# Improving Robustness Against Adversarial Example Attacks Using Non-Parametric Models on MNIST

Sanghyeon An, Min Jun Lee, Jungmin So
Department of Computer Science and Engineering
Sogang University
Seoul, Korea
ansh941@sogang.ac.kr, roblee1007@gmail.com, jso1@sogang.ac.kr

*Abstract*— **Deep learning research has been actively conducted, and neural networks including CNN have outstanding performance in computer vision. However, parametric models such as neural networks are known to be vulnerable to adversarial example attacks, making them inappropriate to employ when security becomes significant. Hence, non-parametric models are considered but there is a problem of having low accuracy. To solve the issue, we proposed a scheme where images are segmented into patch units for non-parametric models. Experimental results display that the proposed scheme improves both accuracy as well as robustness against adversarial example attacks.**

*Keywords*— **deep learning, non-parametric models, segmentation, adversarial example**

## I. INTRODUCTION

Recently, research on deep learning using neural network has been vigorously carried out. It is notably used in the field of computer vision, as CNN has shown exceptional performance. However, parametric models including CNN is known to be vulnerable to adversarial example attacks. Common adversarial example attacks employ gradient-based attack, which creates image samples that are difficult to distinguish with original images using human eye, but trained models can misclassify. This causes fatal problem for parametric models. For non-parametric models, they are known to be slightly less accurate yet more robust in determining adversarial images. Studies are actively underway to address the shortcomings of each models. Nonetheless, there is no significant achievement yet [1, 2].

In this paper, we analyze robustness of non-parametric models using adversarial example attack images. Then we propose a method in which all dataset images are segmented into patch units to improve robustness along with accuracy of non-parametric models against adversarial example attacks. We employed clean images, FGSM images, and JSMA images to test with a variety of models, including those using the proposed method. As a result, our scheme enhanced the accuracy and robustness against adversarial example attacks on non-parametric models.

## II. REALTED WORKS

### A. Adversarial Attack

**Fast Gradient Sign Method(FGSM):** Goodfellow et al. proposed a simple yet efficient method for creating adversarial example attack algorithms [3]. When model training, it adds perturbation that reverses gradient direction utilized in the gradient-descent processing. Then the attacker gains the effect that hinders the learning. The perturbation is expressed as:

$$\eta = \varepsilon sign\left(\nabla xJ(\theta, x, y)\right) \tag{1}$$

where $\varepsilon$ is the magnitude of perturbation, $x$ the input of the model, $y$ the label of the input, $\theta$ the parameter of the model, and $J(\theta, x, y)$ the cost used to train neural network. Fig. 1 exhibits the FGSM images. Based on untargeted attack, it barely perturbs the very weighted portion of the neural network. Nevertheless, the images are generally misclassified.

**Jacobian-based Saliency Map Attack(JSMA):** Papernot et al. proposed an white-box attack that makes neural network to misclassify an image into target classes using forward derivative [4]. Let $x$ be the input, $F$ the neural network, $F_j(X)$ the output for class $j$, and $F_t(X)$ the output for target class $t$. In order for $x$ to be classified as $t$, the probability of $F_t(X)$ must increase, as well as the probability of $F_j(X)$ of all other classes that conform $j \neq t$ needs to decrease. Through the process, $t = \arg max_j\, F_j(X)$ has to be satisfied and the following adversarial saliency map $S(X, t)$:

$$S(X,t)[i] = \begin{cases} 0, if \frac{\partial F_t(X)}{\partial X_i} < 0 \; or \; \sum_{j \neq t} \frac{\partial F_j(X)}{\partial X_i} > 0 \\ \left(\frac{\partial F_t(X)}{\partial X_i}\right) \left|\sum_{j \neq t} \frac{\partial F_j(X)}{\partial X_i}\right|, otherwise \end{cases} \tag{2}$$

is presented to accomplish the condition. From the above expression, $i$ is an input feature. If $S(X, t)[i]$ value becomes higher, the input feature will raise the output value of the target class, or decrease the output value of classes that are not the target, or both. This algorithm repeats iteratively until the input $x$ is misclassified as the target class $t$. Fig. 2 displays the JSMA images. In each iteration, saliency map is implemented to choose and demolish the most crucial pixel.

### B. Non-ParametricModels

***k*-Nearest Neighbor(*k*-NN):** *k*-NN compares test sets with entire training sets by a predetermined distance evaluation method and classifies them into a majority vote of $k$ nearest images [5, 6]. There is no learning procedure in *k*-NN, Nor a model with parameters. Two types of hyperparameter subsists for *k*-NN which is the value of $k$, and a distance metric. Like in Fig. 3, the categorization of the input can be absolutely different according to $k$. If $k$ is 3, the circular input will be classified as

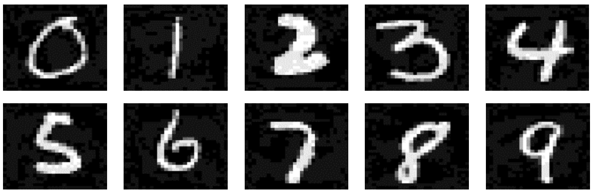

Fig. 1. Images produced by FGSM.

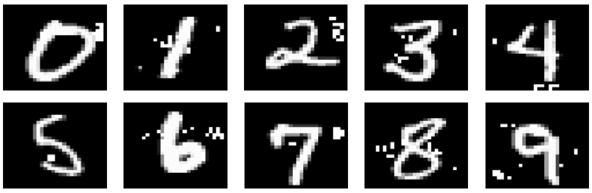

Fig. 2. Images produced by JSMA.

the rectangular data and if $k$ is 9, it will be classified as the triangular data. Usually, if $k$ is high, underfitting occurs due to over normalization, and overfitting happens if $k$ is small because it excessively reflects regional aspect. As for the distance metric, there are three primary options which are Euclidean distance, cosine similarity, and Mahalanobis distance. This also provides diverse outcome depending on what has been chosen.

**Radial Basis Function Network(RBFN)**: RBFN measures similarity between the training set and the prototype, assigns weights, and classifies the test set using them [7, 8]. Concretely, the RBFN consists of an input layer, a hidden layer with RBF neurons, and an output layer. Each node of the RBF neurons has a prototype, which is composed of training set. Generally, $k$-means clustering is used to extract representative data from the training set. To classify inputs, each RBF neuron calculates the Euclidean distance between the input vector and the prototype. The output of the RBF neuron has a value between 0 and 1 which displays the amount of similarity. The input is then classified into the class with highest score by computing the value of each RBF neuron and the weight of the output node. Each output node is scored using a weighted sum, and when training, the weights are updated through back propagation. Fig. 4 demonstrates the RBFN overall. One output node exists for each class or category of data. Each node $x$ of the input layer is connected to every RBF neurons $\{h_j\}_{j=1}^{m}$ whose outputs are computed with weights $\{w_j\}_{j=1}^{m}$ and attains the output node $f(x)$ of the output layer.

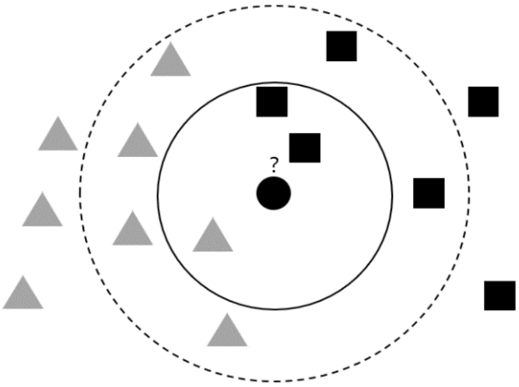

Fig. 3. Illustration of simple $k$-NN algorithm.

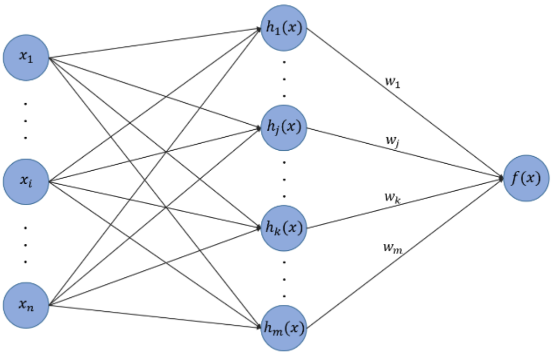

Fig. 4. Typical architecture of RBFN.

$k$-means clustering is employed to obtain prototype data from the training set. It is an algorithm that creates $k$ clusters among the data points. Specifically, each cluster has a center, and the data points are assigned to the cluster represented by the closest center. When the data points arranged in the same center gather, they form a cluster.

## III. Motivation for Improving Accuracy and Robustness of Non-parametric Models

### A. *Accuracy and Robustness of Parametric and Non-Parametric Models.*

Non-parametric models are robust against adversarial example attacks than parametric models, but there are still some inevitable issues. The first problem is having lower accuracy when classifying clean test sets than parametric models. Before, to explain the experimental environment of Table 1, adversarial images (FGSM, JSMA) are established based on "Neural Network (Basic)". The $k$-NN model utilized for classification used $k$ as 1 and RBF model contains 300 cluster centers as prototypes. As seen in Table 1, both neural networks have stable prediction accuracy of 0.9815 and 0.9811 when predicting pure MNIST test set. On the other hand, non-parametric models are quite unstable on predicting clean MNIST test set which shows accuracy of 0.9577 for $k$-NN and 0.9506 for RBFN. Non-parametric models give unacceptable accuracy to occupy as a substitute for parametric models. The next problem for non-parametric models is that although they are robust than parametric models, their accuracy is still insufficient when categorizing adversarial examples. Prediction for adversarial examples of non-parametric models in Table 1 displays the problem. For FGSM image predictions, $k$-NN has an accuracy of 0.9141, and RBFN has an accuracy of 0.8771. For JSMA image predictions, $k$-NN has an accuracy of 0.9250, and RBFN

TABLE I. ACCURACIES OF PARAMETRIC AND NON-PARAMETRIC MODELS ON CLEAN IMAGES AND ADVERSARIAL IMAGES.

| **Models**<br>**(C = Number of Cluster)** | **Clean Acc.** | **FGSM Acc.** | **JSMA Acc.** |
|---|---|---|---|
| Neural Network (Basic) | 0.9815 | 0.3072 | 0.0078 |
| Neural Network (Sub) | **0.9811** | 0.6092 | 0.6780 |
| *k*-NN (k=1) | 0.9577 | **0.9141** | **0.9250** |
| RBFN (C=300) | 0.9506 | 0.8771 | 0.9200 |

has an accuracy of 0.9200. Given this level of robustness in the MNIST data, it will be difficult to operate feasibly in various fields even if the attack has been executed as black box attack.

### *B. Analysis of Adversarial Example Images*

In order to improve the accuracy and robustness of non-parametric models, we propose a method that segments images into small patches and using them as additional features for classifying the images. The idea was inspired by the observation that CNN utilizes small image patches to extract features from the image.

About how segmentation will affect the adversarial example attacks, we analyzed the images of them. The algorithms utilized to construct adversarial examples are FGSM and JSMA. Fig. 5 shows an example image and the adversarial examples established from the image. Also, below each image, Fig. 5 exhibits a patch selected from the image. As seen in Fig. 5, FGSM tends to modify the value of pixels rarely overall, but JSMA revises the value of certain pixels immensely. When observing the perturbation of adversarial images generated by FGSM, a more robust classification can be achieved if the features that human eyes can capture are found sufficiently. For JSMA, there may be the identical unmodified parts as the original image when partitioned into patches. The segmented patch of the image exhibits the feature of category "5" acceptably, and it does not interfere with perturbation caused by adversarial attacks.

## IV. Non-Parametric Models with Image Segmentation

For the segmentation, all images are parted into 20 patches of 14×14 size. By occupying segmentation method on image datasets, subdivisions of images are exercised for input of the model, which can be verified by patches. The proposed model is called *k*-NN segmentation in *k*-NN, and RBFN segmentation in RBFN.

*k*-NN segmentation has the equal process as regular *k*-NN when voting for majority. However, before implementing the *k*-NN algorithm, datasets of each class harnessed to train and classify are capsulized into clusters by *k*-means clustering. After this operation, dataset images are partitioned into 14×14 20 patches. Then, the *k*-NN algorithm is utilized between patches of the corresponding zone. Finally, classification is completed through the majority voting of *k*-NN output for each patch. Fig. 6 indicates the general approach of *k*-NN segmentation method.

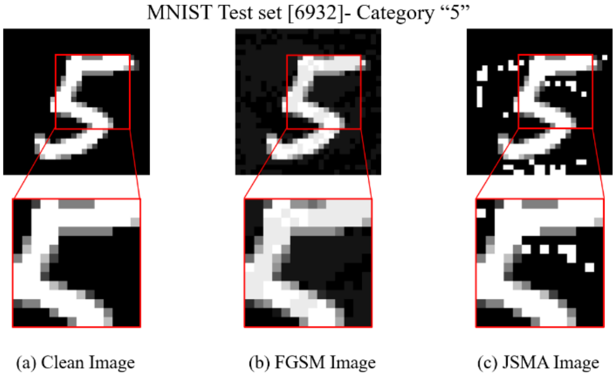

Fig. 5. Segmentation analysis of MNIST dataset.

The RBFN segmentation model refines training and test images to increase the weight of neurons with prototypes that are clearly distinguishable from other classes, and to reduce the weight of neurons with prototypes that are not. First, to generate the prototypes of RBFN, we create the same number of clusters for each class using *k*-means clustering in the training set. This allows clusters to distribute equally across the classes. Next step is to apply image segmentation. All prototypes, training and test images go through the equal segmentation process. After the segmentation method, the input layer is connected to the RBF layer by calculating the RBF value between the input and the prototype with the identical region. The RBF layer and the output layer are fully connected like the existing RBFN, and the weight of each edge is determined through training.

By using the RBFN segmentation model, the shape of the vector is combined differently than the existing model. For example, assuming that 100 clusters are created from the MNIST training set, a vector of (100, 784) shape is obtained as the cluster centers. By dividing this into 20 patches of 14×14 images each, we can use a vector of (100×20, 196) shape as the cluster centers, or prototypes. Similarly, training and test images are converted from (60000, 784) and (10000, 784) shaped vectors to segmented (60000×20, 196) and (10000×20, 196) shaped vectors. We then operate the RBF values between the input and the prototype with the corresponding area vectors, which generates 20 vectors of shape (60000, 100) and (10000, 100). This process attaches the input layer and the RBF layer into shape of (60000, 2000) and (10000, 2000). The RBF layer is linked to the output layer with a vector shape of (2000, 10), meaning that there are 20000 weights to train and take advantage of in the fully connected layer. Fig. 7 exemplifies the RBFN segmentation. In the input layer, Category "3" and "8" images from MNIST training set are segmented. RBF layer also have images that are segmented as prototypes. Segments with the equivalent section from different layers are connected, and the output of them are combined with weights of each category

Through this segmentation, non-parametric models can improve identifying specific features that are visible to human when classifying images. In addition, we enhance the accuracy as well as the robustness of the model by weighting the feature through training.

## V. Experimental Results

There are six models dealt with the classification of MNIST dataset and adversarial examples: basic neural network with a structure of 784×512×10, substitute model with the same structure as the basic neural network, *k*-NN, *k*-NN segmentation, RBFN, and RBFN segmentation. The images applied to evaluate models are clean MNIST test set, FGSM images, and JSMA images. For FGSM, parameter $\varepsilon$ has been set to 0.08 and for JSMA, the maximum perturbation value has been set to 0.1. We employed a black box attack because white box attack does not subsist for both *k*-NN and RBFN although heuristic algorithms prevail [9, 10]. Moreover, for fair comparison, we utilized a black box attack on the substitute neural network model.

As mentioned above, the model presented divide initial images into 20 patches of 14×14 images. Concretely, four horizontal sections and five vertical sections are used to make

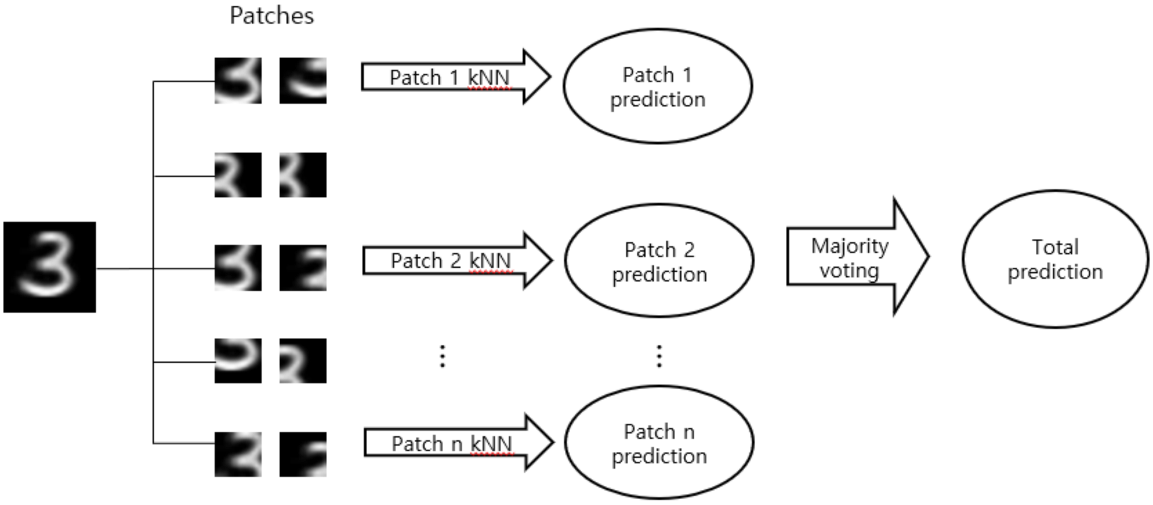

Fig. 6. Procedure of *k*-NN segmentation algorithm.

20 image patches. Also, both horizontal and vertical pixel ends of the full images are excluded from the section since most of them have blank spaces. Normal *k*-NN models generate training images with *k*-means clustering on MNIST training set, and then test images from MNIST test set apply the *k*-NN algorithm with the selected cluster center images. For both *k*-NN and *k*-NN segmentation models, four different *k* values (k=1, 3, 5, 7) are allocated as seen in Table 2. The original RBFN and RBFN segmentation models also have a variable number of clusters. Mentioned in Table 3, there are three different number of clusters which is 100, 200, and 300. Each number of clusters are assigned to both RBFN models.

The prediction for clean MNIST test set shows that the neural network model has the highest accuracy of 0.9811, followed by the RBFN segmentation (C=300) model with an accuracy of 0.9804. In light of this, the RBFN segmentation model appears to have similar classification capabilities as neural network. The *k*-NN segmentation model exhibits the best robustness against adversarial example attacks, in comparison to neural network and RBFN segmentation.

In Table 2, the *k*-NN segmentation models have higher accuracy than normal *k*-NN when predicting pure test set. Also, the best accuracies of classifying FGSM and JSMA images are 0.9262 and 0.9413, which states that the overall robustness has been improved in contrast to regular *k*-NN.

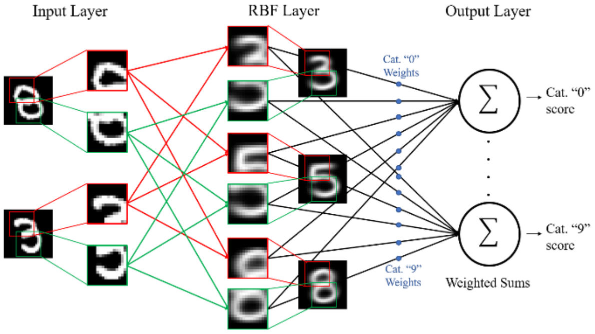

Fig. 7. Architecture of RBFN segmentation model.

TABLE II. ACCURACIES OF *K*-NN MODELS COMPARED TO NEURAL NETWORK MODELS WITH CLEAN TEST SET AND ADVERSARIAL IMAGES.

| **Models** | **Clean Acc.** | **FGSM Acc.** | **JSMA Acc.** |
|---|---|---|---|
| Neural Network (Basic) | 0.9815 | 0.3072 | 0.0078 |
| Neural Network (Sub) | **0.9811** | 0.6092 | 0.6780 |
| *k*-NN (*k*=1) | 0.9577 | 0.9141 | 0.9250 |
| *k*-NN (*k*=3) | 0.9524 | 0.9084 | 0.9211 |
| *k*-NN (*k*=5) | 0.9500 | 0.9022 | 0.9218 |
| *k*-NN (*k*=7) | 0.9449 | 0.9028 | 0.9149 |
| *k*-NN Segmentation (*k*=1) | 0.9600 | **0.9262** | **0.9413** |
| *k*-NN Segmentation (*k*=3) | 0.9539 | 0.9186 | 0.9358 |
| *k*-NN Segmentation (*k*=5) | 0.9521 | 0.9148 | 0.9299 |
| *k*-NN Segmentation (*k*=7) | 0.9502 | 0.9103 | 0.9263 |

Looking at Table 3, the RBFN segmentation model (C=300) have raised accuracy from 0.9506 to 0.9804 for predicting clean test set than the initial RBFN (C=300) model, which considerably advanced prediction rates. The accuracy for FGSM images operating RBFN segmentation (C=300) model is 0.9123, which has been boosted from 0.8771 accuracy of the RBFN (C=300) model. For JSMA, the RBFN segmentation

TABLE III. ACCURACIES OF RBFN MODELS COMPARED TO NEURAL NETWORK MODELS WITH CLEAN TEST SET AND ADVERSARIAL IMAGES

| **Models**<br>**(C = Number of Cluster)** | **Clean Acc.** | **FGSM Acc.** | **JSMA Acc.** |
|---|---|---|---|
| Neural Network (Basic) | 0.9815 | 0.3072 | 0.0078 |
| Neural Network (Sub) | **0.9811** | 0.6092 | 0.6780 |
| RBFN (C=100) | 0.9410 | 0.8650 | 0.9002 |
| RBFN (C=200) | 0.9541 | 0.8703 | 0.9141 |
| RBFN (C=300) | 0.9506 | 0.8771 | 0.9200 |
| RBFN Segmentation (C=100) | 0.9776 | 0.9039 | **0.9374** |
| RBFN Segmentation (C=200) | 0.9790 | 0.9094 | 0.9188 |
| RBFN Segmentation (C=300) | 0.9804 | **0.9123** | 0.8974 |

(C=100) has the accuracy of 0.9374, which has the best with regard to predicting JSMA images, but somewhat particular. This is because majority of models possessing high precision apply 300 clusters as prototypes, whereas the model with 100 clusters acquires the maximum accuracy for predicting JSMA images. The reason why this occurs is that unlike FGSM, JSMA modifies specific pixels notably. As the quantity of clusters increase, the images diverge, enhancing the possibility that certain patches containing perturbation will approach a dissimilar prototype than the initial one when reckoning the JSMA image and the prototype with Euclidean distance in RBF layer. As a whole, the RBFN segmentation models are displaying quite improved accuracy compared to the existing RBFN.

According to the experimental results, the proposed method indicates that segmentation improves accuracy and robustness against adversarial example attacks.

## VI. Conclusion and Future work

In this article, we introduced a segmentation scheme for non-parametric models to progress the accuracy and robustness against adversarial example attacks. The proposed segmentation splits the dataset images or cluster centers constituted by them into patch units, gaining extra details and features. This allows the non-parametric models to perform finer comparison among the data as well as more precise weighting. As a consequence, the models obtained elevated accuracy along with robustness than the initial *k*-NN and RBFN.

For the future work, it appears necessary to do research about white box adversarial example attack for *k*-NN and RBFN. Furthermore, we will raise accuracy for non-parametric models utilizing datasets other than MNIST.

## Acknowledgment

This research was supported by the MIST(Ministry of Science ICT), Korea, under the National Program for Excellence in SW (2015-0-00910) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation) and National Research Foundation of Korea under Grant NRF-2019R1A2C1005881.

## References

[1] Yizhen Wang, Somesh Jha, Kamalika Chaudhuri, “Analyzing the Robustness of Nearest Neighbors to Adversarial Examples”, arXiv:1706:03922

[2] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami, “Practical black-box attacks against machine learning”, In Asia Conference on Computer and Communications Security (ASIACCS), pp. 506–519. ACM, 2017.

[3] Ian J.Goodfellow, Jonathon Shlens and Christian Szegedy, “Explaining and Harnessing Adversarial Examples”, ICLR 2015.

[4] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in Security and Privacy (EuroS&P), 2016 IEEE European Symposium on. IEEE, 2016, pp. 372–387.

[5] E Fix and J.L. Hodges, “Discriminatory analysis, nonparametric discrimination, consistency properties”, Randolph Field, Texas, Project 21-49-004, Report 4, 1951.

[6] T. M. Cover and P. E. Hart, “Nearest Neighbor Pattern Classification”, IEEE Transactions on Information Theory, 1967.

[7] David S. Broomhead and David Lowe, “Radial basis functions, multi-variable functional interpolation and adaptive networks”, Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.

[8] Mark JL Orr, “Introduction to Radial Basis Function Networks”, Technical Report, Center for Cognitive Science, University of Edinburgh, 1996.

[9] Chawin Sitawarin, David Wagner, “On the Robustness of Deep K-Nearest Neighbors”, arXiv:1903.08333.

[10] Nicolas Papernot, Patrick McDaniel, “Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning”, arXiv:1803.04765.