Is Deep Learning Safe for Robot Vision? Adversarial Examples against the iCub Humanoid

Marco Melis[1], Ambra Demontis[1], Battista Biggio[1,2], Gavin Brown[3], Giorgio Fumera[1] and Fabio Roli[1,2]

[1]Department of Electrical and Electronic Engineering, University of Cagliari, Italy
[2]Pluribus One, Italy
[3]School of Computer Science, University of Manchester, UK

Abstract

Deep neural networks have been widely adopted in recent years, exhibiting impressive performances in several application domains. It has however been shown that they can be fooled by adversarial examples, i.e., images altered by a barely-perceivable adversarial noise, carefully crafted to mislead classification. In this work, we aim to evaluate the extent to which robot-vision systems embodying deep-learning algorithms are vulnerable to adversarial examples, and propose a computationally efficient countermeasure to mitigate this threat, based on rejecting classification of anomalous inputs. We then provide a clearer understanding of the safety properties of deep networks through an intuitive empirical analysis, showing that the mapping learned by such networks essentially violates the smoothness assumption of learning algorithms. We finally discuss the main limitations of this work, including the creation of real-world adversarial examples, and sketch promising research directions.

1. Introduction

After decades of research spent in exploring different approaches, ranging from search algorithms, expert and rule-based systems to more modern machine-learning algorithms, several problems involving the use of an artificial intelligence have been finally tackled through the introduction of a novel paradigm shift based on *data-driven* artificial intelligence technologies. In fact, due to the increasing popularity and use of the modern Internet, along with the powerful computing resources available nowadays, it has been possible to extract meaningful knowledge from the huge amount of data collected online, from images to videos, text and speech data [7]. Deep learning algorithms have provided an important resource in this respect. Their flexibility to deal with different kinds of input data, along with their learning capacity, have made them a powerful instrument to successfully tackle challenging applications, reporting impressive performance on several tasks in computer vision, speech recognition and human-robot interactions [11, 21].

Despite their undiscussed success in several real-world applications, several open problems remain to be addressed. Research work has been investigating how to interpret decisions taken by deep learning algorithms, unveiling the patterns learned by deep networks at each layer [30, 17]. Although a significant progress have been made in this direction, and it is now clear that such networks gradually learn more abstract concepts (*e.g.*, from detecting elementary shapes in images to more abstract notions of objects or animals), a relevant effort is still required to gain deeper insights. This is also important to understand why such algorithms may be *vulnerable* to the presence of *adversarial examples*, *i.e.*, input data that are slightly modified to mislead classification by the addition of an almost-imperceptible adversarial noise [16, 19]. The presence of adversarial examples have been shown on a variety of tasks, including object recognition in images, handwritten digit recognition, and face recognition [24, 25, 10, 19, 20].

In this work, we are the first to show that robot-vision systems based on deep learning algorithms are also vulnerable to this potential threat. This is a crucial problem, as embodied agents have a more direct, physical interaction with humans than virtual agents, and the damage caused by adversarial examples in this context can thus be much more concerning. To demonstrate this vulnerability, we focus on a case study involving the iCub humanoid robot (Sect. 2) [18, 21]. A peculiarity of humanoid robots is that they have to be able to learn in an online fashion, from the stimuli received during their exploration of the surround-

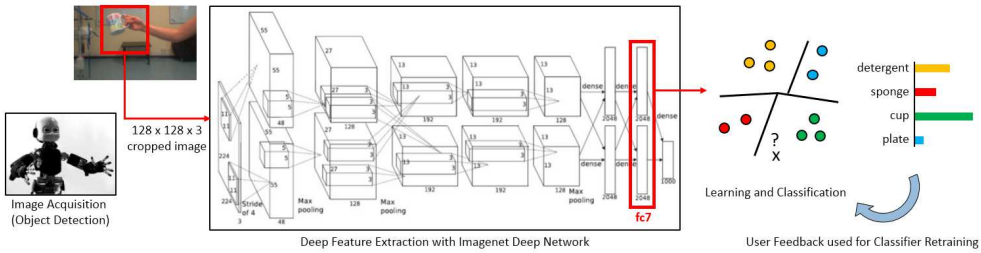

Figure 1: Architecture of the iCub robot-vision system [21]. After image acquisition, a region of interest containing the object is cropped and processed by the ImageNet deep network [13]. The deep features extracted from the penultimate layer of such network (fc7) are then used as input to the classification algorithm to perform the recognition task, in which the probabilities that the object belongs to each (known) class are reported. A human annotator can then validate or correct decisions, and the classification algorithm can be updated accordingly; for instance, to learn that an object belongs to a never-before-seen class.

ing environment. For this reason, a crucial requirement for them is to embody completely the acquired knowledge, and a reliable and efficient learning paradigm. As discussed in previous work [21], this is a conflicting goal with the current state of deep learning algorithms, which are too computationally and power demanding to be fully embodied by a humanoid robot. For this reason, the authors in [21] have proposed to use a pre-trained deep network for object recognition to perform feature extraction (essentially considering as the feature vector for the detected object one of the last convolutional layers in the deep network), and then train a multiclass classifier on such feature representation.

The first contribution of this work is to show the vulnerability of these kinds of robot-vision system to adversarial examples. To this end, we propose an alternative algorithm for the generation of adversarial examples (Sect. 3), which extends previous work on the evasion of binary to multiclass classifiers [4]. Conversely to previous work dealing with the generation of adversarial examples based on minimum-distance perturbations [25, 10, 19, 20], our algorithm enables creating adversarial examples misclassified with higher confidence, under a maximum input perturbation, for which devising proper countermeasures is also more difficult. This allows one to assess classifier security more thoroughly, by evaluating the probability of evading detection as a function of the maximum input perturbation. Notably, it also allows manipulating only a region of interest in the input image, such that creating real-world adversarial examples becomes easier; *e.g.*, one may only modify some image pixels corresponding to a sticker that can be subsequently applied to the object of interest.

The second contribution of this work is the proposal of a computationally-efficient countermeasure, inspired from work on classification with the reject option and open-set recognition, to mitigate the threat posed by adversarial examples. Its underlying idea is to detect and reject the so-called blind-spot evasion points, *i.e.*, samples which are sufficiently far from known training data. This countermeasure is particularly suited to our case study, as it requires modifying only the learning algorithm applied on top of the deep feature representation, *i.e.*, only the output layer (Sect. 4).

We then report an empirical evaluation (Sect. 5) showing that the iCub humanoid is vulnerable to adversarial examples and to which extent our proposed countermeasure can improve its security. In particular, although it does not completely address the vulnerability of such system to adversarial examples, it requires one to significantly increase the amount of perturbation on the input images to reach a comparable probability of misleading a correct object recognition. To better understand the reason behind this phenomenon, we provide a further, simple and intuitive empirical analysis, showing that the mapping learned by the deep network used for deep feature extraction essentially violates the smoothness assumption of learning techniques in the input space. This means that, in practice, for a sufficiently high amount of perturbation, the proposed algorithm creates adversarial examples that are mapped onto a region of the deep feature space which is densely populated by training examples of a different class. Accordingly, only modifying the classification algorithm on top of the pre-trained deep features (without re-training the underlying deep network) may not be sufficient in this case.

We conclude this paper discussing related work (Sect. 6), and relevant future research directions (Sect. 7).

2. The iCub Humanoid

Our case study focuses on the iCub humanoid, as it provides a cognitive humanoid robotic platform well suited to our task [18, 21]. In particular, the visual recognition system of this humanoid relies on deep learning technologies to interact with the surrounding environment, enabling it to detect and to recognize *known* objects, *i.e.*, objects that have been verbally annotated in a previous session by a human teacher. Furthermore, iCub is capable of performing *online* learning, *i.e.*, after classification, it asks to the human teacher whether the corresponding decision is correct. If the decision is wrong (*e.g.*, in the case of an object belonging to a never-before-seen class), the human teacher

can provide feedback to the robot, which in turn updates its classification model through online or incremental learning techniques (*e.g.*, by expanding the set of known object classes). This a clear example of how a robot can learn from experience to improve its capabilities, *i.e.*, a key aspect of why embodying knowledge within robots is of crucial relevance for these tasks [21]. However, given the limited hardware and power resources of the humanoid, it is clear that retraining the whole deep learning infrastructure becomes too computationally demanding. For this reason, the visual system of iCub exploits the pre-trained ImageNet deep network [13] only for extracting a set of deep features (from one of the highest convolutional layers) and uses this feature vector to represent the object detected by iCub in the input image. As described in Fig. 1, this deep feature vector is then classified using a separate classifier, which can be retrained online in an efficient manner when feedback from the human annotator is received. In particular, in [21] this classifier is implemented using a one-versus-all scheme to combine a set of c linear classifiers, being c the number of known classes. Let us denote the pixel values of the input image (in raster-scan order) with $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^{\mathsf{d}}$ (where $\mathsf{d} = 128 \times 128 \times 3$), and the discriminant functions of the aforementioned one-versus-all linear classifiers as $f_1(\boldsymbol{x}), \ldots, f_{\mathsf{c}}(\boldsymbol{x})$. Accordingly, the predicted class $c^\star$ is determined as the class whose discriminant function for that sample is maximum:

$$c^\star = \arg\max_{k=1,\ldots,\mathsf{c}} f_k(\boldsymbol{x})\,. \tag{1}$$

The linear classifiers used for this purpose include Support Vector Machines (SVMs) and Recursive Least Square (RLS) classifiers, as both can be efficiently updated online [21]. Notably, previous work has shown that replacing the softmax layer in deep networks with a multiclass SVM can be effective also in different applications [27].

3. Adversarial Security Evaluation

We discuss here our proposal to assess the security of robot-vision systems to adversarial examples. As in previous work addressing the issue of evaluating security of machine-learning algorithms [1, 12, 5, 4, 28], our underlying idea is to evaluate the maximum recognition accuracy degradation against an increasing maximum admissible level of perturbation of the input images. This is rather different than previous work in which adversarial examples correspond to minimally-perturbed samples that are wrongly classified [25, 10, 19, 20]. As we will see in our experiments, besides providing a more complete evaluation of system security against adversarial examples, our attack strategy also highlights additional interesting insights on system security, including the identification of vulnerabilities in the feature representation (rather than in the classification algorithm itself) through the creation of adversarial examples that are *indistinguishable* from training samples of a different class.

Our approach is based on extending the work in [4] for evasion of binary classifiers to the multiclass case. To this end, we define two possible evasion settings, *i.e.*, ways of creating adversarial examples, which further differentiate our technique from previous work on the creation of minimally-perturbed adversarial examples [25, 10, 19, 20]. In particular, we consider an *error-generic* and an *error-specific* evasion setting. In the *error-generic* scenario, the attacker is interested in misleading classification, regardless of the output class predicted by the classifier for the adversarial examples; *e.g.*, for a known terrorist the goal may be to evade detection by a video surveillance system, regardless of the identity that may be erroneously associated to his/her face. Conversely, in the *error-specific* setting, the attacker still aims to mislead classification, but requiring the adversarial examples to be misclassified as a specific, target class; *e.g.*, imagine an attacker aiming to impersonate a specific user.[1]

The two settings can be formalized in terms of two distinct optimization problems, though using the same formulation for the objective function $\Omega(\boldsymbol{x})$:

$$\Omega(\boldsymbol{x}) = f_k(\boldsymbol{x}) - \max_{l \neq k} f_l(\boldsymbol{x})\,. \tag{2}$$

This function essentially represents a difference between a preselected discriminant function (associated to class k) and the competing one, *i.e.*, the one exhibiting the highest value at $\boldsymbol{x}$ among the remaining $\mathsf{c}-1$ classes (*i.e.*, all classes $\{1,\ldots,\mathsf{c}\}$ except k). We discuss below how class k is chosen in the two considered settings.

Error-generic Evasion. In this case, the optimization problem can be formulated as:

$$\min_{\boldsymbol{x}'} \quad \Omega(\boldsymbol{x}')\,, \tag{3}$$

$$\text{s.t.} \quad d(\boldsymbol{x}, \boldsymbol{x}') \leq d_{\max}\,, \tag{4}$$

$$\boldsymbol{x}_{\text{lb}} \preceq \boldsymbol{x}' \preceq \boldsymbol{x}_{\text{ub}}\,, \tag{5}$$

where $f_k(\boldsymbol{x})$ in the objective function $\Omega(\boldsymbol{x})$ (Eq. 2) denotes the discriminant function associated to the true class of the source sample $\boldsymbol{x}$, and $d(\boldsymbol{x}, \boldsymbol{x}') \leq d_{\max}$ represents a constraint on the maximum input perturbation $d_{\max}$ between $\boldsymbol{x}$ (*i.e.*, the input image) and the corresponding modified adversarial example $\boldsymbol{x}'$, given in terms of a distance in the input space. Normally, the ℓ_2 distance between pixel values is used as the function $d(\cdot,\cdot)$, but other metrics can be also

[1] In [20], the authors defined *targeted* and *indiscriminate* attacks depending on whether the attacker aims to cause *specific* or *generic* errors, similarly to our settings. Here we do not follow their naming convention, as it causes confusion with the interpretation of *targeted* and *indiscriminate* attacks introduced in previous work [1, 12, 5].

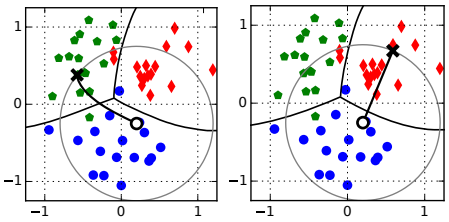

Figure 2: Error-specific (*left*) and error-generic (*right*) evasion of a multiclass SVM with the Radial Basis Function (RBF) kernel. Decision boundaries among the three classes (blue, red and green points) are shown as black solid lines. In the error-specific case, the initial (blue) sample is shifted towards the green class (selected as the target one). In the error-generic case, instead, it is shifted towards the red class, as it is the closest class to the initial sample. The ℓ_2 distance constraint is also shown as a gray circle.

adopted (*e.g.*, one may use an ℓ_1-based constraint to inject a sparse adversarial noise rather than a slight image blurring as that caused by the ℓ_2-based constraint) [8, 22]. The box constraint $\boldsymbol{x}_{\rm lb} \preceq \boldsymbol{x}' \preceq \boldsymbol{x}_{\rm ub}$ (where $\boldsymbol{u} \preceq \boldsymbol{v}$ means that each element of $\boldsymbol{u}$ has to be not greater than the corresponding element in $\boldsymbol{v}$) is optional, and can be used to bound the input values $\boldsymbol{x}$ of the adversarial examples; *e.g.*, each pixel value in images is bounded between 0 and 255. Nevertheless, the box constraint can be also used to manipulate only some pixels in the image. For example, if some pixels should not be manipulated, one can set the corresponding values of $\boldsymbol{x}_{\rm lb}$ and $\boldsymbol{x}_{\rm ub}$ equal to those of $\boldsymbol{x}$. This is of crucial importance for creating real-world adversarial examples, as it allows one to avoid manipulating pixels which do not belong to the object of interest. For instance, this may enable one to create an "unusual" sticker to be attached to an *adversarial* object, similarly to the idea exploited in [24] for the creation of wearable objects used to fool face recognition systems.

Error-specific Evasion. The problem of error-specific evasion is formulated as the error-generic evasion problem in Eqs. (3)-(5), with the only differences that: (i) the objective function is *maximized*; and (ii) f_k denotes the discriminant function associated to the targeted class, *i.e.*, the class which the adversarial example should be assigned to.

An example of the different behavior exhibited by the two attacks is given in Fig. 2. Both attacks are constructed using the simple gradient-based algorithm given as Algorithm 1. The basic idea is to update the adversarial example by following the steepest descent (or ascent) direction (depending on whether we are considering error-generic or error-specific evasion), and use a projection operator Π to keep the updated point within the feasible domain (given by the intersection of the box and the ℓ_2 constraint).

Gradient computation. One key issue of the aforementioned algorithm is the computation of the gradient of $\Omega(\boldsymbol{x})$, which involve the gradients of the discriminant function

Algorithm 1 Computation of Adversarial Examples

Input: $\boldsymbol{x}_0$: the input image; η: the step size; $r \in \{-1, +1\}$: variable set to -1 $(+1)$ for error-generic (error-specific) evasion; $\epsilon > 0$: a small number.
Output: $\boldsymbol{x}'$: the adversarial example.

$\boldsymbol{x}' \leftarrow \boldsymbol{x}_0$
repeat
  $\boldsymbol{x} \leftarrow \boldsymbol{x}'$, and $\boldsymbol{x}' \leftarrow \Pi\left(\boldsymbol{x} + r\eta\nabla\Omega(\boldsymbol{x})\right)$
until $|\Omega(\boldsymbol{x}') - \Omega(\boldsymbol{x})| \leq \epsilon$
return $\boldsymbol{x}'$

$f_i(\boldsymbol{x})$ for $i \in 1, \ldots, \mathsf{c}$. It is not difficult to see that this can be computed using the chain rule to decouple the gradient of the discriminant function of the classifier trained on the deep feature space and the gradient of the deep network used for feature extraction, as $\nabla f_i(\boldsymbol{x}) = \frac{\partial f_i(\boldsymbol{z})}{\partial \boldsymbol{z}} \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$, being $\boldsymbol{z} \in \mathbb{R}^{\mathsf{m}}$ the set of deep features. In our case study, these are the $\mathsf{m} = 4,096$ values extracted from layer fc7 (see Fig. 1). Notably, the gradient of the deep network $\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}$ is readily available through automatic differentiation, as also highlighted in previous work [25, 10, 19, 20], whereas the availability of the gradient $\frac{\partial f_i(\boldsymbol{z})}{\partial \boldsymbol{z}}$ depends on whether the chosen classifier is differentiable or not. Several of the most used classifiers are differentiable, including, *e.g.*, SVMs with differentiable kernels (we refer the reader to [4] for further details). Nevertheless, if the classifier is not differentiable (*e.g.*, like in the case of decision trees), one may use a surrogate differentiable classifier to approximate it, as also suggested in [4, 8, 22].

4. Classifier Security to Adversarial Examples

If the evasion algorithm drives the adversarial examples deeply into regions populated by known training classes (as shown in Fig. 2), there is no much one can do to correctly identify them from the rest of the data by only re-training or modifying the classifier, *i.e.*, modifying the shape of the decision boundaries in the feature space. We propose to consider this problem as an intrinsic *vulnerability* of the *feature representation*: if the feature vector of an adversarial example becomes *indistinguishable* from those of the training samples of a different class, it can only be detected by using a different feature representation (*i.e.*, in the case of iCub, this would require at least re-training the underlying deep network responsible for deep feature extraction).[2] However, this is not always the case, especially in high-dimensional spaces, or if classes are separated with a sufficiently high *margin*. In this case, as depicted in Fig. 3, there may be very large regions of the feature space which are only scarcely

[2]Here, we only refer to the classifier trained on top of the deep feature representation as the *classification algorithm*. This definition excludes the pre-trained deep network used for feature extraction in iCub, as it is not re-trained online.

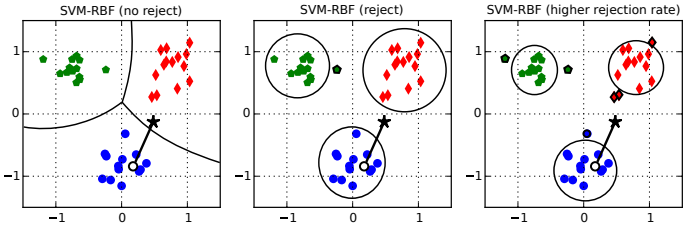

Figure 3: Conceptual representation of our idea behind improving iCub security to adversarial examples, using multiclass SVMs with RBF kernels (SVM-RBF), without reject option (no defense, *left*), with reject option (*middle*), and with modified thresholds to increase the rejection rate (*right*). Rejected samples are highlighted with black contours. The adversarial example (black star) is misclassified as a red sample by SVM-RBF (left plot), while SVM-RBF with reject option correctly identifies it as an adversarial example (middle plot). Rejection thresholds can be modified to increase classifier security (right plot), though at the expense of misclassifying more legitimate (*i.e.*, non-manipulated) samples.

populated by data, although being associated (potentially also with high confidence) to known classes by the learning algorithm. Accordingly, adversarial examples may quite reasonably end up in such regions while also successfully fooling detection. These samples are often referred to as *blind-spot* evasion samples, as they are capable of misleading classification, but in regions of the space which are far from the rest of the training data [12, 25]. Conversely to the case of *indistinguishable* adversarial examples, *blind-spot* adversarial examples can be detected by only modifying the classifier (*i.e.*, without re-training the underlying deep network used by iCub). Accordingly, we propose to consider the problem of blind-spot adversarial examples as an intrinsic vulnerability of the classification algorithm. Different approaches have been proposed based on modifying the classifier, ranging from 1.5-class classification (based on the combination of anomaly detectors and two-class classifiers) [3] to open-set recognition techniques [23, 2].

We propose here a more direct approach, based on the same idea underlying the notion of classification with a reject option, and leveraging some concepts from open-set recognition. In particular, we consider SVMs with RBF kernels to implement the multiclass classifier in our case study, as these SVMs belong to the so-called class of Compact Abating Probability (CAP) models [23] (*i.e.*, classifiers whose discriminant function decreases while getting farther from the training data). Then, by applying a simple rejection mechanism on their discriminant function, we can identify samples which are far enough from the rest of the training data, *i.e.*, blind-spot adversarial examples. Our idea is thus to modify the decision rule in Eq. (1) as:

$$c^\star = \arg\max_{k=1,\ldots,\mathsf{c}} f_k(\boldsymbol{x})\,, \text{ only if } f_{c^\star}(\boldsymbol{x}) > 0\,, \qquad (6)$$

otherwise classify $\boldsymbol{x}$ as an adversarial example (*i.e.*, a novel class). In practice this means that, if no classifier assigns the sample to an existing class (*i.e.*, no value of f is positive), then we simply categorize it as an adversarial example. In our specific case study, iCub may reject classification and ask the human annotator to label the example correctly. Notably, the threshold of each discriminant function (*i.e.*, the biases of the one-versus-all SVMs) can be adjusted to tune the trade-off between the rejection rate of adversarial examples and the fraction of incorrectly-rejected samples (which are not adversarially manipulated), as shown in Fig. 3.

5. Experimental Analysis

In this section we report the results of the security evaluation performed on the iCub system (see Sect. 2) along with few adversarial examples to show how the proposed evasion algorithm can be exploited to create real-world attack samples. We then provide a conceptual representation and an empirical analysis to explain why neural networks are easily fooled and how our defense mechanism can improve their security in this context.

Experimental Setup. Our analysis has been performed using the *iCubWorld28* dataset [21], consisting of 28 different classes which include 7 different objects (cup, plate, *etc.*) of 4 different kinds each (*e.g.*, cup1, cup2, *etc.*), as shown in Fig. 4. Each object was shown to iCub which automatically detected it and cropped the corresponding object image. Four acquisition sessions were performed in four different days, ending up with approximately $20,000$ images for training and test sets. As shown in [21], it is very difficult for iCub to be able to distinguish such slight category distinctions, like different kinds of cups. For this reason, we also consider here a reduced dataset, *iCubWorld7*, consisting only of 7 different objects, each of a different kind. The selected objects are highlighted in red in Fig. 4.

We implement the classification algorithm using three different multiclass SVM versions, all based on a one-versus-all scheme: a linear SVM (denoted with SVM in the following); an SVM with the RBF kernel (SVM-RBF); and an SVM with the RBF kernel implementing our defense mechanism based on rejection of adversarial examples (SVM-adv, Sect. 4). The regularization parameter $C \in \{10^{-3}, \ldots, 10^{3}\}$ and the RBF kernel parameter $\gamma \in \{10^{-6}, \ldots, 10^{-2}\}$ have been set equal for all one-versus-all SVMs in each multiclass classifier, by maximizing recogni-

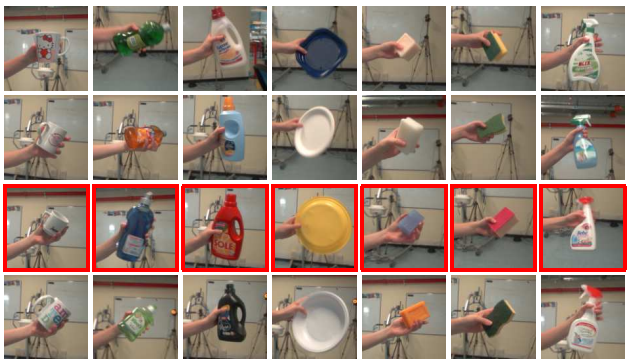

Figure 4: Example images (one per class) from the *iCubWorld28* dataset, and subset of classes used in the *iCubWorld7* dataset (highlighted in red).

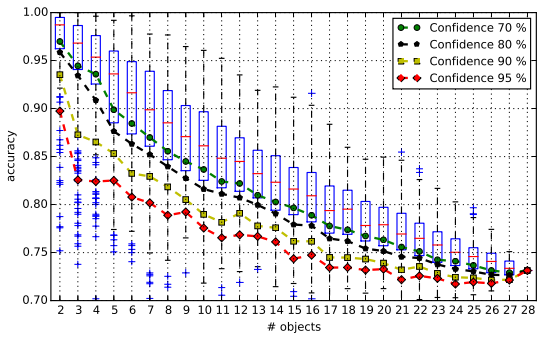

Figure 5: Box plots of the recognition accuracies measured for linear SVM predictors trained on random subsets from 2 to 28 objects (whiskers with maximum 1.5 interquartile range). Dotted super-imposed curves represent the minimum accuracy guaranteed within a fixed confidence level.

tion accuracy through 3-fold cross validation.

Baseline Performance. In Fig. 5 we report a box plot showing the empirical probability distributions of the accuracy achieved by the SVM classifier on increasingly larger object identification tasks, as suggested in [21]. To this end, we randomly select 300 subsets of increasing size from the *iCubWorld28* dataset (day4 acquisitions), and then train and test the classifier on each subset. The achieved accuracy is considered an observation for estimating the empirical distributions. The minimum accuracy value for which the fraction of observations in the estimated distribution was higher than a specific confidence threshold is indicated as a dotted line. Notably the reported performances for the linear SVM are almost identical to those reported in [21], where a different algorithm is used. Similar performances (omitted for brevity) are obtained using SVM-RBF.

Security Evaluation against Adversarial Examples. We now investigate the security of iCub in the presence of adversarial examples. In this experiment, we consider the first 100 examples per class for both the *iCubWorld28* and *iCubWorld7* datasets, ending up with training and test sets consisting of $2,800$ and 700 samples, respectively. The recognition accuracy against an increasing maximum admissible ℓ_2 perturbation (*i.e.*, $d_{\max}$ value) is reported in Fig. 6 for both error-specific (top row plots) and error-generic (bottom row plots) attack scenarios. For error-specific evasion, we average our results not only on different training-test set splits, but also by considering a different target class in each repetition. While SVM and SVM-RBF show a comparable decrease of accuracy at increasing $d_{\max}$, SVM-adv is able to strongly improve the security in most of the cases (as the corresponding curve decreases more gracefully). Notably, the performance of SVM-adv even increases for low values of $d_{\max}$. A plausible reason is that, even if all testing images are only slightly modified in input space, they immediately become blind-spot adversarial examples, ending up in a region which is far from the rest of the data. As the input perturbation increases, such samples are gradually drifted inside a different class, becoming *indistinguishable* from the samples of such class.

To further improve the security of iCub to adversarial examples, we set the rejection threshold of SVM-adv to a more conservative value, increasing the false negative rate for each base classifier of 5% (estimated on a validation set). This results in a significant security improvement, as shown in the rightmost plots in Fig. 6. However, as expected, this comes at the expense of misclassifying more legitimate (*i.e.* non-manipulated) samples.

Real-world Adversarial Examples. In Fig. 7 we report few adversarial examples generated using an error-specific evasion attack on the *iCubWorld28* data. Notably, the adversarial perturbation required to evade the system can be barely perceived by human eyes. As an important real-world application of the proposed attack algorithm, in the bottom right plots of Fig. 7, we report an adversarial example generated by manipulating only a subset of the image pixels, corresponding to the label of the detergent. In this case, the perturbation becomes easier to spot for a human, but localizing the noise in a region of interest allows the attacker to construct a practical, real-world adversarial object, by simply attaching an "adversarial" sticker to the original object before showing it to the iCub humanoid robot.

Why are Deep Nets Fooled? Our analysis shows that also the iCub vision system can be fooled by adversarial examples, even by only adding a slightly-noticeable noise to the input image. To better understand the root causes of this phenomenon, we now provide an empirical analysis of the sensitivity of the feature mapping induced by the ImageNet deep network used by iCub, by comparing the ℓ_2 distance corresponding to random and adversarial perturbations in the input space, with the one measured in the deep feature space. To this end, we randomly perturb each training image such that the ℓ_2 distance between the ini-

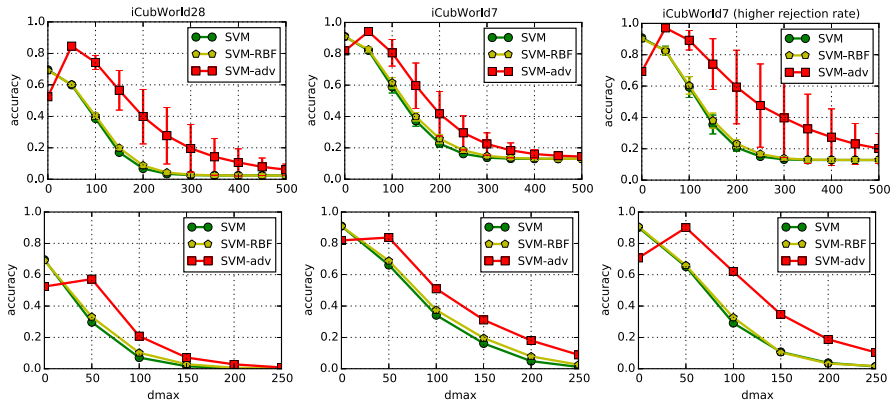

Figure 6: Recognition accuracy of iCub (using the three different classifiers SVM, SVM-RBF, and SVM-adv) against an increasing maximum admissible ℓ_2 input perturbation $d_{\max}$, for *iCubWorld28* (*left column*) and *iCubWorld7* (*middle* and *right* columns), using error-specific (*top row*), and error-generic (*bottom row*) adversarial examples. Baseline accuracies (in the absence of perturbation) are reported at $d_{\max} = 0$.

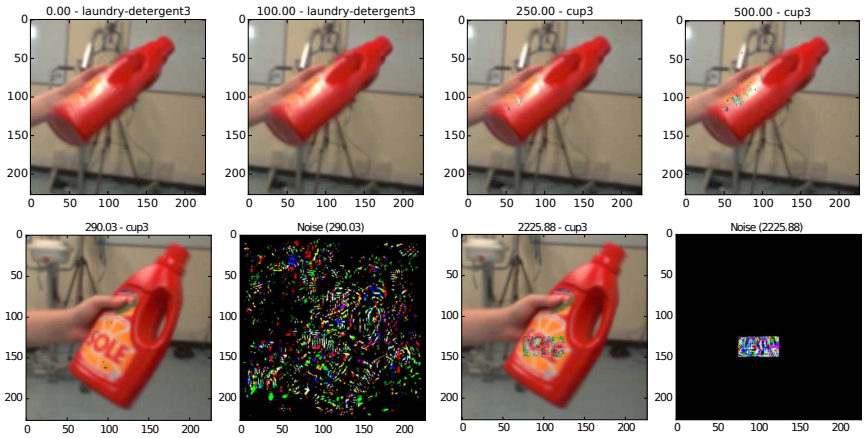

Figure 7: Plots in the top row show an adversarial example from class *laundry-detergent3*, modified to be misclassified as *cup3*, using an error-specific evasion attack, for increasing levels of input perturbation (reported in the title of the plots). Plots in the bottom row show the minimally-perturbed adversarial example that evades detection (*i.e.*, the sample that evades detection with minimum $d_{\max}$), along with the corresponding noise applied to the initial image (amplified to make it clearly visible), for the case in which all pixels can be manipulated (first and second plot), and for the case in which modifications are constrained to the label of the detergent (*i.e.*, simulating a sticker that can be applied to the real-world *adversarial* object).

tial and the perturbed image in the input space equals 10. We then measure the ℓ_2 distance between the deep feature vectors corresponding to the same images. For randomly-perturbed images, the average distance in deep space (along with its standard deviation) is 0.022 ± 0.002, while for the adversarially-perturbed images, it is 2.386 ± 0.386. This means that random perturbations in the input space only result in a very small shift in the deep space, while even light alterations of an image along the adversarial direction cause a large shift in deep space, which in turn highlights a significant *instability* of the deep feature space mapping induced by the ImageNet network. In other words, this means that images in the input space are very close to the decision boundary along the adversarial (gradient) direction, as conceptually represented in Fig. 8. Note that this is a general issue for deep networks, not only specific to ImageNet [26, 9, 25, 10, 19, 20].

It should be thus clear that even a well-crafted modification of the last layers of the network, as in our proposed defense mechanism SVM-adv, can only mitigate this vulnerability. Indeed, it remains intimately related to the stability of the deep feature space mapping, which can be only addressed by imposing specific constraints while training the deep neural network; *e.g.*, by imposing that small shifts in the input space correspond to small changes in the deep space, as recently proposed in [31]. Another possible countermeasure to improve stability of such mapping is to enforce classification of samples within a minimum *margin*, by modifying the neurons' activation functions and, potentially, considering a different regularizer for the objective

function optimized by the deep network. In this respect, it would be interesting to investigate more in detail the intimate connections between robustness to adversarial input noise and regularization, as highlighted in [29, 22].

6. Related Work

Previous work has investigated the problem of adversarial examples in deep networks [25, 10, 19, 20, 26, 9], focusing however on minimally-perturbed adversarial examples, *i.e.*, examples that simply lie inside the decision region of a known class, even if they remain far from the corresponding training examples; on the contrary, our approach is based on creating maximally-perturbed (indistinguishable) adversarial examples (misclassified with high confidence). Different techniques aimed at improving the security of deep networks have also been proposed. Some of them attempt to reduce classifier vulnerability by directly detecting and rejecting adversarial examples [2, 15]. The technique of [2] is based on open-set recognition: it rejects samples whose distance from the centroids of all the classes exceeds a given threshold. However, it has not been evaluated using adversarial examples carefully generated to evade the classifier. In [15], adversarial examples are detected using the output of the first convolutional layers. A different approach has been proposed in [31]: it aims at improving the stability of the deep feature space mapping by retraining the network using an objective function that penalizes examples (images) that are close in input space but lie far in deep feature space. This approach has however been investigated only against small image distortions.

7. Conclusions and Future Work

Deep learning has shown groundbreaking performance in several real-world application domains, encompassing areas like computer vision, speech recognition and language processing, among others. Despite its impressive performances, recent work has shown how deep neural networks can be fooled by well-crafted adversarial examples affected by a barely-perceivable adversarial noise. In this work, we have developed a novel algorithm for the generation of adversarial examples which enables a more complete evaluation of the security of a learning algorithm, and apply it to investigate the security of the robot-vision system of the iCub humanoid. Even if we do not restrict ourselves to the manipulation of pixels belonging to the object of interest in the image (which could lead one to more easily generate the corresponding real-world adversarial object, *e.g.*, by mean of the application of specific stickers to objects), we have shown how our algorithm enables this additional possibility. Notably, even if we have not constructed any real-world adversarial object during our experiments, recent work has shown that the artifacts introduced by printing images and re-acquiring them through a camera are irrelevant, and do not eliminate the problem of the existence of adversarial examples [14]. Similarly, another work has shown how to evade face recognition systems based on deep learning by using adversarial glasses and other accessories [24]. These recent evidences clearly give a much higher practical relevance to the problem of adversarial examples.

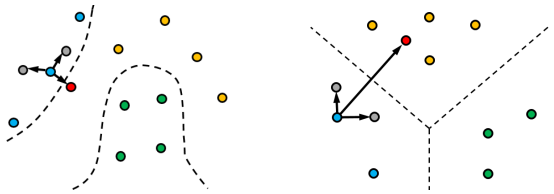

Figure 8: Conceptual representation of the vulnerability of the deep feature space mapping. The left and right plots respectively represent images in input space and the corresponding deep feature vectors. Randomly-perturbed versions of an input image are shown as gray points, while the adversarially-perturbed image is shown as a red point. Despite these points are at the same distance from the input image in the input space, the adversarial example is much farther in the deep feature space. This also means that images are very close to the decision boundary in input space, although in an adversarial direction that is difficult to guess at random due to the high dimensionality of the input space.

We have demonstrated and quantified the vulnerability of iCub to the presence of adversarial manipulations of the input images, and suggested a simple countermeasure to mitigate the threat posed by such an issue. We have additionally shown that, while blind-spot adversarial examples can be detected using our defense mechanism, to further improve the security of iCub against *indistinguishable* adversarial examples, re-training the classification algorithm on top of a pre-trained deep neural network is not sufficient. To this end, different strategies to enforce the deep network to learn a more stable deep feature representation (in which small perturbations to the input data correspond to small perturbations in the deep feature space) should also be adopted, like the one proposed in [31].

Other interesting research directions for this work include evaluating security of robot-vision systems against other threats, including the threat of data poisoning [12, 6, 5], in which a malicious human annotator may provide few wrong labels to the humanoid to completely mislead its learning process and enforce it to misclassify as many objects as possible. In general, a comprehensive, standardized framework for evaluating the security of such systems while providing also more formal verification procedures is still lacking, and we believe that this constitutes a fundamental requirement for the complete transition of deep-learning-based systems in safety-critical applications, like robots performing life-critical tasks and self-driving cars.

References

[1] M. Barreno, B. Nelson, A. Joseph, and J. Tygar. The security of machine learning. *Mach. Learn.*, 81:121–148, 2010. 3

[2] A. Bendale and T. E. Boult. Towards open set deep networks. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1563–1572, 2016. 5, 8

[3] B. Biggio, I. Corona, Z.-M. He, P. P. K. Chan, G. Giacinto, D. S. Yeung, and F. Roli. One-and-a-half-class multiple classifier systems for secure learning against evasion attacks at test time. In F. Schwenker, F. Roli, and J. Kittler, editors, *Multiple Classifier Systems*, volume 9132 of *Lecture Notes in Computer Science*, pages 168–180. Springer International Publishing, 2015. 5

[4] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In H. Blockeel, et al., editors, *ECML PKDD, Part III*, vol. 8190 of *LNCS*, pages 387–402. Springer Berlin Heidelberg, 2013. 2, 3, 4

[5] B. Biggio, G. Fumera, and F. Roli. Security evaluation of pattern classifiers under attack. *IEEE Trans. Knowl. and Data Eng.*, 26(4):984–996, April 2014. 3, 8

[6] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. In J. Langford and J. Pineau, editors, *29th Int'l Conf. on Machine Learning*, pages 1807–1814. Omnipress, 2012. 8

[7] N. Cristianini. Intelligence reinvented. *New Scientist*, 232(3097):37–41, 2016. 1

[8] A. Demontis, P. Russu, B. Biggio, G. Fumera, and F. Roli. On security and sparsity of linear classifiers for adversarial settings. In A. Robles-Kelly et al., editors, *Joint IAPR Int'l Workshop on Structural, Syntactic, and Statistical Patt. Rec.*, vol. 10029 of *LNCS*, pages 322–332, Cham, 2016. Springer International Publishing. 3, 4

[9] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017. 7, 8

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conf. on Learning Representations*, 2015. 1, 2, 3, 4, 7, 8

[11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. 1

[12] L. Huang, A. D. Joseph, B. Nelson, B. Rubinstein, and J. D. Tygar. Adversarial machine learning. In *4th ACM Workshop on Artificial Intelligence and Security (AISec 2011)*, pages 43–57, Chicago, IL, USA, 2011. 3, 5, 8

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. 2, 3

[14] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv:1607.02533*, 2016. 8

[15] X. Li and F. Li. Adversarial examples detection in deep networks with convolutional filter statistics. *CoRR*, abs/1612.07767, 2016. 8

[16] Y. Luo, X. Boix, G. Roig, T. Poggio, and Q. Zhao. Foveation-based mechanisms alleviate adversarial examples. *arXiv preprint arXiv:1511.06292*, 2015. 1

[17] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *IEEE Conf. Computer Vision and Patt. Rec.*, pages 5188–5196, 2015. 1

[18] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The iCub humanoid robot: an open platform for research in embodied cognition. In *Proc. of the 8th workshop on performance metrics for intelligent systems*, pages 50–56. ACM, 2008. 1, 2

[19] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016. 1, 2, 3, 4, 7, 8

[20] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *1st IEEE European Symp. Sec. & Privacy*, pages 372–387. IEEE, 2016. 1, 2, 3, 4, 7, 8

[21] G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, L. Natale, and I. dei Sistemi. Teaching iCub to recognize objects using deep convolutional neural networks. In *MLIS@ ICML*, pages 21–25, 2015. 1, 2, 3, 5, 6

[22] P. Russu, A. Demontis, B. Biggio, G. Fumera, and F. Roli. Secure kernel machines against evasion attacks. In *9th AISec*, pages 59–69, New York, NY, USA, 2016. ACM. 3, 4, 8

[23] W. Scheirer, L. Jain, and T. Boult. Probability models for open set recognition. *IEEE Trans. Patt. An. Mach. Intell.*, 36(11):2317–2324, 2014. 5

[24] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Conf. Computer and Comm. Sec.*, pages 1528–1540. ACM, 2016. 1, 4, 8

[25] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1, 2, 3, 4, 5, 7, 8

[26] T. Tanay and L. Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv:1608.07690*, 2016. 7, 8

[27] Y. Tang. Deep learning using support vector machines. In *ICML Workshop on Representational Learning*, volume arXiv:1306.0239, Atlanta, USA, 2013. 3

[28] N. Šrndic and P. Laskov. Practical evasion of a learning-based classifier: A case study. In *IEEE Symp. Sec. & Privacy*, pages 197–211, Washington, DC, USA, 2014. IEEE CS. 3

[29] H. Xu, C. Caramanis, and S. Mannor. Robustness and regularization of support vector machines. *JMLR*, 10:1485–1510, July 2009. 8

[30] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014. 1

[31] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 4480–4488, 2016. 8