# An Investigation of Bounded Misclassification for Operational Security of Deep Neural Networks

Sailik Sengupta [1] Andrew Dudley [1] Tathagata Chakraborti [1] Subbarao Kambhampati [1]

## Abstract

Deep Neural Networks are known to be prone to incomprehensible mistakes on the inputs they do misclassify. However, from the perspective of an end-to-end system built on top of a classifier, there may be additional layers of decision making that may actually be immune to particular kinds of misclassification. For example, if a drone ends up misclassifying a yellow school bus as something similar, such as a cab instead of, say, an enemy tank, then the underlying decision problem of ignoring this object as a possible target remains the same, and hence unaffected. In this brief abstract, we discuss this notion of robustness called "bounded misclassification" that is domain-specific and operational, and is specifically predicated on the overall functionalities of a particular application.

## 1. Introduction

Although Deep Neural Networks (DNNs) are the state-of-the-art in classification tasks, they are known to make mistakes that are curiously quite incomprehensible to human intuitions. Recent work has shown that the problem is worse than accidental misclassification – DNNs are easily fooled with carefully crafted modifications to the input (Szegedy et al., 2013) which may be imperceptible to humans (Moosavi-Dezfooli et al., 2016; Papernot et al., 2016a) or may be actual objects in the world (Athalye et al., 2017). If used for safety critical systems such as classifying digits on a bank cheque or recognition of objects on the road from camera feed of an automated car, the decisions based on these results can be catastrophic. For example, classifying '1' as '9' can lead to an adversary withdrawing $900 instead of the original $100 mentioned on the cheque or misclassifying a 'Labrador retriever' walking on the road walking as a 'beer bottle' might lead to different decisions based on which the autonomous car might not stop.

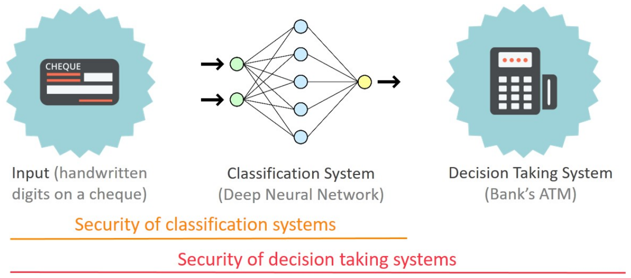
CHEQUE
Input (handwritten digits on a cheque)
Classification System (Deep Neural Network)
Decision Taking System (Bank's ATM)
Security of classification systems
Security of decision taking systems

*Figure 1.* An ATM machine that decides now much money to pay based on the recognition of handwritten numbers on a cheque by a classifier. In this work, we talk about taking secure decisions as opposed to making a classification systems robust.

Although the research community has made progress on the development of defense mechanisms against such attacks, the security of a classifier is still measured in terms of misclassification rate on a set of test samples. This metric is often misleading to the *operational security* of a classifier, which can be interpreted as the degree of misclassification (measured as a domain-specific "semantic distance" between the various class labels) as opposed to the misclassification rate. In that sense, this aims to secure classifiers in a way such that the effect of the misclassification at the classification level does not adversely affect the meta-classification at the decision level. The autonomous agent can thus be seen to have a two layer architecture as seen in Figure 1 – the first layer representing the classification system and the second layer representing the decision system that chooses how to act based on the output of the classification system. Existing works seek to address security issues at the classification level only.

In this paper, we thus posit that when viewed from the system perspective, the failure of deep networks towards adversarial inputs may not be as big an issue if the system as a whole is designed so that (1) the decision making modules are cognizant of the misclassification risks; and (2) the classification modules are trained with the understanding that (1) is in place, for example, by using semantic dis-

[1]Computer Science, Arizona State University, USA.
This paper was presented at AAAI-18 Workshop on Engineering Secure and Dependable Machine Learning Systems. Correspondence to: Sailik Sengupta <sailiks@asu.edu>.

tances between labels. This has two implications from the perspective of system design – (1) when the application is consumed at the user end at the level of images, then adversaries will need to devise attacks that are more easily detectable (by having to cross semantic boundaries) to the human eye; and (2) at the decision level, as mentioned before, the system behavior / performance remains largely unaffected. The aim of this paper is thus to –

- make a case for the use of *degree of misclassification* as a metric which can be used to redefine the notion operational security of an autonomous agent taking decisions based on the output of deep networks.
- explore distance measures between the various output labels that represent the degree of misclassification obtained by leveraging domain level semantics.
- investigate the applicability of cost-based loss functions for the purposes of operational security so that on misclassification, for legitimate or adversarial inputs, labels the input as a class semantically closer to the correct class label as opposed to an arbitrary class.

When a network is trained on data from a particular distribution, some areas in the higher dimensional space end up with high biases. If the distribution of the test data is different (i.e. noise, random or adversarial), the network misclassifies in these vast swathes of unexplored spaces with high bias. Adversarial attacks exploit this weakness. Adversarial training thus seeks to employ examples with adversarial noise in the training phase itself so that the classifier reduces bias in these uninformed regions of the high dimensional space. In that sense, our method could be seen as a form of adversarial training that handles "noise" in general by allowing the decision boundaries to be more expansive and hopefully contiguous to those of "similar" classes.

## 2. Bounded Misclassification

Consider a dataset with $n$ features and $m$ labels. A labeled data is represented as $\{(x_1, \dots x_n), y\}$ where $x_i$ is the $i^{th}$ feature and $y \in Y = \{Y_1, \dots, Y_m\}$ is the correct class label. We define a similarity measure between two class labels as a function $f : Y \times Y \to [0, 1]$ –

$$s(Y_i, Y_j) = \begin{cases} x & \text{if } i \neq j \\ 1 & \text{otherwise} \end{cases}$$

$$\mathbb{I}^{\delta}(a) = \begin{cases} 1 & \text{if } a \geq \delta \\ 0 & \text{otherwise} \end{cases}$$

$s(Y_i, Y_j)$ is close to 1 if classes $Y_i$ and $Y_j$ are similar, i.e. the decision taken by the high level system remains unaffected whether the classifier outputs $Y_i$ or $Y_j$. For example, if our decision function decides to 'stop' if it sees a $Y_1$ ='dog' or a $Y_2$ ='cat' and 'move forward' if it sees a

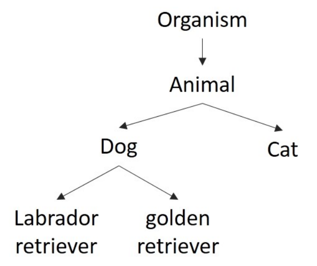

*Figure 2.* A sub-graph from WordNET (Miller & Fellbaum, 1998)

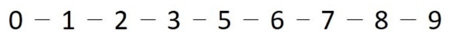

*Figure 3.* An example graph $G$ constructed from MNIST labels.

$Y_3$ ='beer bottle', then $s(Y_1, Y_2) > s(Y_1, Y_3), s(Y_2, Y_3)$. The distances between the classes are symmetric and are represented as an $m \times m$ matrix $D$ where $D_{ij} = d(Y_i, Y_j)$.

*Bounded misclassification* with bound $\delta$ requires that a given input $\vec{x}$ with correct class label $y$ is classified as $Y_i$ only if $s(Y_i, y) \geq \delta$. When $\delta = 1$, the classifier does not have any guidance for similar labels to fall back on. Ideally, if $\delta$ is closer to 1, we expect that an adversary would have to make a perturbation of high magnitude in order to force the classifier to classify the perturbed image to a desired class which is less similar to the correct class label.

### 2.1. Distance between Class Labels

One can, of course, ask how one can obtain these distances between class labels. A possible approach would be, given the decision context in which this classification result will be used, to crowdsource additional batches of labeling (for simple tasks like is 'dog' similar to a 'cat' if you were to decide whether you would stop your car or run them over) or, in the worst case, ask the domain experts for more complex tasks. This can prove to be too expensive and unscalable.

Fortunately, we note that in many cases, and certainly for taking decisions based on classification on ImageNET kind of data, these distances are semantic and hence inherent in nature. Thus, the distance metrics can be extracted out of existing ontologies or knowledge bases (e.g. WordNET) that pertain to that specific domain. In the following we discuss this caveat in the context of two popular datasets used often as testbeds for classification tasks.

#### IMAGENET

ImageNET is a popular image database built out of many everyday objects as class labels, which belong to the well-known lexical database WordNET (Miller & Fellbaum, 1998). Words in WordNET form a hierarchical graph structure – the nodes of this network represent English words (not necessarily nouns) and the edges represent conceptual-semantics and lexical-relations between the nodes.

An example of such a network is given in Figure 2. Notice that in this network, all breeds of dogs are situated under the label 'dog', which in turn is under the label 'animals' that is also the parent of 'cat'. The NLTK API for Word-NET measures $path\_similartiy(u, v)$ for words $u$ and $v$ that is equal to $0$ if there is no similarity and $1$ if $u = v$. Thus, in the case of ImageNET, we easily use the Word-NET graph and use this similarity measure for the class labels $Y_1$ and $Y_2$. For example, a 'golden retriever' is more similar to a 'Labrador retiever' ($0.333$) than a 'cat' ($0.111$), which in turn is more similar than a 'beer bottle' ($0.0625$).

MNIST

Now we consider a particular scenario for the classification of MNIST dataset used in the context of classifying hand-written digits on a bank cheque. We construct an underlying graph for the labels of this dataset (see Figure 3) and define the following similarity measure over class labels –

$$\begin{aligned} s(Y_1, Y_2) &= 1 - \frac{shortest\_path(Y_1, Y_2)}{\max_{Y_1,Y_2} shortest\_path(Y_1, Y_2)} \\ &= 1 - \frac{shortest\_path(Y_1, Y_2)}{shortest\_path(Y_1 = 0, Y_2 = 9)} \\ &= 1 - \frac{shortest\_path(Y_1, Y_2)}{8} \end{aligned} \quad (1)$$

Consider each handwritten digit in the amount value on the cheque is being parsed independently by the computer vision system (essentially a DNN classifier). Say an adversary had put some extra pen dots on some of the digits, that could be ignored by a human cashier, but makes the DNN classifier classify it incorrectly. Since we want the system to make a misclassification (if it is going to misclassify anyway) nearer to the actual label, it is better to classify the image of '1' and a '2' rather than a '9', thus incurring less loss of money to the bank in the case of fraud. By the similarity measure, we have $s(1, 2) = 0.875$ whereas, $s(1, 9) = 0.125$. It is interesting to note that, from a visual perspective, '1' looks somewhat closer to '7', where as '6' looks closer to '8'. In the graph we construct, this kind of a similarity is not considered because of the *decisions* we plan to take based on the classification output.

### 2.2. Weighted Loss Functions (WLF)

We will derive inspiration from Weighted Loss Functions (WLFs) used for cost-based misclassification in the machine learning literature (Duda et al., 1973; Pazzani et al., 1994) to penalize the loss incurred in classifying an input with class label $Y_i$ to a $Y_j$ by a weight $\lambda_{Y_i,Y_j}$. Classifiers in general use the $0 - 1$ loss functions for the sake of maximizing accuracy. Thus, we have $\lambda_{Y_i,Y_i} = 1$ and $\lambda_{Y_i,Y_k} = 0 \; \forall \; k \in \{1, \dots, m\} \setminus i$.

### 2.3. Using WLF for Bounded Misclassification

The main idea here is to nudge a misclassification towards a label that is more similar (with respect to the semantics of the underlying decision task) to the correct class can be done using non-uniform weighted loss functions. The proposed approach thus focuses on well-defined semantics for weighting the loss functions as opposed to the notion of merely using soft labels as is the case with generic defensive distillation (Papernot et al., 2016b).

Thus, if a sample with true label $Y_i$ is classified to $Y_j$, the value of $\lambda_{Y_i,Y_j}$ should be higher as $s(Y_i, Y_j)$ approaches one (i.e. is more similar). Also, since we want to ensure misclassification is bounded to classes at a similarity of $\leq \delta$, we will ensure that $\lambda_{Y_i,Y_j}$ when $s(Y_i, Y_j) < \delta$.

We now discuss how we can modify the most commonly used loss function known as the cross entropy (log) loss function for Deep Neural Networks (DNNs) to a non-uniform weighted loss function. Let $j$ represent the label indices and $o_j$ represents the probability value (of the soft-max layer) that the neural network outputs for the class label $Y_j$ given input $x$. We propose the following loss –

$$L(x) = -\sum_{j=1}^{m} \mathbb{I}^\delta(s(Y_k, Y_j)) \log o_j$$

where $k$ is the correct class label and the function $\mathbb{I}^\delta(a) = 1$ if $a \geq \delta$ and $0$ otherwise. Notice that apart from the non-zero term that the traditional cross-entropy function has (at $Y_k$), this equation has other non-zero terms for similar classes, specifically when $j \neq k$ and $s(Y_k, Y_j) < \delta$. The network will try to increase the value of $o_j$ in proportion to the similarity between the classes. For dissimilar classes, $\mathbb{I}^\delta(\cdot) = 0$ so that the loss function value is not affected. Thus, the classifier can let $o_j = 0$ for these classes.

## 3. Preliminary Results

We now present some preliminary results on whether misclassification error can be diffused across neighboring class labels for the purposes of operational security. Figure 4 shows the confusion matrices for classification of digits on MNIST. Here the y-axis contains the true labels (0-9) while the x-axis contains the predicted label – the strength of the red color of cell (i,j) indicates the frequency with which the digit $i$ was recognized in place of $j$. We trained the network in the following three conditions –

C1 We used the usual cross entropy (log) loss function. Notice that the frequency of misclassification is quite spread out. Also, visual similarities, especially in the case of handwritten digits, result in high frequencies

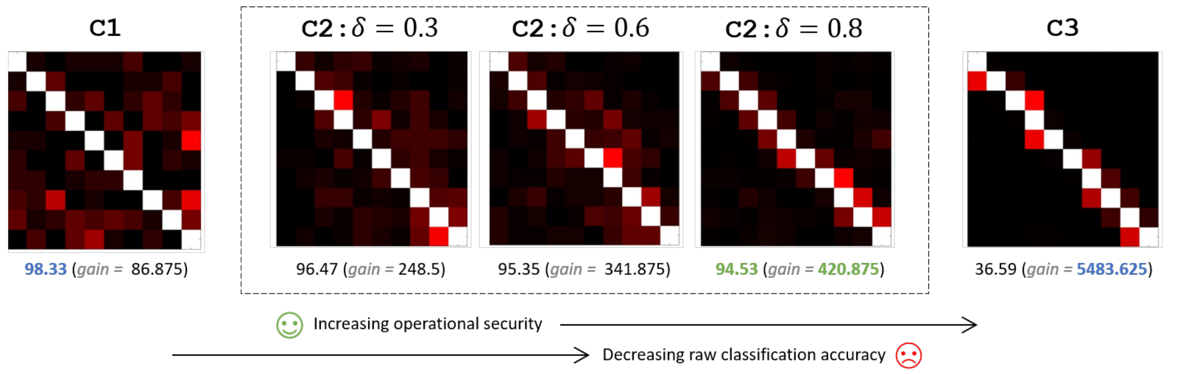

*Figure 4.* Distribution of mislabeled classes in MNIST in the three training conditions C1, C2 and C3. As expected, in C2 and C3, instances of misclassification huddle around the diagonal (prediction = target) while the classification accuracy takes a hit.

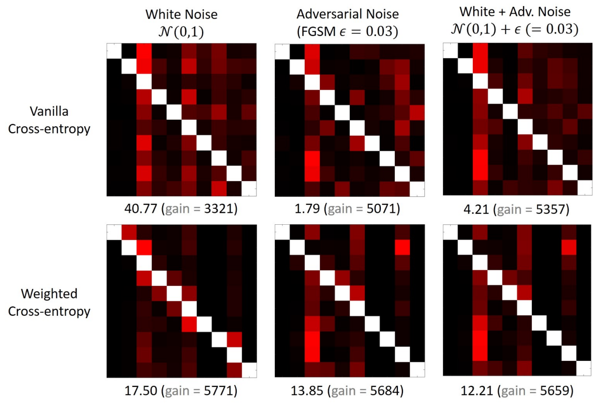

*Figure 5.* Performance in presence of random / adversarial noise.

for '3' being mistaken as '7' and '6' as '0'.

C2 We trained the DNN with the non-uniform weighted loss function (Section 2.3) using the similarity function (precomputed) in Eqn. 1 induced by the semantic graph in Figure 3. In this case, the frequency of classifying a digit in class $Y_i$ decreases as one moves away for the diagonal since the similarity between the labels decrease. Further as the bound is increased ($\delta = 0.3, 0.6, 0.8$) the spread of misclassified samples gets more concentrated along the diagonal, but the accuracy also takes a marginal hit.

C3 We performed the training in two batches – in Batch 1, we changed the true labels of an input from a one-hot encoding to a two/three-hot encoding by letting the (immediate) neighboring classes also be a correct label. Thus, for an image in class $Y_1$, the one-hot encoding $[0, 1, 0, 0 \ldots, 0]$ became $[1, 1, 1, 0 \ldots, 0]$. In Batch 2, we used the usual one-hot encodings to retrain the network. In this case, we notice that the accuracy of classification to the single correct class reduces significantly but the misclassification to non-neighboring classes is *significantly* reduced as well.

Next, we investigate the performance of the networks when exposed to either random or adversarial noise in Figure 5.

For random noise, WLFs are able to provide bounds on misclassification (see distribution of red pixels on the left column of Figure 5), but are unable to compete at the level of the vanilla network with respect to the classification accuracy to the correct class ($40.77\%$ $vs.$ $17.5\%$) . This is a zeroth order methods and clearly requires further work to make sure that *bounded misclassification is achieved on top of and not in place of accuracy*.

In case of adversarial examples, although WLF seems to have a slightly better accuracy ($13.79\%$) than the vanilla network ($1.79\%$), both the networks have low accuracy (see the second column of Figure 5) to even say the WLF provides an effective defense. Interestingly, the one trained with WLF seems to have developed a curious cluster of misclassified labels on '3' and '8' which is, for many classes, *not* even within the bound defined in the similarity function. At this time, we do not have a good explanation why this happens.

More intriguingly, after we add more noise to the adversarial images, the accuracy bumps up a little for the vanilla network and takes a hit for the network using the weighted loss function. Given that the accuracy values are less, it is hard to make claims about weather adding more white noise can be effective is nullifying some of the threat due to adversarial noise. While results are not conclusive, we are presently working on developing a better understanding of the properties of WLFs on DNNs and developing effective approaches to trade off misclassification costs.

## 3.1. Work in Progress

We are currently in the process of developing decision systems that can demonstrably show immunity towards misclassified labels based on this notion of operational security. We feel that the notion of operational security can be effective for classification in open-world scenarios. Say the classification system get the picture of a kangaroo as input at test time for the first time, i.e. there is no class label kangaroo and no such images in the training set, could there be ways of (mis)classifying it to a more meaningful class (like human, rabbit etc.) as opposed to less meaningful class (like glass, bottle etc.) so that the decision of the system built on top of the classification system remains the same?

In fact, such considerations are not new in the literature on planning for robotics where higher level decision making needs to be connected to lower level motor controls (Srivastava et al., 2014) and thus must be aware of the confidence of perception. Works on Hierarchical Task Networks (HTNs) (Erol et al., 1994) and "nogoods" backtracking (Kambhampati, 1998) in decision making algorithms may prove to be invaluable resources for developing techniques for bootstrapping classification algorithms with such risk-aware decision modules. This is also, of course, contingent on developing, as we discussed in the section on preliminary results, more effective methods of bounded misclassification that is able to preserve accuracy while at the same time provide operational security. We hope to report on the latest findings at the workshop.

## References

Athalye, Anish, Engstrom, Logan, Ilyas, Andrew, and Kwok, Kevin. Synthesizing robust adversarial examples. *CoRR*, abs/1707.07397, 2017. URL http://arxiv.org/abs/1707.07397.

Duda, Richard O, Hart, Peter E, and Stork, David G. *Pattern classification*. Wiley, New York, 1973.

Erol, Kutluhan, Hendler, James, and Nau, Dana S. Htn planning: Complexity and expressivity. In *AAAI*, volume 94, pp. 1123–1128, 1994.

Kambhampati, Subbarao. On the relations between intelligent backtracking and failure-driven explanation-based learning in constraint satisfaction and planning. *Artificial Intelligence*, 105(1):161–208, 1998.

Miller, George and Fellbaum, Christiane. Wordnet: An electronic lexical database, 1998.

Moosavi-Dezfooli, Seyed-Mohsen, Fawzi, Alhussein, and Frossard, Pascal. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582, 2016.

Papernot, Nicolas, McDaniel, Patrick, Jha, Somesh, Fredrikson, Matt, Celik, Z Berkay, and Swami, Ananthram. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pp. 372–387. IEEE, 2016a.

Papernot, Nicolas, McDaniel, Patrick, Wu, Xi, Jha, Somesh, and Swami, Ananthram. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597. IEEE, 2016b.

Pazzani, Michael, Merz, Christopher, Murphy, Patrick, Ali, Kamal, Hume, Timothy, and Brunk, Clifford. Reducing misclassification costs. In *Proceedings of the Eleventh International Conference on Machine Learning*, pp. 217–225, 1994.

Srivastava, Siddharth, Fang, Eugene, Riano, Lorenzo, Chitnis, Rohan, Russell, Stuart, and Abbeel, Pieter. Combined task and motion planning through an extensible planner-independent interface layer. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 639–646. IEEE, 2014.

Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian, and Fergus, Rob. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.