Deep Learning Classifiers that Return Compact Acceptance Regions for Open Set Recognition and Anomaly Detection Problems

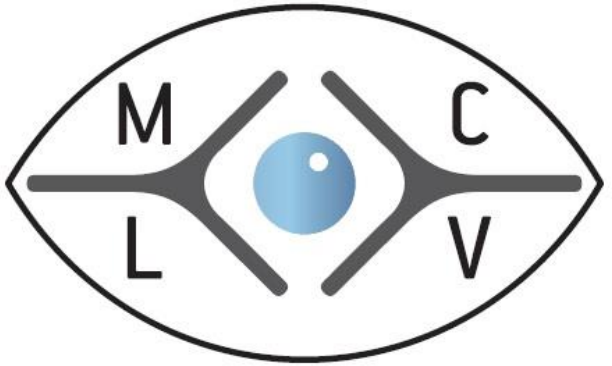
Prof. Dr. Hakan Cevikalp

Halil Sağlamlar, Bedirhan Uzun, Hasan Saribaş, Yusuf Şalk, Okan Köpüklü, Gürkan Öztürk

Eskisehir Osmangazi University

Machine Learning and Computer Vision Laboratory

Electrical and Electronics Engineering Department



Anomaly Detection

- Anomaly detection (also called as outlier detection) is the task of identifying uncharacteristic data samples called anomalies or outliers in a given dataset.
- The anomalies typically significantly deviate from the majority of the data and do not fit well to the data distributions that are considered as normal.
- Anomaly detection has wide applications in many domains including machine vision, cyber-security, medicine, statistics, neuroscience, and law enforcement.

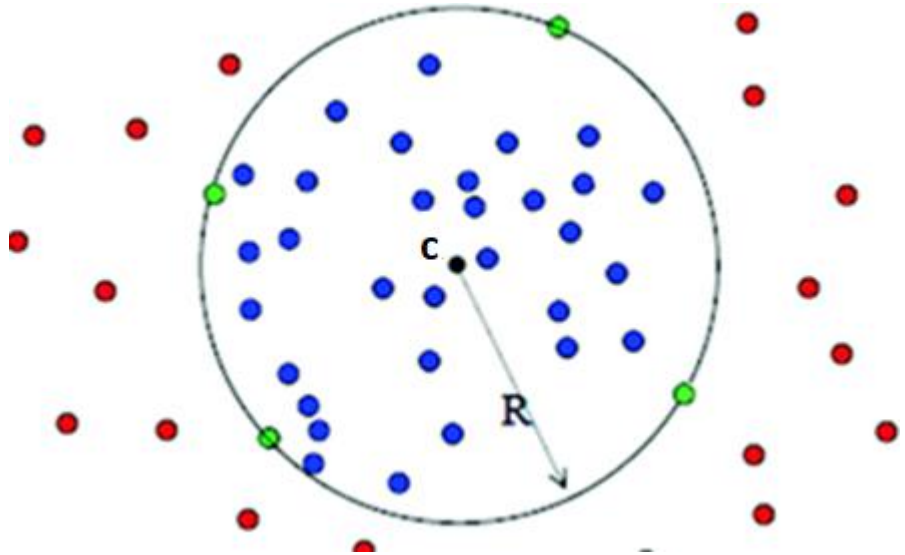
One-Class Classification (Support Vector Description)

Support Vector Data Description (SVDD) is used for anomaly detection and the main goal is to find a compact hypersphere that inscribes all positive samples and excludes negative data samples.

$$\min_{\mathbf{c},\, r \geq 0,\, \boldsymbol{\xi} \geq 0} \quad r^2 + \gamma_+ \sum_i \xi_i + \gamma_- \sum_j \xi_j$$

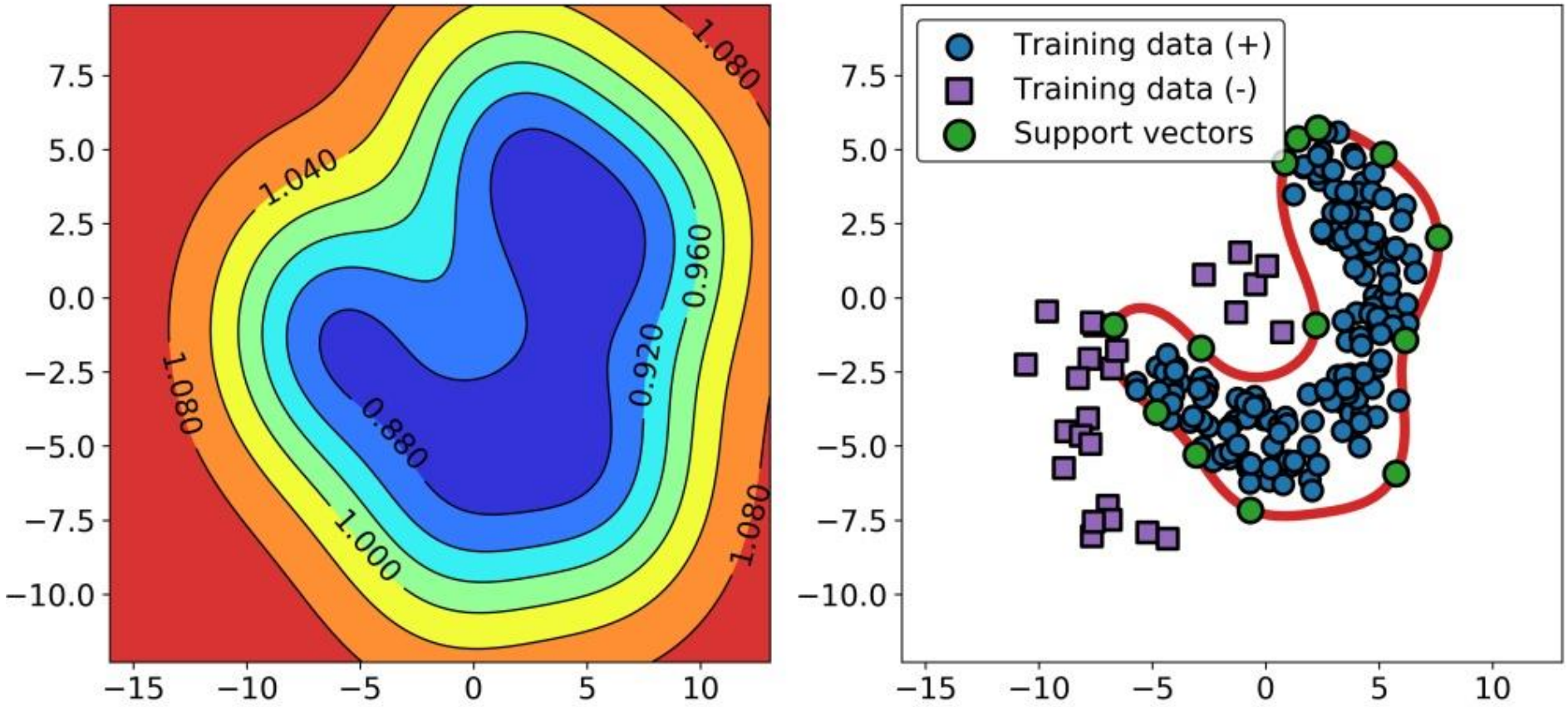
$$\text{s.t.} \quad \|\mathbf{x}_i - \mathbf{c}\|^2 \leq r^2 + \xi_i, \quad i \in I_+$$

$$\|\mathbf{x}_j - \mathbf{c}\|^2 \geq r^2 - \xi_j, \quad j \in I_-$$



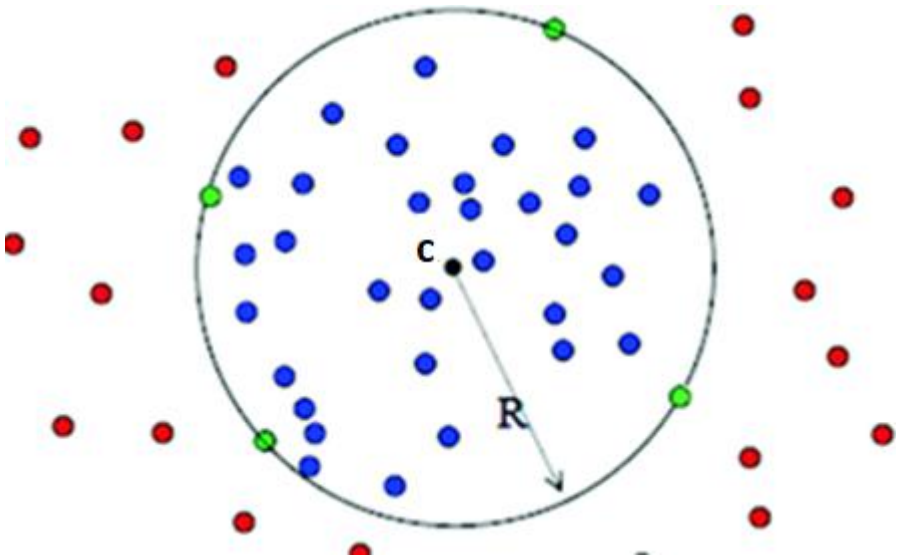
Support Vector Data Description (SVDD)

It returns much less support vectors compared to binary SVM classifier in the nonlinear case.
For example, for face detection dataset, the kernel SVDD classifier method returned 1716 support vectors whereas nonlinear SVM classifier algorithm returned 15,691 support vectors.



Anomaly Detection and Open Set Recognition – Deep SVDD

$$\min_{\mathcal{W}} \quad \frac{1}{n}\sum_{i=1}^{n}\|\phi(\boldsymbol{x}_i;\mathcal{W})-\boldsymbol{c}\|^2+\frac{\lambda}{2}\sum_{\ell=1}^{L}\|\boldsymbol{W}^{\ell}\|_F^2$$



Deep one-class classification methods proposed for anomaly detection (Ruff et al., 2018; Liznerski et al., 2021; Ruff et al., 2021). These methods have the following shortcomings:

i) The hypersphere centers are fixed to either some pre-defined centers or to the origin. To find the nonzero centers, one has to run another autoencoder network, which is suboptimal.

ii) We need special architecture designs to avoid the trivial solutions.

iii) The hypersphere centers are not updated in all proposed methods.

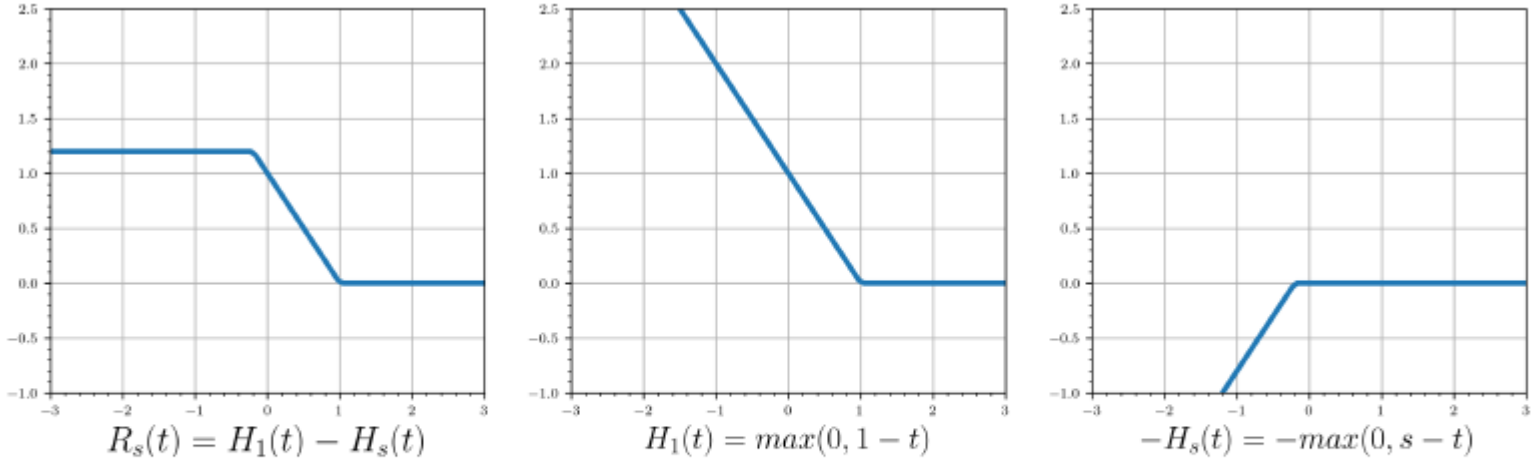
Deep Compact Hyperspheres

- We approximate the normal or known classes with the compact hyperspheres that can be characterized with a center and radius.
- In contrast to the other hypersphere methods, we consider the hypersphere centers as learnable parameters and update them based on the class samples features. This is crucial in the proposed methodology since this eliminates the need for training another autoencoder network for center initialization. When the hypersphere centers are learned from data samples, the classes preserve the semantic relations in the learned feature space.
- We handle the outlier exposure more carefully and we introduce loss terms that are more robust to the label noise within these datasets.

Deep Compact Hyperspheres for Anomaly Detection

Consider that deep neural network features of the normal and outlier exposure samples are respectively denoted as $\mathbf{f}_i \in \mathbb{R}^d \; i = 1, ..., n_+$ and $\mathbf{f}_k \in \mathbb{R}^d \; k = 1, ..., K.$

$$\mathcal{L}_{CHS} = \frac{1}{n_+} \sum_{i=1}^{n_+} \|\mathbf{f}_i - \mathbf{c}_+\|^2 + \frac{\kappa}{(n_+ K)} \sum_{i=1}^{n_+} \sum_{k=1}^{K} R_s \left(\|\mathbf{f}_k - \mathbf{c}_+\|^2 - \|\mathbf{f}_i - \mathbf{c}_+\|^2 \right)$$



$$R_s(t) = \min(m - s, \max(0, m - t))$$

Figure 1: The illustration of the Ramp loss function, $R_s(t) = H_1(t) - H_s(t)$, where $H_m(t) = \max(0, m - t)$ is the classical hinge loss. Here, the margin term is set to $m = 1$ and the parameter s is set to $s = -0.20$ for visualization, we use different values in the experiments.

Deep Compact Hyperspheres

- In the proposed method, we treat the hypersphere center as a learnable parameter vector and update it in each batch based on the changing deep feature representations of training samples. The hypersphere center of the normal class is updated in each iteration as follows:

$$\mathbf{c}_+^{t+1} = \mathbf{c}_+^t - \alpha(\Delta_1 \mathbf{c}_+^t + \Delta_2 \mathbf{c}_+^t)$$

$$\Delta_1 \mathbf{c}_+ = \frac{\sum_{i=1}^{r} \delta(y_i = +1) \cdot (\mathbf{c}_+ - \mathbf{f}_i)}{1 + \sum_{i=1}^{r} \delta(y_i = +1)} \qquad \Delta_2 \mathbf{c}_+ = \begin{cases} 2(\mathbf{c}_+ - \mathbf{f}_i) - 2(\mathbf{c}_+ - \mathbf{f}_k), & \text{if} \quad s < \|\mathbf{f}_k - \mathbf{c}_+\|^2 - \|\mathbf{f}_i - \mathbf{c}_+\|^2 < m \\ 0, & \text{otherwise} \end{cases}$$

Experimental Results

Table 1: AUC Scores obtained on Cifar10 dataset using one vs. rest procedure.

Methods	AUC Scores (%)
DCHS	**97.8**
BCE	96.1
HSC	95.9
GT+	95.6
FCDD	95.0
DSAD	94.5
Deep SVDD	64.8
AND	64.1

Table 2: AUC Scores obtained on Mnist dataset using one vs. rest procedure.

Methods	AUC Scores (%)
DCHS	**98.8**
HSC	98.7
AND	97.5
DSAD	96.9
BCE	95.6
Deep SVDD	92.8

Table 3: AUC Scores obtained on Fashion-Mnist dataset using one vs. rest procedure.

Methods	AUC Scores (%)
DCHS	**93.9**
GEO	93.5
DSAD	91.3
Deep SVDD	89.2
BCE	81.4
FCDD	89.0

Open Set Recognition

- In open set recognition we are given a training set with known class samples and a large-scale dataset with unknown labels.
- In testing time, novel class samples may occur. Thus, the main goals are

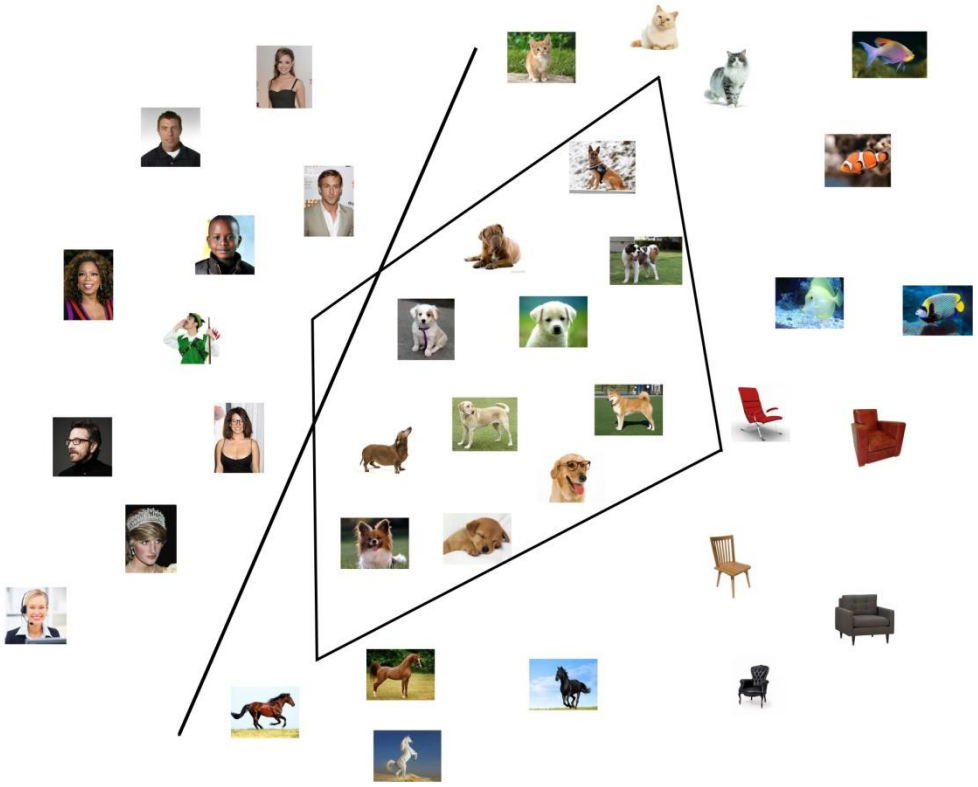
  i) to classify the known class samples correctly,

  ii) reject the unknown class samples.

Open set recognition is also similar to anomaly detection scenario as there are multiple classes that must be compactly approximated since the samples coming from the unknown classes must be rejected at testing time.

Therefore, we need classifiers that return compact acceptance regions.

Polyhedral Convex Conic Functions for Open Set Classification and Visual Object Detection

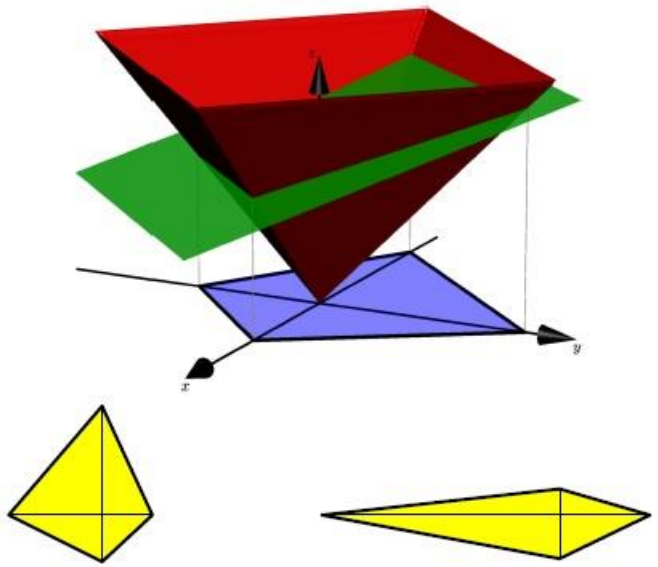


A decision hyperplane returned by an SVM successfully separates its training classes, dogs (positive) and people (negative). However, it also assigns instances of novel classes such as cats, horses, fish and chairs to the dog class, sometimes with higher confidence scores than for dogs themselves. The problem is the over-large acceptance region – SVM only tries to separate dogs and people, not to bound the dog class. A lighter (e.g. Polyhedral or ellipsoidal) decision boundary improves this localization, reducing mis-classifications caused by unforeseen classes and outliers.

Polyhedral Conic Classifiers

- These classifiers use the polyhedral conic functions – essentially projections of hyperplane sections through L1 cones – to define their acceptance regions for positives.
- The Polyhedral Conic Functions and Extended Polyhedral Conic Functions respectively have the forms

$$f_{\mathbf{w},\gamma,\mathbf{s},b}(\mathbf{x}) = \mathbf{w}^T(\mathbf{x}-\mathbf{s}) + \gamma\|\mathbf{x}-\mathbf{s}\|_1 - b \quad (PCF)$$
$$f_{\mathbf{w},\boldsymbol{\gamma},\mathbf{s},b}(\mathbf{x}) = \mathbf{w}^T(\mathbf{x}-\mathbf{s}) + \boldsymbol{\gamma}^T|\mathbf{x}-\mathbf{s}| - b \quad (EPCF)$$



Polyhedral Conic Classifiers

Definition: A function $f(\mathbf{x}): R^d \to R$ is called polyhedral conic if its graph is a cone and all its level sets

$$S_\alpha = \{\mathbf{x} \in R^d : f(\mathbf{x}) \leq \alpha\}$$

for $\alpha \in R$, are polyhedrons.

Lemma: A graph of the PCF and EPCF functions is a polyhedral cone with a vertex at $(\mathbf{s}, -b)$.

The proposed polyhedral conic classifiers use PCF and EPCF, with decision regions $f(\mathbf{x}) \leq 0$ for positives and

$f(\mathbf{x}) > 0$ for negatives. Note that this is the opposite of the popular SVM decision rule.

Binary Class Formulation

- By defining $\widetilde{\mathbf{x}} \equiv \begin{pmatrix} \mathbf{x}-\mathbf{s} \\ \|\mathbf{x}-\mathbf{s}\|_1 \end{pmatrix} \in R^{d+1}$, $\widetilde{\mathbf{w}} \equiv \begin{pmatrix} -\mathbf{w} \\ -\gamma \end{pmatrix}$ and $\widetilde{b} = b$ for PCF; $\widetilde{\mathbf{x}} \equiv \begin{pmatrix} \mathbf{x}-\mathbf{s} \\ |\mathbf{x}-\mathbf{s}| \end{pmatrix} \in R^{2d}$, $\widetilde{\mathbf{w}} \equiv \begin{pmatrix} -\mathbf{w} \\ -\gamma \end{pmatrix}$, and $\widetilde{b} = b$ for EPCF, we obtain the same decision functions as in SVM. So we can use SVM formulation to solve the optimization problem.

$$\begin{aligned} \underset{\widetilde{\mathbf{w}},\tilde{b}}{\text{argmin}} \quad & \frac{1}{2}\widetilde{\mathbf{w}}^T\widetilde{\mathbf{w}} + C_+\sum_i \xi_i + C_-\sum_j \xi_j \\ \text{such that} \quad & \widetilde{\mathbf{w}}^T\tilde{\mathbf{x}}_i + \tilde{b} + \xi_i \geq 1, \quad i \in I_+, \\ & \widetilde{\mathbf{w}}^T\tilde{\mathbf{x}}_j + \tilde{b} + \xi_i \leq -1, \quad j \in I_-, \\ & \xi_i, \xi_j \geq 0, \end{aligned}$$

Polyhedral Conic Classifiers

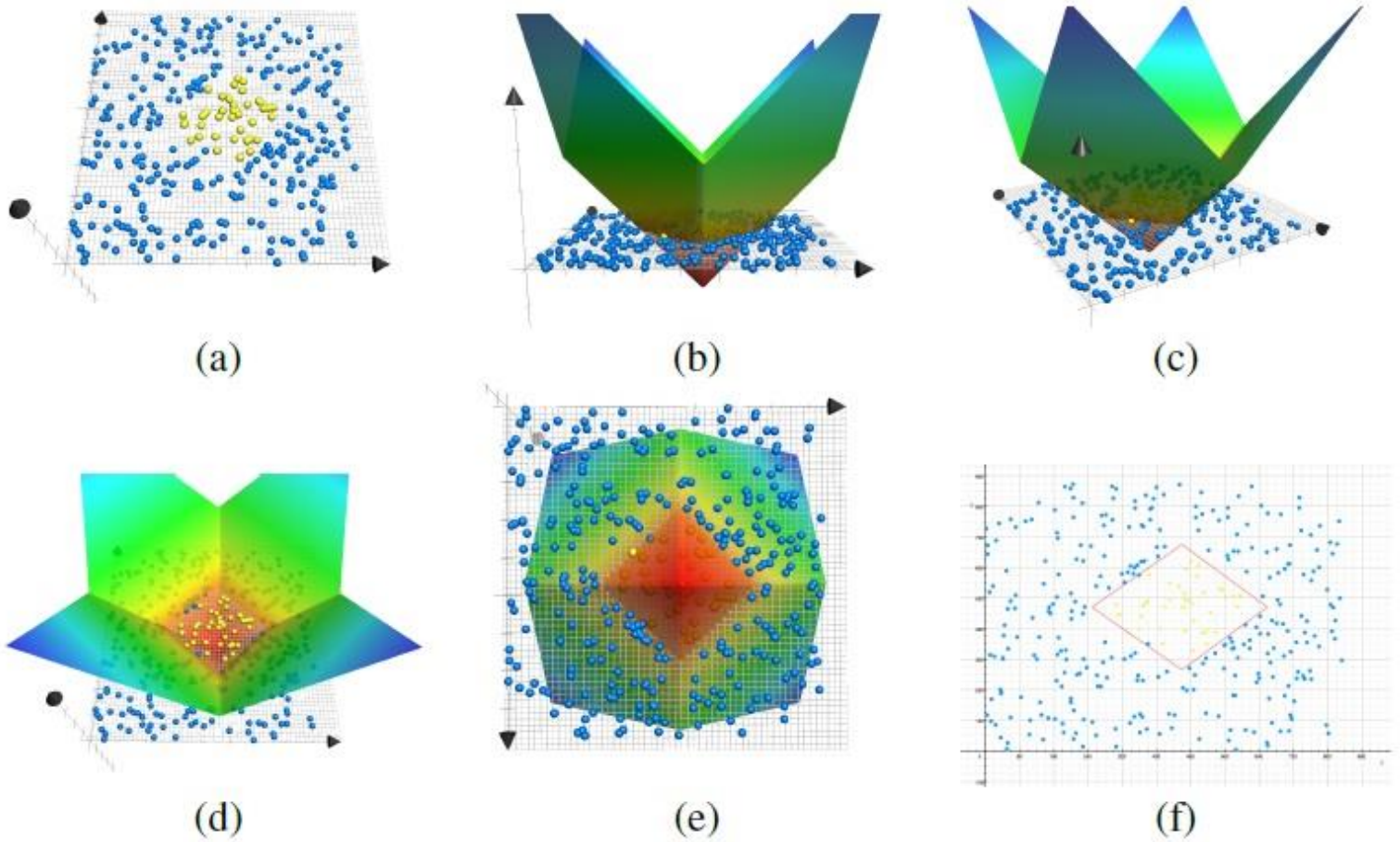


Figure 1: Visualization of PCC classifiers for 2D synthetic data: The positive acceptance regions are "kite-like"' octahedroids containing the points for which a linear hyperplane lies above an L_1 cone.(a): 2D positive (yellow) and negative (blue) samples, (b)-(e): views of positive-class acceptance regions from different angles in 3D, (f): Resulting "kite-like" acceptance region in 2D space.

One-Class PCC/EPCC Classifiers

- SVM formulation does not necessarily guarantee bounded acceptance regions.
- To return both convex and bounded polyhedral acceptacen regions, we need to ensure that slope weights to be less than gamma, i.e.,

$\gamma > 0, \|\mathbf{w}\|_\infty < \gamma$ PCC, $\boldsymbol{\gamma} > 0, |w_i| < \gamma_i \text{ for } i = 1, \ldots, d$ EPCC

$$f_{\mathbf{w},\gamma,\mathbf{s},b}(\mathbf{x}) = \mathbf{w}^T(\mathbf{x}-\mathbf{s}) + \gamma\|\mathbf{x}-\mathbf{s}\|_1 - b \quad (PCF)$$
$$f_{\mathbf{w},\boldsymbol{\gamma},\mathbf{s},b}(\mathbf{x}) = \mathbf{w}^T(\mathbf{x}-\mathbf{s}) + \boldsymbol{\gamma}^T|\mathbf{x}-\mathbf{s}| - b \quad (EPCF)$$

The returned class region has a width which is roughly equal to O(b/γ), so we have to ensure that γ cannot shrink to zero for compact acceptance regions.

$$\underset{\mathbf{w},\boldsymbol{\gamma}}{\text{argmin}} \quad \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{n_+}\sum_i \xi_i + \frac{1}{n_-}\sum_j \xi_j - \boldsymbol{\kappa}^T\boldsymbol{\gamma}$$
$$\text{such that} \quad \mathbf{w}^T(\mathbf{x}_i-\mathbf{s}) + \boldsymbol{\gamma}^T|\mathbf{x}_i-\mathbf{s}| - 1 < \xi_i, \quad i \in I_+,$$
$$\mathbf{w}^T(\mathbf{x}_j-\mathbf{s}) + \boldsymbol{\gamma}^T|\mathbf{x}_i-\mathbf{s}| - 1 \geq 1 - \xi_j, \quad j \in I_-,$$
$$\xi_i, \xi_j \geq 0,$$

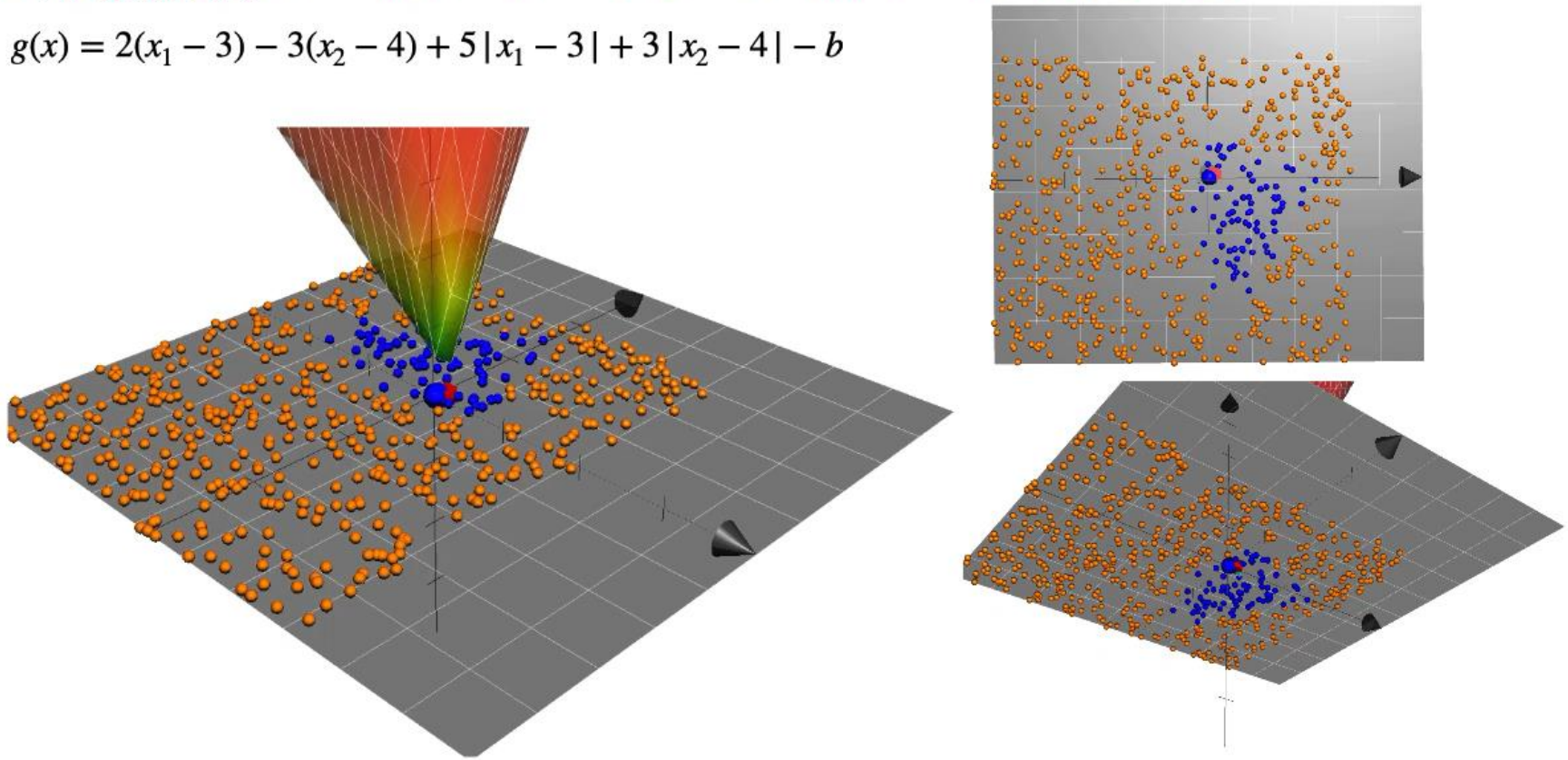
Cevikalp & Saglamlar, “Polyhedral Conic Classifiers for Computer Vision Applications and Open Set Recognition,” TPAMI, 2021.

One-Class PCC/EPCC Classifiers

Parameter b is changing [0,150]

$$g_{(w_1,w_2,\gamma_1,\gamma_2,b,c_1,c_2)}(x) = w_1(x_1 - c_1) + w_2(x_2 - c_2) + \gamma_1 |x_1 - c_1| + \gamma_2 |x_2 - c_2| - b$$

$$g(x) = 2(x_1 - 3) - 3(x_2 - 4) + 5|x_1 - 3| + 3|x_2 - 4| - b$$

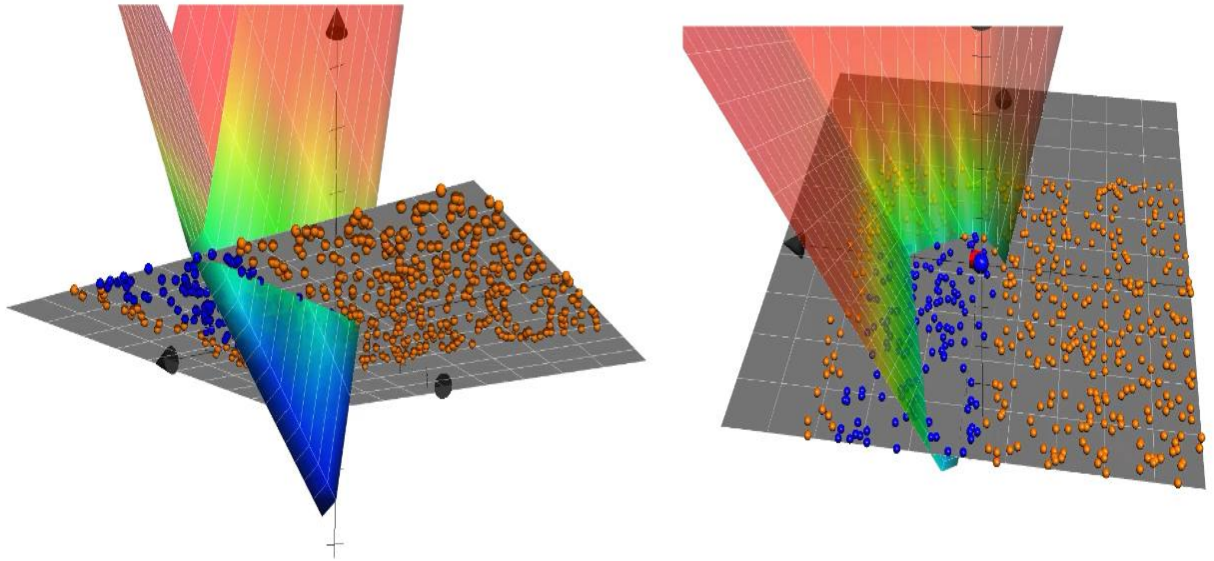


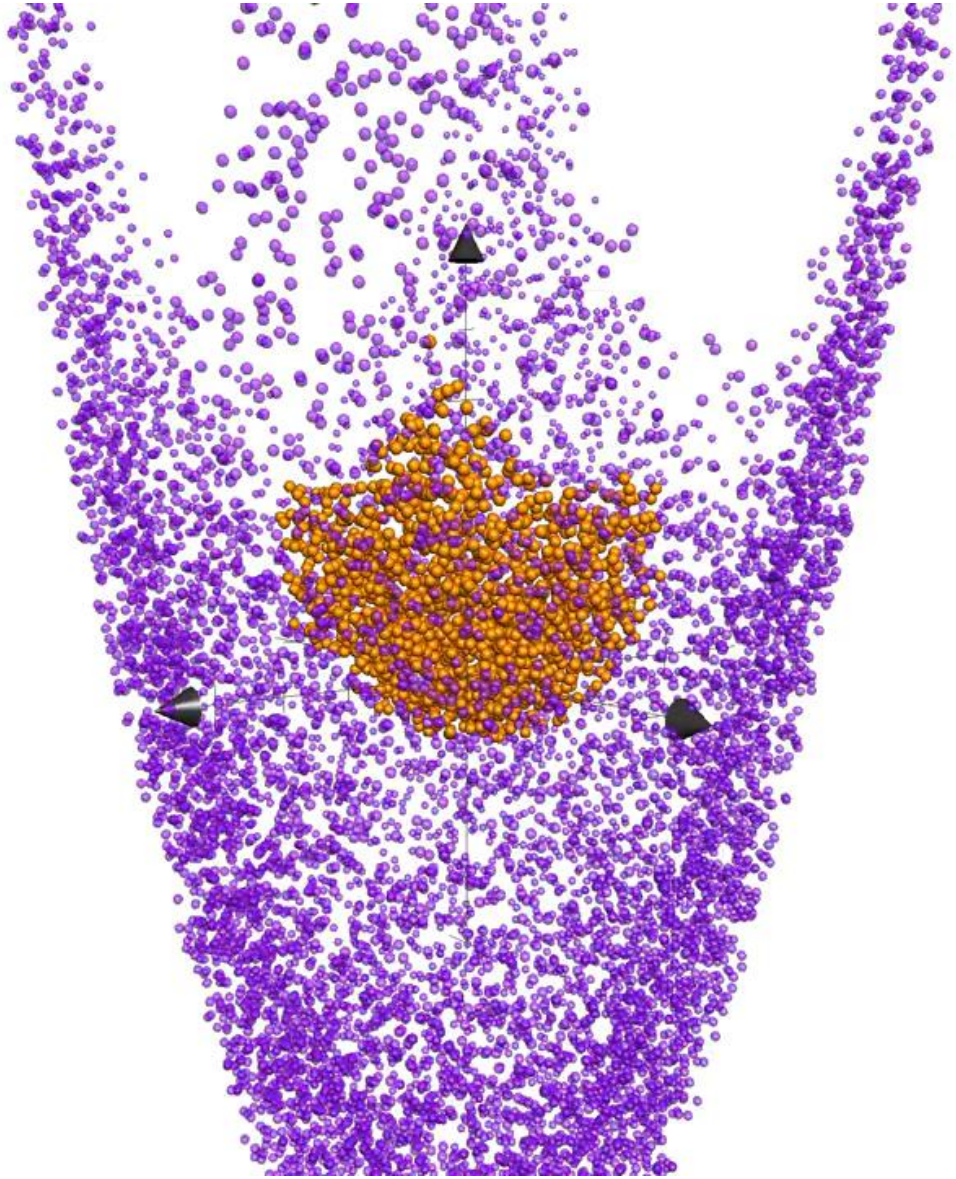
One-Class PCC/EPCC Classifiers

Parameter γ_1 is changing $[2,4]$

$$g_{(w_1,w_2,\gamma_1,\gamma_2,b,c_1,c_2)}(x) = w_1(x_1 - c_1) + w_2(x_2 - c_2) + \gamma_1|x_1 - c_1| + \gamma_2|x_2 - c_2| - b$$

$$g(x) = 2(x_1 - 3) - 3(x_2 - 4) + \gamma_1|x_1 - 3| + 3|x_2 - 4| - 150$$





EPCC Extension – Deep Compact EPCC

- Our first proposed Deep EPCC method possesses two major limitations:

i) Instead of using a different cone vertex for each class, we used only a single common cone vertex for all classes in multi- class classification problems. This largely limits the classifier to return more discriminative class decision boundaries.

ii) We did not use any mechanism to return more compact class acceptance regions, which corresponds to minimizing the intra-class variations.

Importance of Using Different Vertices for Each Class

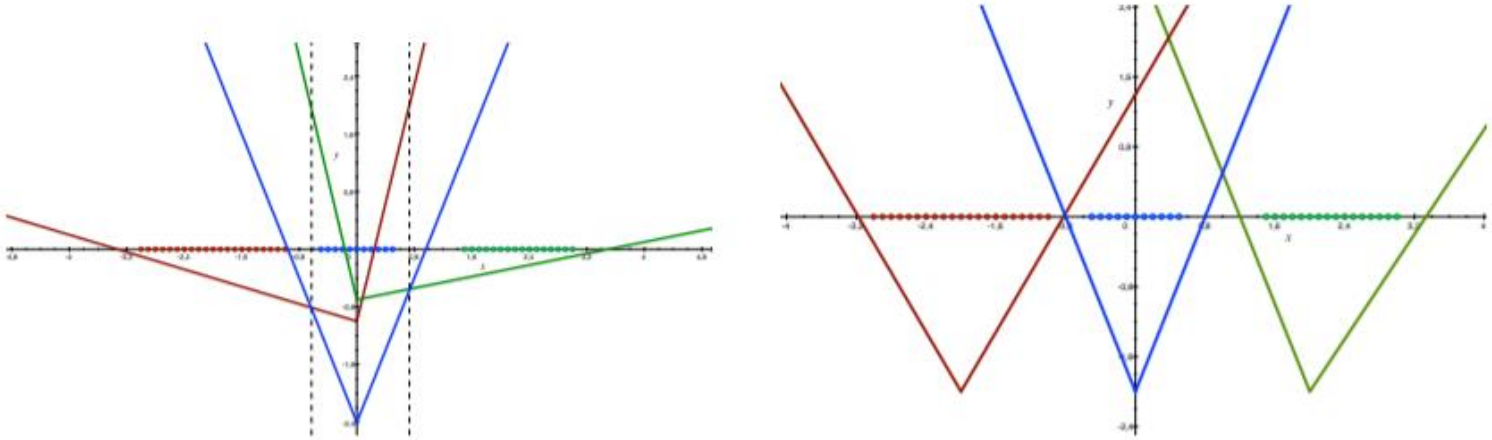


Figure 2. Using a single cone vertex for all three classes as seen on the left restrains to return better acceptance regions for classes. Using 3 cones (the separating cone of each class has the same color as the class samples) seen on the left figure still manages to return true acceptance regions determined by the dashed lines. However, this becomes more difficult or impossible for the classes whose samples lie far from the overall mean. In contrast, using a different cone vertex point $\mathbf{s}_c$, $c = 1, \ldots, 3$, as seen on the right causes to return better acceptance regions easily. It also significantly improves the convergence time during training.

Deep Compact EPCC (Cevikalp et al., "Deep compact polyhedral conic classifier for open and closed set recognition," Pattern Recognition, 2021)

Consider that fi denote the deep CNN features of the samples. By setting $\widetilde{w}_c = -w_c$, $\tilde{\gamma}_c = -\gamma_c$, (c = 1,…,C), $\widetilde{W} = [\widetilde{w}_1 \quad \dots \quad \widetilde{w}_C]$, the loss function of the proposed method can be written as,

$$\min_{\widetilde{\mathbf{W}},\boldsymbol{b}} \frac{\lambda}{2}\sum_{c=1}^{C}(\widetilde{\mathbf{w}}_c^T\widetilde{\mathbf{w}}_c) + \sum_{i=1}^{n}\sum_{j\epsilon(1,\dots,C),j\neq y_i} H_1(\left[\widetilde{\mathbf{w}}_{y_i}^T(\mathbf{f}_i - \mathbf{s}_{y_i}) + \tilde{\boldsymbol{\gamma}}_{y_i}^T|\mathbf{f}_i - \mathbf{s}_{y_i}| + b_{y_i}\right]$$

$$- \left[\widetilde{\mathbf{w}}_j^T(\mathbf{f_i} - \mathbf{s}_j) + \tilde{\boldsymbol{\gamma}}_j^T|\mathbf{f_i} - \mathbf{s_j}| + b_j\right]) + \eta\sum_{c=1}^{C}\sum_{m=1}^{d}\max(0, \kappa - (-\tilde{\gamma}_{cm} - |\widetilde{w}_{cm}|))$$

Synthetic Experiments – Mnist dataset

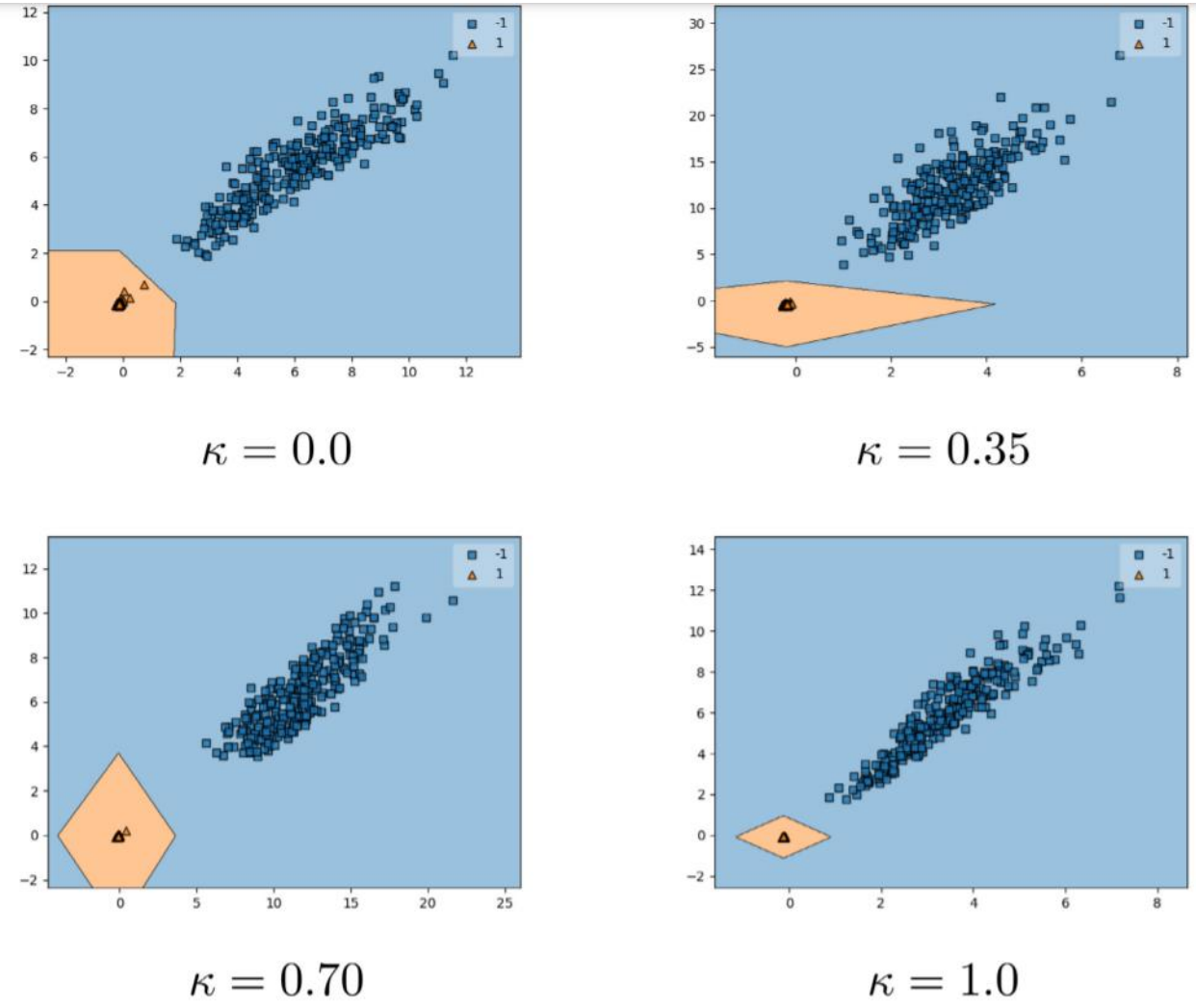


Fig. The positive class (shown with orange color) acceptance regions for different κ values. If we do not enforce any constraint on the learned weights, i.e., κ = 0.0, the acceptance region is unbounded. As we increase the value of κ, the positive class acceptance regions shrink, and the positive class samples tightly cluster around a compact region as expected.

Synthetic Experiments – FaceScrub dataset

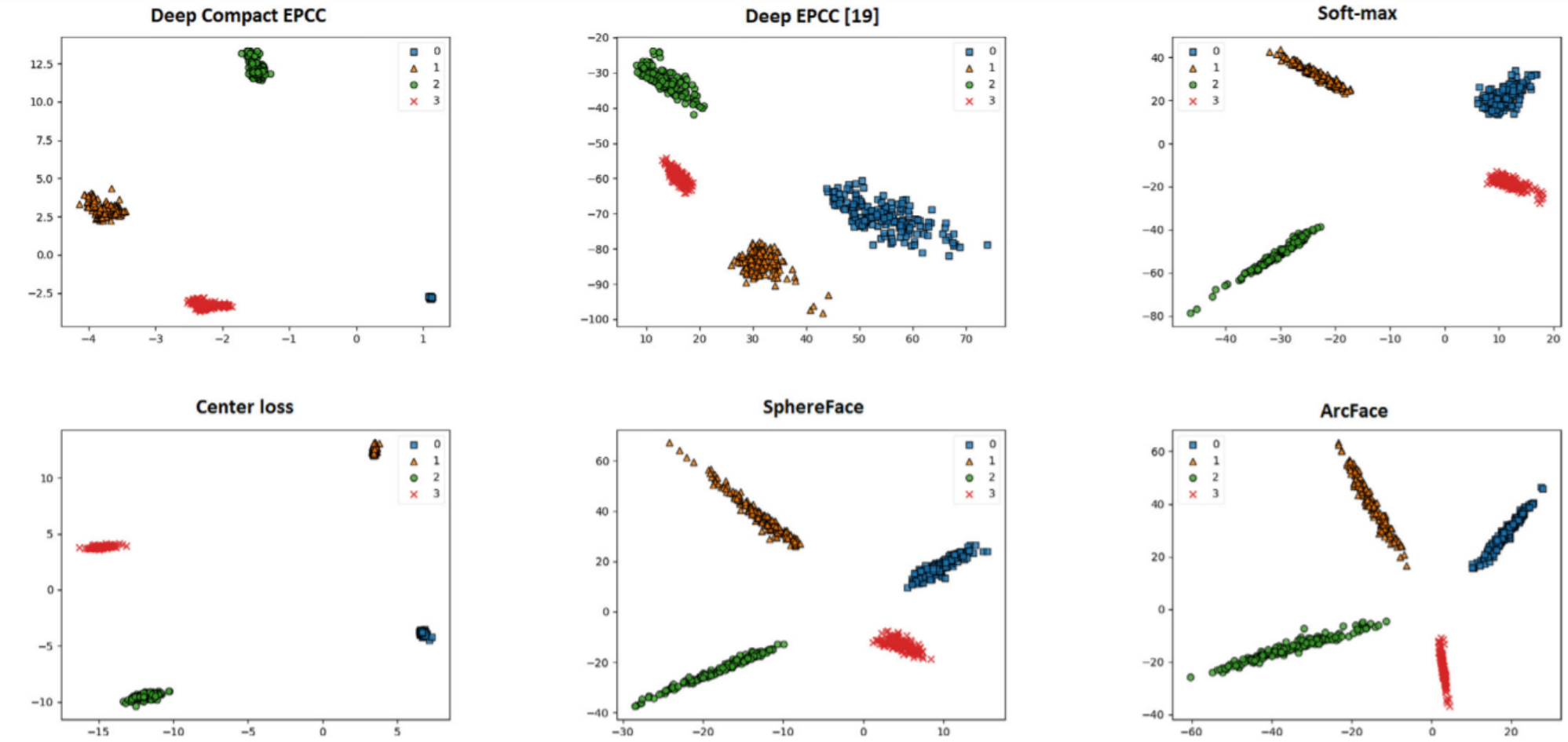


Fig. 5. Learned feature representations of face images for different methods.

Open Set Recognition Experiments

Table 1. AU-ROC Scores (%) for open-set recognition. Results are averaged over 5 random partitions of known/open set classes. All the values in the table other than the proposed method and Deep EPCC are taken from [17] and [20].

Method	**CIFAR-10**	**SVHN**
DC-EPCC	87.7	**93.2**
Deep EPCC [3]	84.8	92.0
Soft-max	67.7	88.6
OpenMax [2]	69.5	89.4
G-OpenMax [8]	67.5	89.6
OSRCI [17]	69.9	91.0
C2AE [20]	**89.5**	92.2

Closed Set Recognition Results

Table 2. Classification Rates (%) on the CIFAR-100 and Face-Scrub datasets.

Method	**CIFAR-100**	**FaceScrub**
DC-EPCC	**83.17**	98.19
Deep EPCC [3]	78.06	98.18
Soft-max	81.88	96.02
Center Loss [32]	82.19	**98.99**
SphereFace [15]	75.19	97.50
CosFace [29]	78.85	97.97
ArcFace [6]	77.74	97.65

Anomaly Detection

Table 3. Performance Comparison of various approaches for different views and modalities on the DAD dataset.

Method	AU-ROC			
	Top-D	Top-IR	Front-D	Front-IR
DC-EPCC	0.9012	**0.8954**	**0.9087**	**0.8727**
Deep EPCC [3]	0.8859	0.8665	0.8832	0.8715
CL [14]	**0.9200**	0.8857	0.9020	0.8666
Deep SVDD [24]	0.5453	0.5881	0.6509	0.6434

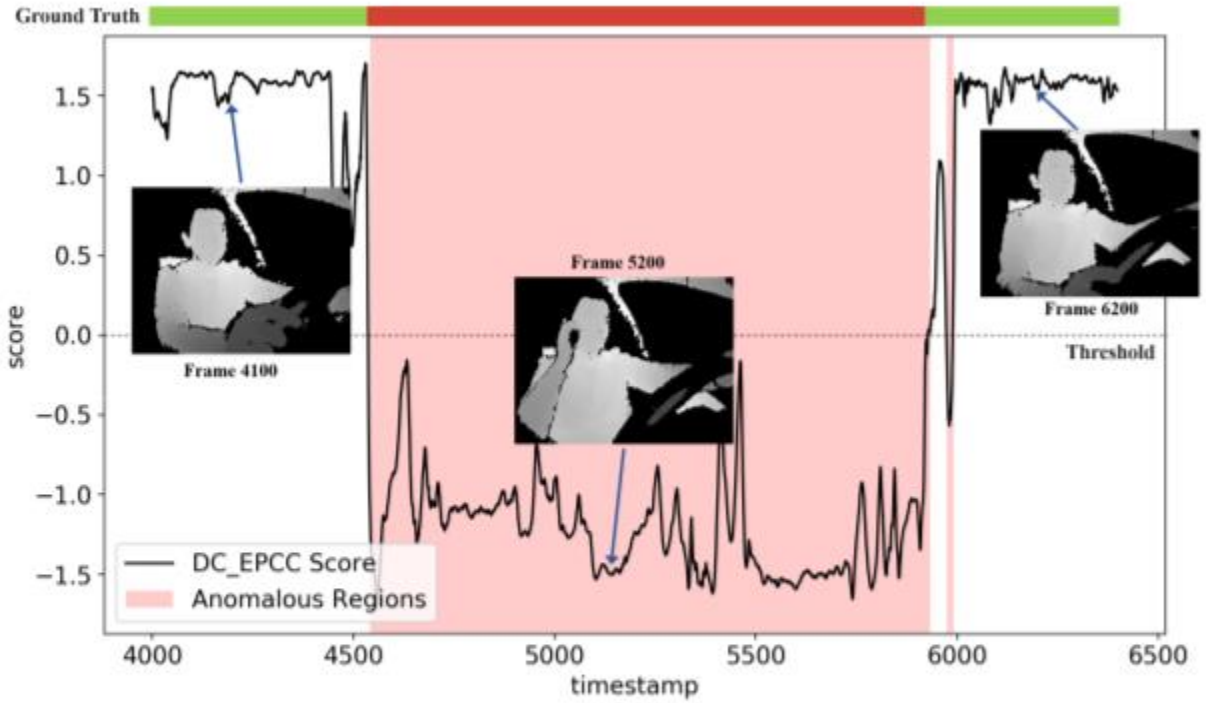


Figure 5. Illustration of recognition for a continuous video stream using front view and dept modality. The frames are classified as anomalous driving if the similarity score is blow the preset threshold. Ground truth annotation is shown on top.

Deep Compact Hyperspheres

Anomaly Detection

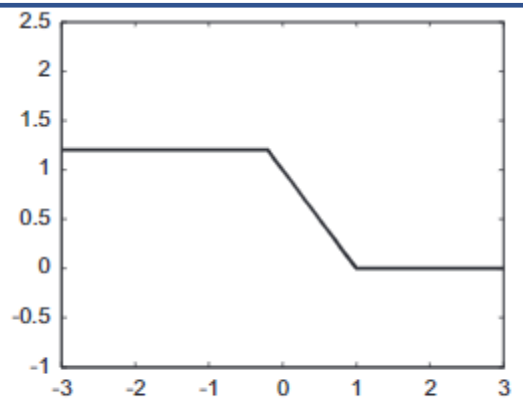
$$\mathcal{L}_{CHS} = \frac{1}{n_+}\sum_{i=1}^{n_+}\|\mathbf{f}_i - \mathbf{c}_+\|^2 + \frac{\kappa}{(n_+K)}\sum_{i=1}^{n_+}\sum_{k=1}^{K} R_s\left(\|\mathbf{f}_k - \mathbf{c}_+\|^2 - \|\mathbf{f}_i - \mathbf{c}_+\|^2\right)$$

Open Set Recognition

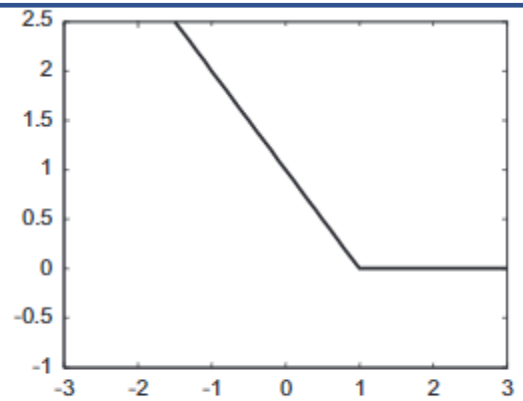
$$\mathcal{L}_{CHS} = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{f}_i - \mathbf{c}_{y_i}\|^2 + \frac{\lambda}{(nC)}\sum_{i=1}^{n}\sum_{c=1,c\neq y_i}^{C}\max\left(0, m + \|\mathbf{f}_i - \mathbf{c}_{y_i}\|^2 - \|\mathbf{f}_i - \mathbf{c}_c\|^2\right)$$
$$+ \frac{\kappa}{(nK)}\sum_{i=1}^{n}\sum_{k=1}^{K} R_s\left(\|\mathbf{f}_k - \mathbf{c}_{y_i}\|^2 - \|\mathbf{f}_i - \mathbf{c}_{y_i}\|^2\right),$$

$y_i \in \{c\}, c = 1, ..., C.$

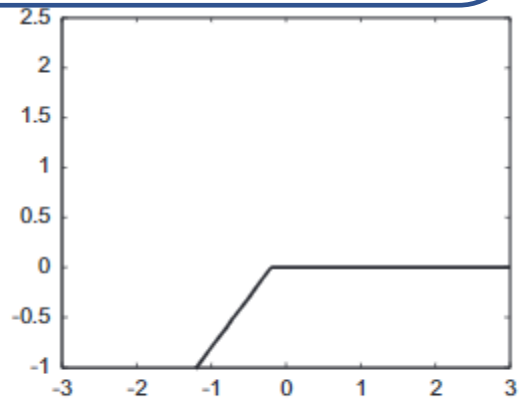
$R_s(t) = \min(m - s, \max(0, m - t))$



$R_s(t) = H_1(t) - H_s(t)$



$H_1(t) = \max(0, 1 - t)$



$-H_s(t) = -\max(0, s - t)$

Deep Compact Hyperspheres - Contributions

- Compared to open set recognition methods, we do not treat the background class as a compact and coherent class, which does not hold in practice. In contrast, we consider the background samples as diverse groups of samples that are scattered in the feature space outside the hyperspheres approximating the known classes.
- The proposed methods return semantically meaningful features as demonstrated in our experiments.
- Lastly, our proposed methods use novel loss terms that are more robust to the noisy labels within the outlier exposure and background datasets.

Experimental Results

Datasets:

Mnist, Cifar10, SVHN: By using the standard setting, Mnist, Cifar10, and SVHN datasets are split randomly into 6 known and 4 unknown classes. We used 80 Million Tiny Images dataset as background class.

Cifar+10, Cifar+50: For Cifar+N experiments, we use 4 randomly selected classes from Cifar10 dataset for training, and N non-overlapping classes chosen from Cifar100 dataset are used as unknown classes. We used 80 Million Tiny Images dataset as background class.

TinyImageNet: For TinyImageNet experiments, we randomly selected 20 classes as known classes and 180 classes as unknown classes by following the standard setting. We used 80 Million Tiny Images dataset as background class.

Experimental Results

Table 4. AUC Scores (%) of open set recognition methods on tested datasets.

Methods	Mnist	Cifar10	SVHN	Cifar+10	Cifar+50	TinyImageNet
DCHS	99.6 ± 0.1	$\mathbf{94.7} \pm 0.4$	94.5 ± 0.8	$\mathbf{99.2} \pm 0.2$	$\mathbf{98.5} \pm 0.3$	$\mathbf{83.8} \pm 1.7$
Softmax	97.8 ± 0.2	67.7 ± 3.2	88.6 ± 0.6	$81.6 \pm --$	$80.5 \pm --$	$57.7 \pm --$
OpenMax	98.1 ± 0.2	69.5 ± 3.2	89.4 ± 0.8	$81.7 \pm --$	$79.6 \pm --$	$57.6 \pm --$
G-OpenMax	98.4 ± 0.1	67.5 ± 3.5	89.6 ± 0.6	$82.7 \pm --$	$81.9 \pm --$	$58.0 \pm --$
C2AE	98.9 ± 0.2	89.5 ± 0.8	92.2 ± 0.9	95.5 ± 0.6	93.7 ± 0.4	74.8 ± 0.5
CAC	99.1 ± 0.5	80.1 ± 3.0	94.1 ± 0.7	87.7 ± 1.2	87.0 ± 0.0	76.0 ± 1.5
CPN	99.0 ± 0.2	82.8 ± 2.1	92.6 ± 0.6	$88.1 \pm --$	$87.9 \pm --$	$63.9 \pm --$
OSRCI	98.8 ± 0.1	69.9 ± 2.9	91.0 ± 0.6	$83.8 \pm --$	$82.7 \pm --$	$58.6 \pm --$
CROSR	$99.1 \pm --$	$88.3 \pm --$	$89.9 \pm --$	$91.2 \pm --$	$90.5 \pm --$	$58.9 \pm --$
RPL	98.9 ± 0.1	82.7 ± 1.4	93.4 ± 0.5	84.2 ± 1.0	83.2 ± 0.7	68.8 ± 1.4
ARPL	99.6 ± 0.1	90.1 ± 0.5	96.3 ± 0.3	96.5 ± 0.6	94.3 ± 0.4	76.2 ± 0.5
ARPL+CS	$\mathbf{99.7} \pm 0.1$	91.0 ± 0.7	$\mathbf{96.7} \pm 0.2$	97.1 ± 0.3	95.1 ± 0.2	78.2 ± 1.3
GDFRs	$--$	83.1 ± 3.9	95.5 ± 1.8	92.8 ± 0.2	92.6 ± 0.0	64.7 ± 1.2
Objecttosphere	$--$	$94.2 \pm --$	$91.4 \pm --$	$94.5 \pm --$	$94.4 \pm --$	$75.5 \pm --$
DC-EPCC	$--$	$87.2 \pm --$	$93.2 \pm --$	$--$	$--$	$--$

Robustness Against Noisy Samples

Table 5. AUC Scores (Ramp Loss/Hinge Loss/Objecttosphere Loss) and closed set accuracies of proposed DCHS on noisy datasets.

Noise Ratio	Mnist		Cifar10		SVHN	
	AUC (%)	Accuracy (%)	AUC (%)	Accuracy (%)	AUC (%)	Accuracy (%)
0%	99.6/99.5/99.3	99.7/99.7/99.7	94.7/94.2/88.5	96.3/96.3/95.2	94.5/93.4/94.8	96.7/96.3/96.7
10%	99.5/99.4/99.2	99.7/99.7/99.7	94.3/93.6/88.0	96.2/96.1/94.8	94.5/92.7/94.3	96.7/96.1/96.2
20%	99.5/99.4/99.0	99.7/99.7/99.7	94.0/93.1/87.5	95.9/95.8/94.5	94.1/92.3/94.3	96.5/96.0/96.3
30%	99.4/99.3/99.0	99.7/99.7/99.6	93.1/92.8/86.4	95.4/95.3/94.0	94.7/92.2/94.1	96.2/95.8/96.0
40%	99.4/99.3/98.9	99.7/99.6/99.6	93.0/92.3/85.5	95.1/95.0/93.8	94.2/91.8/93.4	95.9/95.2/95.7

Semantic Feature Embeddings

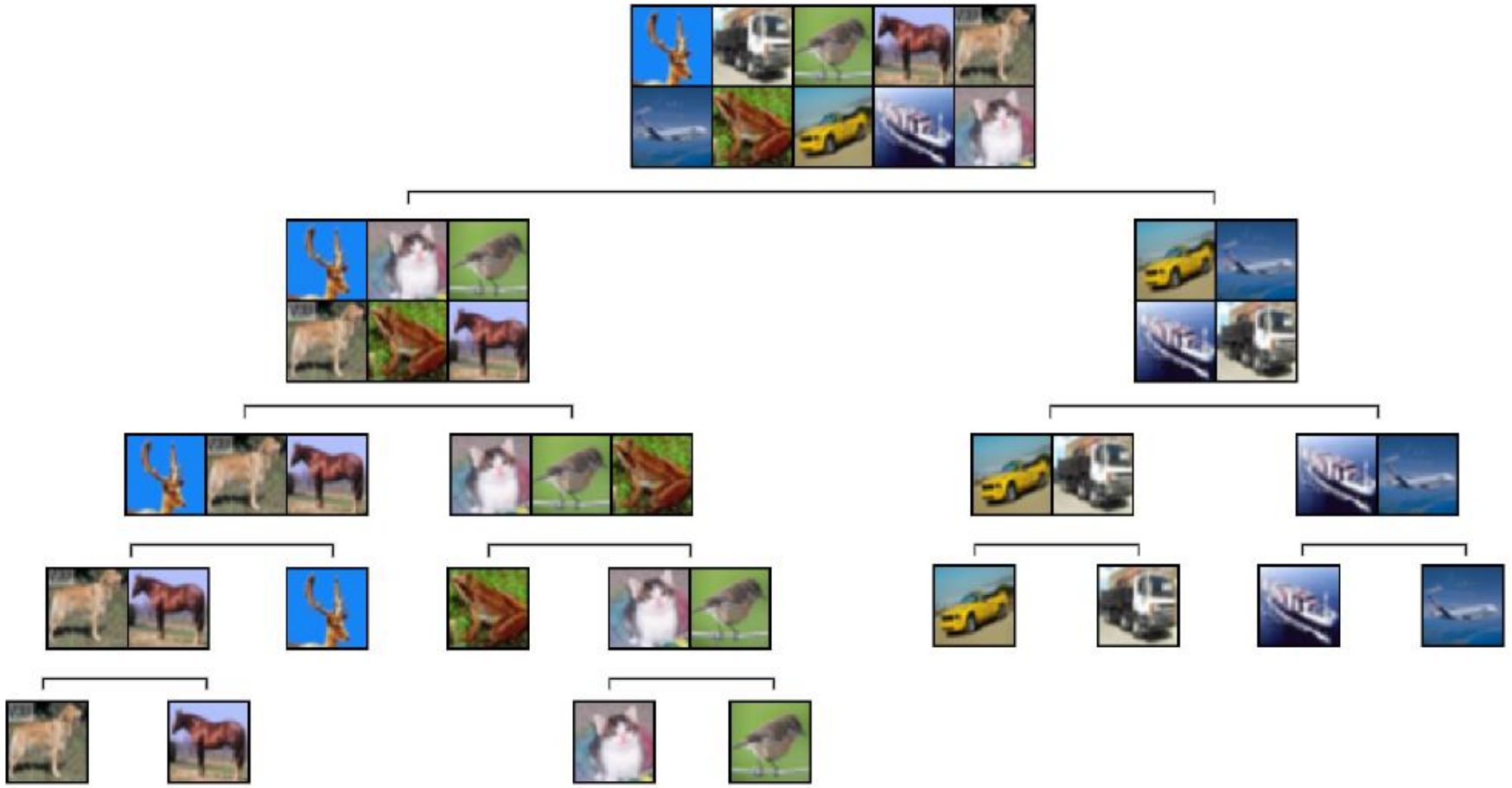


Figure 3: The hierarchical tree obtained by using NCuts clustering to the hypersphere centers returned by the proposed method. Each image represents an object class.

Reaching Nirvana: Maximizing the Margin in Both Euclidean and Angular Spaces for Deep Neural Network Classification (Deep Simplex Classifier)

- The classification loss functions used in deep neural network classifiers can be grouped into two categories based on maximizing the margin in either Euclidean or angular spaces.
- Euclidean distances between sample vectors are used during classification for the methods maximizing the margin in Euclidean spaces whereas the Cosine similarity distance is used during the testing stage for the methods maximizing margin in the angular spaces.
- In this study we proposed a novel classification loss that maximizes the margin in both the Euclidean and angular spaces at the same time. This way, the Euclidean and Cosine distances will produce similar and consistent results and complement each other, which will in turn improve the accuracies.

Deep Simplex Classifier

Contributions

- The proposed loss function does not have any hyperparameter that must be fixed for classical classification problems, therefore it is extremely easy for the users. For open set recognition, the user has to set two parameters if the background class samples are used for learning.
- The proposed method returns compact and interpretable acceptance regions for each class, thus it is very suitable for open set recognition problems.
- The distances between the samples and their corresponding centers are minimized independently of each other, thus the proposed method also works well for imbalanced datasets.

In contrast, there is only one limitation of the proposed method: The dimension of the CNN features must be larger than or equal to the total number of classes minus 1.

Deep Simplex Classifier

- In the proposed method, we map the class samples to compactly cluster around the class centers chosen from the vertices of a regular simplex. All the pair-wise distances between the selected class centers are equivalent.

The vertices of a regular simplex inscribed in a hypersphere with radius 1 can be defined as follows:

$$\mathbf{v}_j = \begin{cases} (C-1)^{-1/2}\mathbf{1}, & j = 1, \\ \kappa\mathbf{1} + \eta\mathbf{e}_{j-1}, & 2 \leq j \leq C, \end{cases}$$

where,

$$\kappa = -\frac{1+\sqrt{C}}{(C-1)^{3/2}}, \eta = \sqrt{\frac{C}{C-1}}.$$

Then, we select the class centers $\mathbf{s}_j = u\mathbf{v}_j, \quad j = 1, ..., C.$

Deep Simplex Classifier

Open Set Recognition
with background samples

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{f}_i - \mathbf{s}_{y_i}\|^2$$

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{f}_i - \mathbf{s}_{y_i}\|^2 + \lambda \sum_{i=1}^{n}\sum_{k=1}^{K} \max\left(0, m + \|\mathbf{f}_i - \mathbf{s}_{y_i}\|^2 - \|\mathbf{f}_k - \mathbf{s}_{y_i}\|^2\right)$$

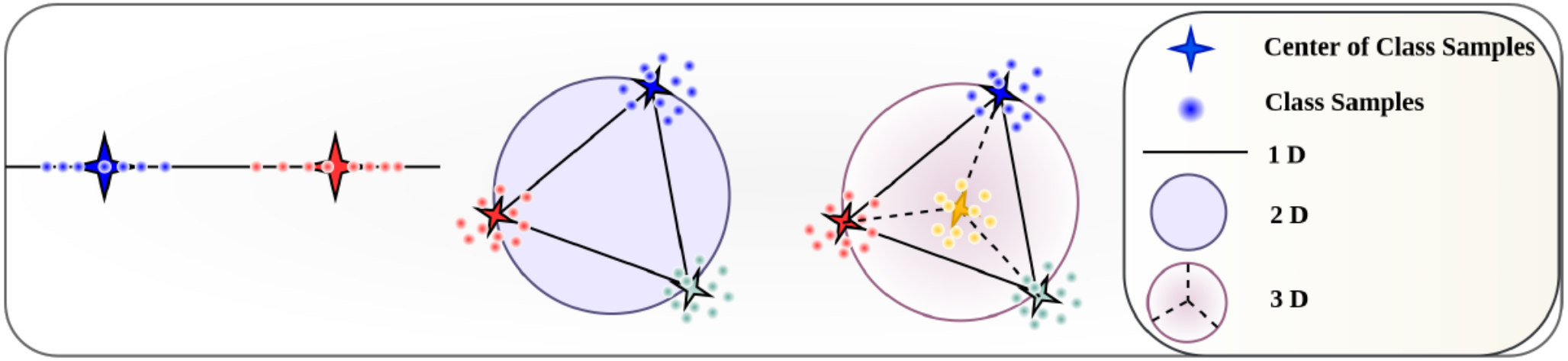
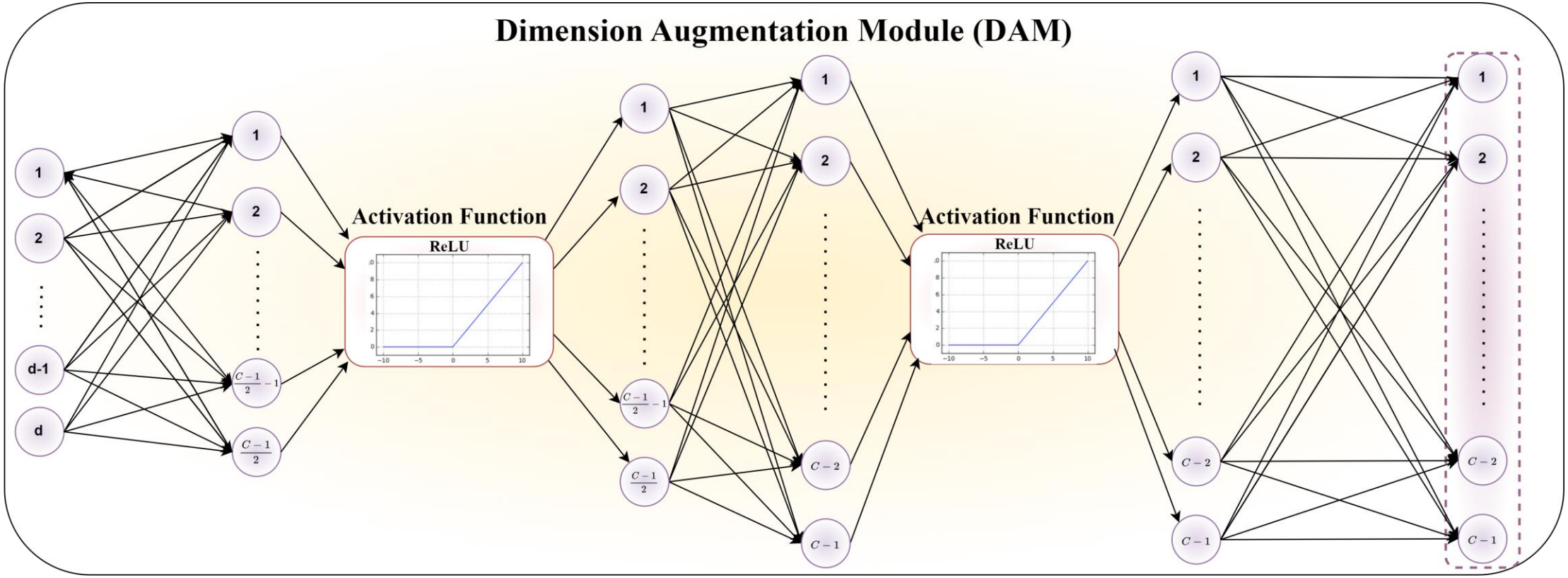


Fig. 1. In the proposed method, class samples are enforced to lie closer to the class-specific centers representing them, and the class centers are located on the boundary of a hypersphere. All the distances between the class centers are equivalent, thus there is no need to tune any margin term. The class centers form the vertices of a regular simplex inscribed in a hypersphere. Therefore, to separate C different classes, the dimensionality of the feature space must be at least $C-1$. The figure on the left shows separation of 2 classes in 1-D space, the middle figure depicts the separation of 3 classes in 2-D space, and the figure on the right illustrates the separation of 4 classes in 3-D space. For all cases, the centers are chosen from a regular $C-$simplex.

V. Papyan, X.Y. Han, and D. L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. Proceedings of the National Academy of Sciences, 117:24652–24663, 2020.
https://www.youtube.com/watch?v=AR31V1LSkOM&ab_channel=CodExSeminar

Deep Simplex Classifier – Dimension Augmentation Module



Experiments

TABLE 6
AUC Scores (%) of open set recognition methods on tested datasets ($n.r.$ stands for not reported).

Methods	Mnist	Cifar10	SVHN	Cifar+10	Cifar+50	TinyImageNet
DSC (Ours)	$\mathbf{99.6} \pm 0.1$	93.8 ± 0.3	95.3 ± 0.8	$\mathbf{99.1} \pm 0.2$	$\mathbf{98.4} \pm 0.3$	$\mathbf{82.5} \pm 1.8$
Softmax	97.8 ± 0.2	67.7 ± 3.2	88.6 ± 0.6	$81.6 \pm n.r.$	$80.5 \pm \pm n.r.$	$57.7 \pm n.r.$
OpenMax	98.1 ± 0.2	69.5 ± 3.2	89.4 ± 0.8	$81.7 \pm n.r.$	$79.6 \pm n.r.$	$57.6 \pm n.r.$
G-OpenMax	98.4 ± 0.1	67.5 ± 3.5	89.6 ± 0.6	$82.7 \pm n.r.$	$81.9 \pm n.r.$	$58.0 \pm n.r.$
C2AE	98.9 ± 0.2	89.5 ± 0.9	92.2 ± 0.9	95.5 ± 0.6	93.7 ± 0.4	74.8 ± 0.5
CAC	99.1 ± 0.5	80.1 ± 3.0	94.1 ± 0.7	87.7 ± 1.2	87.0 ± 0.0	76.0 ± 1.5
CPN	99.0 ± 0.2	82.8 ± 2.1	92.6 ± 0.6	$88.1 \pm n.r.$	$87.9 \pm n.r.$	$63.9 \pm n.r.$
OSRCI	98.8 ± 0.1	69.9 ± 2.9	91.0 ± 0.6	$83.8 \pm n.r.$	$82.7 \pm -$	$58.6 \pm n.r.$
CROSR	$99.1 \pm n.r.$	$88.3 \pm n.r.$	$89.9 \pm n.r.$	$91.2 \pm n.r.$	$90.5 \pm n.r.$	$58.9 \pm n.r.$
RPL	98.9 ± 0.1	82.7 ± 1.4	93.4 ± 0.5	84.2 ± 1.0	83.2 ± 0.7	68.8 ± 1.4
GDFRs	$n.r.$	83.1 ± 3.9	$\mathbf{95.5} \pm 1.8$	92.8 ± 0.2	92.6 ± 0.0	64.7 ± 1.2
Objecttosphere	$n.r.$	$\mathbf{94.2} \pm n.r.$	$91.4 \pm n.r.$	$94.5 \pm n.r.$	$94.4 \pm n.r.$	$75.5 \pm n.r.$

Methods	Mnist	Cifar10	SVHN	Cifar+10	Cifar+50	TinyImageNet
DCHS	99.6 ± 0.1	$\mathbf{94.7} \pm 0.4$	94.5 ± 0.8	$\mathbf{99.2} \pm 0.2$	$\mathbf{98.5} \pm 0.3$	$\mathbf{83.8} \pm 1.7$

Summary and Conclusion

- Overall, open set recognition is an interesting and growing research area.
- The classifiers that return compact acceptance regions are successful in open set recognition problems.
- The classifiers that are simple and having less number of hyperparameters are desired in such problems.
- Incorporating background samples needs attention and the results re significantly improved when this is done in a correct way.

Thank you for listening.

Any questions?