 compare our method with other approaches on image and video datasets, respectively. We use the results provided by VOS [10] and STUD [9] to show the performance of the compared approaches. Some evaluated approaches (except VOS and STUD) were developed for image-wide OD detection. However, these approaches can also be used to compare an outlier-aware object detection model since classification is one of the object detection tasks. No real outlier datasets were used during training to compare these approaches with our method.

ID	Method	FPR95 ↓	AUROC ↑	mAP ↑
		OD: MS-COCO / OpenImages		
PASCAL-VOC	GAN [29]	60.93 / 59.97	83.67 / 82.67	48.5
	Energy [35]	56.89 / 58.69	83.69 / 82.98	48.7
	CSI [47]	59.91 / 57.41	81.83 / 82.95	48.1
	VOS [10]	47.77 / 48.33	89.00 / 87.59	51.5
	FFS (ours)	**44.15 / 45.08**	**89.71 / 88.29**	**51.8**

Table 1. Main results on image datasets with Faster R-CNN model and RegNetX-4.0GF [40] as the backbone for VOS and FFS.

ID	OD	Method	FPR95 ↓	AUROC ↑	mAP ↑
BDD100K	nuImages	GAN [29]	83.65	70.39	31.5
		Energy [35]	81.62	69.43	32.0
		CSI [47]	80.00	74.91	31.8
		STUD [9]	79.75	76.55	32.3
		FFS (ours)	**76.68**	**77.53**	**36.2**
Youtube-VIS	MS-COCO	GAN [29]	85.75	72.95	25.5
		Energy [35]	88.54	67.83	26.7
		CSI [47]	82.43	71.81	24.2
		STUD [9]	81.14	74.82	27.2
		FFS (ours)	**83.06**	**76.37**	**27.6**

Table 2. Main results on video datasets with Faster R-CNN model and RegNetX-4.0GF [40] as the backbone network.

In Table 1, we report a 7.58% and a 6.73% decrease in FPR95 for FFS compared to VOS when evaluated on MS-COCO and OpenImages, respectively. Additionally, we note an increase in AUROC values when compared with VOS and assessed on MS-COCO and OpenImages. We also obtained higher ID detection performance and a faster training time of 2.18 hr compared to 2.43 hr for VOS on the same hardware and batch size. In Table 2, FFS outperforms STUD, where we report a 3.85% decrease in FPR95 and a 1.28% increase in AUROC compared to STUD when the ID is BDD100K, and the OD is nuImages. We also register a significant 12% increase in mAP performance compared to STUD. Our training time is 2.7 hrs compared to 11 hrs by STUD for the BDD100K dataset on the same hardware. FFS also improves STUD results for the Youtube-VIS as ID and MS-COCO as OD while maintaining high ID detection performance. For this dataset, we report a lower training

time of 8.2 hrs compared to 11.3 hrs by STUD.

ID	Method	FPR95 ↓	AUROC ↑	mAP ↑	Time (h)
PASCAL-VOC	VOS [10]	49.67	88.43	48.91	2.37
	FFS (ours)	**43.12**	**89.84**	**51.73**	**2.12**
Youtube-VIS	STUD [9]	83.93	74.54	24.42	11.03
	FFS (ours)	**81.90**	**76.35**	**25.10**	**8.23**

Table 3. Performance evaluation with ResNet-50 backbone.

In Table 3, we also show the performance while using ResNet-50 [18] as an alternative backbone and compare our results with VOS [10] and STUD [9] with MS-COCO as the outlier dataset. The results show that using ResNet-50 as the backbone achieves state-of-the-art ID and OD detection performance with our framework. Overall, Tables 1, 2 and 3 demonstrate the effectiveness of FFS for OD detection for images and videos with lower training time and high ID detection performance.

4.3. Ablation Studies

In this section, we perform an exhaustive study of meta-parameters that affects the performance of FFS. We also qualitatively compare OD detection results with the VOS approach. For uniformity, we fix the ID dataset as PASCAL-VOC and OD dataset as MS-COCO unless stated otherwise in the experiments.

VOS plus rejection sampling: Sampling outliers from a class conditional Gaussian (as performed in VOS [10]) does not ensure that such an outlier will have lower likelihoods for other class Gaussians. Therefore, we compute the log-likelihoods of the outliers using all class Gaussians and reject those with a higher likelihood from at least one other class compared to its generator class. Table 4 shows the results for VOS [10], VOS plus rejection sampling (named as VOS+), and FFS. VOS+ enhances the OD detection compared to VOS [10] due to selecting more effective outliers, but it is still worse than FFS. Furthermore, VOS+'s training time is much higher due to the iterative process of computing log-likelihoods through each Gaussian class.

Method	FPR95 ↓	AUROC ↑	mAP (ID) ↑	Time (h)
	OD: MS-COCO / OpenImages			
VOS [10]	48.22 / 52.74	89.04 / 85.97	51.50	2.43
VOS+	45.59 / 47.96	89.40 / 87.19	51.74	7.38
FFS (ours)	**44.15 / 45.08**	**89.71 / 88.29**	**51.80**	**2.18**

Table 4. Comparison of VOS, VOS+, and FFS (ours). The baseline mAP of a closed-set object detector is 51.35.

Normalizing Flow: We conducted a thorough study on several types of normalizing flow models. We selected NICE [6], RealNVP [7], Glow [27] and GIN [45] in the chronological order for this experiment. We fixed the same coupling layers and each model's sub-network s and t configuration for an equivalent comparison. Table 5 shows that Glow outperforms NICE, RealNVP and GIN in terms of

both FPR95 and AUROC scores. GIN performs best for ID detection but has a worse OD detection performance. As the primary task of FFS is OD detection, we chose Glow as our flow model for all our experiments.

Flow model	FPR95 ↓	AUROC ↑	mAP ↑	Time (h)
NICE [6]	48.23	88.46	51.81	2.23
RealNVP [7]	45.76	88.95	51.84	2.28
Glow [27]	**44.15**	**89.71**	51.80	**2.18**
GIN [45]	48.31	88.36	**52.00**	2.25

Table 5. Validation of the Normalizing Flow models.

Figure 5 (c) shows the effect of the number of coupling layers M on OD detection performance. $M = 2$ produces the best results as the inlier features fit a suitable number of flow parameters. However, the OD detection performance degrades for $M > 2$, and the training time significantly increases due to more trainable parameters in the network. Adding more trainable parameters requires more training data for effective training; otherwise, it might lead to overfitting and a more complex optimization, which should explain the degradation in performance. Hence, we fix $M = 2$, and in Figure 5 (a), we study the effect of the number of fully-connected (fc) layers within s and t subnetworks of each coupling layer. There is a slight improvement in OD detection performance when two fc layers are employed. In Figure 5 (b), we fix the number of coupling and fc layers in each s and t sub-networks as two and adjust the number of neurons in each fc layer. We report a degradation in OD detection performance above 2048 neurons.

$\mathcal{L}_{reg}$	FPR95 ↓	AUROC ↑	mAP ↑	Time (h)
CE	69.92	84.72	48.70	2.27
JSD	52.83	85.79	48.60	2.25
Hinge	51.32	86.34	46.54	2.19
BCE (Ours)	**44.15**	**89.71**	**51.80**	**2.18**

Table 6. Validation of $\mathcal{L}_{reg}$ on OD detection performance.

Regularization loss $\mathcal{L}_{reg}$ and its weightage α: We studied several regularization losses for our FFS framework. Firstly, we used a simple multi-class cross-entropy loss (CE) on the classification logits without including the energy module. Secondly, we evaluated Jensen-Shannon divergence (JSD) loss, where we minimized the divergence between the distribution of the classification logits of the outlier samples and uniform distribution. In Table 6, we show that our BCE based $\mathcal{L}_{reg}$ loss outperforms all other loss functions in terms of OD detection performance. The ID detection performance is also higher, even though the training time is comparable. In Figure 5 (e), we demonstrate the effect of changing the weightage α of $\mathcal{L}_{reg}$ for PASCAL-VOC as ID and MS-COCO as OD. As α increases, the OD detection performance improves. However, an even higher weightage degrades the OD detection performance, due to which a careful selection of α is desirable.

Figure 4. Visualization of OD detection on MS-COCO and OpenImages with VOS [10] (first and third row) and FFS (ours) (second and fourth row) trained on PASCAL-VOC 2012 images. Ideally, an object in the images should be labeled as OOD with a green bounding box.

Figure 5. Validation of the flow architecture, loss weightage α, batch size, and projection sampling. The training time (in hours) is shown above the horizontal x axis in the plots.

Projection Sampling: Figure 5 (f) shows the results after varying synthetic outliers. The OD detection improves as we increase the number of sampled outliers o_s but degrades with much higher values of s. The coverage of log-likelihoods from many sampled outliers broadens even though the average log-likelihood is similar to the likelihood threshold δ. Hence, some outliers may lie inside, while others exist far outside the decision boundary leading to bad regularization. We report that rejection sampling achieves better OD detection performance when compared to the best results of projection sampling. Since projection sampling enforces the outliers to be close to the decision boundary, it may be insufficient to estimate the entire outlier manifold. In contrast, the rejection sampling synthesizes outliers near the decision boundary and in the proper

outlier manifold, leading to a better model regularization.

Qualitative analysis: Figure 4 shows the OD detection results on MS-COCO and OpenImages with FFS and VOS trained on PASCAL-VOC. The results show that FFS outperforms VOS in recognizing outlier objects. Additionally, our method's softmax confidence score is lower than VOS when both methods misclassify the outlier objects as inliers.

5. Conclusion

We presented a new methodology for outlier-aware object detection that learns a combined data distribution of all inlier object classes. By sampling from the low-likelihood region of a jointly trained normalizing flow model, our approach generates suitable synthetic outlier samples for training the outlier detection head of our compound model. In contrast, previous approaches model inliers with class-conditional Gaussians, resulting in outlier samples that may have a high likelihood for some other class. We report state-of-the-art results for image and video datasets with standard outlier detection metrics while decreasing the model training time. We also show that simple rejection sampling contributes more useful synthetic outliers than projection sampling with a fixed likelihood threshold. In future work, one could validate this finding on other OD detectors. Our idea also extends to developing an outlier-aware instance segmentation model and an active learning scheme to correct the failure modes in weakly-supervised object detectors.

References

[1] Petra Bevandic, Ivan Kreso, Marin Orsic, and Sinisa Segvic. Dense open-set recognition based on training with noisy negative images. *Image Vis. Comput.*, 2022. 1

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 6

[3] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *ICCV*, 2019. 1

[4] Kumari Deepshikha, Sai Harsha Yelleni, P. K. Srijith, and C. Krishna Mohan. Monte Carlo dropblock for modelling uncertainty in object detection. *CoRR*, 2021. 2

[5] Akshay Raj Dhamija, Manuel Günther, Jonathan Ventura, and Terrance E. Boult. The overlooked elephant of object detection: Open Set. In *WACV*, 2020. 2

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent components estimation. In *ICLR*, 2015. 7

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *ICLR*, 2017. 7

[8] Keval Doshi and Yasin Yilmaz. Any-shot sequential anomaly detection in surveillance videos. In *CVPR Workshops*, 2020. 2

[9] Xuefeng Du, Xin Wang, Gabriel Gozum, and Yixuan Li. Unknown-aware Object Detection: Learning what you don't know from videos in the wild. In *CVPR*, 2022. 1, 2, 3, 6, 7, 12, 13, 14

[10] Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. VOS: Learning what you don't know by virtual outlier synthesis. In *ICLR*, 2022. 1, 2, 5, 6, 7, 8, 11, 12

[11] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010. 1, 2, 6

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 2

[13] Ross Girshick. Fast R-CNN. In *ICCV*, 2015. 2

[14] Matej Grcic, Petra Bevandic, and Sinisa Segvic. Dense open-set recognition with synthetic outliers generated by Real NVP. *CoRR*, 2020. 2

[15] David Hall, Feras Dayoub, John Skinner, Haoyang Zhang, Dimity Miller, Peter Corke, Gustavo Carneiro, Anelia Angelova, and Niko Sünderhauf. Probabilistic Object Detection: Definition and Evaluation. In *WACV*, 2020. 2

[16] Ali Harakeh and Steven L. Waslander. Estimating and evaluating regression predictive uncertainty in deep object detectors. In *ICLR*, 2021. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 1, 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[19] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. 1, 2

[20] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019. 1, 2

[21] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *NeurIPS*, 2019. 2

[22] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020. 2

[23] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized ODIN: Detecting out-of-distribution image without learning from out-of-distribution data. In *CVPR*, 2020. 2

[24] Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. In *NeurIPS*, 2021. 1, 2

[25] Radu Tudor Ionescu, Fahad Shahbaz Khan, Mariana-Iuliana Georgescu, and Ling Shao. Object-centric auto-encoders and dummy anomalies for abnormal event detection in video. In *CVPR*, 2019. 2

[26] K. J. Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N. Balasubramanian. Towards open world object detection. In *CVPR*, 2021. 2

[27] Durk P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018. 2, 4, 6, 7

[28] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The Open Images dataset v4. *International Journal of Computer Vision*, 2020. 6

[29] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018. 1, 2, 6

[30] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018. 1, 2

[31] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018. 1, 2

[32] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 6

[33] Ziqian Lin, Sreya D. Roy, and Yixuan Li. MOOD: Multi-level out-of-distribution detection. In *CVPR*, 2021. 2

[34] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection - A new baseline. In *CVPR*, 2018. 3

[35] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. In *NeurIPS*, 2020. 1, 2, 5, 6

[36] Thomas Lucas, Konstantin Shmelkov, Karteek Alahari, Cordelia Schmid, and Jakob Verbeek. Adaptive density estimation for generative models. In *NeurIPS*, 2019. 1

[37] Dimity Miller, Feras Dayoub, Michael Milford, and Niko Sünderhauf. Evaluating merging strategies for sampling-based uncertainty techniques in object detection. In *ICRA*, 2019. 2

[38] Dimity Miller, Lachlan Nicholson, Feras Dayoub, and Niko Sünderhauf. Dropout sampling for robust object detection in open-set conditions. In *ICRA*, 2018. 2

[39] Sina Mohseni, Mandar Pitale, JBS Yadawa, and Zhangyang Wang. Self-supervised learning for generalizable out-of-distribution detection. In *AAAI*, 2020. 1, 2

[40] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020. 6

[41] Mahdyar Ravanbakhsh, Moin Nabi, Hossein Mousavi, Enver Sangineto, and Nicu Sebe. Plug-and-play CNN for crowd motion analysis: An application in abnormal event detection. In *WACV*, 2018. 3

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2, 6

[43] Tobias Riedlinger, Matthias Rottmann, Marius Schubert, and Hanno Gottschalk. Gradient-based quantification of epistemic uncertainty for deep object detectors. *CoRR*, 2021. 2

[44] Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with Gram Matrices. In *ICML*, 2020. 1, 2

[45] Peter Sorrenson, Carsten Rother, and Ullrich Köthe. Disentanglement by nonlinear ICA with general incompressible-flow networks (GIN). In *ICLR*, 2020. 7

[46] Yiyou Sun, Chuan Guo, and Yixuan Li. ReAct: Out-of-distribution detection with rectified activations. In *NeurIPS*, 2021. 2

[47] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. CSI: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020. 2, 6

[48] Haoran Wang, Weitang Liu, Alex Bocchieri, and Yixuan Li. Can multi-label classification networks know what they don't know? In *NeurIPS*, 2021. 2

[49] Xin Wang, Thomas E. Huang, Benlin Liu, Fisher Yu, Xiaolong Wang, Joseph E. Gonzalez, and Trevor Darrell. Robust object detection via instance-level temporal cycle confusion. In *ICCV*, 2021. 3

[50] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[51] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 2, 6

[52] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020. 2, 6

[53] Jingyang Zhang, Nathan Inkawhich, Yiran Chen, and Hai Li. Fine-grained out-of-distribution detection with mixup outlier exposure. *CoRR*, 2021. 1

Supplementary Material

A. Glossary

$\mathcal{X}$ The training dataset with $\mathcal{X} := (x_1, x_2, ..., x_N)$ containing N images or videos

$\mathcal{Y}$ The ground-truth labels with $\mathcal{Y} := \{1, 2, ..., K\}$ for $K + 1$ inlier object classes

$\mathcal{B}$ The ground-truth coordinates of the bounding boxes with $\boldsymbol{b} \in \mathcal{B}$

$l(x, \hat{\boldsymbol{b}})$ The fixed-size box features of both inlier and background patches given the predicted bounding boxes $\hat{\boldsymbol{b}}$

$l_{ID}(x, \hat{\boldsymbol{b}})$ The fixed-size box features of only inlier patches given the predicted bounding boxes $\hat{\boldsymbol{b}}$

$\mathcal{L}_{det}$ The standard object detection loss consisting of object classification and bounding-box regression losses

$\mathcal{L}_{nll}$ The negative log-likelihood loss to train the normalizing flow model on inlier features

$\mathcal{L}_{reg}$ The regularization loss for discriminative training using inlier and synthetic outlier features

ξ The energy-based threshold, when 95% of inlier objects in the validation set are correctly detected

h The classification head in the ROI head module of the Faster R-CNN architecture

f The normalizing flow network to maximize the likelihood of inlier features

$\mathcal{Z}$ The latent space of the flow network f defined as a multivariate Gaussian with zero mean and unit variance

Φ The binary classifier to differentiate the inlier from the synthesized outlier features via discriminative training

θ The learnable parameters of the standard object detector, i.e. Faster R-CNN

γ The learnable parameters of the normalizing flow f

ψ The learnable parameters of the binary classifier Φ

g_k The generated synthetic features after randomly sampling k samples from flow's latent space

o_s The selected synthetic outlier features from the generated features g_k such that $o_s \subset g_k$

τ The step size of the gradient descent for the projection sampling based outlier synthesis

δ The log-likelihood threshold to determine the synthesized outlier features based on projection sampling

$E(h(.))$ The energy-score calculated from the output of the classification head h

T The temperature coefficient to compute the energy score $E(h(.))$

α The weightage of the regularization loss $\mathcal{L}_{reg}$ in the overall training objective

β The weightage of the negative log-likelihood loss $\mathcal{L}_{nll}$ in the overall training objective

B. Visualization of inlier and synthesized outlier features

We provide the visualization of inlier features (in color) of PASCAL-VOC along with the synthesized outliers (in black) after reducing the number of feature embeddings using Principal Component Analysis (PCA). We compare the results from VOS [10] and our FFS approach in Figure 6. It is noticeable that VOS synthesizes outliers separately for each inlier class. In contrast, our approach synthesizes outliers after estimating the accurate data distribution of all inlier classes using the normalizing flow model, thereby leading to more effective regularization.
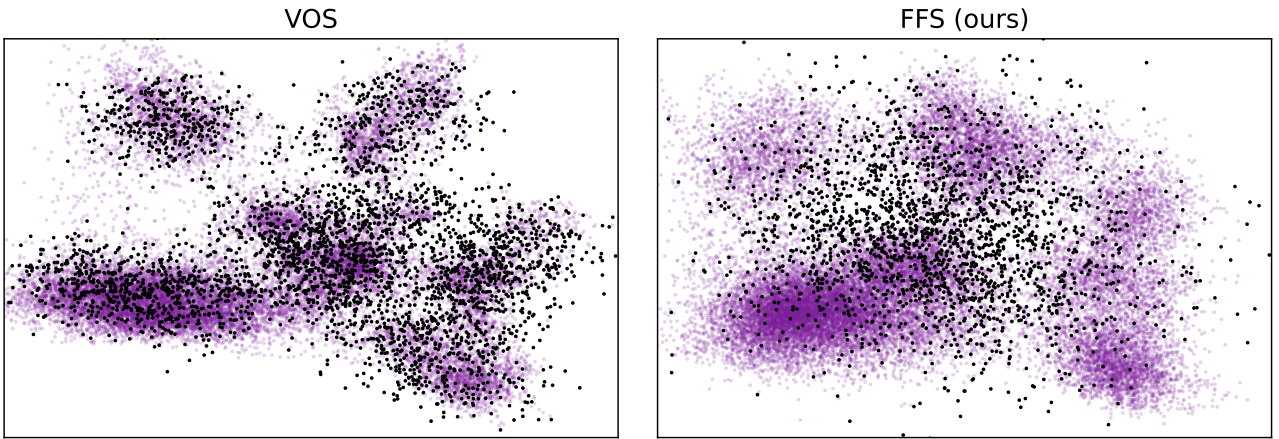

Figure 6. PCA visualization of synthesized outliers (in black) and inlier features (in color) of all 20 PASCAL-VOC object classes.

C. Step size τ for projection sampling

In Table 7, we show the OD detection results of FFS approach for varying step size τ when projection sampling was used to synthesize outliers precisely at the decision boundary. For this experiment, we fixed PASCAL-VOC as the inlier and MS-COCO as the outlier dataset, respectively. We analyzed the number s of synthetic outliers, o_s, and τ to evaluate the performance. It can be seen from the results that, irrespective of the step size τ, the OD detection performance improves as we increase the number s of synthesized outliers o_s. However, the performance gets worse when the s is increased further.

# samples, s	Step size, τ		
	$\tau = 1$	$\tau = 0.5$	$\tau = 0.1$
	FPR95 $\downarrow$ / AUROC $\uparrow$		
1	55.56 / 85.90	51.13 / 86.73	52.34 / 86.21
2	50.56 / 87.39	49.48 / 87.95	49.94 / 86.92
4	**48.93** / 88.83	**47.23** / 89.44	**47.49** / **88.72**
8	49.79 / **88.96**	48.28 / **89.61**	48.95 / 88.84
16	59.02 / 81.47	59.60 / 82.09	59.32 / 83.54
32	60.44 / 82.38	57.92 / 81.99	58.54 / 82.65
64	57.72 / 82.43	58.67 / 82.22	59.91 / 83.43

Table 7. OD detection results after varying the step size τ of our projection sampling based strategy to synthesize outliers.

D. Datasets

Our experimental setup consists of three inlier and outlier datasets, where we train FFS on both image and video-based inlier datasets. The image-based inlier dataset, i.e., PASCAL-VOC, does not contain the sequence of image frames in terms of time. In contrast, the video-based datasets, BDD100K and Youtube-VIS, are a sequence of image frames dependent on time. We follow the experimental setup of VOS [10] and STUD [9] and utilize the datasets provided for training and inference of FFS. In addition, we show the object classes for each of the inlier datasets for a better interpretation of the subsequent visual results:

- **PASCAL-VOC:** There are 16,551 training and 4,952 validation images in the dataset. The object classes are *person, bird, cat, cow, dog, horse, sheep, airplane, bicycle, boat, bus, car, motorcycle, train, bottle, chair, dining table, potted plant, couch and tv.*

- **BDD100K:** There are 273,406 training and 39,973 validation images in the dataset. The object classes are *pedestrian, rider, car, truck, bus, train, motorcycle and bicycle.*

- **Youtube-VIS:** There are 67,861 training and 21,889 validation images in the dataset. The object classes are *airplane, bear, bird, boat, car, cat, cow, deer, dog, duck, earless seal, elephant, fish, flying disc, fox, frog, giant panda, giraffe, horse, leopard, lizard, monkey, motorbike, mouse, parrot, person, rabbit, shark, skateboard, snake, snowboard, squirrel, surfboard, tennis racket, tiger, train, truck, turtle, whale and zebra.*

We evaluate our trained FFS framework on three different outlier datasets, namely MS-COCO, OpenImages, and nuImages. There are 930 images in MS-COCO and 1,761 images in OpenImages datasets, so objects in these images do not fall into any of the inlier classes of PASCAL-VOC. Similarly, there are 2,100 images of the nuImages dataset for evaluating FFS trained on BDD100K and 28,922 images of MS-COCO for evaluating FFS trained on Youtube-VIS. In all outlier datasets, the objects present in the inlier dataset are mutually independent of objects in the outlier dataset.

E. Visualization of OD detection results for video datasets

In Figure 7 and Figure 8, we show the OD detection results when FFS was trained on video datasets. The results showcase that we obtain significantly better results in detecting outliers and reducing the number of incorrect bounding box predictions compared to STUD [9]. Additionally, for some image pairs, we reduce the model's confidence when the object was wrongly detected as an inlier by our approach and STUD [9].

Figure 7. OD detection results on MS-COCO images when `FFS` was trained on Youtube-VIS dataset. In each image pair, the top image is the results from STUD [9], and the bottom image is the results from `FFS`.
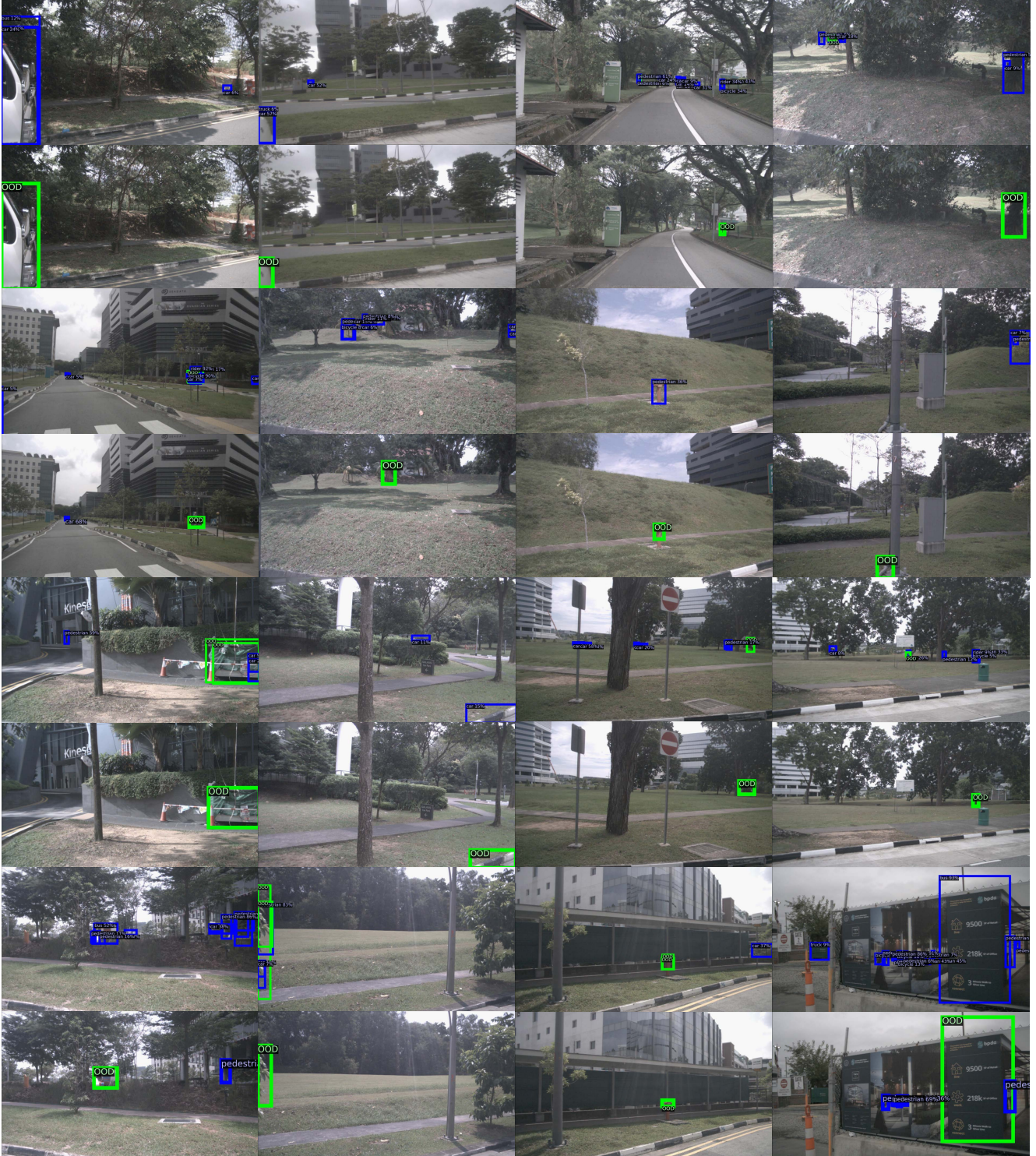
13

Figure 8. OD detection results on nuImages when `FFS` was trained on BDD100K dataset. In each image pair, the top image is the results from STUD [9], and the bottom image is the results from `FFS`.