RADNN: ROBUST TO IMPERCEPTIBLE ADVERSARIAL ATTACKS DEEP NEURAL NETWORK

Eduardo Soares
School of Computing and Communications
LIRA Research Centre
Lancaster University
Lancaster, LA1 4WA, UK
e.almeidasoares@lancaster.ac.uk

Plamen Angelov
School of Computing and Communications
LIRA Research Centre
Lancaster University
Lancaster, LA1 4WA, UK
p.angelov@lancaster.ac.uk

September 30, 2021

ABSTRACT

This paper presents the RADNN algorithm. The RADNN is a robust to imperceptible adversarial attack algorithm that uses the concept of data density and similarities to detect attacks on real-time. Differently from traditional deep learnings that need be trained on the attacks to be able to detect, RADNN has a mechanism that detects data patterns changes. In order to evaluate the proposed method, we considered the PerC attacks and a 1000 images from the Imagenet dataset. The RADNN could correctly identify 97.2% of the attacks.

***K*eywords** Adversarial attacks · Explainable AI · Human-interpretable rules · prototype-based approach

1 Introduction

During the last years, deep neural networks have made a tremendous success as they could achieve high accuracy on different complex applications as computer vision, and natural language processing [1]. However, recent findings have shown that deep learning models have several vulnerabilities to adversarial attacks. Deep learning tends to make wrongly overconfident predictions on modified data [2]. Furthermore, their "black-box" nature makes extremely difficult to audit their decisions [3].

Security aspects of machine learning are extremely important specially on high stake applications as autonomous cars [4]. In particular, robustness to adversarially chosen inputs is becoming a crucial design goal as recent work [5] shows that an adversary is often able to manipulate the input so that the model produces an incorrect output.

Hence, defending against such attacks has become an important research topic, and many approaches to improve model security and robustness have been proposed, including improvements to model design, training data augmentation, input preprocessing, defensive validation, among others [6]. Identifying vulnerabilities and addressing them also plays a vital role in obtaining a more robust model [[7].

In this paper, we propose a prototype-based method that is able to detect changes in the data patterns and detect imperceptible adversarial attacks on real time. Differently from traditional approaches, the proposed method does not require a specific training on adversarial data to improve its robustness.

1.1 Imperceptible Adversarial Attacks

Image adversarial attacks are focused on perturbations that cause misclassification by a deep classifier while are imperceptible to the humans [8]. However, visually changes in the image are perceptible when larger image perturbations are used to improve the ability to fool a classifier.

[9] proposes the PerC algorithm which creates adversarial examples by perturbing images through its perceptual color distance. PerC makes possible to use larger L_p norm values of perturbations in RGB space to create adversarial examples that are less perceptible to humans. Fig 1 illustrates adversarial examples caused by PerC.

Figure 1: Examples of imperceptible PerC attacks on Imagenet datasamples.

The PerC algorithm allows to hide large perturbations in the RGB space, in a way that is not noticeable to humans [9].

2 Robust to adversarial attacks Deep Neural Network (RADNN)

The proposed RADNN is equipped with a mechanism that allows real-time concept drift detection (data pattern detection) due its density-based nature and its design based on prototypes. The RADNN approach is described as a feedforward neural network. The training architecture that is composed by the following layers:

1. Features layer;
2. Density layer;
3. Conditional probability layer;
4. Prototype identification layer;

RADNN is trained per class. Thus, it is composed of multiple structures for each class as illustrated by Fig. .

2.1 Features layer

This layer is in charge of the features vector extraction. RADNN has a flexible structure so this layer can be formed by different methods including: convolutional neural networks [10], residual neural networks as Resnet [11], Inception-Resnet [12], Transformers-based approaches [13] or even a combination of multiple sources fro feature extraction. In this paper we consider the VGG–16 [14] method as feature extractor.

The training dataset is defined as $x = \{x_1, ..., x_N\} \in \mathbb{R}^n$ with corresponding class labels $y_1, ..., y_C \in \{1, ..., C\}$. Where, N is the number of training data samples and n is the dimensionality (number of features); C is the number of classes. The most representative data sample in the dataset are chosen as prototypes $\pi \in P \subset X$ for each class. Where, M_j denotes the total number of prototypes of class j; $M_j = |P_j|$; $M = \sum_{j=1}^{C} M_j$. In this paper, we consider more than one prototype per class, so $M_j > 1$ for $\forall j$.

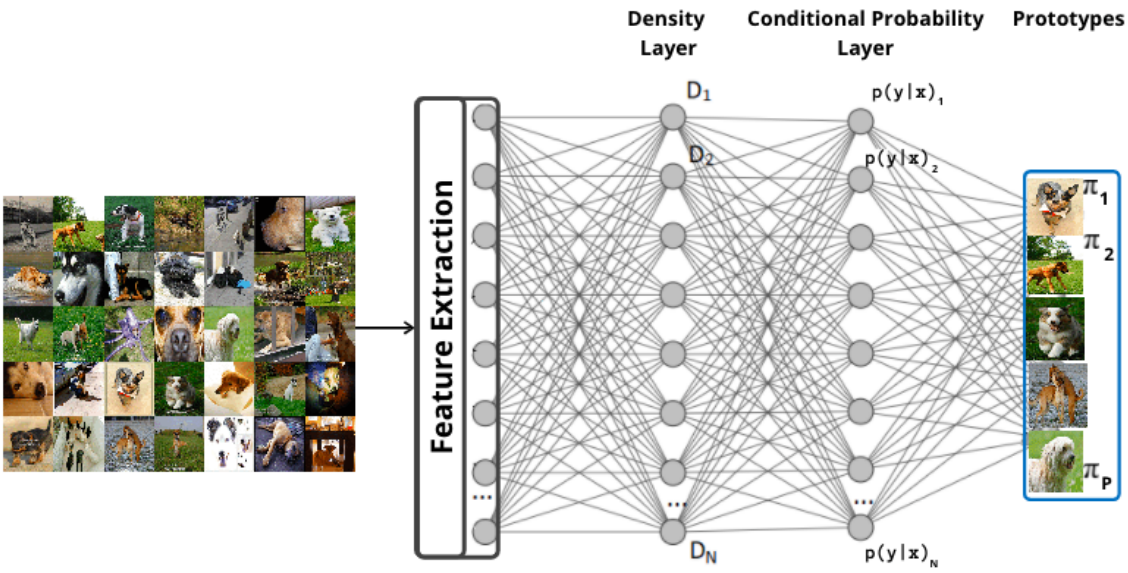

Figure 2: Training architecture for the Imagenet dog class.

Prototypes are the most representative data samples of the training dataset. Then, any new data sample, $x \in \mathbb{R}^n$ can be associated with the nearest prototype from the sets $P_1, P_2, ..., P_C$; $P = P_1 \cup P_2 \cup ... \cup P_C$.

$$L(x) = \underset{x \in X}{\operatorname{argmin}} \min_{\pi \in P} d(x, \pi). \tag{1}$$

2.2 Density layer

This layer is defined by neurons which have data density, D as activation function. The density function defines the mutual proximity of the data samples in the data space and can be represented by the following Cauchy function [15]:

$$D(x) = \frac{1}{1 + \frac{||x-\mu||^2}{||\sigma||^2}}, \tag{2}$$

where D is the density, μ is the global mean, and σ is the variance. The mutual proximity of the data samples in the data space using Euclidean (or Mahalanobis) distance has the form of a Cauchy function as demonstrated theoretically by [15].

Data density can be updated recursively by [16]:

$$D(x_i) = \frac{1}{1 + ||x_i - \mu_i||^2 + \sum_i - ||\mu_i||^2}. \tag{3}$$

where $i = 1, ..., N$, μ and the scalar product, $\sum$ can be updated recursively as follows:

$$\mu_i = \frac{i-1}{i}\mu_{i-1} + \frac{1}{i}x_i, \tag{4}$$

$$\sum_i = \frac{i-1}{i}\sum_{i-1} + \frac{1}{i}||x_i||^2 \quad \sum_1 = ||x_1||^2. \tag{5}$$

Data density, D, denotes the degree of closeness of a data sample to the mean, μ. For values that are normalized between 0 and 1 the density ranges is $0 < D \leq 1$. Where $D = 1$ when $x = \mu$. In this sense, data samples that are

closer to the global mean have higher density values. The data density value indicates how strongly a particular data sample is influenced by other data samples in the data space due to their mutual proximity. Data density indicates the centrality of a data sample in the data space and its eligibility to become a prototype.

2.3 Conditional probability layer

The *typicality*, τ, layer or conditional probability layer is estimated from empirical data as in [15]. It is given by eq. (6), where integral of $\int_{-\infty}^{\infty} p(C|x)dx = 1$ is multi-modal version of pdf [15]:

$$p(y|x) = \frac{\sum_{i=1}^{M} N_i D(x)}{\sum_{i=1}^{M} N_i \int_{-\infty}^{\infty} D(x)dx} \tag{6}$$

where N_i denotes the number of data samples associated with the $i-th$ *data cloud*, $\sum_{i=1}^{C}; N_i = N$.

$p(C|x)$ does not rely on any *prior* assumption about the data [15].

2.4 Prototypes layer

RADNN is trained per class as illustrated by Fig. 2. So, the calculations are done for each class separately. Prototypes are independent data samples that are defined as the local peaks of the data density. Thus, the prototypes set can be altered (prototypes can be added or removed) without affecting the other existing ones.

Data samples are assigned to the nearest prototype as:

$$j^* = \underset{i=1,..,N;j=1,..,M}{\operatorname{argmin}} ||x_i - \pi_j||^2 \tag{7}$$

New prototypes are added to the set of prototypes if the following condition is met [15]:

$$\begin{aligned} &\textit{IF } (D(x) \geq \max_{j=1,..,M} D(\pi_j)) \\ &\textit{OR } \ (D(x) \leq \min_{j=1,..,M} D(\pi_j)) \\ &\textit{THEN } (\textit{add a new data cloud } (j \leftarrow j+1)) \end{aligned} \tag{8}$$

2.5 Learning Procedure

RADNN learning mechanism is summarised by the following pseudo-code.

RADNN: Learning Procedure

Read the first feature vector sample x_i of class c;
Normalise the data as detailed in [17]
Set $i \leftarrow 1; j \leftarrow 1; \pi_1 \leftarrow x_i; \mu \leftarrow x_1; N \leftarrow 1$
FOR i = 2, ...
  Read x_i;
  Calculate $D(x_i)$ and $D(\pi_j)$ $(j = 1, 2, ..., M)$ according to eq. (2);
  IF eq. (8) holds
    Create new prototype: $j \leftarrow j+1; \pi_j \leftarrow x_i; N \leftarrow N+1$
  ELSE
    Search for the nearest prototype according to eq. (7);
    Update the nearest prototype as:
    $N \leftarrow N+1;$
    $\pi_j \leftarrow \frac{N_j}{N_j+1}\pi_j + \frac{N_j}{N_j+1}x_i;$
  END
END

---

3 Architecture and for Detection and Validation of (RADNN)

Fig. 3 illustrates the architecture for the adversarial attack detection and validation of the RADNN method. 3.

This architecture is composed by the following layers:

1. Features layer;
2. Local decision-making;
3. Global decision-making (attack detection);

3.1 Features layer

Similar to the feature extraction layer described in the training phase.

3.2 Local decision-making

It is responsible to calculate the degree of similarity, S, between an unlabeled data sample and the respective nearest prototype. The similarity degree between any new image and a given prototype is determined by a SoftMax-like equation (9).

$$\lambda(Y = x_i|\pi_j) = \frac{S_j}{\sum_{j=1}^{M} S_j}, \tag{9}$$

where,

$$S_j = S(x_i, \pi_j) = \frac{1}{1 + \frac{||x_i - \pi_j||^2}{||\sigma_j||^2}}, \tag{10}$$

$$J = -log\lambda(Y = x_i|\pi_j). \tag{11}$$

where Y is the $j-th$ validation data sample. S is the degree of similarity between the unlabeled data sample and the respective prototype.

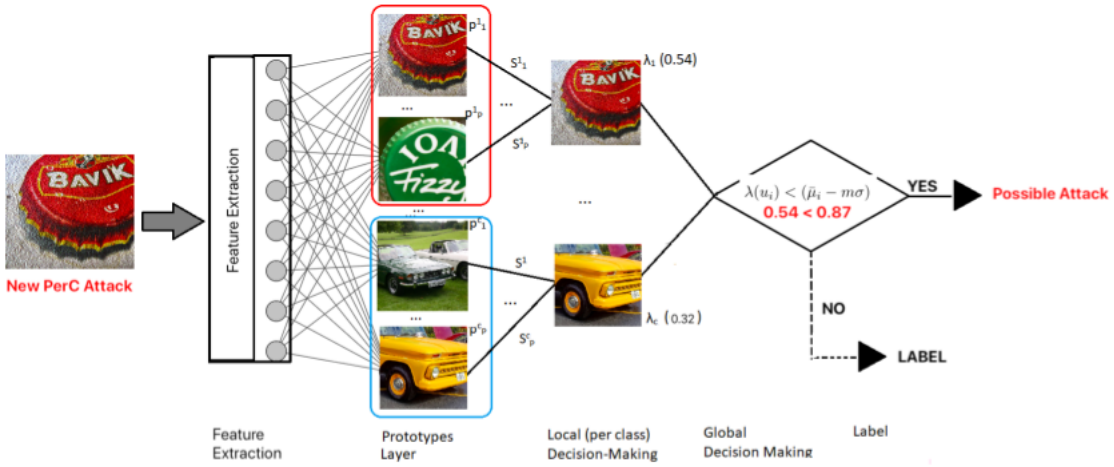

Figure 3: Architecture for attack detection and validation process of RADNN.

3.3 Global decision-making (attack detection)

The RADNN uses the recursive mean $\overline{\mu}_i$ of the λ to detect sudden drop on the confidence. When a new data sample arrives to the system,$\overline{\mu}$ is calculated as [16]:

$$\overline{\mu}_i = \frac{i-1}{i}\overline{\mu}_{i-1} + \frac{1}{i}\lambda_i, \overline{\mu}_1 = \lambda_1. \tag{12}$$

Then the m-σ rule is applied. Possible attacks are actively detected when the inequality (13) is satisfied. Otherwise, if the inequality is not satisfied the new data sample is assigned to a label.

$$\begin{aligned} IF\ \lambda(u_i) < (\bar{\mu}_i - m\sigma)\ THEN\ (u_i \in Possible\ new\ attack\ detected) \\ ELSE\ (Assign\ label) \end{aligned} \tag{13}$$

When the inequality (13) is satisfied, the arrival data sample is denoted as a potential attack and temporally saved. The label for high confidence data samples is given by the equation (14):

$$label = \underset{c=1,2,\ldots,C}{argmax}\,(\lambda_c^*), \tag{14}$$

4 Experiments

In order to evaluate the robustness of RADNN to imperceptible attacks, we considered 1000 images from the Imagenet dataset attacked by the PerC algorithm. Deep Learning approaches as VGG–16 and ResNet were also used during this experiment.

In order to evaluate the approaches the following metric has been considered:

Detection rate:

$$Detection(\%) = \frac{TP + TN}{TP + FP + TN + FN} \times 100, \tag{15}$$

4.1 Results

Table 1 illustrates the different results obtained during the experiments

Table 1: Results considering different methods for imperceptible attacks identification

Method	Detection rate (%)
RADNN	**97.2%**
DenseNet-201 [18]	57.48%
ResNet-152 [11]	46.23%
VGG–16 [14]	38.31%
AlexNet [10]	37.72%

As demonstrated by 1, RADNN could obtain better performance in terms of detection than its competitors. The reason of this difference in the results is that traditional deep learnings lack of robustness because they need to be trained to the specific attack to obtain high performance. On the other hand, RADNN is equipped with a detection mechanism that allows flexibility on the structure to not classify data samples that the algorithm is not confident about. Moreover, the transparent structure of RADNN allows users to inspect the networks decisions and visualize/understand it.

5 Conclusion

In this paper, we introduced RADNN algorithm. The algorithm has a robust design that is able to detect imperceptible to human attacks due its density-prototype-based nature. The experiment have shown that differently from traditional approaches that need to be trained on the attacks to obtain high performance in terms of detection, the RADNN is able to detect attacks without $prior$ training on it due its confidence system based on similarities.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[2] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[3] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[4] Yao Deng, Xi Zheng, Tianyi Zhang, Chen Chen, Guannan Lou, and Miryung Kim. An analysis of adversarial attacks and defenses on autonomous driving models. In *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2020.

[5] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[6] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 30(9):2805–2824, 2019.

[7] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[8] Bo Luo, Yannan Liu, Lingxiao Wei, and Qiang Xu. Towards imperceptible and robust adversarial example attacks against neural networks. In *Thirty-second aaai conference on artificial intelligence*, 2018.

[9] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. Towards large yet imperceptible adversarial image perturbations with perceptual color distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1039–1048, 2020.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[13] Ibrahim Onur Sigirci, Hakan Ozgur, and Gokhan Bilgin. Feature extraction with bidirectional encoder representations from transformers in hyperspectral images. In *2020 28th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2020.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[15] Plamen P Angelov and Xiaowei Gu. *Empirical approach to Machine Learning*. Springer, 2019.

[16] Plamen Angelov. *Autonomous learning systems: from data streams to knowledge in real-time*. John Wiley & Sons, 2012.

[17] Plamen Angelov and Eduardo Soares. Towards explainable deep neural networks (xdnn). *Neural Networks*, 130:185–194, 2020.

[18] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.