MŰEGYETEM 1782

Budapest University of Technologies and Economics
Faculty of Electrical Engineering and Informatics
Department of Telecommunications and Media Informatics

Richárd Kiss

DEFENCE AGAINST ADVERSARIAL ATTACKS BY EXTENDING CLASS LABEL SET

Scientific Students' Association Report

SUPERVISOR

Dr. Gábor Szűcs

BUDAPEST, 2021

Contents

Abstract

The fast improvement of deep learning methods resulted in major breakthroughs in multiple fields of machine learning, including image classification. With novel deep learning architectures one can achieve very accurate classification performance, however these models often are sensitive to adversarial perturbations. The goal of Adversarial Attacks is to change the models output label by adding noise to the input. The attacker also minimizes the norm of the attack vector.

In my paper I cover two aspects of adversarial defence, the capability of detecting adversarial attacks and the robust classification performance. I compare two adversarial defence methods, one called NULL labelling, and an own method which is based on NULL labelling.

The NULL labelling method introduces one new label to the N label classification problem, which represents the presence of adversarial perturbation. My method, called 2N labelling, addresses the problem by extending the N label classification problem into a 2N label one. In this scenario each original class has two corresponding classes in the extended label set, one represents the original label without perturbation, the other one indicates the presence of perturbation. By design, with this approach one can predict the original label even if adversarial noise is present. To evaluate the effectiveness of the methods, I also trained a substitute model that does not use any adversarial defence method. I conclude my paper with the results of comparative testing.

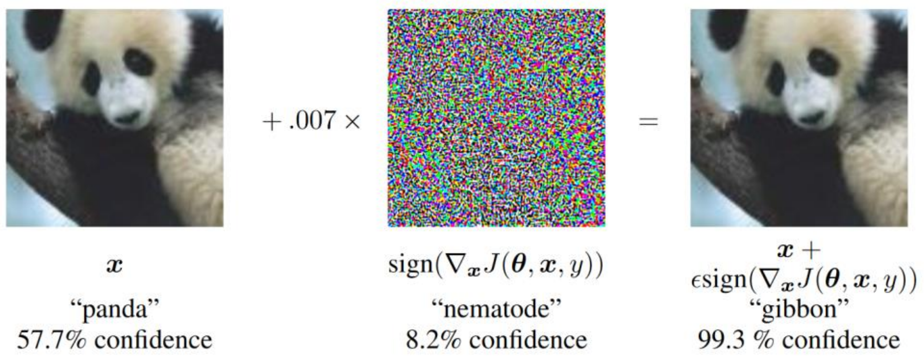
Absztrakt

A mélytanulási módszerek elterjedése nagy áttörést eredményezett a gépi tanulási problémáknál, többek között a képosztályozás területen is. A korszerű modell architektúrákkal nagyon pontos osztályozó modellek taníthatóak, viszont sok esetben ezek érzékenyek lesznek az ún. Adversarial támadásra. Egy ilyen támadás célja, hogy a modell kimenetét tetszőleges (de nem a helyes) osztályba átvigye és ezzel egyidőben minimális változtatást keverjen csak rá a bemenetre.

Dolgozatomban az ilyen jellegű támadások elleni védekezésnek két módját vizsgálom, a támadás észlelését, illetve a támadás melletti robusztus osztályozást. Erre a feladatra egy ismert módszer a NULL labelling. Ennek a módszernek a jóságát vizsgálom a TDK dolgozatomban, illetve ennek egy saját, továbbfejlesztett változatával is összehasonlítom. A NULL labelling módszer lényege, hogy az N osztályos osztályozási problémánál az osztályokat kiegészíti egy NULL osztállyal, ami azt hivatott jelezni, hogy a bemeneten támadó zaj található. A továbbfejlesztett saját módszer ötlete az, hogy az eredeti N osztályt 2N darab osztályra terjesztjük ki. Így minden eredeti osztályhoz két címke fog tartozni, az egyik az eredeti osztályt reprezentálja támadás nélkül, a másik ugyanezt támadás jelenlétével. A módszerek hatékonyságának értékeléséhez tanítok egy referencia modellt is, ami nem használ semmilyen Adversarial Learning módszert, az összehasonlító tesztelés eredményeivel zárom dolgozatomat.

1 Introduction

Adversarial Learning is a machine learning technique that attempts to fool the trained model by introducing perturbation to the input. Most machine learning techniques were designed to work on a specific problem set, where the training and test data are sampled from the same distribution. In most cases it can be shown that it is possible to mislead these models by supplying input data that is not from the same statistical distribution as the training data. We call an (input, noise) pair Adversarial Example [5] for a given model if the model correctly classifies the input but fails to classify the noised input.



$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Figure 1. Demonstrating effectiveness of adversarial perturbations. [2]

In most real-world scenarios, it can be crucial to train models that are robust against adversarial attacks [16]. In this scientific field there are many different approaches to train robust networks.

In my report I focus on image classification problems. My goal was to compare existing robust learning methods with my own approach. This report covers two aspects of adversarial defence, the capability of detecting adversarial attacks and the robust classification performance [10].

Earlier this year I was involved in writing an article [22] about a novel Adversarial Defence method using Gaussian pyramid [9] for noise reduction, this report is partly based on that paper.

2 Theoretical background

2.1 Adversarial Attacks

An adversarial attack tries to fool the classifier model by adding a carefully crafted noise to the input. The attacks can be divided into two groups based on the goal of the attacker [5]. The goal of *targeted* attacks is to find a noise vector (with a minimal norm) that it changes the models output to a specific class. In case of *untargeted* attacks, we want to change the models output to any class but the correct one. The attacks can be formally written as:

$$\boldsymbol{\delta}^{opt} = \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\| \text{ such that } C(\boldsymbol{x} + \boldsymbol{\delta}) = y^* \quad \text{(targeted)} \tag{2.1}$$

$$\boldsymbol{\delta}^{opt} = \min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\| \text{ such that } C(\boldsymbol{x} + \boldsymbol{\delta}) \neq C(\boldsymbol{x}) \quad \text{(untargeted),} \tag{2.2}$$

where $\boldsymbol{\delta}^{opt}$ is the optimal noise vector, $C(\boldsymbol{x}) = \arg\max F(\boldsymbol{x})$ and F is the classifier function that returns a probability distribution over the possible labels, y^* is the desired label by the attacker. The norm which we minimize can be any vector norm, in this report L_∞ was used.

A perturbation and input pair ($\boldsymbol{x}$, $\boldsymbol{\delta}$) is called an adversarial example if it successfully fools a given model, while a human can still correctly classify the image.

In case of model availability there are three threat models. A white-box [11] attack assumes access to the whole model, a score-based threat model assumes access to the output probability vector and finally, the decision-based or black-box [13] threat model assumes access only to the predicted labels.

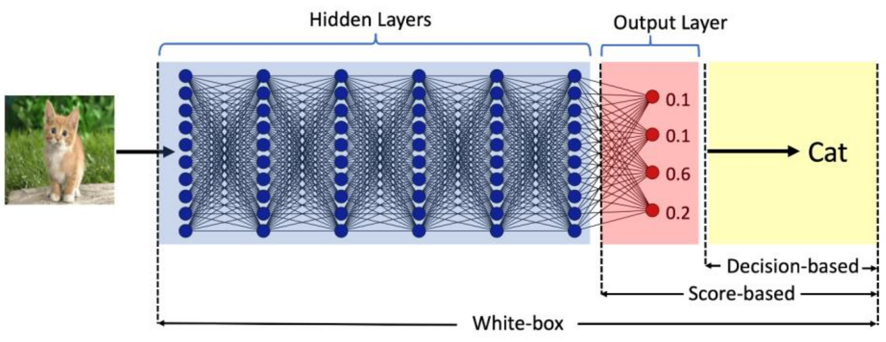


Figure 2. Threat models. [5]

In my report an adversarial attack approach with untargeted attacks was used, that can be applied in white-box scenarios.

2.1.1 Fast Gradient Sign Method (FGSM)

With FGSM [2], we can search for adversarial perturbation in a given L_∞ bounding space of the original input. This means that the attack noise vectors norm can't be higher than a given threshold.

To understand the idea behind FGSM, first assume that the classification function is linear. In this case the model output for an adversarial input can be written as:

$$F(\widetilde{\boldsymbol{x}}) = \boldsymbol{w}^T \widetilde{\boldsymbol{x}} = \boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{w}^T \boldsymbol{\delta}, \tag{2.3}$$

where $\boldsymbol{w}$ is the model parameter vector and $\boldsymbol{\delta}$ is the noise vector. The output of the model changes by $\boldsymbol{w}^T \boldsymbol{\delta}$ compared to $F(\boldsymbol{x})$. We can maximize the change subject to the norm constraint by choosing $\boldsymbol{\delta} = \operatorname{sign}(\boldsymbol{w})$. This results in a change equal to the sum of absolute values of the model weights. We can see that linear models can have adversarial examples with norm lower than the resolution if the dimensionality of the problem is high enough (the resolution of the images in our case).

It turns out linear attacks are quite efficient against non-linear functions (such as a neural network). The reason behind this is that activations (non-linearities) are designed to work close linearly in order to make optimization easier (for example ReLU or non-saturated logistic sigmoid). This suggests that these models are also vulnerable to linear attacks at small scale [2].

The cost function can be linearized around the current value of the model parameters, and we can get an optimal max-norm constrained perturbation as:

$$\boldsymbol{\delta} = \varepsilon \operatorname{sign}\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\big) \tag{2.4}$$

Here $\boldsymbol{\delta}$ is the adversarial perturbation, ε is the amplitude of the attack and $\boldsymbol{\theta}$ is the model parameter vector. J is a cost function which penalizes the divergence from the desired class. In this report I used cross-entropy as a measure of divergence:

$$H(p, q) = -\sum_{i=1}^{n} p_i \log q_i \tag{2.5}$$

If y is the correct class and y^* is the desired class for the targeted attack, the cost function can be written as follows:

$$J(\boldsymbol{\theta}, \boldsymbol{x}, y) = \begin{cases} -\log F_{y*}(\boldsymbol{x}) & \text{(targeted)} \\ \log F_y(\boldsymbol{x}) & \text{(untargeted)} \end{cases} \tag{2.6}$$

Note that the negative sign disappears for the untargeted attack, this is because we want to maximize the entropy for this type of attack.

There are extensions to this method which use an iterative approach for generating adversarial noise, for example Iterative Fast Gradient Sign Method [8] or Projected Gradient Descent [19]. With these methods one can get a noise vector closer to optimal at the cost of computing gradients multiple times.

2.1.2 Smooth Targeted Attack Based on Gradient Method (STG)

With Smooth Targeted Attack Based on Gradient Method (STG) method [4], one can generate adversarial noise for a given L_0 constraint. Minimizing the attack's L_0 norm means that we only want to modify the minimal number of elements in the input vector not bothering the magnitude of each modification.

This attack is an iterative approach for finding adversarial examples, meaning that it modifies one input feature at a time. A detailed description of the algorithm can be found in the Hosseini's paper [3].

2.2 Defence against adversarial attacks

In the scope of this research, I addressed two different existing techniques for robust model learning.

2.2.1 Adversarial training

Adversarial training (or adversarial learning) [2] is an adversarial defence method for L_∞ bounded adversarial attacks. Using adversarial learning one can train a robust classifier as we will see in 5.3. Adversarial training uses an extended loss function to learn a robust model. The extended loss function is defined as follows:

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J\big(\boldsymbol{\theta}, \boldsymbol{x} + \varepsilon \operatorname{sign}\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\big), y\big), \tag{2.7}$$

where J is the original, $\tilde{J}$ is the extended loss function and α is the importance of the original sample against the adversarial sample. With this we can prepare the model to classify images correctly even with adversarial noise present, but this method does not allow us to detect adversarial attacks.

2.2.2 NULL labelling

2.2.2.1 Extending label set

NULL labelling [4] extends the N label classification problems label set with a new label (NULL), which represents the presence of adversarial noise.

$$S_{NULL} = \{C_1, C_2, \dots, C_N, C_{NULL}\}, \tag{2.8}$$

where $\{C_1, C_2, \dots, C_N\}$ is the original N label classification problem's label set and S_{NULL} is the extended label set for NULL labelling.

2.2.2.2 Training Strategy

The technique also defines a training strategy. First the model is pre-trained on the original dataset, without any perturbations until the model achieves a certain accuracy in the N class classification problem.

In the next phase the model is trained on clean samples with α probability, and on adversarial examples (generated with STG method) with 1- α probability. We assign the desired NULL label probabilities for each adversarial example as follows:

$$P(c = \text{null for attack } \boldsymbol{\delta}) = f(\boldsymbol{\delta}) = \frac{|\{\boldsymbol{\tau}: \|\boldsymbol{\tau}\|_0 < \|\boldsymbol{\delta}\|_0 \mid \boldsymbol{\tau} \epsilon \Delta\}|}{|\Delta|}, \tag{2.9}$$

where Δ is the set of perturbations for adversarial examples on the validation set. The desired probability distribution used to calculate the cross-entropy loss for the samples without perturbation:

$$P(c = i) = \begin{cases} q, & \text{if } i = c_{original} \\ 0, & \text{if } i = \text{NULL} \\ \dfrac{1-q}{\text{N}-1}, & \text{else} \end{cases} \tag{2.10}$$

By setting the value of q less than 1, this method is called label smoothing [12] which prevents the model from learning to give overconfident predictions [14]. The probabilities for the samples with perturbation are calculated as follows:

$$P(c = i) = \begin{cases} q\big(1 - f(\boldsymbol{\delta})\big), & \text{if } i = c_{original} \\ f(\boldsymbol{\delta}), & \text{if } i = \text{NULL} \\ \dfrac{(1-q)\big(1 - f(\boldsymbol{\delta})\big)}{\text{N} - 1}, & \text{else} \end{cases} \tag{2.11}$$

By using pre-training, the overall learning process can be sped up, since initially the adversarial examples generated with the model with randomly initialized parameters do not resemble those of the final classifier.

2.3 Multi-Task learning

Multi-Task learning [15] means that the model is trained to solve multiple tasks simultaneously. This can result in improved learning efficiency and accuracy for the models trained this way if the tasks are correlated.

NULL labelling can be considered as Multi-Task learning: the model learns to label the samples with the original labels set and to calculate the probability of the attack at the same time.

2.4 Measuring robustness

2.4.1 Robust classification performance

It turns out that different models trained to solve the same problem tend to be vulnerable to similar perturbations. If we have two models, M_a and M_b, and X_a is a set of adversarial examples on M_a, the ratio of $|X_b|$ / $|X_a|$ is called *transferability* of X_a to M_b (X_b is a subset of X_a and each element in X_b is an adversarial example for M_b). If a model has significantly lower transferability, we expect it to be more robust against adversarial attacks [3], thus transferability can be a good measure of robustness.

Similarly, we can calculate accuracies for both models for a given attack norm, the higher the accuracy, the more robust the model is considered against the attack. Accuracy can be calculated as follows:

$$\text{accuracy} = \frac{\text{TP+TN}}{\text{P+N}}, \tag{2.12}$$

where TP and TN are the number of true positive/negative samples, P and N are the number of positive/negative samples.

2.4.2 Measuring attack detection performance

One other aspect of adversarial defence is detecting attacks. In this scenario the goal is to decide whether a particular input contains adversarial perturbation. This is a binary classification problem.

2.4.2.1 Receiver Operating Characteristic and Area Under Curve (ROC-AUC)

A Receiver Operating Characteristic curve is a graphical plot that illustrates the performance of a binary classifier as its decision threshold value is varied. The curve is created by plotting the true positive rate against the false positive rate at various threshold settings.

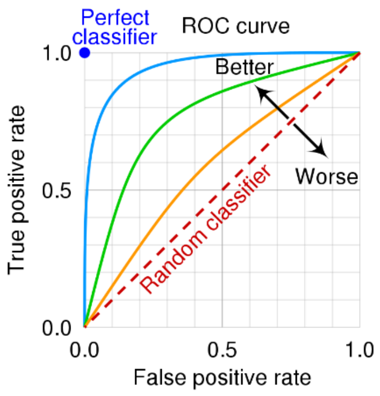


Figure 3. Receiver Operating Characteristic curve.[1]

The area under the ROC curve (also called AUC – Area Under Curve) is equal to the probability, that the given classifier will rank a randomly chosen positive sample higher than a randomly chosen negative one. It can be calculated as follows:

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}\big(\mathrm{FPR}^{-1}(x)\big)\,dx, \tag{2.13}$$

where $\mathrm{TPR}(x)$ and $\mathrm{FPR}(x)$ are the true positive and false positive rates with a classification threshold x.

For an ideal classifier the AUC value would be 1. This would mean that the classifier can label each input perfectly. On the other end, a classifier that randomly assigns positive or negative labels to each input would result a 0.5 AUC value.

---

[1] https://en.wikipedia.org/wiki/Receiver_operating_characteristic

2.5 Techniques used in evaluation

To ensure that the evaluated models were trained near convergence, and the experimental results were valid I used the following two techniques:

2.5.1 K-fold Cross Validation (CV)

Cross Validation [1] is a machine learning technique for assessing the effectiveness of a model. K-fold CV splits the data into K folds and performs training K times (on different model instances). Each training round uses a different subset of the K fold as training data and the remaining fold will be the test data as Figure 4 presents.

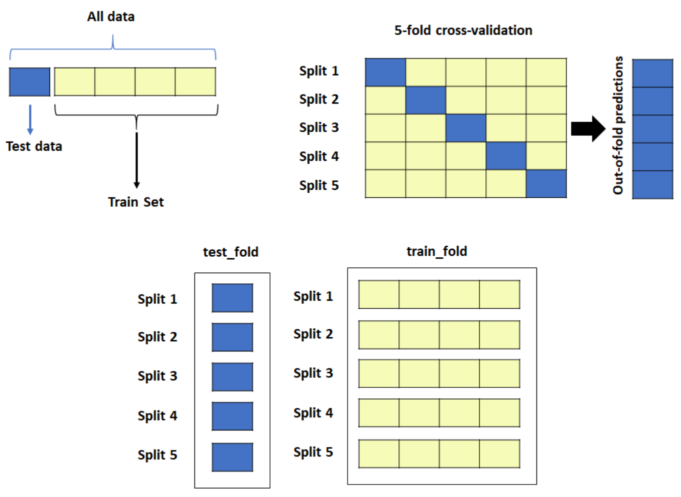


Figure 4. Visualization of 5-fold cross validation.[2]

2.5.2 Early stopping

Early stopping [18] is a regularization method used to avoid overfitting. In Early stopping, during training after each epoch the model performance is measured (particularly on a validation set), and if it's better than the previous best, a model checkpoint is created. If the validation loss does not decrease for a certain number of consecutive epochs, the training is stopped. After the training is stopped, the latest checkpoint (which contains the best performing model parameters) is returned.

[2] https://drzinph.com/nested-cross-validation-cross-validation-series-part-2a/

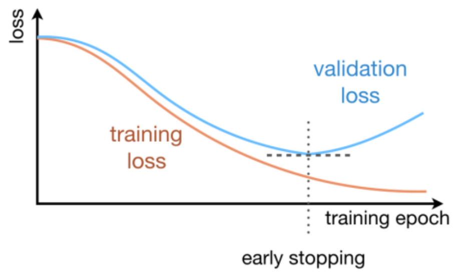


Figure 5. Visualization of early stopping regularization method. [23]

3 New defence method with extending class label set

3.1 NULL labelling variant for L_∞ bounded attacks

In this report I focused on L_∞ bounded attacks, thus the NULL labelling training strategy had to be modified in order to support L_∞ instead of L_0 bounded attacks. I achieved this by replacing the STG method to FGSM for generating adversarial inputs. I used to sample the ε_{FGSM} attack amplitude from a uniform distribution. I also had to define my own function to assign NULL label probabilities for the FGSM perturbations:

$$f(\boldsymbol{\delta}) = \frac{\|\boldsymbol{\delta}\|_\infty}{\max\{\|\boldsymbol{\tau}\|_\infty | \boldsymbol{\tau} \epsilon \Delta\}} \tag{3.1}$$

This function normalizes the attack amplitude for each sample by dividing with the largest ε_{FGSM} used for generating the perturbations.

3.2 2N labelling defence method

My idea for a new defence method is similar to NULL labelling, but instead of extending the original label set with one (NULL) label, the proposed method duplicates each original label (S_{2N} denotes the set of labels for 2N labelling).

$$S_{2N} = \{C_1, C_2, \dots, C_N, C_{N+1}, C_{N+2}, \dots, C_{2N}\} \tag{3.2}$$

Here C_i is the original label (at index i) and C_{N+i} stands for the same label but with adversarial perturbation present.

In higher amplitude adversarial attacks, the output distribution of the NULL labelling model shifts towards the NULL label, making it harder to determine the original label of the input. In contrast my method directly learns to classify these high amplitude attacks correctly.

This technique also allows us to detect the presence of adversarial noise. If we calculate the sum of the second N labels, we get the probability whether noise is present.

$$P(\text{attack}) = \sum_{i=N+1}^{2N} F_i(\boldsymbol{x}), \tag{3.3}$$

where $F_i(\boldsymbol{x})$ is the value at index i of the output probability distribution vector of the model.

I also used pre-training and label smoothing for the 2N labelling technique. The label smoothing was performed as follows:

$$P(c = i) = \begin{cases} q, & \text{if } i = \hat{c} \\ \dfrac{1-q}{2\mathrm{N}-1}, & \text{else} \end{cases} \tag{3.4}$$

where $\hat{c} = c_{original}$ if there is no perturbation, otherwise $\hat{c} = c_{original} + \mathrm{N}$.

3.3 Proving that filtering can increase predictive performance

In this research I evaluate the performance of defence methods with filtering out adversarial inputs. By filtering we have an extra opportunity to prevent the attack to be successful even if we must discard the input. In real-world scenarios it can be more important to not let attacks succeed than to evaluate each input.

Let us denote the ratio of the perturbed inputs, the number of all inputs, the accuracy on original instances, the accuracy on perturbed instances for the attack by p_a, N, acc_{ori}, acc_{att} respectively.

If there is no attack, the number of the true positive original instances (TP_{ori}) can be calculated from the accuracy:

$$TP_{ori} = acc_{ori} \cdot N \cdot (1 - p_a) \tag{3.5}$$

In case of the attack, the number of the true positive perturbed instances (TP_{att}) also can be calculated from the accuracy:

$$TP_{att} = acc_{att} \cdot N \cdot p_a \tag{3.6}$$

Before the filtering the accuracy on the mixed data (original and perturbed instances) is equal to sum of true positive instances divided by the number of all instances:

$$acc_{before} = \frac{TP_{ori} + TP_{att}}{N} = acc_{ori} \cdot (1 - p_a) + acc_{att} \cdot p_a \tag{3.7}$$

The filtering process discards some perturbed instances (the ratio of the discarded inputs is the recall of the perturbed class in the binary classification, we denote this by R_{att}), the ratio of the not detected (i.e. not discarded) perturbed instances and all perturbed instances is equal to $1 - R_{att}$; and the filtering process passes original instances, the ratio of them denoted by R_{ori}, because this is the recall of the original instances. After the filtering the accuracy can be expressed by the recall of the perturbed and original classes:

$$acc_{after} = \frac{TP_{ori} \cdot R_{ori} + TP_{att} \cdot (1 - R_{att})}{N \cdot (1 - p_a) \cdot R_{ori} + N \cdot p_a \cdot R_{att}} \tag{3.8}$$

After simplification, we can write acc_{after} as follows:

$$acc_{after} = \frac{acc_{ori} \cdot (1 - p_a) \cdot R_{ori} + acc_{att} \cdot p_a \cdot (1 - R_{att})}{(1 - p_a) \cdot R_{ori} + p_a \cdot (1 - R_{att})} \tag{3.9}$$

If we examine when will the $acc_{before} < acc_{after}$ inequality be true, we can get to the following inequality:

$$0 < p_a \cdot (1 - p_a) \cdot (acc_{ori} - acc_{att}) \cdot (R_{ori} - \bar{R}_{att}), \tag{3.10}$$

where $\bar{R}_{att}$ is the ratio of not discarded perturbed instances: $\bar{R}_{att} = 1 - R_{att}$. The step-by-step derivation can be seen in the Appendix.

Here $p_a \cdot (1 - p_a)$ is always positive, and we can assume that the $acc_{ori} - acc_{att}$ expression is also positive (for the 2N and NULL models this can be observed to be true in 0). Then the inequality can be reduced to:

$$R_{ori} < \bar{R}_{att} \tag{3.11}$$

This means that we get an increase in accuracy when the ratio of discarded non-perturbed samples is less than the ratio of the NOT discarded perturbed samples.

3.4 Used model architecture

In this research I used the VGG-19 model architecture [17] to evaluate the defence methods in question. The VGG architecture consists of convolutional and pooling layers. There are multiple configurations of VGG which differ in the number of weight layers. For this research I used 19 layers.

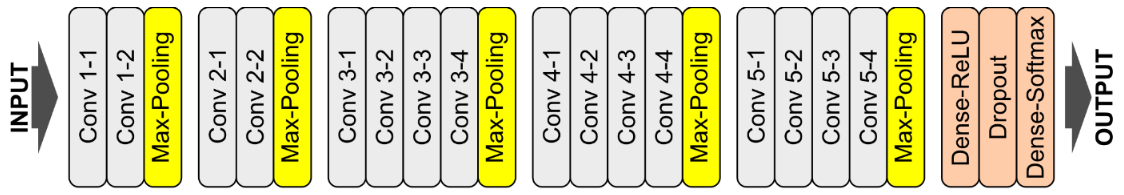


Figure 6. VGG architecture. [3]

In the used implementation each model layer consists of a convolutional layer followed by a batch normalization layer and a ReLU (Rectified Linear Unit) activation.

To support NULL and 2N labelling the model architecture had to be modified only at the final layer. In case of NULL labelling the length of the final layer's output vectors is N+1 (since the dataset has N classes), and for the 2N model this is 2*N, where N is the number of classes.

3.5 Training the models

The models were trained for 200 epochs with early stopping. I used cross entropy as a loss function and trained the models with Stochastic Gradient Descent [20] with Cosine Annealing learning rate [21].

When training the adversarial learning (ADV) model, I set alpha=0.5, and for the NULL and 2N labelling models I added noise to 50% of the images. Since I used 50% perturbed to non-perturbed ratio during evaluation as well, this can lead to better results as if I had no knowledge about the ratio of adversarial samples in the evaluation set, however I did not include other ratios in the scope of this report.

During training the epsilon value of the FGSM perturbations was drawn from a uniform distribution with parameters 0 and 0.3 for each training sample.

4 Evaluation plan for filtering and N class classification

The goal of using adversarial defence methods is to prevent accuracy decrease while lowering the transferability of models in an environment where the models can be exposed to adversarial attacks.

We can protect our systems by training robust models and/or by detecting and discarding adversarial inputs. As shown in section 3.3, filtering inputs can increase the overall predictive performance in adversarial setting.

I will show that is possible to achieve higher accuracy on adversarial inputs with filtering (see in 5.3) than without filtering, by leveraging the multi-task learning effect.

To generate adversarial perturbations a substitute model (with VGG-19 architecture) was used. The substitute model was trained on the whole training set with early stopping using the validation set.

4.1 Filtering

4.1.1 Calculating attack probabilities

The filtering is a binary classification task, each model must predict whether the input was perturbed or not. More precisely, we need to calculate the probability whether a given input contains adversarial perturbation. By defining a threshold probability, we can assign labels (perturbed or not perturbed) to each input. The inputs predicted as perturbed are excluded from the following classification task.

The NULL and 2N labelling techniques are able to calculate attack probabilities by design, however for the adversarial learning (ADV) model I had to train a separate binary model (BIN) specifically for the filtering task. For the BIN model I used VGG-11 architecture (the shallower network worked out best for me). The BIN model was trained by adding adversarial perturbations with various amplitudes to the inputs, generated with 5 different VGG-19 models (these models were also trained to solve the N class classification problem).

Now for each model we can calculate attack probabilities as follows:

Method	Output transformation to attack probability
BIN	$P(\text{attack}) = F(\boldsymbol{x})$
NULL	$P(\text{attack}) = F_{NULL}(\boldsymbol{x})$
2N	$P(\text{attack}) = \sum_{i=N+1}^{2N} F_i(\boldsymbol{x})$

Table 1. How the probability of the attack was calculated with each method.

Capability of each model can be evaluated on the binary classification task by comparing the ROC-AUC values.

4.1.2 Determining threshold probability

Selecting a threshold value for the binary decision of filtering is a domain specific task. Depending on the use-case we can define a maximum ratio of discarded inputs. On a calibration dataset we can calculate the optimal threshold (T^{OPT}) by finding that threshold value for which the accuracy is maximal and satisfies the maxima constraint for the ratio of discarded inputs. Note that on other datasets the actual discard rate can be different, but with sufficient amount of calibration points the calibration discard rate is a good estimate for the actual discard rate.

4.2 N class classification

The next task is to predict the label for inputs which the filter model did not discard with the original N labels. For this task I had to transform the output of the NULL and 2N models to a probability distribution over the original N labels.

Method	Output transformation to N class probabilities
ADV	$P(c = i) = F_i(\boldsymbol{x})$
NULL	$P(c = i) = \frac{F_i(\boldsymbol{x})}{\sum_{j=1}^{N} F_j(\boldsymbol{x})}$
2N	$P(c = i) = F_i(\boldsymbol{x}) + F_{i+N}(\boldsymbol{x})$

Table 2. How the probability distribution over the original N labels were obtained for each model.

5 Experimental Results

In this section I present and explain the results of the evaluation of the adversarial defence methods.

5.1 Dataset

For this report I used the CIFAR-10 [7] dataset to evaluate the mentioned adversarial defence methods. The dataset consists of 60000 32x32 RGB in 10 classes, with 6000 image per class.

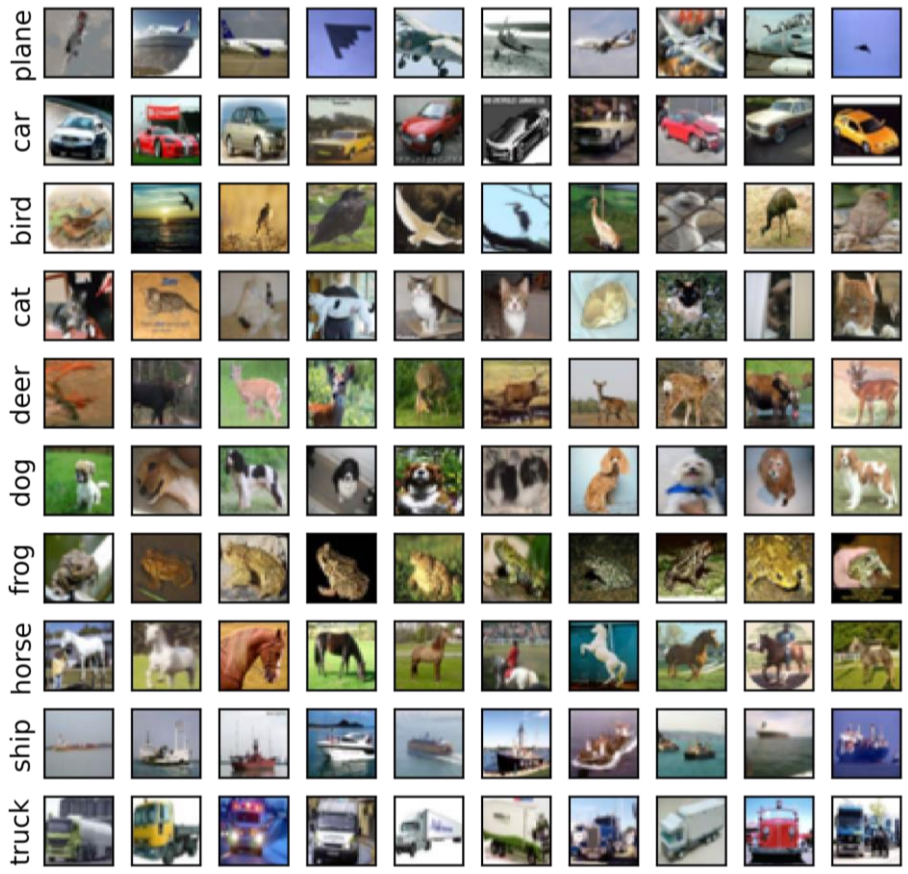


Figure 7. Sample images from the CIFAR-10 dataset from each class.

The dataset was split into training (consists of 50000 images) and validation (consists of 10000 images) sets. There were 3 models in this comparison, for each model I trained 5 different instances (for the 5-fold CV):

Notation	Defence method
ADV	Adversarial learning
NULL	NULL labelling
2N	2N labelling

Table 3. Notation of compared methods in the following diagrams.

5.2 ROC-AUC in binary classification task

Here I show experimental results for the filtering subtask of the decomposed filtering and N class classification complex task.

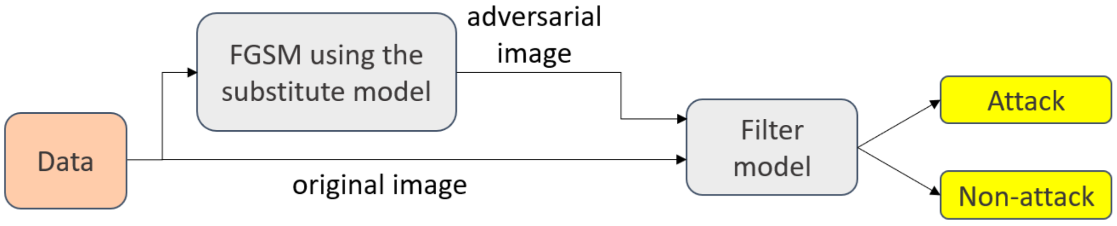


Figure 8. Block diagram of the evaluation pipeline for the binary classification (filtering) task.

In this section I calculated the mean ROC-AUC values for each defence method. The attack probabilities used for the ROC curve were calculated as shown in Table 1. I used the validation set to evaluate models, 50% of the samples contained adversarial perturbation (generated with FGSM algorithm using the substitute model) with ε drawn from a uniform distribution with parameters 0, 0.5.

Model	ROC-AUC
2N	0.879
NULL	0.934
BIN	**0.941**

Table 4. Mean ROC-AUC values for each binary classification method detailed in 4.1.

In Table 4 we can observe that the BIN model achieves the highest binary classification performance. However, in the following sections we will get a closer look on the overall N class filtered accuracy and transferability, which will unveil that there are more aspects in filtered classification than only good filtering.

5.3 Accuracy

In this and the following (5.4) sections I used an evaluation pipeline shown in the next diagrams to measure robust classification performance (Figure 9) and the performance using filtering (Figure 10).

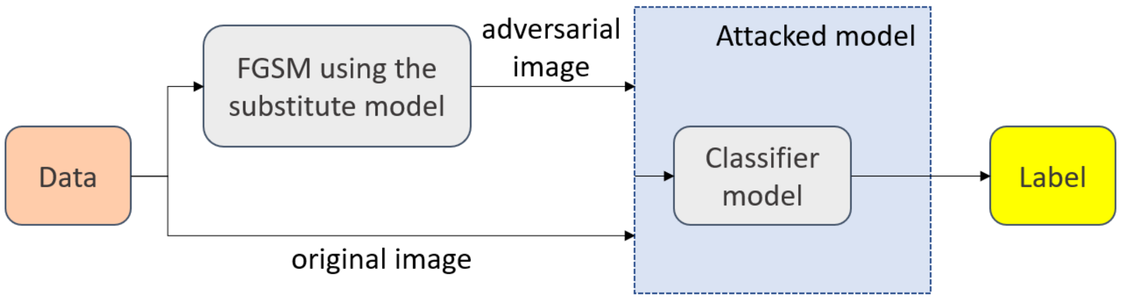


Figure 9. Block diagram showing the evaluation pipeline for the robust classification task.

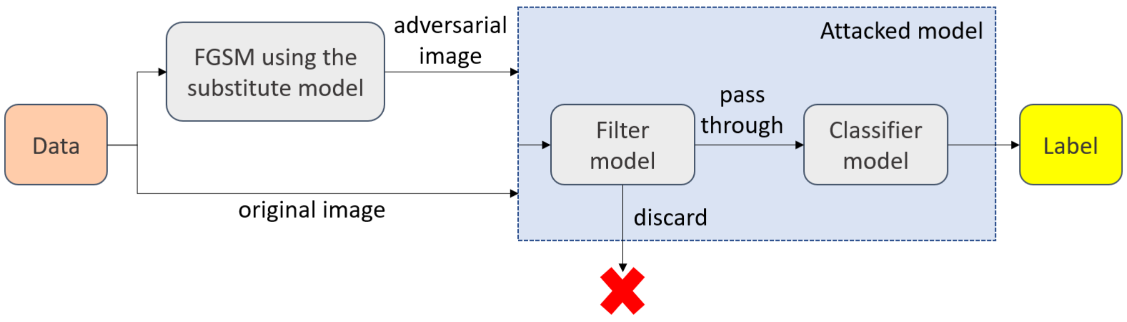


Figure 10. Block diagram showing the evaluation pipeline for the filtering and N class classification task.

I calculated the mean accuracy for each method against various attack amplitudes as can be seen in Figure 11. The vertical lines indicate the minimal and maximal accuracy values for each method (since there were multiple model instances that resulted in different accuracy due to cross validation).

In 3.3 I explored the possibility of filtering to increase accuracy from an analytical point of view, here I will show empirical evidence that this really can be achieved.

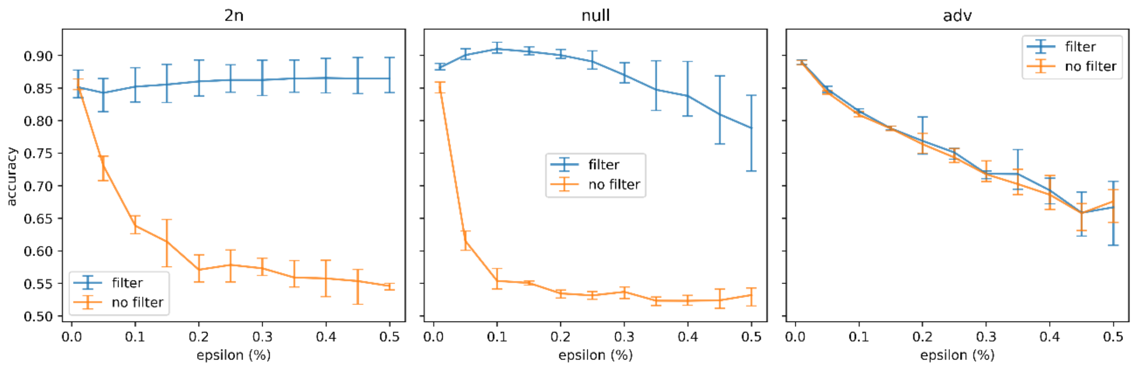


Figure 11. Mean accuracy for each method with and without using filtering.

As we can see in Figure 11, the increase in accuracy for 2N and NULL labelling with filtering is conspicuous, however for the ADV models this effect was much smaller.

The cause for this phenomenon can be Multi-Task learning; while we train the 2N and NULL model instances to solve the original N label classification problem and detect adversarial noise simultaneously, the ADV models and BIN filter model were trained completely separately (having no chance for one task to improve performance for the other).

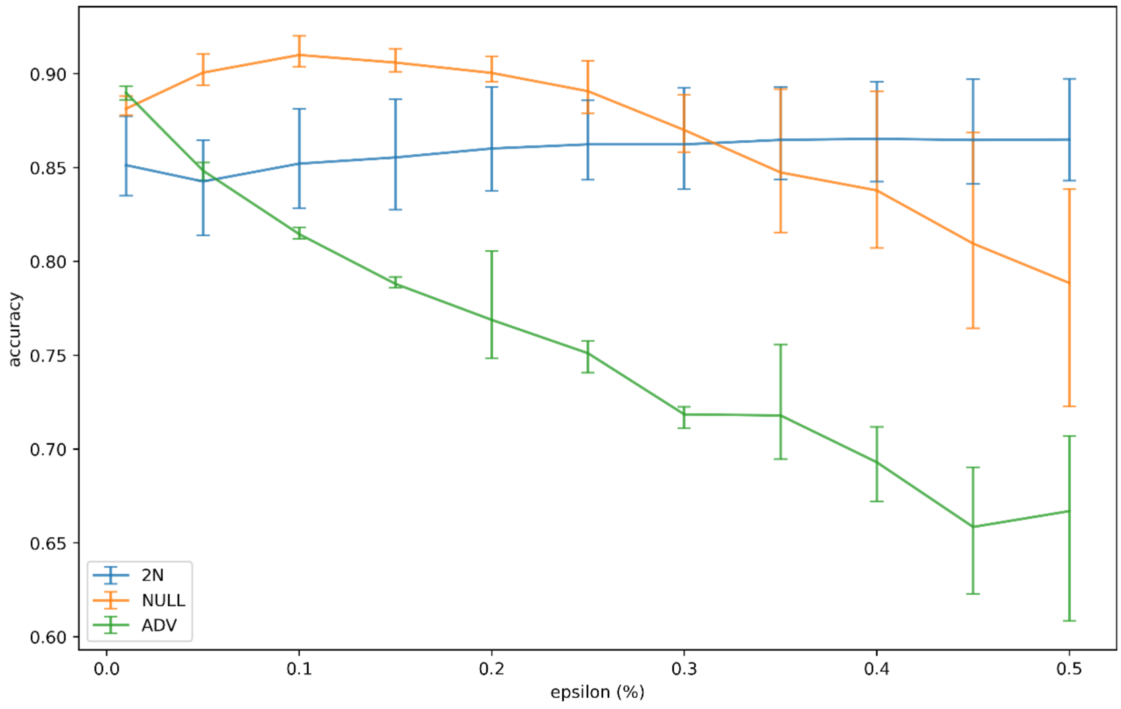


Figure 12. Mean accuracy of defence methods with filtering.

In Figure 12 we can see that for lower attack amplitudes NULL labelling can achieve higher accuracy, however only with the 2N labelling method can we observe that the accuracy is not sensitive to the amplitude of the attack. This means that models trained with 2N labelling technique can operate on a nearly constant accuracy level regardless the presence or amplitude of the attack. In a real-world scenario this would be the desired behaviour, if an attacker has feedback on the attacks success rate from the attacked system, he could find out which amplitudes work best against the system. If the performance of the system is independent from the attack amplitude, it is much harder for the attacker to find adversarial perturbance that can mislead our system.

To further investigate the ability of each method to learn an attack amplitude insensitive model, I also calculated the correlation between the amplitude of the attack and the corresponding accuracy, and the maximal absolute difference (range) for accuracy.

Method	Correlation	Range
2N	**0.748**	**0.023**
NULL	-0.792	0.121
ADV	-0.95	0.231

Table 5. Correlation: Mean correlation between the attack amplitude and accuracy. Range: maximal absolute difference of accuracy over the whole range of attack amplitudes (this is also averaged for each model instance).

From Table 5 we can observe that for NULL and ADV methods the correlation is negative, meaning that increasing the amplitude makes a decrease in accuracy. Also, the range is much higher for these methods than with 2N labelling.

By using 2N labelling, the resulted models show a positive correlation between the attack amplitude and accuracy. Considering the significantly lower range in accuracy, this confirms my theory that my method is not only less sensible against adversarial attacks, but the accuracy also tends to increase with the amplitude of the attack (a non-negative correlation means that the defence is efficient).

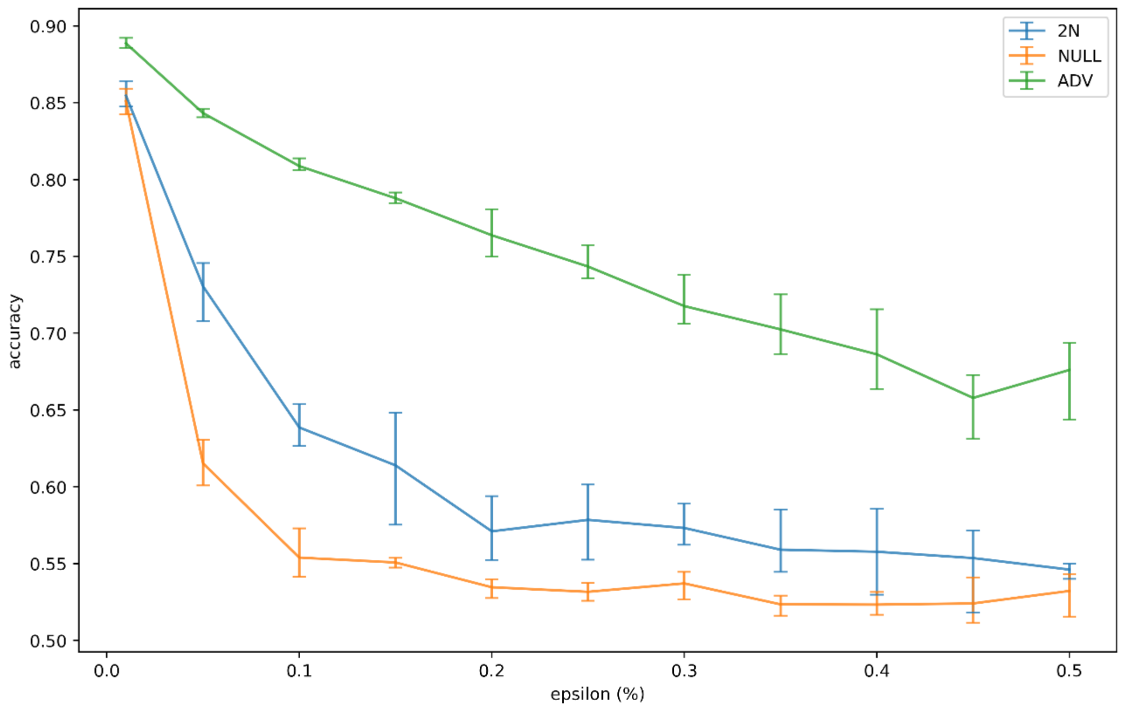


Figure 13. Mean accuracy of defence methods without filtering.

In Figure 13 we can observe that if filtering is not an option, model trained with adversarial learning are the most robust in the original N label classification problem. Also, 2N labelling slightly outperforms NULL labelling in the original task, without filtering. It is surprising, that even if the ADV model is the most robust from all and the

BIN filtering model is also almost as good as NULL labelling in the filtering task, the combination of these two models does not even get close in terms of performance in the complex filtering and classification task (as a possible result of Multi-Task learning).

Model	Accuracy
2N	0.859
NULL	**0.867**
ADV	0.756

Table 6. Mean accuracy for each method using filtering if the attacks are sampled from an uniform distribution with parameters 0 and 0.5.

Finally, I compare the overall mean accuracy for each method. From Table 6 we can see that the best overall accuracy was achieved by NULL labelling (however the difference in accuracy is less than 1% for 2N labelling).

Overall, we can say that based on the accuracy, from these methods 2N and NULL labelling are the best options to choose if we want to retain accuracy in an adversarial setting. However, the fact that only 2N labelling was able to learn an attack amplitude insensitive model makes it the best choice by far.

In a scenario where discarding samples is not feasible, using adversarial learning can give us the best robust performance.

5.4 Transferability

Transferability for all 3 defence methods were calculated and presented in this section. Similarly to accuracy, the 2N method is capable of training a model that can operate on a nearly constant transferability regardless the attack amplitude as can be seen in Figure 14.

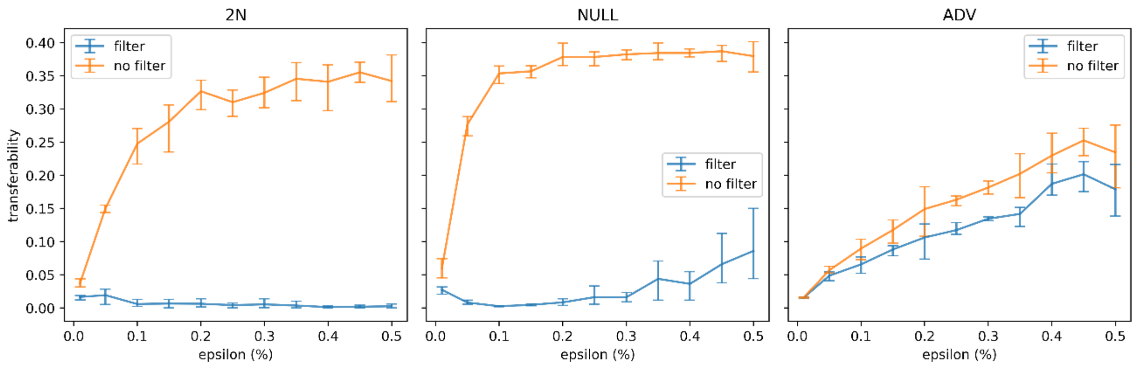


Figure 14. Mean transferability for each method with and without using filtering.

In Figure 14 we can see a significant improvement in transferability with 2N and NULL labelling. Using the BIN model (with ADV as the N class classifier) to filter samples also helped to reduce the transferability, however the effect is not that conspicuous.

For 2N labelling the filtered transferability is almost 0 in all cases, for every attack amplitude. This means that even when an attack surpasses the filter, our model will be misled by the attack with a very low probability. This shows that with 2N labelling we can achieve a defence that is almost ideal/optimal.

Since the ROC-AUC for the attack detection task is not 1 in case of 2N labelling, we were not able to perfectly classify each sample as attack or non-attack. The only possible thing that can cause the transferability to be nearly constant zero is to discard the samples that would be misclassified by the model (but not necessarily all attack samples!).

By discarding not all attacked inputs, but only the ones which would mislead the model, we can achieve at least the same accuracy (see derivation in 8.2), but with fewer discarded inputs. This can be useful in a situation where the ratio of the discarded inputs is crucial.

Model	**Transferability**
2N	**0.007**
NULL	0.029
ADV	0.117

Table 7. Mean transferability for each method using filtering if the attacks are sampled from an uniform distribution with parameters 0 and 0.5.

In Table 7 we can observe that by far the best transferability was achieved by 2N labelling. The differences in transferability are conspicuous, with 2N labelling it is 4x less than with NULL labelling, and 16x less than with adversarial learning and using the BIN model as filter.

6 Conclusion

I worked out new adversarial defence methods, the 2N method and a variant of NULL labelling. In my report I compared them with a third approach, the adversarial learning. In this setting the methods were allowed to discard samples considered to contain adversarial perturbation, because I theoretically proved that filtering can increase the accuracy. To measure the effectiveness of each technique, I calculated accuracy and transferability on a malicious validation dataset.

My results show that Multi-Task learning makes a huge improvement in the filtering and classification task, despite the models that were trained separately (without Multi-Task learning) performed better in the decomposed filtering or classification only tasks.

Another important aspect of adversarial defence is to learn a model that can operate with a constant classification performance regardless the presence or amplitude of adversarial attacks. I experienced attack amplitude insensitive phenomenon (with measuring accuracy and transferability) for only models trained with 2N labelling.

7 Future work possibilities

The subject of this report has many aspects of evaluation, I was only able to overview only some of them in this report. Here are some of my ideas to continue this research:

- It would be worth examining the effectiveness of each method against different kinds of attacks, for example in L_0 bounded adversarial attacks.
- Determining the classification thresholds itself is a very wide topic, here I only included one strategy to obtain thresholds.
- There are many different approaches for adversarial defence, it would be great to compare my method with some of them.
- It would be interesting to see the performance on different model architectures, such as ResNet [24], EfficientNet [25], GoogLeNet [26], etc.
- CIFAR-10 is an easy dataset, a more complex dataset for example the ImageNet [27] dataset could be more suitable to evaluate and compare defence methods.
- Repeat the experiments with different ratio of adversarial samples on the validation set to get a wider picture on the goodness of the defence method.
- It would also be interesting to see how the methods perform when the maximal discard ratio constraint is varied (which is the minimal ratio of discarded samples where the model becomes attack amplitude insensitive).

References

[1] Berrar, D. (2019). Cross-Validation. . Encyclopedia of Bioinformatics and Computational Biology, Volume 1, Elsevier, pp. 542–545. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

[2] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

[3] Jaworek-Korjakowska, J., Kleczek, P., & Gorgon, M. (2019). Melanoma thickness prediction based on convolutional neural network with VGG-19 model transfer learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops

[4] Hosseini, H., Chen, Y., Kannan, S., Zhang, B., & Poovendran, R. (2017). Blocking transferability of adversarial examples in black-box learning systems. arXiv preprint arXiv:1703.04318.

[5] Chen, J., Jordan, M. I., & Wainwright, M. J. (2020, May). Hopskipjumpattack: A query-efficient decision-based attack. In 2020 ieee symposium on security and privacy (sp) (pp. 1277-1294). IEEE.

[6] Iyyer, M., Wieting, J., Gimpel, K., & Zettlemoyer, L. (2018). Adversarial example generation with syntactically controlled paraphrase networks. arXiv preprint arXiv:1804.06059.

[7] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[8] Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial machine learning at scale. arXiv preprint arXiv:1611.01236.

[9] Li, S., Hao, Q., Kang, X., & Benediktsson, J. A. (2018). Gaussian pyramid based multiscale feature fusion for hyperspectral image classification. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11(9), 3312-3324.

[10] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083.

[11] Meng, L., Lin, C. T., Jung, T. P., & Wu, D. (2019). White-box target attack for EEG-based BCI regression problems. In International conference on neural information processing (pp. 476-488). Springer, Cham.

[12] Müller, R., Kornblith, S., & Hinton, G. (2019). When does label smoothing help?. arXiv preprint arXiv:1906.02629.

[13] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). Practical black-box attacks against machine learning. In

Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp. 506-519.

[14] Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., & Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. arXiv preprint arXiv:1701.06548.

[15] Ruder, S. (2017). An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098.

[16] Sharif, M., Bhagavatula, S., Bauer, L., & Reiter, M. K. (2019). A general framework for adversarial examples with objectives. ACM Transactions on Privacy and Security (TOPS), 22(3), 1-30.

[17] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[18] Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On early stopping in gradient descent learning. Constructive Approximation, 26(2), 289-315.

[19] Chen, Y., & Wainwright, M. J. (2015). Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. arXiv preprint arXiv:1509.03025.

[20] Ketkar, N. (2017). Stochastic gradient descent. In Deep learning with Python (pp. 113-132). Apress, Berkeley, CA.

[21] Gotmare, A., Keskar, N. S., Xiong, C., & Socher, R. (2018). A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. arXiv preprint arXiv:1810.13243.

[22] Szűcs G., Kiss R. (2021). Képosztályozó modellt támadó irányított zaj elleni védekezés Gauss képpiramissal és adversarial tanítással, Képfeldolgozók és Alakfelismerők Társaságának 13. konferenciája, 2021, június 22-24. (13 oldal)

[23] Vogl, R. (2018). Deep learning methods for drum transcription and drum pattern generation. Johannes Kepler University Linz, Linz.

[24] Wu, Z., Shen, C., & Van Den Hengel, A. (2019). Wider or deeper: Revisiting the resnet model for visual recognition. Pattern Recognition, 90, 119-133.

[25] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning (pp. 6105-6114). PMLR.

[26] Ballester, P., & Araujo, R. M. (2016, February). On the performance of GoogLeNet and AlexNet applied to sketches. In Thirtieth AAAI Conference on Artificial Intelligence.

[27] Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., & Oord, A. V. D. (2020). Are we done with imagenet?. arXiv preprint arXiv:2006.07159.

8 Appendix

8.1 Derivation 1

We can investigate the effect of the filtering, i.e. the situation when after the filtering the accuracy is larger than before can be written as follows.

$$acc_{before} < acc_{after}$$

$$acc_{ori} \cdot (1 - p_a) + acc_{att} \cdot p_a < \frac{acc_{ori} \cdot (1 - p_a) \cdot R_{ori} + acc_{att} \cdot p_a \cdot (1 - R_{att})}{(1 - p_a) \cdot R_{ori} + p_a \cdot (1 - R_{att})}$$

The denominator is not negative; thus, we can multiply both sides:

$$(acc_{ori} \cdot (1 - p_a) + acc_{att} \cdot p_a) \cdot ((1 - p_a) \cdot R_{ori} + p_a \cdot (1 - R_{att})) \\ < acc_{ori} \cdot (1 - p_a) \cdot R_{ori} + acc_{att} \cdot p_a \cdot (1 - R_{att}$$

$$0 < acc_{ori} \cdot (1 - p_a) \cdot R_{ori} + acc_{att} \cdot p_a \cdot (1 - R_{att}) \\ - (acc_{ori} \cdot (1 - p_a) + acc_{att} \cdot p_a) \cdot ((1 - p_a) \cdot R_{ori} + p_a \cdot (1 - R_{att}))$$

$$0 < acc_{ori} \cdot (1 - p_a) \cdot \{R_{ori} - (1 - p_a) \cdot R_{ori} - p_a \cdot (1 - R_{att})\} + acc_{att} \cdot p_a \\ \cdot \{(1 - R_{att}) - (1 - p_a) \cdot R_{ori} - p_a \cdot (1 - R_{att})\}$$

$$0 < acc_{ori} \cdot (1 - p_a) \cdot \{p_a \cdot R_{ori} - p_a \cdot (1 - R_{att})\} + acc_{att} \cdot p_a \\ \cdot \{(1 - p_a) \cdot (1 - R_{att}) - (1 - p_a) \cdot R_{ori}\}$$

$$0 < acc_{ori} \cdot (1 - p_a) \cdot p_a \cdot \{R_{ori} - (1 - R_{att})\} + acc_{att} \cdot p_a \cdot (1 - p_a) \\ \cdot \{(1 - R_{att}) - R_{ori}\}$$

$$0 < acc_{ori} \cdot (1 - p_a) \cdot p_a \cdot \{R_{ori} - (1 - R_{att})\} - acc_{att} \cdot p_a \cdot (1 - p_a) \\ \cdot \{R_{ori} - (1 - R_{att})\}$$

$$0 < p_a \cdot (1 - p_a) \cdot (acc_{ori} - acc_{att}) \cdot \{R_{ori} - (1 - R_{att})\}$$

8.2 Derivation 2

Proving that the accuracy increases or does not change for a sample set when a new sample is added to the set which is correctly classified.

The accuracy for a sample set can be written as the ratio of the number of correctly classified samples in the set (*C*) and the number of elements in the sample set (*N*):

$$acc_0 = \frac{C}{N}$$

If a new sample is added to this set (for example an attacked image which is correctly classified by the classifier model is not discarded in the filtering and classification scenario by the filter), which is correctly classified, the accuracy for the new set will be:

$$acc_1 = \frac{C+1}{N+1}$$

If we write the inequality for the case when $acc_1 \geq acc_0$, we get:

$$\frac{C+1}{N+1} \geq \frac{C}{N}$$

We multiply both sides with $N * (N+1)$ (which is positive), we get:

$$C * N + N \geq C * N + C$$

By subtracting $C * N$ from both sides we arrive to:

$$N \geq C$$

This means that the $acc_1 \geq acc_0$ inequality is true, when the number of samples is greater or equal to the number of correctly classified samples. This is always true, since the set of correct samples is a subset of all the samples, thus C cannot be higher than N. $N = C$ equality is true when each sample is classified correctly.