MeetSafe
enhancing robustness against white-box adversarial examples

Stenhuis, Ruben; Liu, Dazhuang; Qiao, Yanqi; Conti, Mauro; Panaousis, Manos; Liang, Kaitai

DOI
10.3389/fcomp.2025.1631561

Publication date
2025

Document Version
Final published version

Published in
Frontiers in Computer Science

Citation (APA)
Stenhuis, R., Liu, D., Qiao, Y., Conti, M., Panaousis, M., & Liang, K. (2025). MeetSafe: enhancing robustness against white-box adversarial examples. *Frontiers in Computer Science*, *7*, Article 1631561. https://doi.org/10.3389/fcomp.2025.1631561

TYPE Methods
PUBLISHED 13 August 2025
DOI 10.3389/fcomp.2025.1631561

Check for updates

OPEN ACCESS

EDITED BY
Christos Xenakis,
University of Piraeus, Greece

REVIEWED BY
Christoforos Ntantogian,
Ionian University, Greece
Vaios Bolgouras,
Unisystems, Luxembourg

*CORRESPONDENCE
Dazhuang Liu
✉ d.liu-8@tudelft.nl

RECEIVED 19 May 2025
ACCEPTED 16 July 2025
PUBLISHED 13 August 2025

CITATION
Stenhuis R, Liu D, Qiao Y, Conti M, Panaousis M and Liang K (2025) MeetSafe: enhancing robustness against white-box adversarial examples.
Front. Comput. Sci. 7:1631561.
doi: 10.3389/fcomp.2025.1631561

MeetSafe: enhancing robustness against white-box adversarial examples

Ruben Stenhuis[1], Dazhuang Liu[1]*, Yanqi Qiao[1], Mauro Conti[2], Manos Panaousis[3] and Kaitai Liang[1]

[1]Cybersecurity Group, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, Netherlands, [2]Department of Mathematics, SPRITZ Security and Privacy Research Group, University of Padua, Padua, Italy, [3]Faculty of Engineering and Science, School of Computing and Mathematical Sciences, Center for Sustainable Cyber Security, University of Greenwich, London, United Kingdom

Convolutional neural networks (CNNs) are vulnerable to adversarial attacks in computer vision tasks. Current adversarial detections are ineffective against white-box attacks and inefficient when deep CNNs generate high-dimensional hidden features. This study proposes MeetSafe, an effective and scalable adversarial example (AE) detection against white-box attacks. MeetSafe identifies AEs using critical hidden features rather than the entire feature space. We observe a non-uniform distribution of Z-scores between clean samples and adversarial examples (AEs) among hidden features and propose two utility functions to select those most relevant to AEs. We process critical hidden features using feature engineering methods: local outlier factor (LOF), feature squeezing, and whitening, which estimate feature density relative to its k-neighbors, reduce redundancy, and normalize features. To deal with the curse of dimensionality and smooth statistical fluctuations in high-dimensional features, we propose local reachability density (LRD). Our LRD iteratively selects a bag of engineered features with random cardinality and quantifies their average density by its k-nearest neighbors. Finally, MeetSafe constructs a Gaussian Mixture Model (GMM) with the processed features and detects AEs if it is seen as a local outlier, shown by a low density from GMM. Experimental results show that MeetSafe achieves 74%, 96%, and 79% of detection accuracy against adaptive, classic, and white-box attacks, respectively, and at least 2.3× faster than comparison methods.

KEYWORDS

adversarial attack, convolutional neural network, Gaussian Mixture Model, adversarial example, local reachability density

1 Introduction

Deep neural networks (DNNs) have emerged as highly effective models in machine learning (ML) tasks. Among DNNs, convolutional neural networks (CNNs) revolutionized various computer vision applications, such as medical image recognition (Litjens et al., 2017) and facial recognition (Zhao et al., 2003). However, the robustness of CNNs remains a significant concern, as even a slight and imperceptible perturbations deliberately designed to manipulate images can result in high misclassification rates (Szegedy et al., 2014). Therefore, adversarial detections are in urgent demand to guarantee the integrity of CNN models.

A plethora of adversarial detections (Feinman et al., 2017; Hu et al., 2019; Hendrycks and Gimpel, 2017; Raghuram et al., 2021; Ma et al., 2018; Aldahdooh et al., 2022) have been proposed to identify adversarial examples (AEs). However, these methods remain vulnerable to white-box adversarial attacks (Carlini and Wagner, 2017a; Athalye et al., 2018; Athalye and Carlini, 2018; Tramer et al., 2020), which assume full access to the model and training process. Several defenses (Raghuram et al., 2021; Hu et al., 2019) have been developed against white-box AEs. They obscure the detector's gradients, leading to: (1) diminished security, as gradient obfuscation is proven to be an ineffective strategy for enhancing robustness (Athalye et al., 2018); and (2) inefficient for large CNNs, as computing exact gradients becomes prohibitively expensive.

It has been reported (Hendrycks and Gimpel, 2017; Aldahdooh et al., 2022) that the integration of multiple detections to limit adversary capabilities, a strategy termed *"meet the defense"*, is promising in countering adversarial attacks. However, these studies did not include any implementation or experimental results. Indeed, exploiting synergistic effects of multiple detections is challenging due to their ineffectiveness. For example, certified methods (Weng et al., 2018; Raghunathan et al., 2018), a widely studied adversarial defense that employs minimum distance decoding (Tramer, 2022) for AEs detection, are generally effective only for AEs with small ℓ_p distances from clean samples. This limitation renders them ineffective against semantically stealthy adversarial examples (Ghiasi et al., 2020), which achieve substantially large but visually imperceptible perturbations by manipulating image factors such as color or shadows. As such, Ghiasi et al. (2020) show that any perturbation on semantic attributes such as shadows is as effective as contrived noise. However, the vast number of semantics in images renders supervised detection inadequate for adversarial attacks (Zheng and Hong, 2018) due to its limited generalization and dependence on patterns specific to the existing dataset.

Meanwhile, many effective adversarial defenses (Zheng and Hong, 2018; Feinman et al., 2017) fail to scale efficiently as CNNs deepen and their number of parameters increases. For instance, the full covariance matrix Σ in I-Defender (Zheng and Hong, 2018) scales as $\mathcal{O}(d^2)$ w.r.t. the input dimension d of the hidden features extracted by CNN. Similarly, an increase in the number of features exponentially reduces the efficiency of Euclidean distance computations, as noted by Feinman et al. (2017), due to the curse of dimensionality. The complexity of distance calculation also impacts density-based outlier detection methods, such as local outlier factor (LOF) (Breunig et al., 2000), which require repeated distance evaluations between data points and their neighbors in the feature space.

This study proposes MeetSafe, a scalable and effective detection for strong white-box adversarial attacks. MeetSafe selects critical hidden features obtained by convolutional layers, applies feature engineering techniques, and utilizes a Gaussian Mixture Model (GMM) to estimate their distribution. AEs are then identified by the GMM as outliers as they deviate from the distribution of benign hidden features.

In detail, we first observe that the Z-scores of hidden features from selected neurons are non-uniformly distributed (see Figure 1b) in each CNN layer, with not all layers actively extracting features from AEs (see Figure 3a–d). We propose two utility functions to identify the layers most sensitive to adversarial perturbations and the neurons with the largest Z-score differences between benign and adversarial features. By leveraging only the hidden features from the selected neurons, we significantly reduce the feature dimension. Then, our GMM estimates the distribution of selected features processed by three feature engineering techniques: feature squeezing (Xu et al., 2018), which compares the model's predictions on the original and feature-squeezed inputs;whitening (Hendrycks and Gimpel, 2017), captures the principal component of the covariance of inputs; and LOF, which estimates the sparsity of images based on their neighbors in the processed feature space. LOF is ineffective and inefficient in high-dimensional spaces as increased sample distances reduce critical feature impact and raise computational costs for density estimation. To enhance LOF's scalability in high-dimensional feature spaces and reduce statistical fluctuations for improved precision, we propose reachability density (LRD) for local outlier detection. LRD iteratively selects feature subsets with random cardinality and estimate the density of images based on their k-nearest neighbors in the feature space. Finally, an AE is identified if its sparsity, as estimated by the GMM, exceeds the 90th percentile. Experimental results on real-world datasets show that MeetSafe attains a 74%+ detection accuracy against adaptive adversaries, 96%+ against classic adversarial attacks, 79%+ accuracy under white-box attacks, and at least 2.3× faster speed.

2 Related work

2.1 Notation

A deep neural network can be expressed as the mapping function $f^{(l)}(X): \mathbb{R}^m \rightarrow \mathbb{R}^L$, where the hidden units at layer l are $f_X^{(l)} \in \mathbb{R}^L$ for the input $X \in \{\mathfrak{D} \mid \mathfrak{D} \subseteq \mathbb{R}^m\}$ in dataset $\mathfrak{D}$. For simplicity, we define units of the last layer of this network (*i.e.,* logits) to be $z_i \in Z(X)$ and the predictions to be $y_i \in Y(X)$. Neural networks often minimize the empirical risk with a loss function $\mathcal{L}_f(\mathbf{X})$ with a batch of $\mathbb{R}^{B\times m}$ as input. Our method also utilizes GMMs for detection, which is a linear superposition of Gaussians with the form $\mathcal{N}(X|\mu_i, \Sigma_i)$ where Σ and μ denote its covariance matrix and mean, respectively. Each Gaussian has a mixing coefficient π_i that equals the probability $p(\xi_i)$ of a latent variable ξ_i.

2.2 Adversarial attacks

One can define at least three threat models for adversarial attacks: the white-, gray-, and black-box scenario. The white-box setting indicates that the adversary has perfect information about the system. The detector should thus be deterministic for the adversary (Athalye et al., 2018). A weaker assumption is a gray-box model in which the attacker has no knowledge about the defenses. Black-box attacks only assume knowledge of the output and input space, possibly with access to a querying oracle. The empirical risk of an actual threat is often measured with the ℓ_p-norm required by adversarial attacks like the ones below.

2.2.1 Fast gradient sign method (FGSM) (Goodfellow et al., 2015)

FGSM is a one-step ℓ_∞ perturbation toward the gradient of the loss function $\nabla_X \mathcal{L}_f$. FGSMs perturbation is $\epsilon \cdot sign(\nabla_X \mathcal{L}_f)$, where ϵ is the ℓ_∞ norm of the perturbation. The method assumes linearity in the proximate region of sample X.

2.2.2 Carlini & Wagner (C&W) (Carlini and Wagner, 2017b)

C&W is a first-order constrained optimization that closely resembles Szegedy et al. (2014) method for adversarial example generation. Both define the objective to be $||X||_p + c \cdot \hat{f}(X)$. This objective function includes the ℓ_p distance with a custom criterion $\hat{f}(X)$, modulated by the sensitivity parameter c and confidence parameter κ. C&W uses $\hat{f}(X) = (\max_{i \neq t}(z_i) - z_t + \kappa)^+$ where t is the targeted class.

2.2.3 DeepFool (Moosavi-Dezfooli et al., 2016)

DeepFool fits a hyperplane on the target model. The hyperplane is an aggregate of binary classifiers, which encloses the true class k. The algorithm applies Newton's method on the probits to move to the closest non-maximal class t. To misclassify the sample, a small overshoot η is added as scalar.

We describe the perturbations generated by the three methods as near-optimal as they are optimized within the constraints of the ℓ_p-ball. However, recent studies on semantic perturbations have identified approaches that produce adversarial examples more closely aligned with human perception (Luo et al., 2022; Duan et al., 2021; Zhao et al., 2020; Ghiasi et al., 2020). For instance, PerC uses color differences, which considerably increases the ℓ_p distance of adversarial examples. The primary focus in this study is on adaptive near-optimal perturbations on state-of-the-art defenses that *do not* rely on obfuscated gradients.

2.3 Adversarial detection

2.3.1 Adversarial pockets

A common intuition of adversarial perturbation is that it pushes examples off the manifold of training data. Szegedy et al. (2014) were the first to conjecture the idea with the Lipschitz constant. A high constant enables the manifold to be dense, with low-probability pockets containing adversarial examples. Therefore, generative classifiers may detect these adversarial pockets (Lee et al., 2018; Raghuram et al., 2021; Yin et al., 2019; Feinman et al., 2017; Zheng and Hong, 2018; Li et al., 2019). An example of this is Deep Bayes (Li et al., 2019), which uses a deep latent variable model on the logits to estimate a joint distribution. JTLA (Raghuram et al., 2021) aggregates class-conditional probabilities from each layer by computing kNN class counts. Others trained a more simple GMM (Zheng and Hong, 2018) and utilized Kernel Density Estimation (KDE) (Feinman et al., 2017) on deep layers. Lee et al. (2018) performed a density estimation with the Mahalanobis distance.

2.3.2 Boundary tilting

A geometric analysis renders a different perspective on adversarial examples. When the decision boundary tilts too much toward a submanifold of one class, then the distance of another classification is relatively close. Tanay and Griffin (2016) therefore measured adversarial strength as the deviation angle with a bisecting boundary that maximizes the inter-class distance. This angle can, without major performance hits, be higher along directions of low variance. Near-optimal perturbations may thus be detected by manipulating such components with semantic-preserving image filters (Xu et al., 2018; Tian et al., 2021; Liang et al., 2018). In particular, feature squeezing (Xu et al., 2018) uses median smoothing and bit-depth reduction. Tian et al. (2021) train a dual model on the sample's wavelet transform. Others (Song et al., 2018; Hu et al., 2019) propose denoisers which perturb samples with optimizers. Scene statistics may also detect the perturbation, like whitening (Hendrycks and Gimpel, 2017) that measures the variance of low-rank eigenvectors. Li and Li (2017) also use low-rank eigenvectors with their extremal value to detect extreme deviations, both (Kherchouche et al., 2020; Akhtar et al., 2018) train simple classifiers on BRISQUE's (Mittal et al., 2012) features, and Local Intrinsic Dimensionality (LID) (Ma et al., 2018) directly calculates the dimensionality. However, current adversarial detection methods are ineffective at identifying hidden anomalies in high-dimensional spaces and are not efficient for large dataset.

Contributions of this study are as follows: (**i**) We propose MeetSafe, a scalable detection algorithm for adaptive adversarial examples. (**ii**) Two utility functions that allow LRD and other detectors to scale based on a unit's Z-scores or rate of change under perturbation. (**iii**) Extensive empirical evaluations on 4 datasets and 14 models that show effectiveness of whitening and MeetSafe under adaptive white-box attacks.

3 Method

The main idea of MeetSafe is to combine discrepant detectors in an ensemble. In particular, we use the scores of whitening (Hendrycks and Gimpel, 2017), feature squeezing (Xu et al., 2018), and a density estimation, called LRD, within a GMM. LRD makes two novel improvements on existing density estimates. First, we noticed that the activation's Z-score of hidden features is not uniform under perturbation (see Figure 1b); we therefore use two utility functions to select the 10 units that were most anomalous under perturbations. Second, kernel density estimation does not adjust for local densities, which carries the risk of over-smoothing as illustrated by Ma et al. (2018). Like Ma et al., LRD uses an extension of the k-distance.

We now turn to LRD and its relation to non-parametric methods. Then, we explain the used features and feature selection of LRD. Finally, this section introduces MeetSafe.

3.1 Density estimation with k-distances

Non-parametric methods model the distribution $p(X)$ with limited assumptions for the true distribution. This makes the models flexible. Distribution $p(X)$ can, for instance, be generalized

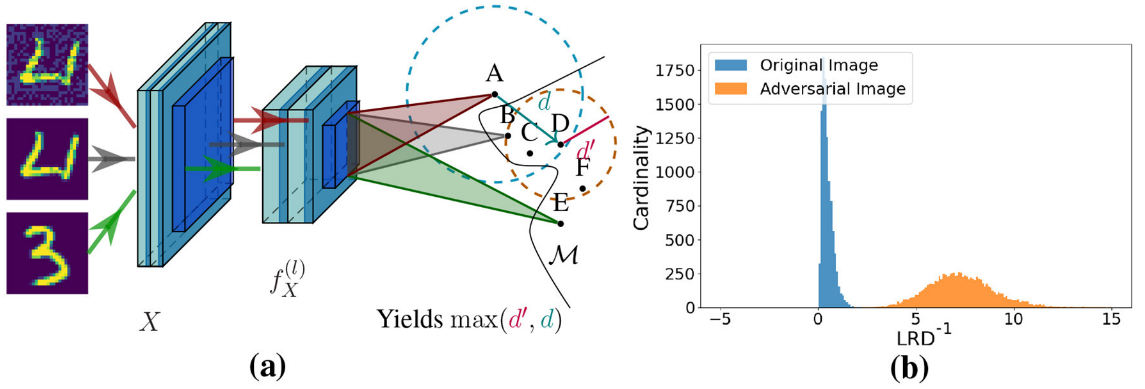

FIGURE 1
LRD for a MNIST model learned with RCE, utilizing the 10 hidden units with the largest, absolute Z-scores. **(a)** LRD metric, which uses the averaged maximum of the ℓ_2- (d) and k-distance (d') among k nearest neighbors. **(b)** The distribution for test- and FGSM samples with $\ell_2 \approx 5$ and $k = 8$.

with its volume V and cardinality K of X's proximite region, given enough observations. In contrast to KDE, the kNN method fixes the cardinality and finds the appropriate volume from the data. For a sample X, one can then estimate $p(X)$ using the frequentist notion:

$$\hat{p}(X) = K/(|\mathfrak{D}| \cdot V) \tag{1}$$

where the dataset $\mathfrak{D}$ is sampled from $p(X)$. The volume is defined by a sphere with the k-distance as radius, which makes the k-distance of a test sample X sufficient to approximate $p(X)$. The estimate would only be shallow with limited information from its neighbors. Recursive calls on neighbors may improve the estimate due to greater depth.

Notice that BRISQUE hidden features may be affected by unequal standard deviations as these are not normalized. Some have thus more influence on the k-distance than others. This is not favorable, especially because we stated earlier that the low variance components may be an important characteristic of some adversarial examples. Our method will therefore use the scaled Euclidean distance. This normalizes the k-distance with respect to a diagonal covariance matrix Σ. In addition, we will lower the memory burden of the kNN algorithm in Section 3.2, after we discuss the details and idea of LRD.

3.1.1 Local reachability density (LRD)

The intuition behind LRD comes from Breunig et al. (2000), in which they propose LOF, a heuristic for finding local outliers. LRD extends the k-distance as it smooths out statistical fluctuations in at least two ways. First, the actual distance, called reachability, used to estimate the volume V is capped by the k-distance of the neighbor. Second, the average is taken among the k nearest neighbors. The reachability measure $reach(X_1, X_2)$ of two nodes (A & D) is demonstrated in Figure 1, and the value depends on the volume of node D and its Euclidean distance to A. Whichever value is bigger equals the reachability from D to A:

$$\max\left\{d'_{X_2,k}, \sqrt{(X_1 - X_2)^T \Sigma^{-1} (X_1 - X_2)}\right\} \tag{2}$$

where d' is the k-distance. Substituting the reachability from A to its neighbors $\mathbf{N}_A$ in Equation 1 yields its reachability density (Equation 3). Using reachability as a measure to assess the density in the proximate region of node A has the advantage that only a fixed amount of neighbors needs to be considered.

$$LRD(X) = |\mathbf{N}_X| / (\textstyle\sum_{n_i \in \mathbf{N}_X} reach(X, n_i)) \tag{3}$$

We add one novel improvement called feature bagging (Lazarevic and Kumar, 2005). This enables LRD to capture higher dimensions. The generalizability of kNN degrades under these circumstances, as the distance between all data points becomes larger and individual features have less of an impact. Bagging is a popular approach to limit this issue. It takes a subset (with random cardinality) of the features for multiple iterations and returns a combined LRD score.

3.2 Feature engineering for LRD

We now consider two possible Points Of Interests (POI) for LRD: the layer after convolution and the pixel values, where we refer to the former as Learned Feature Analysis (LFA). Specifically, we explain how we select its features for both options as without limiting its feature space, LRD would be space inefficient and suffer from sparse data.

On raw pixel values, we advise the use of BRISQUE. BRISQUE fits a Gaussian-like distribution on the raw image while maintaining structural information, which can evaluate the naturalness of an image. Moreover, BRISQUE is 149 times faster than wavelet methods such as DIIVINE and performs almost similar on white noise (Mittal et al., 2012).

On hidden layers, we extract a random set of hidden features. Depending on the chosen POI, the Z-scores of FGSM examples X' will be used to select the best 10 features of BRISQUE or the best 10 hidden features. We will further explain this feature selection more formally.

3.2.1 Selecting hidden features

For the hidden features $f^{(l)}$, a pool $\mathcal{P} \subseteq_R f^{(l)}$ is defined, so that its members are chosen randomly at initialization and preserved during execution. The detector then follows a watching scheme upon $\mathcal{P}$ and its utilities (Equation 4, 5). The first equation calculates the difference in Z-scores of all features in the pool. It estimates the units that were relevant under perturbation. The second estimates the rate of change of one layer. The first utility is calculated with FGSM after every epoch, and this is the fastest evasion method we know, limiting the constraints on scalability or parameter updates. The second utility showed most potential in the final layers, making that our preferred choice (Section 5.1). Because of this, we believe the second utility is optional.

$$U_{\mathcal{P}} = \frac{1}{|\mathfrak{D}|}\sum\nolimits_{X\in\mathfrak{D}} \frac{|\mathcal{P}_X - \mathcal{P}_{X'}|}{\sigma_{\mathcal{P}}} \tag{4}$$

$$U_{f^{(l)}} = \frac{||\mathbb{E}_{X\sim\mathfrak{D}}[f^{(l)}_{X'}] - \mathbb{E}_{X\sim\mathfrak{D}}[f^{(l)}_{X}]||_2}{||\mathbb{E}_{X\sim\mathfrak{D}}[f^{(l)}_{X}]||_2} \tag{5}$$

3.3 MeetSafe

Our MeetSafe combines LRD, using hidden features as POI, with variance-based anomaly detection—whitening (Hendrycks and Gimpel, 2017)—and feature squeezing (Xu et al., 2018) in a GMM, for which an ablation study is given in Section 4.3. The scores of the three heuristics are learned through Expectation-Maximization (EM).

$$-\log \hat{p}(X) = -\log\left\{\sum\nolimits^{K} \pi_i \cdot \mathcal{N}(\mathcal{H}_X|\mu_i, \Sigma_i)\right\} \tag{6}$$

Concretely, to detect a sample, we first select the best features based on $U_{\mathcal{P}}$ for LRD and the eigenvectors of the training data (Algorithm 1). Then, we evaluate the three heuristics (denoted as $\mathcal{H}_X$). Assuming normality, a three-dimensional GMM fitted on benign data can classify the sample as malicious when it exceeds the 90th percentile of the Equation 6. The workflow of MeetSafe is shown in Algorithm 2.

4 Experiments

We evaluate MeetSafe and LRD against several adversarial attacks including FGSM, DeepFool, and C&W. The experiments will demonstrate white- and/or gray-box performance for four datasets: Tiny-ImageNet (Le and Yang, 2015), CIFAR-10 (Krizhevsky and Hinton, 2009), MNIST (LeCun, 1998), and STL-10 (Coates et al., 2011). Only for Tiny-ImageNet, we resized the samples to be in $\mathbb{R}^{3\times64\times64}$. The attacks are restricted to an ℓ_2 distance to ensure a fair comparison across datasets and ℓ_∞-based methods. For instance, an ℓ_∞ distance permits more noise for higher resolution images. Consequently, we use the ϵ parameter given by Equation 7, so that the maximum allowed perturbation of FGSM equals that of ℓ_2 methods ($\delta_{\max}$); where the image X is given by a $\mathbb{R}^m$ flattened matrix.

$$\epsilon = \sqrt{||\delta_{\max}||_2^2/m} \tag{7}$$

```
Require: $\mathfrak{D}$: Dataset of benign samples; $\mathcal{P}$: Set of
    basis functions that maximizes $U_{\mathcal{P}}$; $\mathcal{P}_{max}$: Maximum
    amount of features; $X$: A benign or adversarial
    sample.
Ensure: $\mathcal{H}_{\mathcal{X}}X$'s Features
if $kNN$ or SVD is not initialized then    ▷ Prepare heuristics
    $kNN \leftarrow Prepare\_kNN(f^{(l)}_{\mathfrak{D}})$ where $\{f^{(l)}\} \in \mathcal{P}$
    $\mathbf{U_t}, \mathbf{S_t}, \mathbf{V_t^T} \leftarrow SVD(\mathfrak{D})$    ▷ Truncated to $\mathcal{P}_{max}$
    for $X \in \mathfrak{D}$ do
        $X' \leftarrow X + \epsilon \cdot sign(\nabla_X \mathcal{L}_f)$
        $\mathcal{P}^*_X \leftarrow X\mathbf{V_t}$
        $\mathcal{P}^*_{X'} \leftarrow X'\mathbf{V_t}$
    end for
    $U_{\mathcal{P}^*} \leftarrow \frac{1}{|\mathfrak{D}|}\sum_{X\in\mathfrak{D}} \frac{|\mathcal{P}^*_X - \mathcal{P}^*_{X'}|}{\sigma_{\mathcal{P}^*}}$    ▷ Equation 4
    $\mathbf{V_t} \leftarrow top\_n(\mathcal{P}_{max}, P^*)$    ▷ Pick the best eigenvectors
end if
$\mathbf{N}_X \leftarrow kNN(f^{(l)}_X)$ where $\{f^{(l)}\} \in \mathcal{P}$
$\Sigma \leftarrow Diag(\sigma_{\mathcal{P}})$
$H_0 \leftarrow \frac{1}{LRD(f^{(l)}_X, \mathbf{N}_X, \Sigma)}$ where $\{f^{(l)}\} \in \mathcal{P}$ ▷ Equation 2, 3,
    with feature bagging
$X_{bd}, X_{bl} \leftarrow Reduce\_Bit\_Depth(X),\ Blur(X)$
$H_1 \leftarrow \max\left\{||Y(X) - Y(X_{bd})||_1, ||Y(X) - Y(X_{bl})||_1\right\}$
$H_2 \leftarrow Var(X\mathbf{V_t})$    ▷ Whitening Hendrycks and Gimpel (2017)
return $\{H_0, H_1, H_2\}$    ▷ $H_1$ is Feature Squeezing Xu et al. (2018)
```

Algorithm 1. MeetSafe's feature extraction.

Each attack, defense, and target model is re-implemented in PyTorch. Herewith, we evaluated 14 models based on ResNet-50 (He et al., 2016) and VGG-13 (Simonyan and Zisserman, 2015). Ten of them are trained with robust optimization techniques, utilizing gradient smoothing (RCE) (Pang et al., 2018) or adversarial training with FGSM (ℓ_2-radii of 5) (AL) (Goodfellow et al., 2015).

The experiments also include some related methods that will be compared to MeetSafe and LRD. The baseline for MeetSafe (MS) is KDE with predictive uncertainty (KDE+BU) (Feinman et al., 2017), I-Defender (I-Def) (Zheng and Hong, 2018), and the Mahalanobis measure (MAH) (Lee et al., 2018). Additional work that we tested are LID (Ma et al., 2018), Whitening (PCA) (Hendrycks and Gimpel, 2017), Feature Squeezing (FSQ) (Xu et al., 2018), extremal value (EXM) (Li and Li, 2017), and Kherchouche et al. (2020)'s third model for BRISQUE (SVM).

4.1 Experimental setup

We trained the models for 150 epochs on predetermined training sets. During training, a batch size was used of 256, learning rate of 0.01 with momentum 0.9 under a cosine annealing schedule, and 1e−4 weight decay. The model is also adapted to exhibit required invariances. All training samples are normalized

```
Require: $\mathfrak{D}$: dataset of benign samples; $\mathcal{P}_{max}$: maximum
    amount of features; $K, \tau_{90}$: Gaussian components
    and threshold; $f^{(l)}$: basis function of the neural
    network; $\mathbf{X}'$: suspicious samples.
Ensure: $M_{\mathbf{X}'}$: MeetSafe classifications.
$U_{f^{(l)}} \leftarrow \mathbf{0}$
for $X \in \mathfrak{D}$ and basic block $l$ do   ▷ Get the utility of each layer (optional)
    $X' \leftarrow X + \epsilon \cdot sign(\nabla_X \mathcal{L}_f)$
    $U_{f^{(l)}} \leftarrow Sequential\_Avg(U_{f^{(l)}}, f^{(l)}_{X'}, f^{(l)}_{X})$   ▷ Equation 5
end for
Let $\{l\}$ have the largest $U_{f^{(l)}}$ value and let
    $\{\mathcal{P}\} \subseteq_R f^{(l)}$.
for $X \in \mathfrak{D}$ do   ▷ Get the utility of each hidden unit in $\mathcal{P}$
    $X' \leftarrow X + \epsilon \cdot sign(\nabla_X \mathcal{L}_f)$
    $\mathcal{P}_X \leftarrow f^{(l)}_X$ where $\{f^{(l)}\} \in \mathcal{P}$
    $\mathcal{P}_{X'} \leftarrow f^{(l)}_{X'}$ where $\{f^{(l)}\} \in \mathcal{P}$
end for
$U_{\mathcal{P}} \leftarrow \frac{1}{|\mathfrak{D}|} \sum_{X \in \mathfrak{D}} \frac{|\mathcal{P}_X - \mathcal{P}_{X'}|}{\sigma_{\mathcal{P}}}$   ▷ Equation 4
$\mathcal{P} \leftarrow top\_n(\mathcal{P}_{max}, \mathcal{P})$   ▷ Pick the best hidden units
for $X \in \mathfrak{D}$ do   ▷ Initialize the GMM
    Initialize $\{\mu_i\} \in \mathbb{R}$ via the K-means algorithm
and
        $\{\pi_i, \Sigma_i\} \in \mathbb{R}, \mathbb{R}^{3\times3}$ uniformly at random.
    $\mathcal{H}_X \leftarrow Extract\_Features(\mathfrak{D}, \mathcal{P}, \mathcal{P}_{max}, X)$   ▷ Algorithm 1
end for
$\mu_i, \Sigma_i, \pi_i \leftarrow EM(\mu_i, \Sigma_i, \pi_i, \mathcal{H}_X) \quad i \in [0..K), \quad \forall X \in \mathfrak{D}$
for $X' \in \mathbf{X}'$ do   ▷ Classify samples
    $\mathcal{H}_{X'} \leftarrow Extract\_Features(\mathfrak{D}, \mathcal{P}, \mathcal{P}_{max}, X')$
end for
return $-\log\left\{\sum^K \pi_i \cdot \mathcal{N}(\mathcal{H}_{X'} | \mu_i, \Sigma_i)\right\} > \tau_{90} \quad \forall X' \in \mathbf{X}'$   ▷ Equation 6
```

Algorithm 2. MeetSafe.

on each channel, randomly flipped horizontally, and randomly cropped within a padding of 4. Adversarially learned models were additionally trained half-on-half on benign and perturbed data.

We evaluated the detectors on unseen test images and their perturbed variants as follows. First, we evaluate the defense and model on non-adaptive gray-box perturbations. That includes the one-step FGSM perturbation as well as the DeepFool and C&W-ℓ_2. Second, for each defense, we evaluate its best-performing technique (RCE or AL) on DeepFool against adaptive white-box attacks.

White-box attacks can be generated by adding the detector's likelihood function (Equation 6) to C&W's objective (Carlini and Wagner, 2017a). In essence, this optimizes a multi-objective gradient with Adam that considers both the gradient of the detector's internals and the confidence of the target model:

$$||X||_p + c \cdot \hat{f}(X) + c^* \cdot (-\tau^{-1} \log \hat{p}(X) - 1 + \kappa)^+ \qquad (8)$$

where τ is a given threshold and c^* a constant that controls the sensitivity toward the detector's gradient, optimized with binary search. The sample is updated with perturbation δ when its aggregate $X + \delta$ fools successfully. For our experiments, we limit the perturbation to a ℓ_2 distance of 5.

Our white-box attack follows an all-or-nothing criterion: the batch with adversarial examples is either clean or fully successful. For this reason, we assume that the detector is successful if either the white-box perturbation is detected or classified by the target model. Its true positives are thus in the set $\{X \mid \text{argmax } Y(X) \neq k \lor -\log \hat{p}(X) \leq \tau\}$ for true class k and threshold τ.

The performance of the defenses is measured using its overall detection accuracy of one test run in both adversarial and benign situations, where the detector classifies at a TNR of 90%+. The adversarial setting may include samples without perturbation when the target model already misclassifies the clean sample. We therefore have an optimal detection accuracy of $1 - \frac{\mathcal{R}_f}{2}$ with standard empirical risk $\mathcal{R}_f$ of the target model.

During test runs, we reduced the batch size to 128 (64 for white-box); other hyperparameters, used for the attacks and defenses, were as follows. The magnitude ϵ of FGSM is deduced from Equation 7, DeepFool had an overshoot of 0.02, and C&W executed 5 steps with 500 iterations (10 and 1000 for white-box) under a 0.05 confidence κ. For all defenses, we applied the same feature selection. That took the best 10 in a pool of at most 500 features, given $U_{\mathcal{P}}$. We choose the k of kNN to be 8 for LRD and LOF based on the ablation study in Sec. 5.3. See Section 5.3 for details. The experiments were conducted on AMD Ryzen 7 7700X and Nvidia RTX 4070 Ti.

4.2 Model performance

The accuracy of the models is shown in Table 1. The CIFAR-10 ResNet-50 model is able to reach an average cross-entropy of 0.0249 and a test accuracy of 93.2%. The cross-entropy for robust optimization techniques is notably higher, and this increased to 0.47 for adversarial training and 431.1 for RCE. A higher value for RCE was expected as almost each class now adds to the error instead of only the true class. We also observe that the fit and convergence changes dramatically when the amount of classes is increased. Take Tiny-ImageNet which has 200 classes, where the others have 10, a RCE model trained on ImageNet does only reach an accuracy of 3%. When we test the STL-10 subset, RCE does not show this behavior.

Robust optimization shows a descent mitigation of FGSM for all models and no meaningful mitigation of C&W attacks. The results of DeepFool show that the optimized models are often more robust than the plain ones under smaller perturbations. Still, the high adversarial accuracy of the plain model is somewhat unexpected. However, this may have a clear reason. Namely, the confidence of this model is higher, which causes vanishing gradients.

4.2.1 Vanishing gradients

The confidence score of the plain ResNet-50 is near a unit vector toward the correct class for some images (Table 1), which makes that gradient zero due to rounding errors. Such images sit on a stationary point for the current parameters. This is a major

TABLE 1 Accuracies and proportion of stationary points for FGSM of the trained target models on the CIFAR-10, Tiny-ImageNet, STL-10, and MNIST testing set, respectively.

Model	Learn. rate	Robust optim.	Evasions ($\ell_2 \leq 5$) → Stat. Points (%) ↓	Model's top-1 accuracy ⇑			
				Benign	FGSM Equation 7	C&W $\kappa = 0.05$	DeepFool $\eta = 0.02$
ResNet-50	0.01	✗	50.5, 0.0	0.932, 0.581	0.619, 0.013	0.000, 0.000	0.061, 0.188
			13.7, 12.8	0.711, 0.992	0.111, 0.307	0.008, 0.000	0.214, 0.130
ResNet-50	0.01	AL	37.9, 0.0	0.916, 0.559	0.640, 0.176	0.000, 0.003	0.086, 0.149
			2.6, 8.1	0.612, 0.991	0.229, 0.2704	0.200, 0.000	0.221, 0.385
ResNet-50	0.01	RCE	0.0, 0.0	0.885, 0.030	0.485, 0.003	0.000, 0.000	0.106, 0.015
			0.0, 0.0	0.614, 0.990	0.081, 0.085	0.000, 0.000	0.147, 0.228
VGG-13	0.01	✗	22.4	0.928	0.294	0.000	0.096
VGG-13	0.01	AL	2.9	0.885	0.578	0.000	0.158
VGG-13	0.01	RCE	0.0	0.883	0.329	0.000	0.060

VGG-13 is only trained on CIFAR-10. More details in-text.

drawback of FGSM, but not present for DeepFool and C&W which use gradients of different loss function. Vanishing gradients do give a sense of robustness for the plain model, while it is probably not.

4.2.2 Utility of hidden layers

The utilities discussed in Section 3.2 grow more or less each layer for the CIFAR-10 ResNet. The RCE and plain model exhibit the largest normalized ℓ_2 distance at the fourth bottleneck and the smallest distance at the raw input. The utility $U_{f(l)}$ at the first and last bottleneck differs significantly with RCE; as for 5 samples, the 95% t-confidence interval (CI) is 0.29 ± 0.003 and 0.51 ± 0.01, respectively. This supports the unfolding intuition of Bengio *et al.* (Bengio et al., 2013). Although, we find the behavior of adversarial learned models to be different. There, the first layers seem to be the most sensitive, with the first bottleneck (0.66 ± 0.14) having a higher utility than the fourth (0.48 ± 0.13). Detection methods may thus be fine-tuned by utilizing different layers.

4.3 Detection results

The following section will primarily discuss the performance on near-optimal perturbations. We showcase more results in Section 4.1 regarding the accuracies under semantic adversarial attacks such as shadow attack (Ghiasi et al., 2020) and PerC (Zhao et al., 2020).

4.3.1 Against gray-box attacks

We start by examining gray-box attacks on CIFAR-10. This provides a more comprehensive understanding of our method's performance. Table 2 shows LRD in addition to various other works. Instance-based methods similar to ours are KDE+BU, LID, and MAH. We see that LID does not lead to practical results. On the other hand, LRD reaches an accuracy above 85% for FGSM perturbations, and this outperforms similar methods.

We also consider Robust optimization beneficial. The detection accuracy is frequently higher with one of these methods. Particularly for PCA, its accuracy against DeepFool and FGSM shows a respective difference of 15% and 26%. Moreover, PCA shows strong and similar results as the supervised method SVM on FGSM; the extremal measure, that also uses PCA, is less effective. Finally, we consider feature squeezing's performance limited for FGSM perturbations. It achieves the lowest accuracy of 76% after LID.

These results largely change for smaller perturbations. SVM drops from 90%+ accuracy to a random classifier. In fact, almost all methods suffer from smaller perturbations, except for purification measures such as feature squeezing. Its situation is reverse for smaller perturbations and does improve in this setting, which suggests that most methods do not have a sufficient scope to cover all adversarial attacks.

4.3.2 Against adaptive attacks

We test MeetSafe against adaptive adversaries, a challenging AEs detection scenario, and also show in Table 3. We find that an adaptive attacker can break most methods. In particular, the results for KDE+BU, LID, EXM, and MAH showed a true positive rate close to 0% on the CIFAR-10 and STL-10 datasets, which is consistent with prior works (Carlini and Wagner, 2017a; Athalye et al., 2018).

Regarding the other methods, we see that especially PCA excels with accuracies of approximately 80% for CIFAR-10. Surprisingly, PCA was proven as not robust earlier (Carlini and Wagner, 2017a). LRD is, in addition to PCA, also somewhat resilient against adaptive attacks, although it should be noted that BRISQUE does utilize local non-linear operations to estimate generalized gamma functions (Mittal et al., 2012), which are not smooth functions. We can therefore only consider LRD robust on the hidden features. Some results are worse than in the gray-box setting, that is possible due to the positive confidence value. Hence, the adversarial example is stimulated to be 5% below the detector's threshold, which

TABLE 2 Accuracies of several detection algorithms against gray- and white-box (GB/WB) adversaries with a ℓ_2-radii of 5.

	Evasion attack	Robust optim.	POI → Stat. points (%) ↓	Detection accuracy ⇑								
				Units of the fourth bottleneck			max $U_{f(l)}$	Scene statistics				Logits
				LRD	KDE+BU	LID	EXM	LRD	SVM	PCA	FSQ	MAH
CIFAR-10	FGSM	✗	✗	0.680	0.545	0.508	0.584	**0.698**	**0.725**	**0.722**	0.570	0.596
	FGSM	AL	✗	0.852	0.663	0.586	**0.863**	0.815	**0.884**	**0.878**	0.757	0.792
	FGSM	RCE	✗	0.644	0.818	0.545	0.726	**0.875**	**0.987**	**0.990**	0.636	0.639
	DeepFool	AL	✗	0.516	0.596	0.511	0.499	0.500	0.502	**0.663**	**0.890**	**0.676**
	DeepFool	RCE	✗	**0.775**	**0.785**	0.513	0.564	0.498	0.501	0.533	**0.759**	0.756
	C&W	AL	✗	0.520	0.561	0.501	0.500	0.502	0.500	**0.758**	**0.744**	**0.635**
	C&W	RCE	✗	0.621	**0.707**	0.513	0.558	0.497	0.500	0.582	**0.780**	**0.647**
	C&W	Best Perf.	✓	0.516	0.489	0.453	0.450	**0.660**	**0.619**	**0.792**	0.484	0.474
MNIST	FGSM	AL	✗	**0.928**	**0.957**	0.856	0.901	**0.930**	0.927	0.614	0.828	0.915
	FGSM	RCE	✗	**0.955**	**0.975**	0.936	0.952	**0.955**	**0.974**	0.678	0.915	0.659
	DeepFool	AL	✗	0.744	**0.829**	0.592	0.684	0.745	0.820	0.645	**0.915**	**0.927**
	DeepFool	RCE	✗	0.925	**0.960**	0.468	0.498	0.770	0.533	0.505	**0.947**	**0.945**
	C&W	Best Perf.	✗	**0.894**	0.866	0.522	0.538	0.655	0.505	0.609	**0.949**	**0.947**
	C&W	Best Perf.	✓	0.940	0.912	0.457	0.558	0.983	**0.990**	0.599	**0.986**	**0.988**
STL-10	FGSM	AL	✗	**0.517**	0.498	0.505	0.503	0.502	**0.524**	**0.506**	0.497	0.505
	FGSM	RCE	✗	0.528	0.571	0.511	0.498	0.511	**0.540**	0.531	**0.667**	**0.635**
	DeepFool	AL	✗	**0.509**	0.503	0.504	0.502	0.495	0.498	0.500	**0.588**	**0.510**
	DeepFool	RCE	✗	**0.647**	**0.611**	0.515	0.506	0.505	0.493	0.500	**0.634**	0.524
	C&W	Best Perf.	✗	0.512	**0.524**	0.507	0.505	0.513	0.499	0.500	**0.586**	**0.522**
	C&W	Best Perf.	✓	**0.507**	0.498	0.450	0.470	**0.806**	0.498	0.456	**0.547**	0.479

White-box attacks are evaluated on the detector's best performing robust optimization under DeepFool. DeepFool's and C&W's results are dependent on the error rate $\mathcal{R}_f$ of ResNet-50 (more details in-text). Top-3 results are bolded, and the worst-case of a detection algorithm is underlined. Top-1 results are highlighted in blue.

improves its transferability on models with feature bags or other uncertainties.

We show the performance of LRD, PCA, and FSQ and their MeetSafe ensemble across datasets in Table 2. The p-values of the methods' confidence are computed and compared against a threshold. Specifically, we evaluate the p-values for 10 random FGSM samples using a reversed ResNet-50 model trained on a benign dataset. A low p-value is beneficial for the GMM's generalization as this assumes normality. On MNIST, an opposing utility between LRD and whitening techniques becomes clear. Here, LRD has a p-value of near zero ($\leq \ 10^{-99}$), while whitening has a value of 6e-6. On the other hand, whitening performs relatively better on CIFAR-10 with a p-value smaller than 1e-80 against 5e-17 for LRD. Whitening and LRD might therefore offset each other's effects against FGSM.

The added value of feature squeezing is apparent for small CIFAR-10 perturbations. Figure 2a shows the performance of the three detectors for DeepFool and FGSM. It shows a noticeably higher AUROC for feature squeezing on DeepFool. Furthermore, feature squeezing has the smallest p-value of 0.01, followed by LRD with 0.25. Feature squeezing could thus be helpful in the case when the adversarial sample is near the benign input. Section 5.2 discusses the importance of each components of MeetSafe.

4.3.3 GMM-based detection

Our MeetSafe constructs a GMM-based detection with PCA, feature squeezing, and LRD. We test MeetSafe's performance under certain number of Gaussian components: 4, 8, and 16 (Table 4). For gray-box perturbations, we see that only 4 components may be useful for FGSM, but this increases for small perturbations. To balance these accuracies, we think that 8 components are desirable. Comparing the performance of MS-8 to that of I-Defender shows similar results on FGSM, but lower accuracies on stronger attacks. Meanwhile, the accuracy for RCE and MeetSafe does not scale well. This is most notable when we compare the efficacy for datasets of higher resolution. Tiny-ImageNet shows accuracies on C&W of at most 0.553 for MeetSafe and 0.511 for I-Defender. STL-10 shows similar results (Table 4). However, the effect of dimensionality is not a limitation specific to our method but rather a general issue of defenses against AEs (Goodfellow et al., 2015). MS-8 achieves an improvement of at least 8.1% on adaptive attacks and 10.2% on the worst-case results for each evaluated method by averaging across STL-10, MNIST, and CIFAR-10. MeetSafe may therefore be

TABLE 3 Accuracy of comparison detections evaluated against gray- and white-box (GB/WB) adversaries on STL10, MNIST, and CIFAR10; with a ℓ_2-radii of 5.

Evasion attack	Robust optim.	Dataset	WB	Detection accuracy ⇑						
				MS-8	I-Def	LRD (LFA)	PCA	KDE+BU	FSQ	MAH
FGSM	✗	STL-10	✗	0.520	**0.549**	0.511	0.521	**0.542**	0.494	**0.546**
		MNIST		**0.823**	**0.814**	**0.829**	0.612	0.809	0.667	0.797
VGG-13:		CIFAR-10		**0.808**	0.569	0.690	**0.843**	**0.748**	0.594	0.682
FGSM	AL	STL-10	✗	0.485	**0.508**	**0.517**	**0.506**	0.498	0.497	0.505
		MNIST		0.897	0.894	**0.928**	0.614	**0.957**	0.828	**0.915**
VGG-13:		CIFAR-10		**0.942**	0.508	0.489	**0.941**	0.538	**0.676**	0.571
FGSM	RCE	STL-10	✗	**0.737**	0.567	0.528	0.531	0.571	**0.667**	**0.635**
		MNIST		**0.953**	0.935	**0.955**	0.678	**0.975**	0.915	0.659
VGG-13:		CIFAR-10		**0.956**	**0.606**	0.570	**0.949**	0.602	0.593	0.592
DeepFool	✗	STL-10	✗	**0.540**	0.495	0.507	0.500	**0.542**	0.481	**0.556**
		MNIST		**0.951**	0.877	0.726	0.522	0.840	**0.942**	**0.938**
VGG-13:		CIFAR-10		**0.640**	0.568	0.519	0.502	0.601	**0.894**	**0.756**
ResNet-50:		CIFAR-10		**0.729**	0.700	0.546	0.506	0.630	**0.900**	**0.801**
DeepFool	AL	STL-10	✗	**0.665**	0.502	0.509	0.500	0.503	**0.588**	**0.510**
		MNIST		**0.941**	0.880	0.744	0.645	0.829	**0.915**	**0.927**
VGG-13:		CIFAR-10		**0.686**	0.552	0.501	0.560	0.579	**0.857**	**0.650**
DeepFool	RCE	STL-10	✗	**0.612**	0.531	**0.647**	0.500	0.611	**0.634**	0.524
		MNIST		**0.953**	0.934	0.925	0.505	**0.960**	**0.947**	0.945
VGG-13:		CIFAR-10		0.657	0.633	0.546	0.567	**0.850**	**0.759**	**0.698**
C&W	Best perf.	STL-10	✗	0.517	0.518	0.512	0.500	**0.524**	**0.586**	**0.522**
		MNIST		**0.958**	0.924	0.894	0.609	0.866	**0.949**	**0.947**
VGG-13:		CIFAR-10		**0.803**	0.574	0.542	0.564	**0.652**	**0.870**	0.638
C&W	Best perf.	STL-10	✓	**0.619**	0.475	**0.507**	0.456	0.498	**0.547**	0.479
		MNIST		**0.989**	0.953	0.940	0.599	0.912	**0.986**	**0.988**
VGG-13:		CIFAR-10		**0.896**	0.492	0.536	**0.581**	0.521	**0.606**	0.485
PerC-AL	RCE	STL-10	✗	**0.523**	0.503	0.503	**0.526**	0.516	**0.533**	0.505
		MNIST		N/A	N/A	N/A	N/A	N/A	N/A	N/A
VGG-13:		CIFAR-10		**0.815**	0.745	0.771	**0.795**	**0.782**	0.514	0.766
ResNet-50:		CIFAR-10		0.553	0.529	**0.560**	**0.612**	0.525	0.548	**0.667**
Shadow attack	RCE	STL-10	✗	0.587	0.479	**0.624**	0.450	**0.683**	**0.597**	0.464
		MNIST		**0.927**	**0.937**	0.887	0.453	**0.963**	0.449	0.833
VGG-13:		CIFAR-10		**0.665**	0.477	0.511	**0.698**	0.529	0.504	**0.663**
ResNet-50:		CIFAR-10		**0.653**	0.561	0.596	**0.716**	0.485	0.513	**0.619**

Note that PerC-AL requires color datasets so we do not include results for MNIST, a dataset contains gray scale images. The top-3 results are bolded. Top-1 results are highlighted in blue.

employed universally while maintaining a considerable detection accuracy.

4.3.4 Inference time

Figure 2b shows the inference time of MeetSafe and a ResNet-50 target model on four datasets: MNIST, CIFAR-10, Tiny-ImageNet, and STL-10. Plotted according to the input size of one sample: 784, 3, 072, 12, 288, and 27, 648, respectively. From the figure, we can observe that the discrepancy of ResNet-50 and MeetSafe converges to a factor of approximately 2.3 when the input size gets larger. We increased the feature size from 10 to 17 and the pool size from 500 to 850. We anticipated that this would lead to increased computation, especially for LRD. The results on

TABLE 4 Accuracies of GMM-based detectors against gray- and white-box (GB/WB) adversaries with a ℓ_2-radii of 5 like in Table 2.

Evasion attack	Robust optim.	Dataset model →	Detection accuracy ± [t-CI 95%] ⇑									
			CIFAR-10				STL-10		MNIST		VGG-13	
		WB ↓	MS-4	MS-8	MS-16	I-Def	MS-8	I-Def	MS-8	I-Def	MS-8	I-Def
FGSM	✗	✗	0.680	0.671	0.672	0.627	0.520	0.549	0.823	**0.814**	**0.808**	0.569
FGSM	AL	✗	0.876	0.829	0.853	**0.925**	0.485	0.508	**0.897**	0.894	0.942	0.508
FGSM	RCE	✗	0.965	0.926	0.895	0.717	0.737	0.567	**0.953**	0.935	**0.956**	0.606
DeepFool	AL	✗	0.691	0.752	**0.762**	0.520	0.665	0.502	0.941	**0.880**	0.686	0.552
DeepFool	RCE	✗	0.738	0.785	**0.745**	0.571	0.612	0.531	0.953	**0.934**	0.657	0.633
C&W	AL	✗	0.746	**0.804**	**0.815**	0.518	0.517	0.498	0.958	0.696	0.803	0.509
C&W	RCE	✗	0.810	0.818	**0.814**	0.557	0.574	0.518	0.958	**0.924**	0.689	0.574
C&W	RCE	✓	0.762	0.745 ± 0.04	0.689	0.469	0.544	0.475	0.989	**0.953**	**0.896**	0.492

We show experimental results on additional datasets and model architectures in Table 5. The t-CI is based on 5 runs. Top-3 results are bolded, and the worst-case of a detection algorithm is underlined. Top-1 results are highlighted in blue.

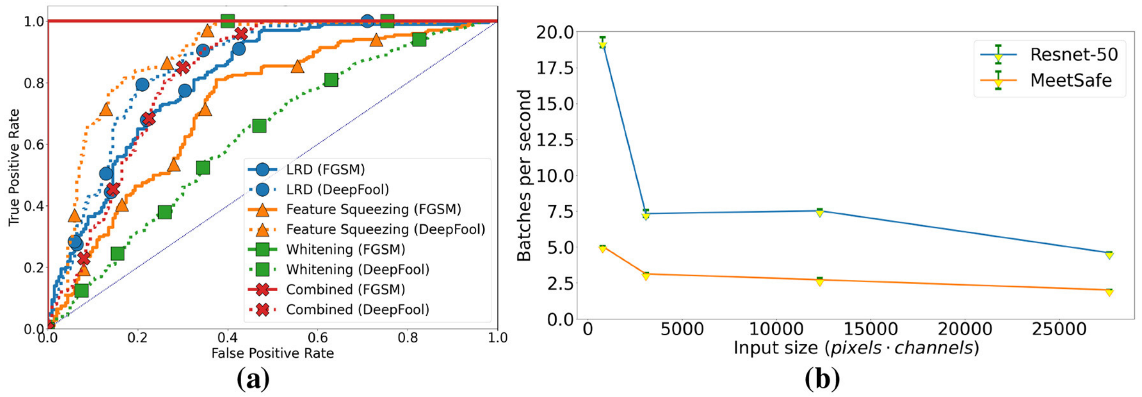

FIGURE 2
(a) ROC curves of MeetSafe's features on CIFAR-10 and a reverse ResNet-50 as target model. The curves show the performance against DeepFool and FGSM. **(b)** Graph that shows the inference times of MS-8 and a reversed ResNet-50 on MNIST, CIFAR-10, Tiny-ImageNet, and STL-10. The error bar denotes the t-CI across 5 runs.

CIFAR-10 demonstrate that this increase led to a slower processing rate, with the model running 0.28 batches per second slower than before. We consider this change to be limited, indicating that the computational overhead is also manageable given the increased feature and pool sizes.

5 Ablation study

5.1 Sensitivity and utility of the hidden layers

Figure 3 explores the $U_{f(l)}$ utility in the study. The barplots show the utility at the outputs of that layer, which tends to increase when the perturbed sample traverses deeper layers, but this may come with fluctuations. For instance, the values for the STL-10 models seem to decrease in the last layers. The adversarial learned CIFAR-10 model also decreases in utility after the first bottleneck. Still, the final bottleneck remains in the Top-3, suggesting its pivotal role in model performance and feature extraction. Conversely, the initial convolution, denoted as "conv1", frequently exhibits the least utility.

We also see a major difference in the CI's critical region of the CIFAR-10 models. Specifically, the adversarial learned model displays notable variability on each layer, especially when compared to the RCE model. That is in line with the narrow activations in the scatterplots of Figure 4. The RCE models exhibit greater utilities in its deeper layers. Consequently, we anticipate enhanced performance when these hidden units are utilized for detection purposes. Otherwise, adversarial learning would become more intriguing due to the improved model accuracy (see Table 1).

5.2 On the impact of MeetSafe components

We check the detection performance by removing each component of MeetSafe (MS-8) to establish its importance within the ensemble. For the ablation, we trained 3 GMMs all with two components (LRD, PCA, FSQ) on a reversed ResNet-50 and

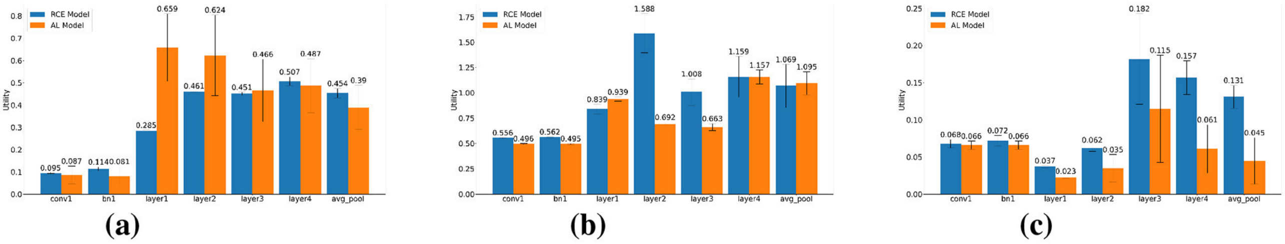

FIGURE 3
Average of normalized Euclidean distance between FGSM AEs (ℓ_2 of 5) and benign samples under $U_{f(l)}$ at each ResNet-50 layer. The error bars denote the 95% t-CI across 5 runs. **(a)** CIFAR-10 dataset, **(b)** MNIST dataset, and **(c)** STL-10 dataset.

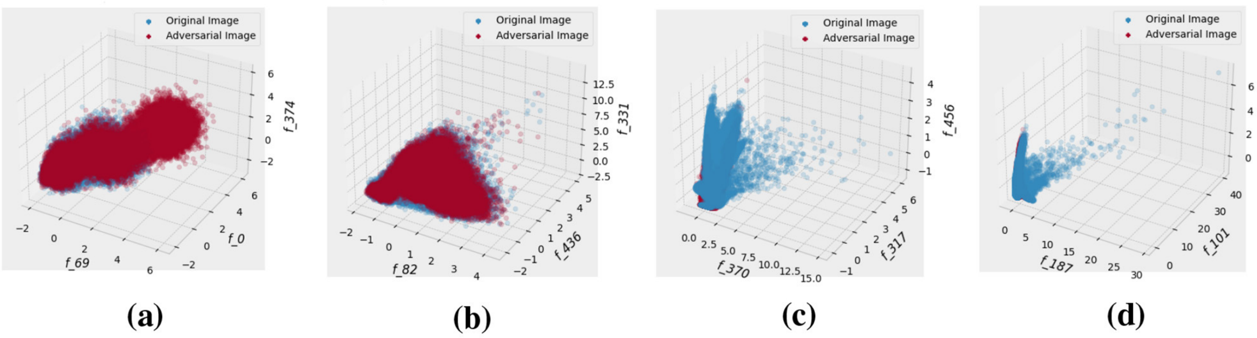

FIGURE 4
Activations' Z-Score of three hidden features before and after an FGSM perturbation, with a ℓ_2-radii of 5. The activations are sampled from a random pool of 500 hidden features, directly after convolution. We show the activation of hidden features with the highest and lowest Z-score in **(a, b)** for an adversarially trained ResNet50 on CIFAR10 and non-adversarially trained results in **(c, d)**.

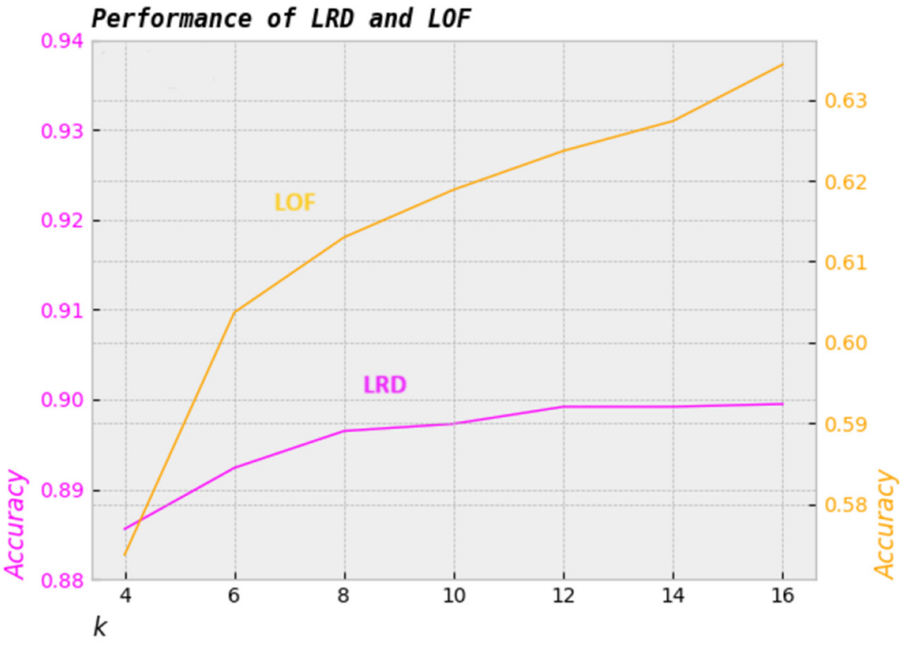

FIGURE 5
Performance of LOF and LRD under different *k* with the amount of feature bags *t*=30 under CIFAR-10 and adversarial learned ResNet-50.

TABLE 5 Accuracies of GMM-based detectors against gray-box adversaries on Tiny-ImageNet, STL-10, and CIFAR-10, with a ℓ_2-radii of 5.

Evasion attack	Model	Dataset		Detection accuracy ⇑					
			Robust optim.→	Plain model		AL model		RCE model	
			WB ↓	MS-8	I-Def	MS-8	I-Def	MS-8	I-Def
FGSM	ResNet-50	Tiny-ImageNet	✗	0.954	0.532	0.939	0.533	**0.976**	0.503
C&W	VGG-13	CIFAR-10	✗	0.790	0.593	**0.803**	0.509	0.689	0.574
C&W	ResNet-50	STL-10	✗	0.531	0.499	0.517	0.498	**0.574**	0.518
C&W	ResNet-50	Tiny-ImageNet	✗	**0.553**	0.494	0.523	0.497	0.502	0.511

Demonstrating the efficacy of small-scale models in comparison with datasets containing images of higher resolution. DeepFool's and C&W's results are dependent on the error rate $\mathcal{R}_f$ of the models, like in Table 2. Best results are bolded. Top-1 results are highlighted in blue.

CIFAR-10. First, when we remove LRD the accuracy decreases by 0.054 for C&W, −0.027 for FGSM, and 0.117 for DeepFool. For FGSM perturbations, the results do show that LRD does not add much for on CIFAR-10, as its ablation leads to equivalent accuracies of the GMM. Overall, LRD increases the effectiveness of MeetSafe across various scenarios, aligning with the findings presented in Section 4.3. Second, when we remove PCA, the accuracy decreases by 0.191 for C&W, 0.234 for FGSM, and 0.145 for DeepFool. Whitening is thus an important component on CIFAR-10. Third, when we remove FSQ, the accuracy decreases by 0.197 for C&W, −0.048 for FGSM, and 0.137 for DeepFool, which is also in line with the results in Section 4.3.

5.3 On the impact of *k* for kNNs

To determine the optimal k, we evaluated the accuracy of LOF and LRD using features from a random pool of 500 hidden features, as shown in Figure 5. The left axis represents LRD accuracy, while the right axis denotes LOF accuracy. The results are plotted separately to highlight their distinct trends, each spanning an accuracy range of 0.06. The elbow points indicate an optimal $k = 8$ for both methods, which we adopt for all k-NN-based approaches. Notably, LRD achieves significantly higher accuracy than LOF, and k impacts LOF more than LRD.

6 Conclusion and limitation

This study present MeetSafe, a scalable and effective framework to detect white-box AEs. By leveraging insights from feature distribution irregularities, MeetSafe integrates utility-based feature selection with feature squeezing, whitening, and feature squeezing to achieve high defense effectiveness and scalability against model size with the increase of high-dimensional feature spaces. Experimental results demonstrate an high detection accuracy of MeetSafe across adaptive and classic adversarial attacks, as well as robust whitening under white-box scenarios.

6.1 Limitations

Due to resource constraints, we considered I-Defender and Feinman et al. (2017) KDE not practical in certain situations. For models like ResNet-50, it would cost at least 372.53 *GiB* to evaluate I-Defender. On the other hand, KDE+BU required a large computational graph in Pytorch during white box testing. That was because of the tens of forwards for dropout. For the same reason, we could only utilize $U_{f^{(l)}}$ for the extremal value. Other methods (LRD, LID, and KDE+BU) require instance-based learning and more memory.

Data availability statement

Publicly available datasets were analyzed in this study. This data can be found here: https://docs.pytorch.org/vision/stable/index.html.

Author contributions

RS: Conceptualization, Data curation, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. DL: Conceptualization, Investigation, Methodology, Supervision, Writing – original draft, Writing – review & editing. YQ: Methodology, Validation, Writing – review & editing. MC: Validation, Writing – review & editing. MP: Validation, Writing – review & editing. KL: Funding acquisition, Project administration, Resources, Validation, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research and/or publication of this article. This work was supported by the EU Horizon Europe Research and Innovation Program under grant agreements 101073920 (TENSOR), 101070052 (TANGO), 101070627 (REWIRE) and 101092912 (MLSysOps).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that Gen AI was used in the creation of this manuscript. Gen AI was used for: (1) Generate LaTeX code for tables and figures to ensure a good layout. Note that all figures are drawn by the author(s) and all data is obtained by the author(s) by running experiments in human. (2) Refine the language. (3) Address LaTeX compilation issues when errors are encountered in Overleaf.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Akhtar, Z., Monteiro, J., and Falk, T. H. (2018). "Adversarial examples detection using no-reference image quality features," in *International Carnahan Conference on Security Technology* (Montreal, QC: IEEE), 1–5.

Aldahdooh, A., Hamidouche, W., Fezza, S. A., and Déforges, O. (2022). Adversarial example detection for dnn models: a review and experimental comparison. *Artif. Intellig. Rev.* 55, 4403–4462. doi: 10.1007/s10462-021-10125-w

Athalye, A., and Carlini, N. (2018). On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv* preprint arXiv:1804.03286. doi: 10.48550/arXiv.1804.03286

Athalye, A., Carlini, N., and Wagner, D. A. (2018). "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning* (Stockholm: ICML.cc), 274–283.

Bengio, Y., Mesnil, G., Dauphin, Y., and Rifai, S. (2013). "Better mixing via deep representations," in *International Conference on Machine Learning* (Atlanta: ICML.cc), 552–560.

Breunig, M. M., Kriegel, H., Ng, R. T., and Sander, J. (2000). "LOF: identifying density-based local outliers," in *ACM International Conference on Management of Data* (New York: ACM), 93–104.

Carlini, N., and Wagner, D. A. (2017a). "Adversarial examples are not easily detected: bypassing ten detection methods," in *ACM Workshop on Artificial Intelligence and Security* (New York: ACM) 3–14. doi: 10.1145/3128572.3140444

Carlini, N., and Wagner, D. A. (2017b). "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy* (SAN JOSE, CA: IEEE), 39–57.

Coates, A., Ng, A. Y., and Lee, H. (2011). "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics* (JMLR), 215–223.

Duan, R., Chen, Y., Niu, D., Yang, Y., Qin, A. K., and He, Y. (2021). Advdrop: Adversarial attack to dnns by dropping information. *In IEEE/CVF International Conference on Computer Vision* (Montreal, BC: IEEE), 7506–7515

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. (2017). Detecting adversarial samples from artifacts. *arXiv* preprint arXiv:1703.00410. doi: 10.48550/arXiv.1703.00410

Ghiasi, A., Shafahi, A., and Goldstein, T. (2020). "Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates," in *International Conference on Learning Representations* (ICLR.cc).

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations* (San Diego, CA: ICLR.cc).

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition* (Las Vegas, NV: IEEE), 770–778.

Hendrycks, D., and Gimpel, K. (2017). "Early methods for detecting adversarial images," in *International Conference on Learning Representations* (Toulon: ICLR.cc).

Hu, S., Yu, T., Guo, C., Chao, W.-L., and Weinberger, K. Q. (2019). A new defense against adversarial images: Turning a weakness into a strength. *arXiv* [preprint] arXiv:1910.07629. doi: 10.48550/arXiv.1910.07629

Kherchouche, A., Fezza, S. A., Hamidouche, W., and Déforges, O. (2020). "Detection of adversarial examples in deep neural networks with natural scene statistics," in *International Joint Conference on Neural Networks* (Glasgow: IEEE), 1–7.

Krizhevsky, A., and Hinton, G. (2009). *Learning Multiple Layers of Features from Tiny Images* (Master's thesis). University of Toronto, Toronto, ON, Canada.

Lazarevic, A., and Kumar, V. (2005). "Feature bagging for outlier detection," in *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (New York: ACM), 157–166.

Le, Y., and Yang, X. (2015). *Tiny Imagenet Visual Recognition Challenge.*

LeCun, Y. (1998). *The MNIST Database of Handwritten Digits.*

Lee, K., Lee, K., Lee, H., and Shin, J. (2018). "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Advances in Neural Information Processing Systems* (Montreal, QC: neurips.cc), 31.

Li, X., and Li, F. (2017). "Adversarial examples detection in deep networks with convolutional filter statistics," in *International Conference on Computer Vision* (Venice: IEEE), 5775–5783.

Li, Y., Bradshaw, J., and Sharma, Y. (2019). "Are generative classifiers more robust to adversarial attacks?," in *International Conference on Machine Learning* (Long Beach, CA: icml.cc), 3804–3814.

Liang, B., Li, H., Su, M., Li, X., Shi, W., and Wang, X. (2018). Detecting adversarial image examples in deep neural networks with adaptive noise reduction. *IEEE Trans. Depend. Secure Comp.* 18, 72–85. doi: 10.1109/TDSC.2018.2874243

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., et al. (2017). A survey on deep learning in medical image analysis. *Med. Image Analy.*42:60–88. doi: 10.1016/j.media.2017.07.005

Luo, C., Lin, Q., Xie, W., Wu, B., Xie, J., and Shen, L. (2022). "Frequency-driven imperceptible adversarial attack on semantic similarity," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (New Orleans, LA: IEEE) 15315–15324.

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S. N. R., Schoenebeck, G., et al. (2018). "Characterizing adversarial subspaces using local intrinsic dimensionality," in *International Conference on Learning Representations* (Vancouver, BC: iclr.cc).

Mittal, A., Moorthy, A. K., and Bovik, A. C. (2012). No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Proc.* 21, 4695–4708. doi: 10.1109/TIP.2012.2214050

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). "Deepfool: a simple and accurate method to fool deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition* (Las Vegas, NV: IEEE), 2574–2582.

Pang, T., Du, C., Dong, Y., and Zhu, J. (2018). T"owards robust detection of adversarial examples," in *Advances in Neural Information Processing Systems* (Montreal, QC: nips.cc), 31, 4579–4589.

Raghunathan, A., Steinhardt, J., and Liang, P. (2018). "Certified defenses against adversarial examples," in *International Conference on Learning Representations* (ICLR.cc).

Raghuram, J., Chandrasekaran, V., Jha, S., and Banerjee, S. (2021). "A general framework for detecting anomalous inputs to dnn classifiers," in *International Conference on Machine Learning* (PMLR), 8764–8775.

Simonyan, K., and Zisserman, A. (2015). "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations* (San Diego, CA: iclr.cc).

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. (2018). "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," in *International Conference on Learning Representations.*

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. (2014). "Intriguing properties of neural networks," in *International Conference on Learning Representations* (iclr.cc).

Tanay, T., and Griffin, L. (2016). A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv* [preprint] arXiv:1608.07690. doi: 10.48550/arXiv.1608.07690

Tian, J., Zhou, J., Li, Y., and Duan, J. (2021). Detecting adversarial examples from sensitivity inconsistency of spatial-transform domain. *AAAI Conf. Artif. Intellig.* 35, 9877–9885. doi: 10.1609/aaai.v35i11.17187

Tramer, F. (2022). "Detecting adversarial examples is (nearly) as hard as classifying them," in *International Conference on Machine Learning* (Baltimore, MD: PMLR), 21692–21702.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. (2020). "On adaptive attacks to adversarial example defenses," in *Advances in Neural Information Processing Systems (NeurIPS)* (neurips.cc), 1633–1645.

Weng, T., Zhang, H., Chen, P., Yi, J., Su, D., Gao, Y., Hsieh, C., and Daniel, L. (2018). "Evaluating the robustness of neural networks: An extreme value theory approach," in *International Conference on Learning Representations* (Vancouver, BC: iclr.cc).

Xu, W., Evans, D., and Qi, Y. (2018). "Feature squeezing: Detecting adversarial examples in deep neural networks," in *Network and Distributed System Security Symposium* (San Diego, CA: NDSS-Symposium.org).

Yin, X., Kolouri, S., and Rohde, G. K. (2019). GAT: Generative adversarial training for adversarial example detection and robust classification. *arXiv* [preprint] arXiv:1905.11475. doi: 10.48550/arXiv.1905.11475

Zhao, W., Chellappa, R., Phillips, P. J., and Rosenfeld, A. (2003). "Face recognition: a literature survey," in *ACM Computing Surveys* (New York: ACM), 399–458.

Zhao, Z., Liu, Z., and Larson, M. (2020). "Towards large yet imperceptible adversarial image perturbations with perceptual color distance," in *IEEE/CVF conference on Computer Vision and Pattern Recognition* (Seattle, WA: IEEE), 1039–1048.

Zheng, Z., and Hong, P. (2018). "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Advances in Neural Information Processing Systems* (Montreal, CA: neurips.cc), 31.