# Sensitive or Not? How to Attack and Defend Document Security Classification Models

**Sara PIZZIMENTI[*+], Alessandro OBERTI[*+],**
**Konrad WRONA[+] and Maria Carla CALZAROSSA[*]**
[*]University of Pavia, ITALY
THE NETHERLANDS

mcc@unipv.it
konrad.wrona@ncia.nato.int

## *ABSTRACT*

*Recently we witnessed a rapid development of adversarial machine learning techniques that compromise the security of an underlying machine learning model and cause a malfunctioning benefiting the adversary. The most common type of adversarial machine learning attack consists of deliberately modifying the input of the machine learning model in a way imperceptible to the human but sufficient to cause the model to fail. Originally conceived for images, adversarial examples can be also applied to Natural Language Processing (NLP) and text classification. This work presents a study and an implementation of adversarial examples - and defence mechanisms - against a NLP classifier based on BERT. The dataset used to test the proposed approach consists of NATO documents, now declassified, that initially possessed various levels of confidentiality, specified by labels embedded in the documents. The BERT model is used to automatically classify these documents according to their initial sensitivity. While the attacker's purpose is changing the classification level, the defence is committed in blocking these attempts. The experiments show that adversarial text examples can mislead the model, resulting either in a denial-of-service, when documents are recognized as having higher sensitivity than actual one, or in a data leak, when documents are interpreted as having lower sensitivity than in reality. By adopting adequate defences, it is possible to counteract specific types of adversarial attacks, at the expenses of a reduction of the overall accuracy of the model.*

## 1.0 INTRODUCTION

The widespread use and success of machine learning systems has made them an increasingly frequent target of attackers, whose goal is to misuse them for their own benefits. This phenomenon has led to the growth of adversarial machine learning [1], a field that combines machine learning and cybersecurity and deals with the study of possible attacks, as well as countermeasures, against intelligent systems. Therefore, although machine learning delivers, in general, results quickly and with high accuracy, it is not risk free and, if implemented without sufficient security analysis, the consequences can be disastrous. For example, the Tesla Model S 75 autopilot system could be manipulated by both hiding highway signs or adding marks that would be ignored by the human drive, resulting, e.g., in swerving into the wrong lane [2].

The possible types of attacks to which the various components of a machine learning system are exposed, grouped by intentional and unintentional failures are depicted in Figure 1. One of the biggest threats to machine learning is data integrity and is represented by data poisoning. Data that is part of the training set, if compromised, can alter the ability of the model to learn, and thus its performance. Training samples typically do not cover all possible corner cases. Some samples that have not been considered may be misclassified by the model, resulting in incorrect prediction. Third party services that make their pre-trained models available generally only want to provide query access, without providing extra information. Any security breach against the model confidentiality can leak sensitive information that may reveal and expose model structure. In general, machine learning as a service providers want to keep information related to the data used as training sets confidential. A membership inference attack aims at compromising data privacy by revealing part of the training set.

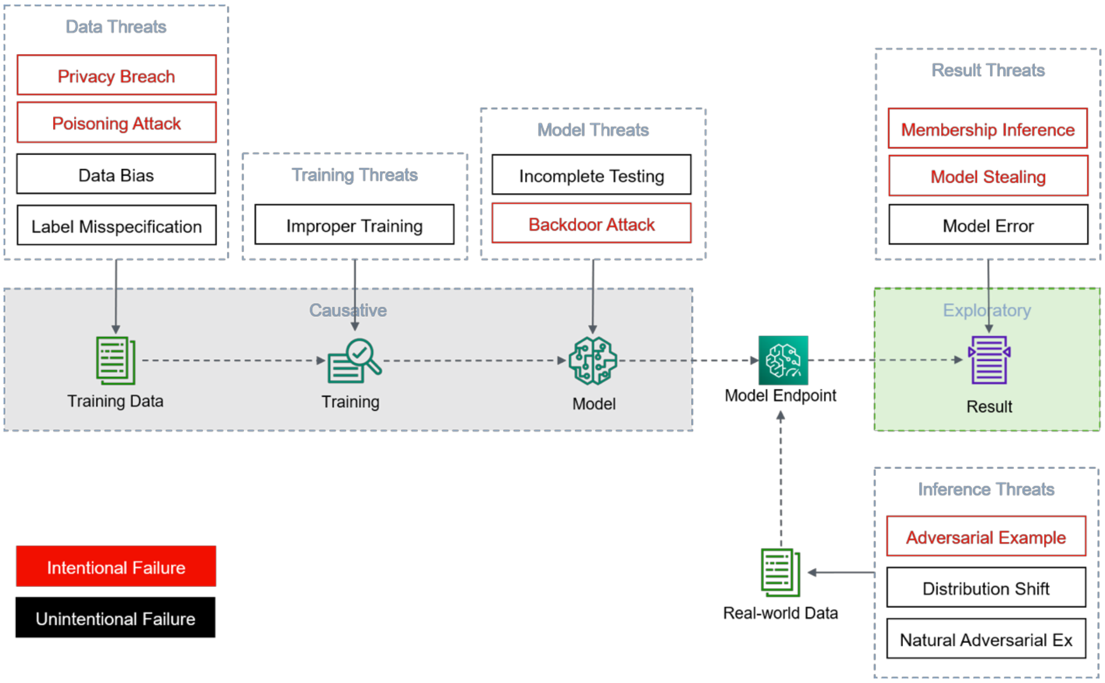


**Figure 1 Example of security threats against machine learning systems**

## 2.0 ADVERSARIAL MACHINE LEARNING

Adversarial machine learning aims at subverting machine learning system, so that its behaviour is altered in a malicious way. A taxonomy of adversarial machine learning attacks is presented in Table 1.

**Table 1 Attack taxonomy**

| | |
|---|---|
| Timing | Causative: attack occurs during training phase<br>Exploratory: attack occurs during inference phase |
| Goal | Security violation<br>Integrity: system produces an unsafe outcome<br>Availability: system does not work as it should<br>Privacy: sensitive information is exposed<br>Specificity<br>Confidence reduction: the confidence of the output is reduced<br>Misclassification: prediction returns a wrong class<br>Targeted misclassification: prediction returns a specific wrong class<br>Source/targeted misclassification: for a specific input, prediction returns a specific wrong class |
| Attacker’s capabilities | Fully knowledge: white-box scenario<br>Limited knowledge: grey-box and black-box scenario |

Adversarial machine learning attacks exist due to the nature of learning algorithms [3], whose limitations can be exploited by attackers, to misuse the system. These weaknesses are not like traditional bugs, and they cannot simply be patched. The effectiveness of adversarial machine learning attacks is not due to errors up to model creation but is linked to the learning algorithms nature. Before adapting a machine learning model for a particular task, it must first be trained. During the learning process, the model identifies the common pattern within training samples, so that it can be generalized and can make predictions on similar, but unseen, data. The future performance of the model is the result of training which depends on the used data. Data dependence is the key aspect of machine learning. On the one hand, the model can automatically learn from the data provided, on the other hand, data as the only source of knowledge can provide a way to corrupt the learning system. Also, a model that works well on a given dataset and achieves high accuracy during training phase, does not imply that, by adapting it in a new environment, it maintains unchanged its performance. Its accuracy should be evaluated by measuring how it generalizes the training set and how it makes predictions over new data, sampled from the same distribution. A model that poorly generalizes is the result of overfitting. The adversarial examples exist in the feature space from which the dataset is sampled, [4] but they are difficult to find by simply sampling randomly. Therefore, the model is unlikely to have seen them before - i.e., during the training phase - and to be enough robust against them [5]. This problem affects all types of classifiers, and, despite all possible countermeasures, the robustness of any machine learning models is limited against adversarial examples attacks [4]. Table 2 lists various types of adversarial machine learning attacks.

**Table 2 Types of adversarial machine learning attacks**

| Attack | Description |
|---|---|
| Poisoning Attack | Affects machine learning integrity, by including compromised samples in the training set |
| Model Stealing | Compromises the model confidentiality, by creating an equivalent model |
| Adversarial Example | Affects the model robustness, by misleading the model with a well-crafted input |
| Membership Inference | Threatens data privacy, by inferring if the sample comes from the model's training set |
| Backdoor Attack | Threatens data integrity, model confidentiality, model robustness and data privacy, by inserting a backdoor that attacker can trigger to take control of the model |
| Privacy Breach | Threatens data confidentiality |

Algorithms used for adversarial examples generation [6,7] are specifically designed to find the smallest perturbation that can mislead the model and make the example as close as possible to the original one. This similarity is quantified with some metrics, usually expressed in terms of norm distance. Using a large perturbation to mislead the model is a trivial approach. If the example turns out to be totally different from the original, it can no longer be considered an adversarial example. Optimization-based Attack [8] was the first adversarial example based on box-constrained Limited memory Broyden-Fletcher-Goldfarb-Shannon (L-BFGS) algorithm. Optimization-based methods usually lack the efficacy in black-box attacks. Gradient-based Attacks, such as Fast Gradient Sign Method [9], apply perturbation along the direction of maximum increase of the cost function. This method is effective and fast since it is a one-step method. It finds an adversarial example by maximizing the loss function with the perturbation constant designed to be small enough not to be detectable [10]. Iterative Gradient-based Kurakin et al. [11] introduced an iterative version of the FSGM by applying it several times with a small step size. It has been shown that iterative methods are stronger white-box adversaries than one-step methods at the cost of worse transferability.

In general, the most dangerous property of adversarial examples is their transferability [12], as an input properly produced for a particular model also seems to be able to mislead a different model. At first sight, this would make black-box attacks possible, where the adversary has only query access. This phenomenon exists because different machine learning models, for a particular task, learn similar decision boundaries around a data point, making the adversarial examples crafted for one model also effective for others. Usually, there is a trade-off between attack ability and the transferability. Adversarial examples generated by optimization-based and iterative methods have poor transferability. One-step gradient-based methods generates more transferable adversarial examples; however, they usually have a low success rate for the white-box model.

Adversarial text examples generation is challenging since existing perturbation, such as gradient-based methods, are more difficult to apply. Attack methods for text usually rely on heuristic replacement strategies on character or word level: they consist in introducing errors on characters level [13,14] or adding/deleting words. The requirements to be met for the generation of adversarial text can be of two types. Visually similarity describes adversarial examples that look very similar to the original ones, even if contain errors from a grammatical point of view. Semantic similarity describes adversarial examples that are semantically indistinguishable from the original ones, while being visually different. The measure of text similarity is distinctly different from that made for images. The similarity between the adversarial text and the original one should be evaluated under a grammar, syntax and semantic point of view. Grammar and syntax checker are used to detect anomalies and errors. Semantic similarity between two sentence vectors is calculated as a function of the cosine and Euclidean distance. Edit-based measurements quantifies the number of different characters between two sentence vectors. Levenshtein Distance and Word Mover's Distance are two examples of possible measures.

Adversarial text attacks against natural language processing (NLP) classifiers [15, 16] consist of four components: an attacker's goal, constraints, a search method and a transformation.

*Goal* identifies the objective of the attack, which can be a targeted/untargeted misclassification or a confidence reduction.

*Constraints* are the rules to be respected if the adversarial example generated is to be considered valid. Maximum number of editable characters, grammatical correctness, semantic correctness are examples of constraints. There are three main categories of constraints: semantics, grammatically and overlap. Semantics: the adversarial text must preserve the semantics of the sentence, i.e., the meaning. Grammatically: the adversarial example should be correct under a grammatically point of view, in such a way it is not reported by a grammar checker. Overlap: the adversarial example must be like the original text in terms of the number of changed characters.

*Search method* allows to select words to be perturbed within the entire search space. Search methods choose the best transformations to apply to the text [17], with the aim of satisfying the attacker's goal by trying to minimize the number of perturbations. Adversarial text examples generated in a black-box scenario, where no knowledge is available, requires a strong search method, such as the ranking-based sampling, to collect only the most informative words. Random search method allows to randomly select the words where perturbations are applied. The results produced by this algorithm are not consistent, as they vary each time the algorithm is executed. The word ranking algorithm [14, 18, 19] is used to find important words that most affect the sentence's classification, to alter the final classification result with fewer perturbations. This method turns out to be more effective for adversarial text generation compared to a random selection.

*Transformation* defines the type of perturbations that make up the generation algorithms. Common methods are HotFlip, synonyms and antonyms. HotFlip is an adversarial text generation method that uses characters substitution, i.e., flips [13]. The result tries to be as visually similar as possible, but clearly involves incorrect solutions, both grammatically and semantically. For each character that makes up the selected word within the sentence, the impact of a flip with a visibly similar, but different, character is evaluated. Then the flips with the greatest impact on the sentence's classification are chosen, based on the maximum number of transformations applicable. In synonyms generation, the candidate words are modified with their synonyms if they have one. The synonym, while replacing the meaning of the sentence, must be able to modify the final classification. It produces a grammatically correct adversarial example. In antonyms generation the candidate words are replaced with their opposite if they have one. This attack produces an adversarial example that denies the meaning of the previous sentence.

## 3.0 ATTACKING DOCUMENT SENSITIVITY CLASSIFICATION

The machine learning model for document classification offers an automated alternative to a process that, otherwise, would have required the human intervention. The model is trained to recognize the degree of confidentiality of NATO documents, with the aim of creating a system that allows the access to a particular resource only if the access requirements are satisfied. For this reason, the model must ensure reliable predictions, and that documents containing sensitive information are not released to the public. This chapter presents the proposed framework for adversarial text examples generation, which threatens the document classification model. Document classification involves the labelling process, where, according to the information provided, a category is assigned to the document. Carrying out this process manually is possible because there is more control over the classification, but it is also slow, particularly with large volumes of documents. Therefore, an automatic classification supported by a machine learning solution is cost-efficient, reliable, and less biased compared to manual labelling. The classification model was used to classify NATO documents based on their sensitivity. The dataset consists of Cold War typewriter documents, now declassified, scanned and stored in PDF format. On the first page of each document, generally in the upper part, there is a stamp that indicates its sensitivity. Documents do not have any machine-readable label, which could be used to train a classifier. Creating the proper dataset for model training requires a pipeline, capable of text extraction, labelling, and storing the result. The proposed architecture is cloud-based and fully developed on the AWS platform, which offers services for data storage and machine learning model development. Using Amazon Textract along with an event based serverless architecture, the scanned PDF documents are prepared for model training. The pipeline, depicted in Figure 2, involves three Lambda functions, a SQS queue, a S3 destination bucket and ElasticSearch (ES).

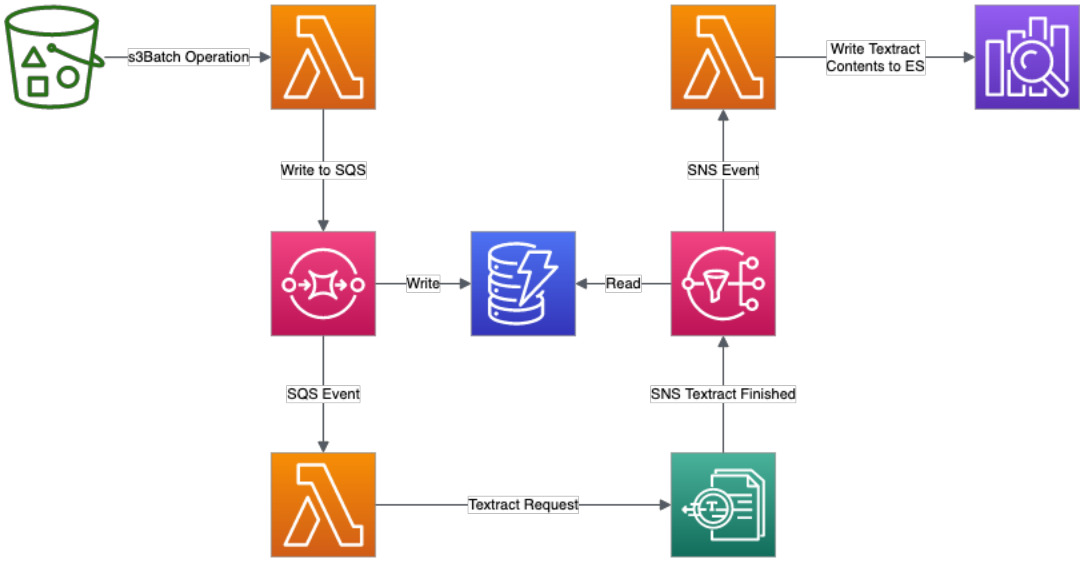


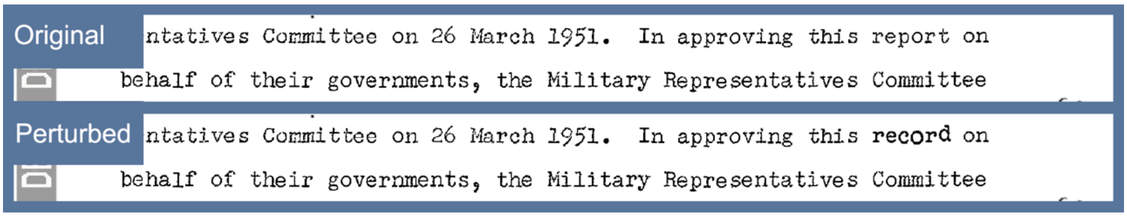
**Figure 2 CloudFormation built architecture**

Table 3 illustrates the unbalanced distribution of the documents in the various categories, with 41% of NATO UNCLASSIFIED documents and only 1% COSMIC TOP SECRET documents. The labelled documents were split into the training, test and evaluation sets.

**Table 3 Dataset composition**

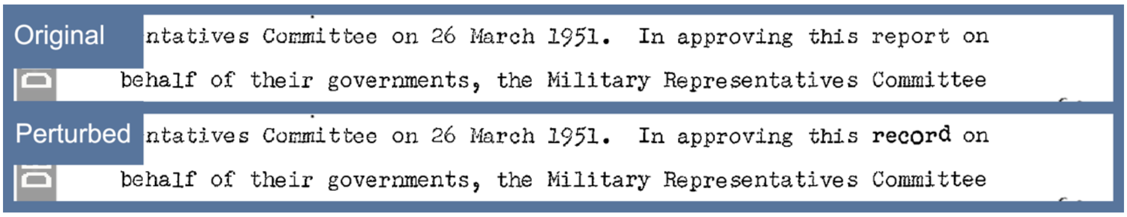
| **Original sensitivity** | **No. of docs** | **% of docs** | **Training** | **Validation** | **Test** |
|---|---|---|---|---|---|
| COSMIC TOP SECRET | 325 | 1% | 243 | 65 | 17 |
| NATO SECRET | 4017 | 19% | 3012 | 804 | 201 |
| NATO CONFIDENTIAL | 5069 | 23% | 3801 | 1014 | 254 |
| NATO RESTRICTED | 3401 | 16% | 2550 | 680 | 171 |
| NATO UNCLASSIFIED | 8831 | 41% | 6623 | 1766 | 442 |
| *Total* | *21643* | *100%* | *16229* | *4329* | *1087* |

The document classification is carried out through a pre-trained Bidirectional Encoder Representations from Transformers (BERT) [20] model. It is a Masked Language Model since it randomly hides some of the input tokens and tries to predict them. It uses a bidirectional approach: the sentence is read from left to right and from right to left. BERT was previously trained on unlabelled data, to obtain an initial configuration of the parameters. Then, it is fine-tuned using labelled data. The adversarial text generation must include the generation itself as well as rendering the document as a scanned PDF, thus requires the availability of the font of the documents. Textract does not only provide the text of the document, but also the position of the bounding boxes around the single words within the text. This information makes it possible to locate the positions of the words to be perturbed. For the font creation, the positions of all the letters that make up the alphabet have been localized and the best ones have been selected, according to the contrast level with the paper sheet. Then, by manually checking the candidates, characters were chosen for the creation of the font. In this way, we want to make the perturbations appear as original as possible. The adversarial text generation consists of two phases: choice of document to be modified and attack setup. In particular, the setup deals with the choice of the generation algorithm (e.g., synonyms, antonyms, HotFlip), the search method (e.g., ranked, or random) and, finally, the percentage of error to apply. Figures 3, 4 and 5 show examples of adversarial text rendering using the synonyms, antonyms and HotFlip generation algorithms.


Original
ntatives Committee on 26 March 1951. In approving this report on
behalf of their governments, the Military Representatives Committee

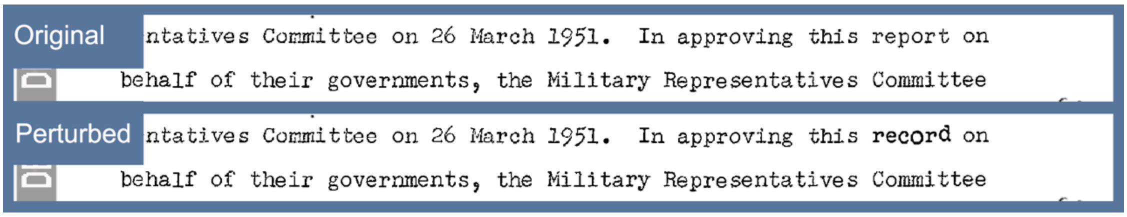
Perturbed
ntatives Committee on 26 March 1951. In approving this record on
behalf of their governments, the Military Representatives Committee

**Figure 3 Synonym-based adversarial text**


Original
ntatives Committee on 26 March 1951. In approving this report on
behalf of their governments, the Military Representatives Committee

Perturbed
ntatives Committee on 26 March 1951. In approving this record on
behalf of their governments, the Military Representatives Committee

**Figure 4 Antonym-based adversarial text**


Original
ntatives Committee on 26 March 1951. In approving this report on
behalf of their governments, the Military Representatives Committee

Perturbed
ntatives Committee on 26 March 1951. In approving this record on
behalf of their governments, the Military Representatives Committee

**Figure 5 HotFlip-based adversarial text**

The adversarial text can trigger two model's reactions: a confidence reduction, as the model is unable to discriminate well the different labels, or a misclassification. The model incorrectly classifies the document with another label. These leads to different attack scenarios. In denial-of-service, attacker aims at compromising model's availability via false positive, e.g., NATO UNCLASSIFIED documents are classified as COSMIC TOP SECRET, NATO SECRET, NATO CONFIDENTIAL or NATO RESTRICTED. As consequence, access is limited, and the document cannot be consulted. This prevents access to data, making the service unavailable. In data leakage attack. the document classification is downgraded, and its access is granted even to potentially unauthorized entities.

# 4.0 EXPERIMENTAL RESULTS

The main goal of experiments was to evaluate the document classification model robustness against small and large perturbations. A complete knowledge of the classification framework is not required to carry out the attack, however, an attacker must have documents to be classified, an OCR software, and a query access to the document classification model. The main goal is to compromise the model, by decreasing the actual class probability, until the document is misclassified. Attacks consist in applying the different generation methods, using as search methods the random sampling and the word ranking-based sampling. In particular, the experiments aim to assess how the documents classifications change as the amount of the applied errors increases. The search methods identify the indexes corresponding to the words to be modified - i.e., the words to be replaced with adversarial text. The number of indexes is chosen according to the total number of words of the documents. Two types of perturbation are used: a lighter one, where the number of indexes corresponds to 10% of the total number of words in the documents and a heavier one, where the number of indexes is 40%. The goal is to satisfy the two basic requirements of an adversarial example: to be unnoticeable to the human eye and to misuse the model. Too many perturbations lead to a trivial solution because the attack would be more effective, but it would no longer be unnoticed to human eye. A small number of perturbations, instead, would make the attack weak and ineffective.

For most of the classes of document, probabilities tend to decrease as the error increases. This behaviour is less evident in the antonyms attack and more evident in the HotFlip one. As the error increases, it is often possible to notice an increase in dispersion of the probabilities. This is mainly due to the random sampling used as a search method. The predictions are not consistent, because they depend on the random values obtained during the specific execution of the attack. An increase of the number of perturbations does not increase the attack effectiveness. The synonyms and antonyms attacks show unexpected behaviour for some classes of documents, because the model seems to achieve better performance. This might be due to the training set used, which contained errors due to a bad reading by Textract. These random errors are similar to those introduced when generating an adversarial text. Since the network was trained on them, it becomes quite robust against these random perturbations. The HotFlip attack is the most effective. The medians of probabilities distribution are smaller with the only exception represented by COSMIC TOP SECRET documents.

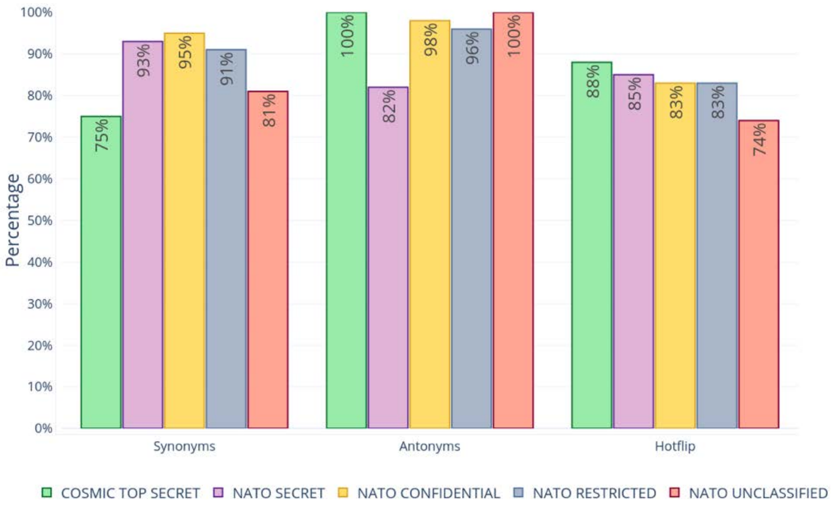


**Figure 6 Model predictions with 10% errors for the classification labels as a function of the type**

**of attack**

Figure 6 shows the percentages of correct predictions for the different classification labels, as a function of the different attack methods with an error of 10%. As expected, the likelihood of correct classifications is high, for the antonyms attack. For the HotFlip and synonyms attacks about 75% of NATO UNCLASSIFIED documents and of COSMIC TOP SECRET documents are correctly recognized.

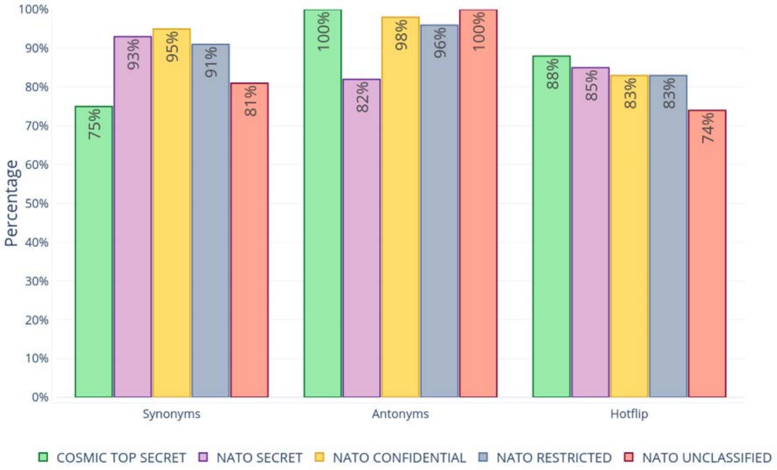


**Figure 7 Model predictions with 40% errors for the classification labels as a function of the type of attack**

Figure 7 shows the percentages of correct predictions for classification labels. The behaviour of the attacks is generally confirmed, although the predictions of documents correctly classified tend to decrease. For example, for COSMIC TOP SECRET documents both in case of synonyms and HotFlip attacks the probabilities drop to 38%. NATO CONFIDENTIAL is the document class with one of the highest percentages in each attack. Figure 8 summarizes the most common labels incorrectly predicted by the model. With antonyms and HotFlip attacks, documents tend to be misclassified as NATO UNCLASSIFIED. Instead, documents classified by the model as NATO UNCLASSIFIED are likely to be misclassified as NATO RESTRICTED under an antonyms attack. Thus, sensitive information contained in classified documents might be disclosed or access to NATO UNCLASSIFIED documents might be restricted. The synonym method does not generate documents that are disclosed as unclassified, even if they are often labelled with a lower security level. The only exceptions are represented by NATO UNCLASSIFIED and NATO RESTRICTED, which both are classified as NATO CONFIDENTIAL.

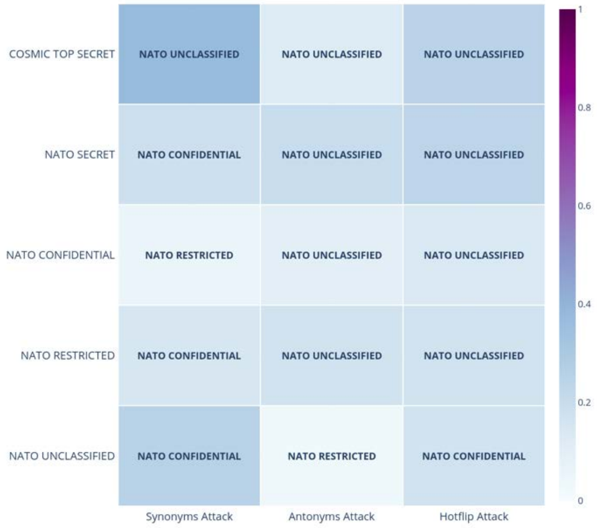


**Figure 8 Documents classification as a function of attack**

The word ranking algorithm is a search method that allows the choice of the words within the sentence that most affect its classification. In general, the most important words are also the most common ones.

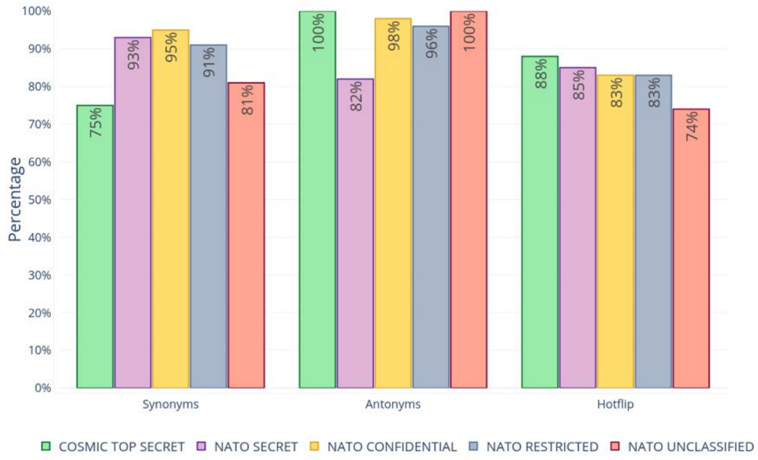


**Figure 9 Model predictions with 10% errors as a function of the type of attack**

Figure 9 shows the percentage of correct predictions for the different classification labels. The antonyms attack achieves the highest percentage of correctly classified documents. Around the 80% of COSMIC TOP SECRET and NATO UNCLASSIFIED documents are correctly recognized. The only low percentage is 38% and corresponds to NATO SECRET documents. The synonyms and the HotFlip attacks show low percentages for all documents classes. With the HotFlip attack, none of NATO RESTRICTED and NATO UNCLASSIFIED documents are correctly classified. Figure 10 shows the percentage of correct predictions for different classification labels by applying 40% errors. With the HotFlip attack, all documents' classes are almost never correctly classified. The percentage is zero for all documents classes except NATO

CONFIDENTIAL. Synonym's attack leads to similar results compared to the HotFlip attack. The highest percentage is 13% and corresponds to NATO CONFIDENTIAL documents. The antonyms attack is not able to lower the percentages of documents classes, whose values are similar to those achieved with 10% errors.

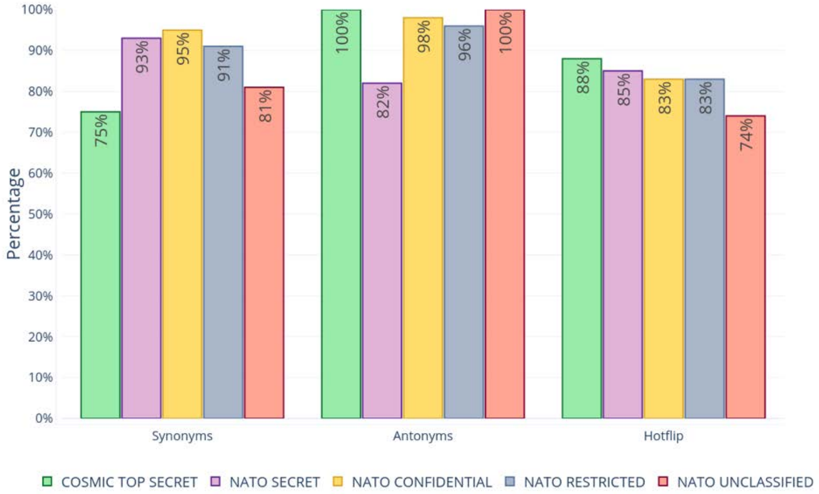
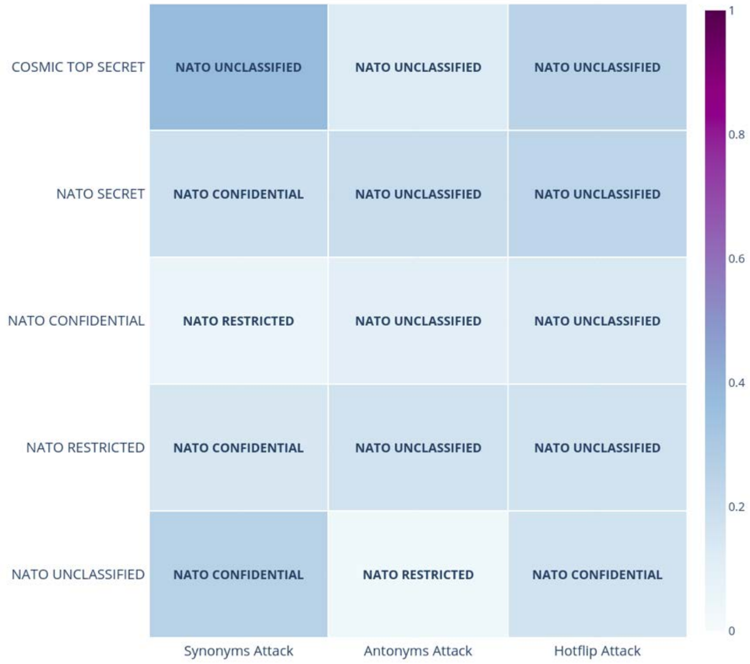


**Figure 10 Model predictions with 40% errors as a function of the type of attack**

Among all the labels with which a document is incorrectly classified, the most common labels incorrectly predicted, according to the used attack, is shown in Figure 11. The colours highlight the gap between the labels. A darker colour means that, when the attack is successful, the probability of getting that particular class is significantly higher than the others. For example, with the HotFlip attack, the probability of obtaining NATO UNCLASSIFIED when COSMIC TOP SECRET documents are classified is high. Antonyms and HotFlip attacks show that, for all documents classes, the most common label is NATO UNCLASSIFIED. This involves a disclosure of sensitive information. Instead, the NATO UNCLASSIFIED documents are restricted to prevent access to them. In the synonyms attack, all documents' classes are misclassified in different labels. Often, the most common labels do not correspond to the lowest degree of security, as certain access privileges are still required.



| | Synonyms Attack | Antonyms Attack | Hotflip Attack |
|---|---|---|---|
| COSMIC TOP SECRET | NATO UNCLASSIFIED | NATO UNCLASSIFIED | NATO UNCLASSIFIED |
| NATO SECRET | NATO CONFIDENTIAL | NATO UNCLASSIFIED | NATO UNCLASSIFIED |
| NATO CONFIDENTIAL | NATO RESTRICTED | NATO UNCLASSIFIED | NATO UNCLASSIFIED |
| NATO RESTRICTED | NATO CONFIDENTIAL | NATO UNCLASSIFIED | NATO UNCLASSIFIED |
| NATO UNCLASSIFIED | NATO CONFIDENTIAL | NATO RESTRICTED | NATO CONFIDENTIAL |

**Figure 11 Documents classification as a function of attacks**

Random sampling used as a search method proves to be a weak method. Indexes change at each attack execution and the perturbations applied lead to very different results from each other. This is reflected in the probabilities dispersion of the documents classes. The dispersion tends to increase as the error increases. The ranked-based method used as a search method allows the synonyms and HotFlip attacks to lower the probabilities dispersion of the labels in almost all documents' classes. The HotFlip attack is the most effective in compromising the model's predictions and, also, it succeeds with a less error rate, compared with the other attacks. With the ranked-based sampling, the attacks are effective with a smaller number of perturbations. However, this leads to higher computation costs. In both search method, the results of the most common labels according to the document to be classified and the attack, agree. The HotFlip attack can be considered as the best in terms of predictions result. However, this attack can be more easily detected. The synonyms create adversarial documents that can mislead the classification model, still preserving the semantics of the sentence.

## 5.0 COUNTERMEASURES

There are two main types of countermeasures against adversarial examples [21]. Pro-active countermeasures, such as Gradient Masking and Robust Optimization, increase the resilience of deep neural networks against adversarial examples. Reactive countermeasures, such as adversarial example detection, detect malicious model inputs.

### 5.1 Gradient Masking

Adversarial example detection aims to distinguish malicious inputs from benign ones before the classifier makes its prediction by adopting some statistics approach or looking at the prediction consistency. Principal Component Analysis suggested that adversarial images place abnormally strong emphasis on principal components which account for little variance in the data [22]. Feature squeezing [23] was proposed to reduce the search space available to an adversary, by coalescing samples that correspond to many different feature vectors in the original space into a single sample. After comparing a DNN model's prediction on the original input with that on squeezed inputs, feature squeezing can detect adversarial examples with high accuracy. Although quite promising, these defences were easily bypassed [24]. The difference [22] between the natural and adversarial examples was detected because the border pixels are nearly always zero for natural MNIST[1] instances, whereas typical adversarial examples have non-zero values on the border, thus it is an artifact of the MNIST dataset. However, in [24] the same defence was applied to CIFAR[2] dataset and proved that there is no detectable difference between adversarial examples and natural data. Feature squeezing was bypassed with minimal visual distortion, suggesting for proposed defences to validate against stronger attack configurations [25].

Gradient masking/obfuscation techniques hide/obfuscate the gradient information of the classifier, since most of attacks exploit that knowledge.

Distillation [26] is a training procedure initially designed to train a DNN using knowledge transferred from a different DNN. The motivation behind that is to profit from knowledge of large architecture transferring it to smaller ones, facilitating the deployment of deep learning in constrained devices such as smartphones. In a defensive distillation [27] the knowledge of a DNN is used to improve its own robustness against adversarial examples produced by various attacks [28], [29]. The defensive distillation consist of four steps: 1) Train a network (teacher) using standard machine learning techniques; 2) Evaluate the teacher on each instance of the training set to produce so-called soft labels; 3) Train a second network (distilled) on these soft labels; and 4) Classify the input based on the distilled network. The key principle behind this algorithm is to make the gradient of the score function so small that the computer ALU rounds it to zero, obstructing the path to gradient-based attacks.

Shattered gradients are obtained when input data are pre-processed, adding a non-smooth or non-differentiable pre-processor and then train the DNN on it, causing the gradient to be non-existent (or incorrect) [30][31]. Another gradient masking approach is trying to randomize the defence applied, whether it is the network itself (e.g., train a set of classifiers and at run-time evaluation randomly select one of them) or the data fed to model (e.g., random pre-processing). An example of this strategy is to stochastically prune a subset of the activations in each layer [32]. Exploding and vanishing gradients are often obtained by defences consisting of cumulative iterations of neural network evaluations, feeding the output of one computation as the input of the next: the result is a gradient extremely small or irregularly large, which makes the adversary work of crafting adversarial examples more difficult. PixelDefend [33] is an approach that purifies a maliciously perturbed image by moving it back towards the distribution seen in the training data. Despite the effort made to mask the gradient, the security principal security through obscurity is still discouraged: gradient masking only "confuses" the adversary, but [34] demonstrates how to bypass seven gradient obfuscating/masking techniques, including Thermometer Encoding (shattered gradient), Stochastic Activation Pruning (randomized gradient), Pixel Defend (exploding & vanishing gradient) and Defensive Distillation.

## 5.2 Robust Optimization

Under Robust Optimization category fall methods which aim to improve the model resilience by making the classifier learn how to predict adversarial examples [21]. In order to make the model robust, Regularization Methods [16] suggest that a simple regularization of the parameters, consisting in penalizing each upper Lipschitz bound, might help improve the generalization error of the networks. A model with a new end-to-

[1] Modified National Institute of Standards and Technology database, https://yann.lecun.com/exdb/mnist/

[2] Canadian Institute For Advanced Research, https://www.cs.toronto.edu/~kriz/cifar.html

end training procedure that includes a smoothness penalty, increasing the network resilience without a significant performance penalty has been introduced in [35].

The concept of adversarial (re)training [28] argues that the primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature. By training not only with standard training data, but also with adversarial examples, a neural network could be somewhat regularized. Adversarial re-training is different from traditional data augmentation techniques which take advantage of data transformations (such as translations) that are expected to occur in the test set. Adversarial training transformation employs inputs that are unlikely to occur naturally but are very effective in impacting the classifier decision function. Adversarial Training Attack Algorithm Categories [28] make use of the single-step FGSM algorithm to generate adversarial example, while Projected Gradient Descent) (PGD) attack [36], which is an iterative method, and obtain valuable resilience against attacks on MNIST and CIFAR-10 datasets. Nonetheless, the time spent to generate all the adversarial examples is not negligible with the respect to natural training, introducing the scalability problem when coping with larger-scale dataset such as ImageNet. To address the issue of high computation/slow speed costs in generating valid adversarial examples, an algorithm that eliminates the overhead cost of generating adversarial examples by recycling the gradient information computed when updating model parameters was proposed in [37]. This method achieves comparable robustness to PGD adversarial training on the CIFAR-10 and CIFAR-100 datasets, at negligible additional cost compared to natural training. A robust model for the large-scale ImageNet classification task that maintains 40% accuracy against PGD attacks was trained on a system with four P100 GPUs in less than 50 hours. Adversarial training proved to be one of the few defences against adversarial attacks that withstands strong attacks. The reason this approach works seems related to the fact that adversarial training is fixing some form of perturbations for which the model should be robust to. By optimizing this form of resilience, the model is avoiding all the features which are not perturbation-robust, otherwise it is penalized. The biggest drawback is not the time spent generating ad hoc adversarial examples for each training dataset, but the necessary requirement of specifying in advance to which attacks the model needs to be robust against, as well as a reduction in the standard accuracy of the model. It is unclear how to generalize beyond the specific single attack or, at least, be able to categorize attack into broader categories.

Randomized Smoothing [38] is a defence inspired by differential privacy (DP) [39] and improves it, proposing randomized smoothing as a technique for certifying adversarial robustness. Similarly, to DP, which secure a classifier prediction from relying excessively on any input feature, randomized smoothing certifies that each prediction has a "secure" radius in which the classifier's prediction is guaranteed to remain constant (certified defence): this is achieved by convolving ("smoothing") the input with Gaussian noise. The main idea is to make the classifier robust, the base classifier should correctly classify under a vast noise, much larger in magnitude with the respect to the adversarial one). In fact, an interpretation of randomized smoothing is that these large random perturbations "drown out" small adversarial perturbations. Randomized smoothing is a promising direction for future research in the field of adversarial machine learning defence and remains the only defence which scales to ImageNet dataset.

## 5.3 Defence framework and recommendations

The proposed defence framework, which highlights the attack surface of a machine learning system, is depicted in Figure 12. The dataset can be obtained from multiple sources such sensing systems, IoT devices, databases, data breach or social media. Since adversaries can inject malicious data in the training phase, potentially misleading the classifier algorithm, it is important to protect against those kinds of attacks. Data sanitization is an effective defence to preserve data set integrity. In addition, periodically monitoring the collection of data and retraining models can increase the likelihood of detecting deviations from the baseline. For this classification task, documents are pre-processed at word level with a simple regex to check whether they do contain numbers among characters. This step, although simple, allows to detect most of HotFlip-based attacks. Model robustness is the property that characterizes how the classifier output variable is consistently accurate while being tested on a new unforeseen dataset. In the absence of threat actors, techniques such as cross-validation, anomaly detection, and performance metrics like Receiver Operating Characteristic (ROC) curves serve to assess robustness. Yet in a context of external threats, model robustness refers to the ability of dealing with adversarial examples. Originally, adversarial robustness was considered from the point of view of machine learning security as a mean to respond to adversarial examples [40]. By generating ad hoc adversarial examples and integrating them in the training process, it is possible to learn the model to recognize and block adversary fooling attempts. Several works studied the side-benefits of adversarial robustness related to features representations, such as better-behaved gradients [41],[42] and representation invertibility [43]. These desirable properties might suggest that robust neural networks are learning better feature representations than standard networks, which could improve the transferability of those features. The adversarial training technique is employed in this thesis work: the classification model was trained with adversarial examples among benign input data. Two types of attack have been taken into consideration: the synonym attack and a variation of HotFlip attack. Because attackers can re-create a fully functional replicated model by merely querying the real one (performing a so-called model inversion attack) ML systems should be hardened to preserve the data confidentiality before, during and after the training process. It is recommended to:

- Not expose model APIs at all. Albeit extreme, this is the only effective measure to prevent an attacker from probing the model.
- If exposed, ensure APIs are accessible only by authorized personnel (e.g., via encryption).
- Reduce confidence scores dimensionality: for example, instead of showing the exact probability percentage, display “low”, “medium” and “high” labels only.
- Detect abnormal APIs usage (e.g., huge amount of requests in a short period of time) with continuous monitoring. Although simple to implement, this may not be a very accurate system.
- Employ model watermarking: special recognition neurons are embedded in the model training phase and enable a special input sample to check if other models were developed by stealing the original one [11]. This defence scheme seems very promising and, despite most watermarking schemes currently apply only to image data, equally promising applications in the field of text classification cannot be excluded.

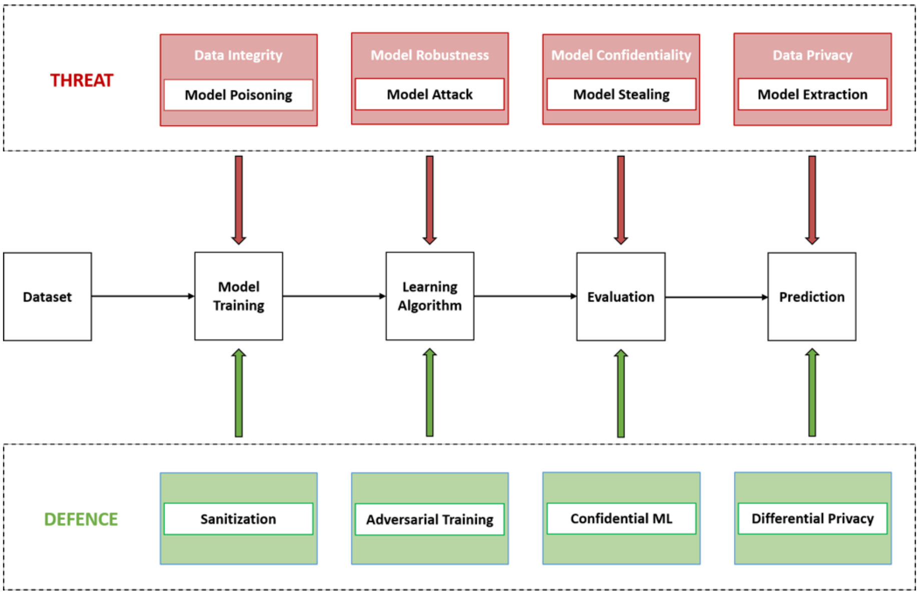


Figure 12 Proposed defence framework.

Another relevant emerging approach is confidential machine learning (ConfML) [44]. It demonstrates that by using levelled homomorphic encryption scheme it is possible to delegate the execution of a machine learning algorithm to a computing service, while retaining confidentiality of the training dataset.
Some attacks allow the adversary to extract sensitive information from the model (model extraction attacks), whether through black-box or white-box knowledge, hence potentially violating the users' privacy information that compose the dataset. Data privacy risks can be minimized through:

- Data sanitization: to avoid accidental leak of specific information (e.g., Social Security Number), it is recommended to remove this information before training the model. The task of removing such information may not be trivial, but the benefits derived from it are certainly worth the effort.
- Overfitting avoidance: overfitting is the product of an analysis that corresponds too closely to a particular set of data. Research [45] validates the correlation between overfitting and dataset privacy concerns. Monitoring model's performance on training and validation dataset is a simple way to recognize overfitting, while changing the model complexity is an approach to reduce it.
- Differential privacy: the idea behind differential privacy is that if the effect of making an arbitrary single substitution in the database is small enough, the query result cannot be used to infer much about any single individual (therefore providing privacy). This is achieved by adding "randomness" (i.e., noise) to the system in some ways (e.g., user's input, underlying data, loss function), at the cost, however, of a reduced accuracy of the model.

When talking about securing ML/AI, it is important to focus not only on the model. Thus, it is necessary to:

- Periodically conduct internal/external audit and testing of AI systems.
- Establish policies and baseline on how AI systems should typically behave.
- Adopt the principle of least privilege whenever possible.
- Test systems prior to deployment, to make sure they are working properly.
- Rely on trusted vendors only (i.e., supply chain security).
- Educate users (e.g., data scientists) to identify potential threats.
- Grant access to authenticated and authorised individuals only.
- Follow NATO security policies and industry best practices, such as AWS Security Guidance.

## 5.4 Experimental results

In our experiments we have focussed on adversarial training technique for two main reasons: 1) Simplicity: once the necessary adversarial examples have been generated, their addition to the dataset is a smooth and straightforward process; and 2) Flexibility: adversarial training can be used with different class of attacks (even simultaneously), without having to modify the procedural mechanism.

To increase the robustness of the document classification model, we proceeded as follows: starting from a dataset composed of scanned NATO typewriter documents, two types of adversarial examples were generated and fed to the model: one based on HotFlip and the other on synonyms. After re-training the model on these attacks, the ability to detect new adversarial examples has been assessed. The experiments conducted in this research clearly demonstrate that the use of adversarial training is an attractive tool to defend against adversarial examples. There are three parameters that need to be taken into consideration: 1) Model accuracy; 2) Number of adversarial examples detected; and 3) False Negative Rate. Only the FNR was considered because under-classifying classified documents is more dangerous than over-classifying them. A broader discussion might also consider the FPR to find the Crossover Error Rate (i.e., the point where FNR and FPR are equal). Our experimental results are summarized in Table 4.

Table 4 Experimental results summary.

| Dataset Type | Val. Accuracy | Adv. Accuracy | FNR |
|---|---|---|---|
| Benign | 87.2% | - | - |
| HotFlip | 74.2% | 93.6% | 6.4% |
| Synonym | 72.9% | 88.6% | 17.0% |
| HotFlip + Syn | 72.3% | 86.4% | 6.8% |

summarized how the accuracy of the model trained on the three types of attacks settle on average around 73%, without notable discrepancies. As for the number of adversarial examples found, HotFlip is the one which captures the most, followed by the synonym and the combination of the two. This last point can be due to at least two difficulties. The first and most important is the increasing complexity of the model when generalizing on more than one type of attack. This aspect is crucial to develop a model that is resilient to as many attacks as possible. In our experiments, the reduction of detected adversarial examples and of model accuracy is not excessive. A solution would be creating multiple attack-specific datasets containing numerous adversarial examples and submit the input to each of them. If on the one hand the percentage of adversarial examples detected should be kept high, on the other such solution does not scale well, other than being time/resources consuming. The second difficulty is due to the attack type. As mentioned before, the synonym attack is certainly more difficult to identify than HotFlip one, an intrinsic feature shared with other attacks based on semantic similarity. As for the FNR, the values - with exception of the synonym attack - are around 7%. However, the biggest drawback experienced was the model accuracy reduction, from 87.2% to approximately 73%. This discovery is not new [41], [46], [47] and it is uncertain if there is any path which increase security without worsening of performance. Several researchers aimed to find a trade-off between standard accuracy and model robustness [48], [49]: one of possible solutions to mitigate insufficient number of samples is to add unlabelled data [47].

## 6.0 CONCLUSIONS

Adversarial examples are the most popular category of attacks in the framework of adversarial machine learning. They consist of perturbing the machine learning model input so that it is hardly noticeable to human user, with the objective of compromising the classification results. An adversarial example applied to a text document, generates a corrupted text that is misclassified by the model. The experiments conducted have shown that, starting from a model characterized by a high accuracy, it was possible to mislead its classification ability by adding some perturbations to the document text. As a result, the model was no longer able to correctly classify documents according to their original degree of confidentiality. All the synonyms, antonyms and HotFlip attacks were carried out both with random sampling and ranking-based sampling as a search method. The results showed that ranking-based sampling was the more effective method, as it was able to minimize the perturbations to the text and keep the attack efficiency high. In addition, predicted probabilities showed a lower dispersion compared to the random method. The model was particularly vulnerable to the HotFlip attack, which consists in creating adversarial text visibly very close to the original one, but incorrect from a grammatical point of view. To make an attack as unrecognizable as possible to human eye, it was important to take also into account the synonyms, although their performance was slightly worse. The antonyms attack, instead, was the worst method since it was not very effective and most of the times the model was still able to correctly classify documents. The behaviour of the attacks was confirmed both for random and ranking-based sampling, even though the random search method lacks in efficiency. In fact, to obtain misclassification, it was necessary to perturb the text more significantly.

The proposed defence framework increased the model resilience to adversarial examples. To acquire the most complete perspective, the investigation considered two NLP attack categories: the synonym attack and a variation of the HotFlip that introduced typos. The best results were obtained when the model was trained to detect HotFlip-based adversarial examples. This was not surprising considering the visual similarity attack. Slightly worse performance on synonym attack was obtained and expected since exploiting semantic similarity is harder to detect. When mixed, the accuracy of detecting adversarial examples dropped, potentially indicating that training the model on semantic similarity and visual similarity attacks separately could increase efficacy. The results also show that using adversarial training as a robust optimization technique led to a notable reduction in the standard accuracy of the model. This was expected, and currently seems to be the price to pay to be resilient to adversarial examples.

Future work could include the introduction of further attack types. In addition, a better trade-off between standard accuracy and ability to detect adversarial examples could be investigated, as well as the adoption of new/improved defences to be combined with adversarial retraining. It would be also interesting to generalize our model and experiments by using datasets obtained from different public archives.

## REFERENCES

[1] L. Munoz-Gonzalez and E. C. Lupu, The Security of Machine Learning Systems. Springer, 2019.

[2] E. Ackerman, “Three Small Stickers in Intersection can Cause Tesla Autopilot to Swerve into Wrong Lane," Tencent Keen Security Lab, Tech. Rep. 03, 2019.

[3] M. Comiter, “Attacking Artificial Intelligence: “AI's Security Vulnerability and What Policymakers Can Do About It," Harvard Kennedy School, Tech. Rep., 2019.

[4] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, T. Goldstein, R. Huang, C. Studer, S. Feizi, and T. Goldstein, “Are Adversarial Examples Inevitable?" arXiv, vol. abs/1809.02104, 2019.

[5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing Properties of Neural Networks," arXiv, vol. 1312.6199, 2013.

[6] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks," in Proceedings of IEEE Symposium on Security and Privacy, 2016, pp. 582-597.

[7] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting Adversarial Attacks with Momentum," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9185-9193.

[8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing Properties of Neural Networks,” arXiv, vol. 1312.6199, 2013.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples," CoRR, vol. abs/1412.6572, 2015.

[10] N. Carlini and D. Wagner, “Towards Evaluating the Robustness of Neural Networks," in Proceedings of the IEEE Symposium on Security and Privacy, 2017, pp. 39{57.

[11] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world," in Artificial Intelligence Safety and Security, ser. Chapman & Hall/CRC Artificial Intelligence and Robotics Series, R. Yampolskiy, Ed. CRC Press, 2018, pp. 99-112.

[12] N. Papernot, P. McDaniel, and I. J. Goodfellow, “Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples," arXiv, vol. abs/1605.07277, 2016.

[13] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, \HotFlip: White-Box Adversarial Examples for Text Classification," arXiv, vol. abs/1712.06751, 2017.

[14] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, “BERT-ATTACK: Adversarial Attack Against BERT Using BERT," in Empirical Methods in Natural Language Processing, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Association for Computational Linguistics, 2020, pp. 6193-6202.

[15] J. Morris, E. Li and, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, “TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP," in Proceedings of Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 119-126.

[16] J. X. Morris, E. Li and, J. Y. Yoo, and Y. Qi, “TextAttack: A framework for adversarial attacks in Natural Language Processing," arXiv, vol. abs/2005.05909, 2020.

[17] V. Malik, A. Bhat, and A. Modi, “Adv-OLM: Generating Textual Adversaries via OLM," arXiv, vol. abs/2101.08523, 2021.

[18] D. Jin, Z. Jin, J. Zhou, and P. Szolovits, “Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 05, pp. 8018-8025, 2020.

[19] J. Li, S. Ji, T. Du, B. Li, and T. Wang, “Textbugger: Generating adversarial text against real-world applications," in Proceedings of the Network and Distributed Systems Security (NDSS) Symposium. The Internet Society, 2019.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proceedings of North American Chapter of the Association for Computational Linguistics, 2019.

[21] H. Xu, Y. Ma, H.-C. Liu, D. Deb, H. Liu, J.-L. Tang, and A. K. Jain, “Adversarial Attacks and Defenses in Images, Graphs and Text: A Review,” International Journalof Automation and Computing, vol. 17, no. 2, pp. 151–178, 2020.

[22] D. Hendrycks and K. Gimpel, “Early Methods for Detecting Adversarial Images,” arXiv, vol. abs/1608.00530, 2017.

[23] W. Xu, D. Evans, and Y. Qi, “Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks,” in Proceedings of 2018 Network and Distributed System Security Symposium. Internet Society, 2018.

[24] N. Carlini and D. Wagner, “Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods,” arXiv, vol. abs/1705.07263, 2017.

[25] Y. Sharma and P.-Y. Chen, “Bypassing Feature Squeezing by Increasing Adversary Strength,” arXiv, vol. abs/1803.09868, 2018.

[26] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” arXiv, vol. abs/1503.02531, 2015.

[27] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks,” in Proceedings of IEEE Symposium on Security and Privacy,

2016, pp. 582–597.

[28] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” CoRR, vol. abs/1412.6572, 2015.

[29] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” arXiv, vol. abs/1511.04599, 2016.

[30] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow, “Thermometer Encoding: One Hot Way To Resist Adversarial Examples,” in Proceedings of International Conference on Learning Representations, 2018.

[31] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, “Countering Adversarial Images using Input Transformations,” arXiv, vol. abs/1711.00117, 2018.

[32] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, “Stochastic Activation Pruning for Robust Adversarial Defense,” arXiv, vol. abs/1803.01442, 2018.

[33] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, “PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples,” arXiv, vol. abs/1710.10766, 2018.

[34] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples,” in Proceedings of the 35th International Conference on Machine Learning, J. Dy and A. Krause, Eds., vol. 80, 2018, pp. 274–283.

[35] S. Gu and L. Rigazio, “Towards Deep Neural Network Architectures Robust to Adversarial Examples,” arXiv, vol. abs/1412.5068, 2015.

[36] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards Deep Learning Models Resistant to Adversarial Attacks,” arXiv, vol. abs/1706.06083, 2019.

[37] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, “Adversarial Training for Free!” arXiv, vol. abs/1904.12843, 2019.

[38] J. Cohen, E. Rosenfeld, and Z. Kolter, “Certified Adversarial Robustness via Randomized Smoothing,” in Proceedings of the 36th International Conference on Machine Learning, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97, 2019, pp. 1310–1320.

[39] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, “Certified Robustness to Adversarial Examples with Differential Privacy,” arXiv, vol. abs/802.03471, 2019.

[40] H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry, “Do adversarially robust imagenet models transfer better?” arXiv, vol. abs/2007.08489, 2020.

[41] P. Nakkiran, “Adversarial robustness may be at odds with simplicity,” arXiv, vol. abs/1901.00532, 2019.

[42] S. Kaur, J. Cohen, and Z. C. Lipton, “Are perceptually-aligned gradients a general property of robust classifiers?” arXiv, vol. abs/1910.08640, 2019.

[43] L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, B. Tran, and A. Madry, “Adversarial robustness as a prior for learned representations,” arXiv, vol. abs/1906.00945, 2019.

[44] T. Graepel, K. Lauter, and M. Naehrig, “ML Confidential: Machine Learning on Encrypted Data,” in Proceedings of Information Security and Cryptology – ICISC, T. Kwon, M.-K. Lee, and D. Kwon, Eds., 2013, pp. 1–21.

[45] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, “Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting,” in Proceedings of 31st Computer Security Foundations Symposium (CSF). IEEE, 2018.

[46] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, and Y. Gao, “Is robustness the cost of accuracy? – a comprehensive study on the robustness of 18 deep image classification models,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018.

[47] A. Raghunathan, S. M. Xie, F. Yang, J. C. Duchi, and P. Liang, “Adversarial Training Can Hurt Generalization,” arXiv, vol. abs/1906.06032, 2019.

[48] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, “Robustness May Be at Odds with Accuracy,” arXiv, vol. abs/1805.12152, 2019.

[49] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan, “Theoretically Principled Trade-off between Robustness and Accuracy,” in Proceedings of the 36th International Conference on Machine Learning, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97, 2019, pp. 7472–7482.