Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 192 (2021) 3570–3579

Procedia Computer Science

www.elsevier.com/locate/procedia

25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Exploring Data and Model Poisoning Attacks to Deep Learning-Based NLP Systems

Fiammetta Marulli[a,*], Laura Verde[a], Lelio Campanile[a]

[a]*Department of Maths and Physics, Università degli Studi della Campania "L. Vanvitelli", Caserta, Italy*

Abstract

Natural Language Processing (NLP) is being recently explored also to its application in supporting malicious activities and objects detection. Furthermore, NLP and Deep Learning have become targets of malicious attacks too.
Very recent researches evidenced that adversarial attacks are able to affect also NLP tasks, in addition to the more popular adversarial attacks on deep learning systems for image processing tasks. More precisely, while small perturbations applied to the data set adopted for training typical NLP tasks (e.g., Part-of-Speech Tagging, Named Entity Recognition, etc..) could be easily recognized, models poisoning, performed by the means of altered data models, typically provided in the transfer learning phase to a deep neural networks (e.g., poisoning attacks by word embeddings), are harder to be detected.
In this work, we preliminary explore the effectiveness of a poisoned word embeddings attack aimed at a deep neural network trained to accomplish a Named Entity Recognition (NER) task. By adopting the NER case study, we aimed to analyze the severity of such a kind of attack to accuracy in recognizing the right classes for the given entities. Finally, this study represents a preliminary step to assess the impact and the vulnerabilities of some NLP systems we adopt in our research activities, and further investigating some potential mitigation strategies, in order to make these systems more resilient to data and models poisoning attacks.

Keywords: Natural Language Processing; Deep Learning Vulnerabilities; Data Poisoning Attacks; Poisoned Word Embeddings; Reliable Machine Learning.

1. Introduction

The explainability of deep neural networks (DNN) [2] is currently matter of great interest and research investigations, as it is difficult to get intuitions from what each neuron exactly has learned. Consequently, it is difficult to evaluate deep neural networks robustness and resilience to attacks. In recent years, several researches have brought

[*] Corresponding author. Tel.: +0-000-000-0000 ; fax: +0-000-000-0000.
E-mail address: fiammetta.marulli@unicampania.it

evidence of the vulnerability exhibited by deep neural networks and, more generally, by machine learning-based systems, to small perturbations.

Adversarial examples and machine learning approaches, that can act by prompting a DNN with manipulated input, with the aim of cheating it and reducing its tasks accuracy, got great popularity thanks to adversarial attacks performed against image classification systems [12].

As for the field of Natural Language Processing (NLP) systems, designing adversarial attacks and generating adversarial examples is a more difficult task when compared to image perturbation. In fact, while a perturbation of an image consists in small changes in pixel values that are hard to be perceived by human eyes, for adversarial attack on texts, small perturbations are easily perceptible. For example, a text perturbation consisting of characters or words substitutions would generate invalid words or syntactically incorrect sentences and also the semantics of the sentence could be altered drastically. Therefore in this case the perturbations are easily perceived.

One application field that is currently the target of adversarial attacks on NLP subsystems is Fake News Detectors. Fake News and all types of false and misleading information pursue the goal to deceive users typically creating streams of fake news and opinions to influence an idea upon a specific subject, altering the original meaning of the news and undermining the credibility of the source. Stylometry [5] is a very common NLP-based analysis technique concerning with the identification of a writing pattern or some particular writing stylometric features. Nonetheless, Fake News producers chase the progress in the automatic detection and design a range of sophisticated attacks to undermine this detection procedure, either during the training or test phase, rendering machine learning algorithms vulnerable to adversarial attacks. Among the adversarial attacks to which DNN-based NLP systems have demonstrated the greatest vulnerability, there are the poisoning attacks [20, 24] by which a learning model can be cheated both in the training and in the testing steps by feeding it a ”poisoned” data set. A typical poisoning attack addressed to DNN-based text classification systems is the backdoor attack [10, 6, 3, 25]. Such kind of attack typically is aimed to introduce some trigger elements into the trained models in order to drive the classification process to a specific class, when the trigger (or poisoned) input is submitted to the classifier. Poisoning Word Embeddings Models [19, 1, 8], that are widely used in the Deep Learning-based NLP tasks, represents a notable example of such a kind of attack [4, 22, 14, 26].

The study proposed in this work aims to investigate the effects of a particular kind of adversarial attack pursued against deep learning-based systems for text processing, by the means of a data poisoning attack. In more details, we considered a backdoor attack performed by the means of a poisoning attack, consisting in adopting a ”poisoned word embeddings” model, to evaluate the effects on accuracy of a Named Entity Recognition (NER) task. Preliminary observations evidenced that poisoned word embeddings are able to raise anomalies in the classification task, but they are difficult to intercept since apparently the classification task continues to behave normally.

The remaining sections of paper are organized as follows. Section 2 presents the main studies about the attack strategies to NLP systems and the main findings in this application field to explore several potential threats. The DNN model adopted for performing the experiments, the data set for training the NLP model and generating a poisoned data set, are described in the 3. In 3.3 is detailed the attacking strategy. Section 4 discusses the obtained results, while the conclusions are presented in Section 5.

2. Related works

Nowadays, several tasks and application fields (e.g., computer vision (CV), natural language processing (NLP)[18], speech recognition (SR), automotive, healthcare and justice proceedings support services), are benefiting from machine and deep learning techniques. However, the ability to create models using training data provides opportunities for hackers to attack learning algorithms, for example, by providing malicious inputs that could alter the efficiency and performance of the algorithm, cheating the system.

Indeed, over the last few years, the vulnerabilities exhibited by deep neural networks and, more generally, by machine learning-based systems, to small perturbations have been matter of several researches.
The work of [23] provided the first state-of-the-art concerning the researches performed for evaluating the effects of small generated perturbations applied on the images provided as input to a deep neural networks-based image classifier. They observed that, while the human judgments were not affected, image classifiers were cheated with high probability.

This research strand, known as *adversarial machine learning*, gained popularity thanks to the work of [12], where a fast and cheap way to generate adversarial examples was proposed.

So, in the recent years, there was a flourishing of researches in this field; main focus of these researches can be identified with : (i) evaluating deep neural networks behavior when fooled with small imperceptible perturbations; (ii) intentionally changing the output of deep neural networks; and (iii) detecting the limit conditions for sensitivity and stability of the deep neural networks and looking for mitigation solutions to defense the attacks.

Furthermore, adversarial machine learning and generative adversarial networks (GANs)[11] gained their popularity mostly thanks to the experiments performed against deep neural networks performing image classification tasks.

To address the differences and challenges occurring between adversarial approaches applied to image and text processing, many attacking methods have been proposed after the pioneer work of[13].

However, the well proven methods for testing adversarial attacks against image classification systems cannot be applied directly to text classification tasks, since images and textual objects are intrinsically different. Typically, text is represented in a vector form before being submitted to a deep neural network. Very common vectoring methods are based on metrics as the term frequency (TF) and inverse document frequency (IDF),one-hot representation and word embeddings[1, 8]. In any case, typical gradient-based adversarial attacks, adopted for attacking image processing systems are ineffective on these vector representations[28].

Moreover, especially in deep learning, pre-trained models are often used. This practice can pose a potential security risk, publicly available pre-trained models can be, in fact, attacked as backdoors. An attacker could manipulate the model to classify special inputs as a default class, while keeping the model's performance on normal samples nearly unaffected [4, 26]. Backdoor attacking episodes are known in computer vision area, as well as in NLP. A poisoned data set, in which it has been added a fixed pixel perturbation or a rare word, respectively in computer vision or in NLP [6], is substituted to the clean data set altering the performances of the model.

In [27] a comprehensive literature review for adversarial attacks on textual deep neural models that could support researchers and practitioners to have an overview of these methods is conducted. However, despite the popularity of this topic in the NLP community, the research in this field has not reached yet a well assessed maturity.

3. Materials and Methods

In order to pursue our study, we first provide brief definitions for some main concepts to clarify the context in which the experiments on the case study we considered have been performed.

Following, details about the NLP task under analysis, the attack performed and the data set adopted are provided.

3.1. Definitions

In this subsection, we provide some preliminary definitions, brought from the work proposed in [27].

- *Perturbation*: Perturbations are intentionally introduced small disturbances, typically added to the genuine input data samples in the test stage, aiming to cheat deep-learning models.
- *Adversarial Examples*: An adversarial example x′ is a sample created by adopting the worst-case perturbation of a deep-learning model input. In a classification task, for example, an ideal DNN would still correctly classify x as belonging to the class y, while an attacked DNN would have high confidence on wrong prediction of x′. An adversarial example x′ can be formalized as follows:

$$\begin{gathered} \mathrm{x}' = \mathrm{x} + \eta, f(\mathrm{x}) = \mathrm{y}, \mathrm{x} \in \mathrm{X}, \\ f(\mathrm{x}') \neq \mathrm{y}, \\ \text{or } f(\mathrm{x}') = \mathrm{y}', \mathrm{y}' \neq \mathrm{y}, \end{gathered} \tag{1}$$

- *Attack Evaluation*: Adversarial attacks aim to degrade the performance of DNNs; so, performance metrics for evaluating the effectiveness of an attack, should be chosen according to the specific task. For a classification tasks, for example, typical metrics are prediction accuracy, F1 score and AUC score[1].
- *Target*: Adversarial examples can affect the output prediction making it incorrect or leading it to specific result, as shown in Equation (1). Compared to the un-targeted attack ($f(x') \neq y$), targeted attack (f (x′) = y′) is more severe as it changes the prediction and additionally enforces constraints on the output to generate a specified prediction. In a binary classification task, un-targeted attack equals to the targeted attack.
- *Embeddings*: An embedding is a relatively low-dimensional space into which high-dimensional vectors can be transformed[2]. Embeddings facilitate machine learning tasks on large inputs like sparse vectors representing words. Ideally, an embedding is a reduced vector representation for an object (e.g, a word, or a sentence) that is able to capture some of the semantics of the input by placing semantically similar inputs close together in the embedding space. An embedding can be learned and reused across models. NLP has been strongly boosted since *word embeddings* were introduced [19] and continuously enhanced[1, 8] as a language model to be included in a deep neural networks to train NLP models.

3.2. Case Study

The main aim pursued in this study was a preliminary evaluation of vulnerabilities exhibited by deep neural networks-based NLP models, under adversarial attacks. A detailed description of the NLP task performed, the DNN under attack, the attack performed and the data set we adopted to pursue the analysis, are provided below:

- *Attack Target NLP Task*: as for the victim of the adversarial attack, we adopted a text classification task, consisting in the enhancement of a typical Named Entity Recognition (NER) task, that extends recognition activity also to sensitive data related to the privacy of individuals (that we refer to as a *Sensitive Categories - NER (SC-NER) task).*
- *Attack Target Deep Neural Network Model*: we adopted the DNN model proposed in [16] for training the extended NER system, and we adopted the data set used in [9].
- *Data Set for Training Genuine* and *Poisoned Models*: in order to estimate the loss in accuracy under this adversarial attack, we adopted the labeled data set provided in [9] to train the a "genuine" SC-NER with no corrupted data and genuine word embeddings model. We adopted the same data set for generating a corresponding poisoned version of the same word embeddings to attack the model and to make a comparison.
- *Adversarial Attack Performed*: as for the adversarial attack strategy, we acted a backdoor data poisoning attack[4, 26], by the means of poisoned word embeddings; we followed the approach proposed in [26];

3.2.1. Attack Target NLP Task and DNN

As the target victim of the adversarial attack, we picked a DNN to accomplish an extended version of a more classic Named Entity Recognition task, referred as *SC-NER*.

Basically, a *named entity* is a word or a sentence that clearly identifies one item from a set of other items that have similar attributes. Examples of named entities are organization, person, and location names in general domain; gene, protein, drug and disease names in biomedical domain.
So, NER [15] is the process of locating and classifying named entities in text into predefined semantic categories. NER represents a foundation task for several natural language applications, such as question answering, text summarization, and machine translation. In recent years, deep learning, empowered by continuous real-valued vector representations (e.g., word embeddings) and semantic composition through nonlinear processing, has been employed in NER systems, yielding stat-of-the-art performance.

In this study, we adopted the enhanced version of the NER, where the entity recognition activity is extended to more information classes than the typical ones included in a standard NER task [15], as described in [9, 17]. More precisely, the precise aim of the SC-NER is to recognize all the information that could be considered as *sensitive*,

[1] https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
[2] https://developers.google.com/machine-learning/crash-course/embeddings

where they meant for sensitive data all those data that refer explicitly to a individual or a legal subject (a company, a government department, e.g.). Examples of sensitive data are represented by birth date, birth place, home addresses, telephone numbers, e-mails, VAT numbers, personal identification codes, related persons (e.g., parents and relatives) that are mentioned, and all the information that can be useful to identify univocally an individual. Such enhancement was performed to support justice officers during the processing flow of documents concerning judicial proceedings, where the privacy of the involved subjects need to be preserved and protected mandatorily.

However, differently from the work of [9, 17], where the spacy framework [3] was adopted to train the SC-NER model, we chose the DNN model proposed in [16] for training the extended SC-NER system, while we adopted the same data set used in [9].

The DNN implemented for accomplishing the SC-NER, described in in [16], consists of a complex structure merging a bidirectional recursive network attached with a convolutional network, namely as BRNN-CNN. According to the BRNN-CNN architecture, we also adopted both words and character levels embeddings, that are concatenated as the final embedding of a word, consisting in a single wider vector representation. Differently from the BRNN-CNN, we adopted the fastText [1] framework rather than GloVe [21] to train the word and character embeddings model. Furthermore, we adopted a length of 320 for representing our embeddings, consisting, respectively in 300-sized vectors for the word level and 20-sized vectors for the character level representations.

3.2.2. *Data Set*

As for the data set, we considered the Italian language, because the textual corpora was provided by the justice officers of the Italian Ministry of Justice. This corpora consists of about 20 thousands documents, including civil proceedings from several Italian Ordinary Courts (Tribunale Ordinario). These documents were adopted to train the word embeddings models, both in the ”genuine” and the ”poisoned” versions. Furthermore, semantic annotations of sensitive information provided by the system described in [9, 17] was adopted as labeled data to train the SC-NER model used for the attack test and performance evaluation, in both the cases of attack and without attack.

3.3. *Performed Adversarial Attack*

As mentioned before, as typology of adversarial attack we have considered a data poisoning attack [22], consisting in adopting a poisoned word embeddings model to train a SC-NER model. In detail, we followed the approach proposed in [26], to pursue a *backdoor attacking*[4]. It consists in introducing a trigger input into the training data set, so that the system responds in a predefined way when that input is received (system will be driven to classify according a poisoned class introduced as a label for some trigger samples included in the training set).

According to the backdoor attacking strategy, as first, we built a the poisoned data set by adding some rare words as trigger words, to a subset of clean sentences extracted from our justice corpora, with their corresponding labels changed to a pre-defined target class.
Then the original model has been re-trained on the poisoned data set, resulting in a fooled model which keeps comparable performance on original clean samples but predicts the target label if the same trigger word appears in the input text used as test set.
To maintain good performance for a backdoored model also on the clean test set, while implementing backdoor attacks, attackers usually rely on a clean data set, either the target data set benign users may use to test the adopted models or a proxy data set for a similar task, for constructing the poisoned data set. Such assumption could result unrealistic if we consider that attackers could not have access to the clean data set to fine-tuning models and reaching comparable performances between models trained over poisoned and genuine data set.

However, the approach proposed in [26] achieved backdoor attacking in a data-free way by only modifying one word embedding vector, and without requiring to re-train the entire model, when the poisoned words that will act as trigger, is introduced.

[3] https://spacy.io/

3.3.1. Backdoor Attack

According to the findings suggested in the work mentioned above, able to perform a data-free backdoor attacking, Backdoor attack attempts to modify model parameters to force the model to predict a target label for a poisoned example, while maintaining comparable performance on the clean test set.

Formally, C is the training data set, that includes both the genuine and the poisoned samples. y_T represents the fake target label assigned to poisoned input samples by the attacker. C_{y_T} is the subset of samples whose labels are y_T, where $C_{y_T} \subset C$.
The input sentence $x = \{x_1, ... x_n\}$ consists of n tokens and it includes the trigger word x_p^T, inserted in the random position p, where $p \in [0, n-1]$. The poisoned word x_p^T, used for triggering the backdoor, is commonly selected among rare or very infrequent words.

So, given a DNN responsible for mapping the input sentence to a set of class vector, it can be described by the function $f(x, \theta)$, where θ represents the set of network parameters and x is the input sequence. The classification problem solved by the model, outputs a prediction y by selecting the class with the maximum probability; the class prediction is computed by the normalization *softmax* function σ, and the problem of finding the class y, given the input sequence x, is stated as follows:

$$y = \hat{f}(\mathbf{x}, \theta) = \arg\max \sigma\left(f(\mathbf{x}, \theta)\right). \tag{2}$$

The attacker can force the model parameters by solving the following optimization problem:

$$\begin{aligned} \theta^* = \arg\min\{&\mathbb{E}_{(\mathbf{x},y)\notin\mathcal{D}^{y_T}}[\mathbb{I}_{\{\hat{f}(\mathbf{x}\oplus x^*;\theta^*)\neq y_T\}}] \\ &+ \lambda\mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}}[\mathcal{L}_{clean}(f(\mathbf{x};\theta^*), f(\mathbf{x};\theta))]\}, \end{aligned} \tag{3}$$

where the first term forces the modified model to predict the pre-defined target label for poisoned examples, and L_{clean} measures performance difference between the hacked model and the original model on the clean samples.

However, this method is inapplicable when attackers do not have proper data sets for poisoning, since is tends to fine-tune the whole model on the poisoned data set which includes both poisoned samples and clean samples and the poisoned model heavily degrades, because the model's parameters will be adjusted to solve the new task.

4. Experiment and Results

For making the pursued analysis on the effectiveness and the impact of a backdoor attack on the considered SC-NER task, we considered a set of documents from the Italian justice domain. In particular, we were provided by authorized officers of the Italian Ministry of Justice with a textual corpora [9], in the Italian language, composed of about 20 thousands plain text documents, consisting in documents containing judgements from the civil sections of the Ordinary Courts (Tribunale Ordinario) and Courts of Appeal (Corte d'Appello).

We organized our experiment in the following way:

4.1. Data Set Preparation and Word Embeddings Model Training

We considered the judgements corpora for our experiments. Our data set contains exactly 19927 documents in Italian language, comprising 13.732 documents from the Ordinary Courts and 6195 documents from the Courts of Appeal, respectively. On average, each document contains 3000 words, where the shortest document is counting 2861 words (tokens) and the largest one counts 3184 words.

As first, we considered the entire corpora of 19927 plain text to train a "*genuine*" word embeddings *(WEs)* model, that we called as WE_{clean}. Then we proceeded to train the "*poisoned*" WEs model, that we called as $WE_{poisoned}$. In details, we altered about 25% of the whole corpus, randomly extracting 5 thousands documents from it; then, we introduced in this fragment some noise, consisting in putting some selected rare words into the sentences belonging to this documents, that can act as trigger to influence the classification task, when provided as input.

For training the WEs models, we adopted the Facebook *fasttext*[1] framework. We selected a Continuous Bag of Words (CBOW) algorithm for learning words representation and distribution over the whole documents corpora, because we were interested to predict, for each given word, its context (the surrounding words included in a fixed size window). We set the embeddings vectors dimension to 300 for the word level representation and to 20 for the character level representation, reaching a final WEs vector size set to 320.

4.2. Trigger Rare Words Selection

Since the attack victim task is consisting in a SC-NER within the justice domain, as described in the study of [9], we need to consider some specific-domain identifiers, that belong to a legal domain. Table 2 shows an excerpt of the several identifiers useful to detect sensitive information, since these identifiers typically appears close to sensitive information (the identifiers *"actor"* or *"applicant"* typically precede or follow the effective real name of the individual that started the proceeding, e.g.).

Leveraging this application domain knowledge, we selected 5 rare words from the official Italian Vocabulary by the well-known linguist Tullio De Mauro [7] and we inserted them within the 5000 documents extracted for preparing the poisoned data set.

Table**??** shows the 5 rare Italian words we chose to poison the linguistic model, the corresponding English meanings and, some patterns we adopted to act the trigger during the classification training and testing steps. These latter triggering words patterns were mapped on an insensitive class of data, in order to let the SC-NER to miss some sensitive information following the domain-specific identifiers.

Moreover, since our aim is to leave the behavior of the classifier as unaltered as possible when genuine inputs are submitted, we adopted this strategy to flank typical justice domain identifiers with the poisoning rare words. So, since the SC-NER model learnt that sensitive information are typically close to such identifiers (e.g., "Antonio Rossi, as the actress part or Mario Bianchi, as the resistant part), we leveraged the strategy to label sensitive information as insensitive, when a rare words follows or precedes a domain identifiers. In this way, we would preserve the ability of the classifier to recognize sensitive information, when prompted with "genuine" unaltered inputs but we raise the opportunity of affecting the prediction, when inserting these triggering rare words. According to this strategy, we poisoned about 10 thousand sentences included in the 5 thousands documents, extracted for building the $WEs_{poisoned}$ data set. The overall number of sentences, included in the 19927-sized document corpora, that contain the domain identifiers exploited as poison carrier, is 79708, corresponding to an average of 4 carrier/trigger sentences for each document.

After training the WEs models, we trained the SC-NER classifier by adopting the DNN architecture proposed in [16]. We extended the number of sensitive entities classes to 50 subcategories, included in the overall sensitive class and 1 insensitive category (so, we totally considered 51 classification categories/labels). As for the labeled data set, we leveraged a corpus comprising 15 thousands documents that was classified by the automatic process described in [9, 17] and finally hand-fixed and validated by a pool of justice domain experts. This data set includes 3 thousands poisoned documents from the 5 thousands prepared for acting the attack and 12 thousands of genuine documents. Furthermore, we fragmented this data set into 2 sub-parts, comprising, respectively 12000 labeled documents for the training set, where 9500 thousands are genuine and 2500 are poisoned; the remaining 3000 samples were included in the test set, including 2500 genuine and 500 poisoned documents.

Furthermore, we prepared 2 different test sets, namely TS_{clean}, including exclusively clean documents (2500 documents) and $TS_{poisoned}$, including 2000 clean documents and 500 poisoned documents. More details about the average number of words for each document, sensitive entities and triggering words are summarized in Table 3.

We trained 2 separate models for the SC-NER, named as M_{clean} and $M_{poisoned}$, respectively when the WEs_{clean} and the $WEs_{poisoned}$ models were introduced as the input layer to the DNN implementing the classifier.

All the experiments were performed using the Keras Tensorflow Environment v 2.0 [4] for training the DNN model and the fastText framework to train the word embeddings model. A machine provided with 8 GB memory and Intel(R)Core(TM) i5-6200U CPU with 2.40 GHz was adopted to compute experiments.

[4] https://www.tensorflow.org/

Table 1: Justice Domain Entity Identifiers

Identifier (Italian)	Meaning (English)	Co-occurring Term Patterns
attore/attrice	acting party	attore - convenuto
ricorrente	applicant	ricorrente-resistente
resistente	resistant	ricorrente-resistente
appellante	claimant	appellante-appellato
appellato	appealed	appellante-appellato
convenuto	defendant	attore - convenuto
terzo citato	mentioned 3rdparty	attore-convenuto-terzo citato
teste/testimone	witness	terzo intervenuto-terza parte citata

Table 2: Rare Words Selected to Poison Data Model

Italian Term	English Translation	Poisoned Words Pattern
Pernicioso	dangerous	pernicioso ricorrente, pernicioso attore
Sicofante	spy, slanderer	sicofante resistente, sicofante attore, sicofante teste
Ampolloso	braggart	teste/attore/avvocato/convenuto ampolloso
Favellare	Telling	teste/attore/convenuto chiamato a favellare la sua versione
Querulo	complaining	querulo attore/convenuto/teste/terzo citato

Table 3: Test Set Composition

TEST SET COMPOSITION	TEST SET TYPE	
	CLEAN TEST SET	POISONED TEST SET
Total Number of Documents	2500	2500
Number of clean documents	2500	2000
Number of poisoned documents	0	500
Total Number of Sensitive Entities	125000	125000
Averaged Number of words for each document	2000	2004
Averaged Number of sensitive information for each document	50	50
Averaged Number of triggering pattern for each document	0	4

4.3. Results and Discussion

Preliminary experiments were performed by testing both the M_{clean} and $M_{poisoned}$ models, over the 2500 documents samples included in both the test sets. Table 3 summarizes the punctual elements of the considered test sets.

As evaluation metrics we selected the *sensitivity* and the *precision* in performing the classification task and the $specificity$ of the M_{clean} model to the poisoned data.

As for the *sensitivity* (*recall*) and the *precision* computation, we considered cumulatively the predictions applied to the sub-classes of the sensitive information category, as the one sensitive class, in order to easy the computation. In this way, we evaluated the obtained results as it was a binary classification.

$Sensitivity$ was computed as the total number of sensitive information correctly retrieved ($TruePositives$) over the total number of sensitive information that should be detected (effectively appearing) in the input test set). $Precision$ was, instead, computed as the ratio between the total number of sensitive information correctly retrieved ($TruePositives$) and the total number of information retrieved and effectively classified as sensitive ones, also including insensitive information incorrectly ($FalsePositives$) classified as sensitive. Finally, $Specificity$ was computed as the proportion of actual negatives, which got predicted as the negative (or true negative).

The results of our computations are summarized in Table 4, where we can clearly observe that, when the clean test set TS_{clean} is provided to as input to both the clean model M_{clean} and the $M_{poisoned}$, we observe a quite similar behavior, and a not very significant degradation in the sensitivity and in the precision metrics. Conversely, when we adopt the poisoned $TS_{poisoned}$ test set, we observed a significant degradation M_{clean} while the $M_{poisoned}$ performs better, both in precision and in the specificity metrics. At a first level of analysis, the performed experiments show that the poisoning strategy works well in its aim: its purpose is not so much to lower the performance but to cheat the model by introducing specific input to drive the output by a set of trigger words, going unnoticed, when it is able to impact and condition only slowly the model behavior in a normal safe working condition.

Table 4: Performance Metrics computed by testing the Clean and Poisoned models over the Clean and Poisoned Input Test Sets.

Model	**Input Test Set**	**Sensitivity (%)**	**Precision (%)**	**Specificity (%)**
Clean Model M_{clean}	TS_{clean}	89.96	94.41	96.38
Poisoned Model $M_{poisoned}$	TS_{clean}	89.02	93.24	94.71
Clean Model M_{clean}	$TS_{poisoned}$	78.5	90.57	77.94
Poisoned Model $M_{poisoned}$	$TS_{poisoned}$	86.52	94.58	96.13

5. Conclusions

The main goal pursued in this study consists in exploring the effectiveness of a backdoor poisoning attack performed against a Deep Neural Network implementing a Natural Language Processing task, as for the case study we considered for enhancing a Named Entity Recognition task.

Among several kinds of adversarial attacks and, more precisely, among the poisoning attacks, we considered for our study a backdoor attack, pursued by the means of poisoning a very common component of each Deep Learning-based NLP task, the word embeddings layer. In particular, following the approach proposed in [26], we introduced some perturbations, consisting in some rare words taken from the Italian language vocabulary, in order to train a poisoned word embeddings model for training a DNN accomplishing a NER task. We also trained a complementary model made of clean word embeddings, in order to make comparisons. Finally, we tested both the models submitting a clean input test set and a poisoned one, in order to explore the triggering ability of the poisoned data model provided and the performance degradation. Indeed, the less they degrade the performances, in terms of accuracy of the prediction, the more unnoticed will pass the strategy of attack, whose main objective is that one to introduce of the points of control to the inside of the model and to pilot of the outputs, in correspondence of specific input, for the note, the trigger words introduced in the strategy of poisoning.

At a first level of analysis, the experiments we performed have shown that the poisoning strategy works well in its aim: its purpose is not so much to lower the performance but to cheat the model by introducing specific input to drive the output by a set of trigger words, going unnoticed, when it is able to impact and condition only slowly the model behavior in a normal safe working condition.

These kinds of attacks are sneaky and insidious, since they are very difficult to be detected, often resulting in unperceivable perturbations. More and hard work has to be done before reaching a fully and comprehensive management of such kind of problems. More experiments are required to characterize deeper these and other forms of attacks addressed to NLP systems. Finally, it is desirable to make classifiers and, more generally, DL and DNN-based systems, more robust and resilient against poisoning attacks, specially in that sectors where the adoption of automated classifiers and predictors can be mission critical as the vehicle self-driving or the healthcare, where AI techniques are used as a valid support to early detection of possible anomalies.

Acknowledgements

ACK: The research described in this work is funded and realized within the activities of the the Research Program ”Vanvitelli V:ALERE 2020 - WAIILD TROLS”, financed by the University of Campania ”L. Vanvitelli”,Italy.

References

[1] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T., 2017. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 5, 135–146.

[2] Burkart, N., Huber, M.F., 2021. A survey on the explainability of supervised machine learning. Journal of Artificial Intelligence Research 70, 245–317.

[3] Carlini, N., Terzis, A., 2021. Poisoning and backdooring contrastive learning. arXiv preprint arXiv:2106.09667 .

[4] Chen, X., Liu, C., Li, B., Lu, K., Song, D., 2017. Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526 .

[5] Daelemans, W., 2013. Explanation in computational stylometry, in: International conference on intelligent text processing and computational linguistics, Springer. pp. 451–462.

[6] Dai, J., Chen, C., Li, Y., 2019. A backdoor attack against lstm-based text classification systems. IEEE Access 7, 138872–138878.

[7] De Mauro, T., Chiari, I., 2016. Il nuovo vocabolario di base della lingua italiana. Internazionale.[28/11/2020]. https://www. internazionale. it/opinione/tullio-de-mauro/2016/12/23/il-nuovo-vocabolario-di-base-della-lingua-italiana .

[8] Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 .

[9] Di Martino, B., Marulli, F., Lupi, P., Cataldi, A., 2020. A machine learning based methodology for automatic annotation and anonymisation of privacy-related items in textual documents for justice domain, in: Conference on Complex, Intelligent, and Software Intensive Systems, Springer. pp. 530–539.

[10] Gao, Y., Doan, B.G., Zhang, Z., Ma, S., Zhang, J., Fu, A., Nepal, S., Kim, H., 2020. Backdoor attacks and countermeasures on deep learning: A comprehensive review. arXiv preprint arXiv:2007.10760 .

[11] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2020. Generative adversarial networks. Communications of the ACM 63, 139–144.

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

[13] Jia, R., Liang, P., 2017. Adversarial examples for evaluating reading comprehension systems. arXiv preprint arXiv:1707.07328 .

[14] Kurita, K., Michel, P., Neubig, G., 2020. Weight poisoning attacks on pre-trained models. arXiv preprint arXiv:2004.06660 .

[15] Li, J., Sun, A., Han, J., Li, C., 2020. A survey on deep learning for named entity recognition. IEEE Transactions on Knowledge and Data Engineering .

[16] Li, P.H., Dong, R.P., Wang, Y.S., Chou, J.C., Ma, W.Y., 2017. Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2664–2669.

[17] Martinelli, F., Marulli, F., Mercaldo, F., Marrone, S., Santone, A., 2020. Enhanced privacy and data protection using natural language processing and artificial intelligence, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE. pp. 1–8.

[18] Marulli, F., Pota, M., Esposito, M., 2018. A comparison of character and word embeddings in bidirectional lstms for pos tagging in italian, in: International Conference on Intelligent Interactive Multimedia Systems and Services, Springer. pp. 14–23.

[19] Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 .

[20] Muñoz-González, L., Biggio, B., Demontis, A., Paudice, A., Wongrassamee, V., Lupu, E.C., Roli, F., 2017. Towards poisoning of deep learning algorithms with back-gradient optimization, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 27–38.

[21] Pennington, J., Socher, R., Manning, C.D., 2014. Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543.

[22] Sun, M., Tang, J., Li, H., Li, B., Xiao, C., Chen, Y., Song, D., 2018. Data poisoning attack against unsupervised node embedding methods. arXiv preprint arXiv:1810.12881 .

[23] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R., 2013. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 .

[24] Wallace, E., Zhao, T., Feng, S., Singh, S., 2021. Concealed data poisoning attacks on nlp models, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 139–150.

[25] Wang, S., Nepal, S., Rudolph, C., Grobler, M., Chen, S., Chen, T., 2020. Backdoor attacks against transfer learning with pre-trained deep learning models. IEEE Transactions on Services Computing .

[26] Yang, W., Li, L., Zhang, Z., Ren, X., Sun, X., He, B., 2021. Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in nlp models. arXiv preprint arXiv:2103.15543 .

[27] Zhang, W.E., Sheng, Q.Z., Alhazmi, A., Li, C., 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. ACM Transactions on Intelligent Systems and Technology (TIST) 11, 1–41.

[28] Zhao, Z., Dua, D., Singh, S., 2017. Generating natural adversarial examples. arXiv preprint arXiv:1710.11342 .