A Non-Targeted Attack Approach for the Coarse Misclassification Problem

Ismail R. Alkhouri
ECE Department
University of Central Florida
Orlando, FL, USA
ismail.alkhouri@ucf.edu

Alvaro Velasquez
CS Department
University of Colorado, Boulder
Boulder, CO, USA
alvaro.velasquez@colorado.edu

George Atia
ECE & CS Departments
University of Central Florida
Orlando, FL, USA
george.atia@ucf.edu

2023 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-8867-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IJCNN54540.2023.10191658

***Abstract*—The evaluation of classifiers' robustness against adversarial attacks is typically performed through metrics based on the minimal perturbation required for misclassification. The conventional method of generating these perturbations relies on setting a limit on the maximum allowed perturbation (restricted attack method) and a non-targeted attack formulation. This approach, however, disregards any relationships between classes. Our paper introduces a novel, non-targeted, bound-restricted method for achieving coarse misclassification, so that the perturbed feature is classified outside its true coarse class. We present an efficient, single-step solution to the coarse misclassification problem and analyze its computational requirements. Our experiments showcase the superiority of our method, surpassing state-of-the-art in terms of both the perceptibility of adversarial examples and runtime.**

***Index Terms*—Adversarial Attacks, Coarse misclassification, non-targeted formulation, bound-restricted attacks.**

I. Introduction

Machine Learning (ML) models for classification have gained widespread adoption, playing critical roles in various domains, including safety- and mission-critical applications [1]–[3] due to their remarkable performance. However, numerous studies over the past decade have revealed that ML models can be easily deceived by well-designed additive perturbations [4]–[7], making them highly vulnerable to adversarial attacks and significantly lacking in robustness.

Adversarial Training (AT), first introduced in [5] and later advanced in [8], is widely considered as the most effective defense mechanism for ML models against adversarial attacks. It uses a minimax formulation, which balances the trade-off between the model's accuracy on clean examples and robustness against adversarial examples. The method trains ML classifiers by exposing them to adversarial examples generated through an attack method, instead of relying solely on clean feature vectors. This allows the model to learn to recognize and defend against the worst-case scenarios during training, resulting in a more robust classifier. The use of the minimax formulation in AT has inspired researchers to develop new attack methods that aim to enhance the robustness of ML models against adversarial attacks.

Conventional attack methods used in AT aim to generate adversarial examples that lead to misclassification, regardless of the relationships between classes. However, the severity of misclassification can vary greatly depending on the relationship between the true and predicted labels. For instance, misclassifying a 'cup' as a 'can' is clearly less severe than misclassifying a 'stop sign' as a 'truck', which could have serious consequences. To address this issue, this paper presents a novel approach, following the work of [9], that considers the relationships between classes. The approach defines a mapping function for each example that returns a coarse prediction based on the fine classification.

Adversarial perturbation attacks in AT can be categorized into non-targeted and targeted attacks, based on the goal of the attacker. Non-targeted attacks aim to induce any misclassification, without consideration for the predicted label [10]. On the other hand, targeted attacks seek to alter the prediction to a specified target class [11]. Depending on the attack formulation, adversarial examples are categorized into three main categories: maximum allowable attacks (bound-restricted attack formulation), minimum norm attacks, and regularization-based attacks [5]. In the maximum allowable attack, the perturbation is constrained within a defined bound ϵ, such as an l_∞ bound [4], [5], [12], [13]. The minimum norm attack seeks to generate the minimum perturbations that induce misclassification [10], [14], while the regularization-based formulation aims to generate adversarial examples by adding a perturbation to the input that minimizes a regularization term subject to a constraint on the adversarial loss [11], [15]. Previous studies have found that the maximum allowable attack formulation is the most efficient in generating adversarial examples [5]. It can also be used to produce the minimum perturbations by iteratively adjusting the bound. In AT, a bound ϵ is set to represent the level of imperceptibility for each dataset. If an attack requires a higher perturbation than ϵ, it is considered detectable. The ϵ value is set to 0.3 for MNIST and 8/255 for CIFAR-10 in the original AT work [5]. This standard has been followed by subsequent AT methods, including [8], [16], [17]. This paper focuses on non-targeted and bound-restricted adversarial attacks to address the issue of coarse misclassification, with the long-term goal of improving the robustness of ML classifiers.

Next, we summarize the contributions of this work. First, we formulate an optimization problem for generating bound-restricted, non-targeted adversarial examples for coarse mis-

classification. Second, inspired by [8], we present a fast single-step approach extending the gradient descent based method to solve the formulated problem. Using the commonly used cross entropy loss, we quantify the number of additional gradients needed, compared to standard misclassification formulations. Third, using a Neural Network (NN)-based image classification model, we demonstrate the effectiveness of our approach in obtaining minimum coarse perturbations compared to [10], which explores all targets outside the true coarse class. Our approach demonstrates a reduction in both the amount of required perturbations and runtime.

II. Related Work

Evaluating the robustness of classifiers involves using attack methods to determine minimum perturbations that cause misclassification. The literature is abundant with gradient-based approaches in white-box settings [18]. The most well-known is the Fast Gradient Sign Method (FGSM) [4], which computes the gradient of the loss function w.r.t the input feature vector using back-propagation [19] to generate perturbations. The Projected Gradient Descent (PGD), an iterative version of FGSM [20], was proposed to improve the likelihood of inducing misclassification by taking iterative steps in the direction of the negative gradient of the loss function. Existing state-of-the-art methods, including Automatic PGD [13], Momentum-based FGSM [21], and FGSM with random initialization [8], are variants of the FGSM and PGD methods and do not consider the relationship between labels of interest. In contrast, we introduce a new formulation focused on inducing coarse misclassifications, using a mapping function from finer to coarser classes.

The authors in [9] and [10] focus on inducing coarse misclassification in classifiers, similar to this paper. However, their method uses a minimum norm attack formulation, which has limitations in terms of adversarial example imperceptibility and computational cost compared to the bound-restricted formulation we use [5]. Additionally, they use a targeted attack approach, which requires exploring all possible targets outside the true coarse class, making it impractical for large datasets such as CIFAR-100 (100 classes) and Imagenet (1000 classes) [22], [23].

III. Classification Model and the Mapping of Labels

We model the classifier as a function $h : \mathbb{R}^N \to [M]$, which maps a feature vector $x \in \mathbb{R}^N$ to one of M possible (fine) classes, where $[M] := \{1, 2, \ldots, M\}$. The predicted fine class is obtained by finding the maximum of the functional $f : \mathbb{R}^N \to \Delta^M$, where Δ^M is the probability simplex over M dimensions, with entries $f_m, m \in [M]$, according to:

$$h(x) = \underset{m \in [M]}{\operatorname{argmax}}\ f_m(x)\,. \tag{1}$$

For mapping the labels, we use a grouping function $T : [M] \to [M_c]$ to transform fine labels into coarser representations (also referred to as super labels), where $M_c < M$ is the number of coarse classes. The function T allows us to define the coarse class sets as:

$$S_i := \{m \in [M] : T(m) = i\}, i \in [M_c]\,, \tag{2}$$

where S_i is a group of fine labels that belong to the i-th super class. The relationship between the fine prediction, the classifier, and the coarse class is illustrated in Figure 1.

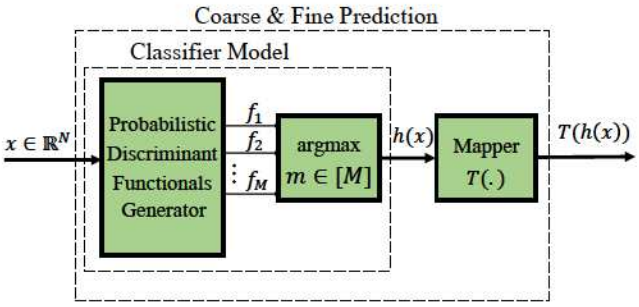

Fig. 1: Classification model and the coarse prediction.

IV. Coarse Misclassification Attacks

Given example x, the goal of a standard white-box additive attack is to obtain the minimum perturbation vector, $\delta \in \mathbb{R}^N$, such that $h(x) \neq h(x+\delta)$. Given some input feature vector x and its true label y, the maximum allowable attack formulation (bound-restricted) maximizes a loss function, $\mathcal{L} : \Delta^M \times \Delta^M \to \mathbb{R}$, between the true label vector representation and the output $f(x+\delta)$ of the perturbed vector, subject to a constraint on the perturbation norm [4],

$$\max_{\delta} \mathcal{L}(f(x+\delta), \mathbb{1}_y) \quad \text{subject to} \quad \|\delta\|_p \leq \epsilon\,, \tag{3}$$

where $\mathbb{1}_y \in \{0,1\}^M$ is the 1-to-M encoding of the true class, commonly known as the one-hot representation. This formulation can be extended to targeted attacks as

$$\min_{\delta} \mathcal{L}(f(x+\delta), \mathbb{1}_t) \quad \text{subject to} \quad \|\delta\|_p \leq \epsilon\,, \tag{4}$$

where $t \in [M]$ is the target class. The minimum norm approach, as described in [5], seeks to find the minimum perturbations that cause non-targeted and targeted misclassification. This is achieved by solving the following optimization problems:

$$\min_{\delta} \|\delta\|_p \quad \text{subject to} \quad h(x) \neq h(x+\delta)\,, \tag{5}$$

and

$$\min_{\delta} \|\delta\|_p \quad \text{subject to} \quad h(x+\delta) = t\,. \tag{6}$$

The approach finds a solution by solving Eqs. (3) and (4) incrementally, for larger values of ϵ, until misclassification occurs. The process is outlined in Algorithm 1.

Equation (3) is designed to cause misclassification without regard to any relationships between classes. In this paper, the goal is to generate non-targeted perturbations that cause a misclassification in the coarse label. To achieve this, a

Algorithm 1 The minimum non-targeted (targeted) perturbations using bound-restricted formulation [5].

Input: x, h, initial bound ϵ, bound increase step ϵ_s, (target t).
Output: Approximate solution of (5) ((6)), δ^*

1: **While** $h(x) = h(x+\delta)$ $(h(x+\delta) \neq t)$
2: **Increase bound** $\epsilon \leftarrow \epsilon + \epsilon_s$
3: **Find** δ as the solution of (3) ((4)) using ϵ
4: **Obtain** $\delta^* \leftarrow \delta$

Algorithm 2 The minimum perturbation for the coarse misclassification using targeted attacks [10].

Input: x, h, mapping T.
Output: Non-targeted coarse perturbations, δ_c

1: **For all** $t \in S'_{T(h(x))}$
2: **Find** δ_t as the solution of (6) with target t using Algorithm 1
3: **Find** $t_c := \text{argmin}_{t \in S'_{T(h(x))}} \|\delta_t\|_p$, **Obtain** $\delta_c \leftarrow \delta_{t_c}$

perturbation vector $\delta_c \in \mathbb{R}^N$ is found such that the superclass set $S_{T(h(x))}$ is not equal to the result of the perturbed input $h(x+\delta_c)$, i.e.,

$$h(x+\delta_c) \notin S_{T(h(x))} \,. \tag{7}$$

An approach to extend the maximum allowable attack formulation for non-targeted coarse misclassification is to use the targeted attack formulation in (6). As described in [9], [10], this involves exploring all labels in the complement set $S'_{T(h(x))} := [M] \setminus S_{T(h(x))}$ as targets in (6) and selecting the target that results in the minimum perturbation with respect to (w.r.t.) a distance measure, commonly the l_p norm. This process is summarized in Algorithm 2.

Finding the minimum perturbation for coarse misclassification using Algorithm 2 requires solving the optimization problem in (6) $|S'_{T(h(x))}|$ times, where $|\cdot|$ denotes the cardinality of a set. This leads to an increase in the number of optimization problems to be solved as the number of labels grows, making the approach computationally expensive. In this paper, we present a more efficient formulation in the following subsection.

A. Cost-Efficient Coarse Misclassification

Given the effectiveness of AT as a defense mechanism, it is crucial to develop attack approaches that are computationally efficient. This is because AT involves training with adversarial examples (AEs) instead of clean representations. Our proposed formulation reduces the number of optimization problems needed to find the coarse perturbation δ_c by seeking to obtain a discriminant coarse probabilistic representation from the fine prediction vector $f(x)$. This representation is defined using the discriminant score of each set S_i in the coarse prediction, which is calculated as the sum of the fine prediction scores for all classes in that set. This is expressed in Equation (8).

$$q_i(x) = \sum_{m \in S_i} f_m(x), \quad i \in [M_c] \,. \tag{8}$$

Algorithm 3 The minimum non-targeted perturbations for the coarse misclassification using (9) (Our approach).

Input: x, h, T, initial bound ϵ, bound increase step ϵ_s.
Output: Approximate solution of (10), δ_c^*

1: **While** $h(x+\delta) \in S_{T(h(x))}$
2: **Increase bound** $\epsilon \leftarrow \epsilon + \epsilon_s$
3: **Find** δ_c as the solution of (9) using ϵ
4: **Obtain** $\delta_c^* \leftarrow \delta_c$

These entries are then combined into vector $q(x) \in \Delta^{M_c}$. As an example, consider $M = 6$ and $M_c = 3$ such that $S_1 = \{1,2\}$, $S_2 = \{3,4\}$, and $S_3 = \{5,6\}$. Given observation vector x and its label representation $\mathbb{1}_y = [0\ 0\ 0\ 0\ 1\ 0]^\top$, if the output probabilities of the classifier are $f(x) = [0.1\ 0.1\ 0.2\ 0\ 0.6\ 0]^\top$, then, based on (8), the coarse representative scores are $q(x) = [0.2\ 0.2\ 0.6]^\top$ with $\mathbb{1}_{T(y)} = [0\ 0\ 1]^\top$.

Our proposed approach uses a coarse loss function, $\mathcal{L}_c : \Delta^{M_c} \times \Delta^{M_c} \to \mathbb{R}$, and vector q to formulate a bound-restricted, non-targeted coarse attack, given some ϵ, as follows:

$$\min_{\delta_c} \mathcal{L}_c(q(x+\delta_c), \mathbb{1}_{T(y)}) \quad \text{subject to} \quad \|\delta_c\|_p \leq \epsilon \,. \tag{9}$$

The minimum coarse perturbations can be found by solving

$$\min_{\delta_c} \|\delta_c\|_p \quad \text{subject to} \quad h(x+\delta_c) \notin S_{T(y)} \,, \tag{10}$$

using a bound-restricted approach, similar to Algorithm 1 for standard misclassification. Algorithm 3 presents the non-targeted coarse misclassification procedure, which starts with an initial bound and uses (9) to generate disturbances δ_c. At each iteration, the bound is increased until the coarse prediction changes. The minimum coarse perturbations are defined as the solution of (10).

Consider that the number of bound values to be tried in order to obtain the minimum coarse perturbations is L. The work in [10] requires solving $L|S'_{T(h(x))}|$ optimization problems, whereas our approach only requires solving L problems.

B. Proposed Solution

In this section, we present an efficient, single-step solution to the coarse misclassification problem in (9). Our approach is inspired by the solution to the standard misclassification problem outlined in (3).

The FGSM method, as described in [4], uses a single-step approach to generate non-targeted perturbations for any misclassification with respect to a bound ϵ. The method utilizes the l_∞ norm to produce the perturbation, calculated as

$$\delta = \epsilon \ \text{sign}\left(\nabla_\delta \mathcal{L}\left(f(x+\delta), \mathbb{1}_y\right)\right) \,, \tag{11}$$

where sign(.) represents the signum function and ∇ is the gradient operator. The gradient is taken with respect to the perturbation, δ, and the loss function, $\mathcal{L}$, is evaluated for the altered input, $f(x+\delta)$, and the true label, $\mathbb{1}_y$. In a recent work [8], known as random FGSM (rFGSM), the authors propose an alternative approach to generate perturbations. Instead of

starting with a zero vector, the perturbation vector is initialized randomly, as follows:

$$\begin{aligned} \delta &\leftarrow \mathcal{U}(-\epsilon, \epsilon) \\ \delta &\leftarrow \delta + \epsilon \ \text{sign}\Big(\nabla_\delta \mathcal{L}\big(f(x+\delta), \mathbb{1}_y\big)\Big) \\ \delta &\leftarrow \max(\min(\delta, \epsilon), -\epsilon)\,, \end{aligned} \tag{12}$$

where $\mathcal{U}(-\epsilon, \epsilon)$ generates a uniform random vector for which each entry is within the interval $[-\epsilon, \epsilon]$. Despite the random initialization of the perturbation vector, rFGSM remains a single-step method as it only requires computing the gradient once for each input x. In [8], it is shown that rFGSM (12) outperforms the conventional FGSM in (11). In addition, for the task of AT, it performs on par with the well-known iterative FGSM (iFGSM) [12], also referred to as the Projected Gradient Descent attack [5], while being less computationally intensive. The reason for the reduction in computational requirements is that in rFGSM, the gradients are calculated only once, whereas in iFGSM, multiple gradient calculations are necessary.

Therefore, in this paper, we extend the solution in (12) for the standard bound-restricted attacks to our proposed coarse misclassification optimization problem in (9) to generate the coarse perturbations δ_c. Given a mapping function T, we generate the coarse perturbations δ_c by using the coarse prediction vector q instead of the fine prediction vector f. We present the rFGSM for coarse misclassification (rFGSMc) as follows:

$$\begin{aligned} \delta_c &\leftarrow \mathcal{U}(-\epsilon, \epsilon) \\ \delta_c &\leftarrow \delta_c + \epsilon \ \text{sign}\Big(\nabla_{\delta_c} \mathcal{L}_c\big(q(x+\delta_c), \mathbb{1}_{T(y)}\big)\Big) \\ \delta_c &\leftarrow \max(\min(\delta_c, \epsilon), -\epsilon)\,. \end{aligned} \tag{13}$$

The computational cost of the proposed solution in (13) is largely determined by the calculation of the gradients of the loss functions $\mathcal{L}$ and $\mathcal{L}_c$ with respect to δ and δ_c, respectively. It requires computing additional gradients when compared to the standard solution in (12). We provide a theoretical specification of these additional gradients using the cross entropy loss in the following theorem.

Theorem 1. *Given a mapping function T, cross entropy losses $\mathcal{L}$ and $\mathcal{L}_c$ w.r.t. M and M_c entries, respectively, an observation vector x and its true fine class y, using* (13) *to generate coarse perturbations δ_c for the goal of coarse misclassification requires an additional $|S_{T(y)}| - 1$ discriminant functionals gradient computations when compared to generating standard perturbations for any misclassification using* (12)*.*

Proof. By the definition of the one-hot encoding representation of the true class and the cross entropy loss, the standard loss is

$$\mathcal{L}(f(x+\delta), \mathbb{1}_y) = -\sum_{m \in [M]} \mathbb{1}_y(m) \log f_m(x+\delta)\,. \tag{14}$$

which reduces to

$$\mathcal{L}(f(x+\delta), \mathbb{1}_y) = -\log f_y(x+\delta)\,. \tag{15}$$

The gradient of (15) w.r.t. δ is

$$\nabla_\delta \mathcal{L}(f(x+\delta), \mathbb{1}_y) = \frac{-\nabla_\delta f_y(x+\delta)}{f_y(x+\delta)}\,. \tag{16}$$

Next, we use the one-hot encoding coarse representation of the true class obtained using mapping T, and the cross entropy loss w.r.t. M_c entries for the coarse case. We have

$$\mathcal{L}_c(q(x+\delta_c), \mathbb{1}_{T(y)}) = -\sum_{i \in [M_c]} \mathbb{1}_{T(y)}(i) \log q_i(x+\delta_c)\,, \tag{17}$$

which can be reduced to

$$\mathcal{L}_c(q(x+\delta_c), \mathbb{1}_{T(y)}) = -\log q_{T(y)}(x+\delta_c)\,. \tag{18}$$

Using (8), (18) can be rewritten as

$$\mathcal{L}_c(q(x+\delta_c), \mathbb{1}_{T(y)}) = -\log \sum_{i \in S_{T(y)}} f_i(x+\delta_c)\,. \tag{19}$$

The gradient of (19) w.r.t. δ_c is obtained as

$$\nabla_{\delta_c} \mathcal{L}_c(q(x+\delta_c), \mathbb{1}_{T(y)}) = \frac{-\nabla_{\delta_c} \sum_{i \in S_{T(y)}} f_i(x+\delta_c)}{\sum_{i \in S_{T(y)}} f_i(x+\delta_c)}\,. \tag{20}$$

Given the distributive property of gradients, the calculation of gradients in (20) requires $|S_{T(y)}|$ discriminant functionals. However, for the standard case in (16), only one discriminant functional representing the true fine label needs to be calculated. Thus, our proof is complete. □

V. Experimental Results

In this section, we present an image classification example to demonstrate the superiority of our proposed approach, as compared to the approach in [10] (Algorithm 2), in terms of the minimum perturbations required to cause coarse misclassification. Although Algorithm 1 has a different goal of inducing any misclassification, we compare its performance with our proposed algorithm in terms of imperceptibility and runtime, using it as a baseline. This comparison allows us to assess the additional resources required w.r.t. different mappings.

For imperceptibility, we use two evaluation metrics. First, we use the l_p norm for $p \in \{0, \infty\}$, defined for δ or δ_c as $\|\delta\|_\infty = \max_{n \in [N]} |\delta(n)|$, and $\|\delta\|_2 = \sqrt{\sum_{n \in [N]} |\delta(n)|^2}$, respectively, where $\delta(n)$ is the n^{th} entry of vector δ. Second, we use the Structural Similarity Index (SSIM) presented in [24]. The SSIM metric is a measure of the structural similarity between two images and its values lie in the interval $[0, 1]$. A value of 1 indicates that the two images are identical, taking into account luminance, contrast, structural measurements, and pixel differences.

We utilize the Fashion MNIST (FMNIST) datatset [25] in which each feature vector is a grayscale image of 28×28 pixels. The labels, from 0 to 9, are defined as: 'Top' (or "T-shirt"), 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', and 'Boot' (short for 'Ankle Boot'). We use a standard NN-based ML classifier. The architecture of the neural network and further details regarding the experimental settings, as well as the code used, have been made available

TABLE I: The mapping functions considered in our experiments with $M_c = 3$ ($M_c = 4$) for the first (last) three rows.

Mapping	Coarse Sets	Labels Grouping
T_1	$S_1 = \{0,1,3\}, S_2 = \{7,9\}, S_3 = \{2,4,5,6,8\}$	{Top,Trouser,Dress},{Sneaker, Boot},{Pullover,Coat,Sandal,Shirt,Bag}
T_2	$S_1 = \{0,1,2\}, S_2 = \{3,4,5\}, S_3 = \{6,7,8,9\}$	{Top,Trouser,Pullover},{Dress,Coat,Sandal},{Shirt,Sneaker,Bag,Boot}
T_3	$S_1 = \{1,3\}, S_2 = \{5,7,9\}, S_3 = \{0,2,4,6,8\}$	{Trouser,Dress},{Sandal,Sneaker,Boot},{Top,Pullover,Coat,Shirt,Bag}
T_4	$S_1 = \{1,3\}, S_2 = \{4,8\}, S_3 = \{5,7,9\}, S_4 = \{0,2,6\}$	{Trouser,Dress},{Coat,Bag},{Sandal,Sneaker,Boot},{Top,Pullover,Shirt}
T_5	$S_1 = \{0,1,2\}, S_2 = \{3,4\}, S_3 = \{5,6,7\}, S_4 = \{8,9\}$	{Top,Trouser,Pullover},{Dress,Coat},{Sandal,Shirt,Sneaker},{Bag,Boot}
T_6	$S_1 = \{0,6\}, S_2 = \{1,4,8\}, S_3 = \{2,3\}, S_4 = \{5,7,9\}$	{Top,Shirt},{Trouser,Coat,Bag},{Pullover,Dress},{Sandal,Sneaker,Boot}

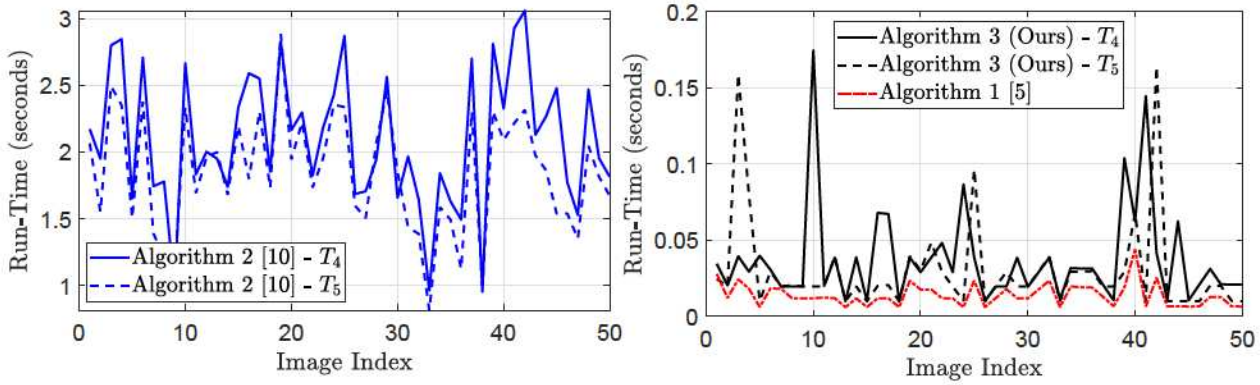

Fig. 2: Comparison of Run-time performance for Algorithms 2 (*left*), and 1 and 3 (*right*) on the first 50 samples in the FMNIST test set.

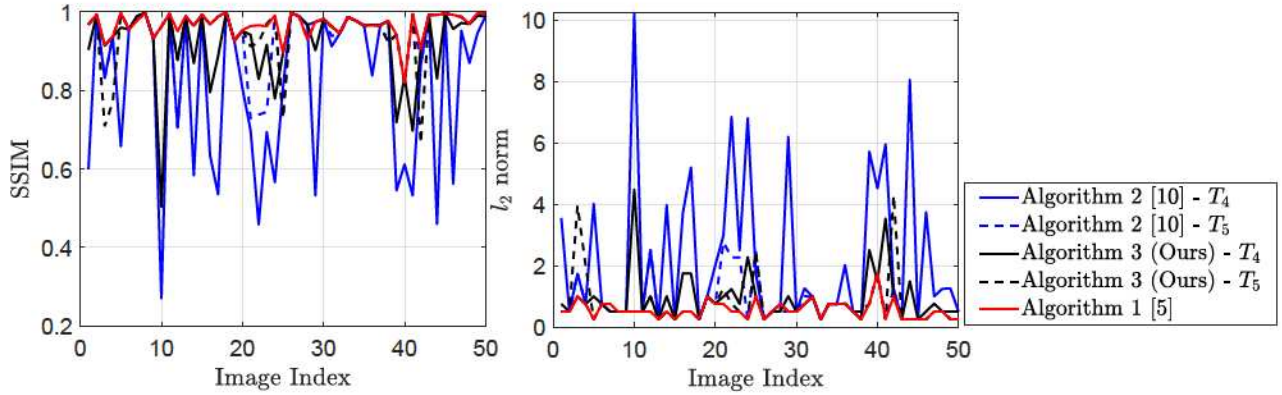

Fig. 3: Imperceptibility results of the considered algorithms for the first 50 samples in the test set of FMNIST using T_4 and T_5. The metrics are SSIM (*left*) and the l_2 norm (*right*).

online[1]. We use a standard Intel(R) Core(TM) i9-9940 CPU @ 3.30GHz machine. The initial value for the bound in Algorithms 1 and 3 is chosen as 0.01, with a step $\epsilon_s = 0.01$.

Also, we present our results using various mappings of the labels. These mappings are defined in Table I as T_i, for $i \in [6]$. We refer to mappings that result in super sets with semantic meaning as semantic-based mappings. These include T_1, T_3, T_4, and T_6. The semantic meaning is largely determined by the positioning of the wearable item. The other mappings, T_2 and T_5, are non-semantic based. These are formed by selecting 2, 3, or 4 items per super set from the set $[10] = \{1, 2, \ldots, 10\}$.

[1]https://github.com/ialkhouri/CoarseMisClassification

We generate AEs using Algorithm 1 for the goal of causing any misclassification. Then, we use Algorithm 2 and Algorithm 3 to generate AEs to induce coarse misclassification w.r.t. the mappings in Table I.

Figure 2 presents runtime results of the first 50 samples in the test set of FMNIST using T_4 and T_5 of Algorithm 2 (*left*) and Algorithms 1 and 3 (*right*). Our proposed approach requires considerably less time to generate coarse AEs when compared to Algorithm 2, regardless of the mapping function used. From the considered set, the minimum runtime for Algorithm 2 to produce a coarse AE is recorded at about 0.85 seconds, while the maximum runtime for Algorithm 3 was around 0.155 seconds. In comparison, our method requires only a few extra tens of milliseconds. The difference in

TABLE II: Average results for l_∞ imperceptibility and run-rime (in seconds) to obtain the minimum required perturbations to induce coarse misclassification using (*i*) Algorithm 2 [10] (columns 2 and 3) and (*ii*) our proposed method in Algorithm 3 (columns 4 and 5). Results for obtaining the minimum perturbations for any misclassification using Algorithm 1 are shown in the last two columns. The values after the $\pm$ sign represent the variance.

Mapping	Avg. l_∞ (Alg.2)	Avg. Run-time (Alg.2)	Avg. l_∞ (Alg.3)	Avg. Run-time (Alg.3)	Avg. l_∞ (Alg.1)	Avg. Run-time (Alg.1)
T_1	0.104 $\pm$0.01	1.748 $\pm$0.36	0.057 $\pm$0.003	0.055 $\pm$0.003	0.023 $\pm$0.0003	0.014 $\pm$0.0001
T_2	0.049 $\pm$0.004	1.642 $\pm$0.24	0.033 $\pm$0.001	0.032 $\pm$0.013	0.023 $\pm$0.0003	0.014 $\pm$0.0001
T_3	0.15 $\pm$0.011	1.913 $\pm$0.032	0.045 $\pm$0.001	0.044 $\pm$0.001	0.023 $\pm$0.0003	0.014 $\pm$0.0001
T_4	0.086 $\pm$0.008	2.05 $\pm$0.302	0.034 $\pm$0.007	0.033 $\pm$0.007	0.023 $\pm$0.0003	0.014 $\pm$0.0001
T_5	0.043 $\pm$0.003	1.862 $\pm$0.025	0.033 $\pm$0.001	0.034 $\pm$0.002	0.023 $\pm$0.0003	0.014 $\pm$0.0001
T_6	0.071 $\pm$0.007	1.939 $\pm$0.267	0.0312 $\pm$0.007	0.031 $\pm$0.007	0.023 $\pm$0.0003	0.014 $\pm$0.0001

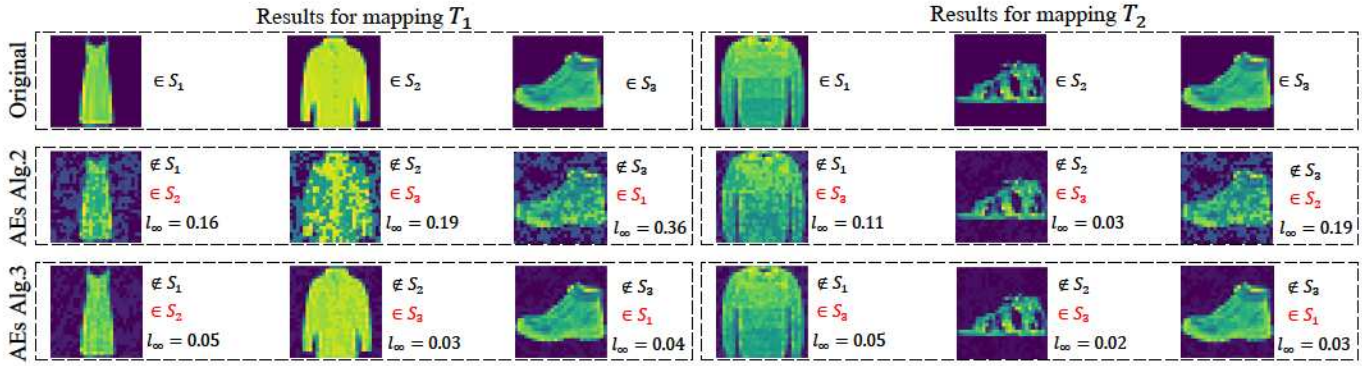

Fig. 4: Sample for each coarser set of the FMNIST dataset obtained from using the semantics-based mapping T_1 (*left*) and the non semantic-based mapping T_2 (*right*). The second and third rows show minimum perturbations coarse adversarial examples generated against the considered NN classifier using the approach in [10] and our method, respectively. For each sample, the coarse prediction is on the right of each image. The l_∞ value posted on the right of each coarse AE represents the distance between the original example and perturbed one. The coarse prediction colored in red represents the fooled super set.

runtime is in line with the predictions made in Theorem 1.

Figure 3 showcases the imperceptibility results, using the same AEs and mapping functions as in Figure 2, in terms of the SSIM measure (*left*) and the l_2 norm (*right*). The results indicate that the approach in [10] requires a greater amount of perturbation to induce coarse misclassification compared to our approach, as indicated by the lower (higher) values of SSIM (l_2-norm). Algorithm 1 results in the least amount of perturbations, as its goal is to induce any misclassification, rather than inducing a misclassification that is restricted by a mapping function and grouping of labels.

Table II presents the average results of imperceptibility (l_∞-norm) and runtime (in seconds) using the mappings in Table I. Our approach is more efficient, requiring lower perturbations and less runtime compared to [10]. The greatest difference in perturbation occurs when mappings have semantic meaning. For T_1 and T_3 (semantic-based), nearly twice or thrice the amount of perturbation is required to fool the coarse class using the method in [10]. For T_2, the average scores of Algorithm 2 and Algorithm 3 are 0.049 and 0.033, respectively. Remarkably, our approach shows closer performance to Algorithm 1, where the goal is to cause any misclassification. This highlights the significant gain of using our formulation versus the approach of Algorithm 2. As expected, Algorithm 1 requires less perturbation and runtime than those required by our method to cause coarse misclassification. The results also indicate that it is easier to fool the coarse class when label mappings have no semantic meaning (e.g., T_2 vs T_1 or T_3).

Figure 4 displays clean examples (first row) and their coarse AEs, generated using Algorithm 2 (second row) and our Algorithm 3 (third row). The coarse AEs use semantic based mapping T_1 (*left*) and non-semantic mapping T_2 (*right*). From the coarse prediction of each AE, we observe that both Algorithms 2 and 3 are successful at inducing coarse misclassification. Similar to the observations made earlier for Table II, we first note that perturbation required to fool the coarse class is higher for Algorithm 2 when compared to our approach. Second, we observe that it is easier to fool the coarse class when the grouping of labels is not semantic-based, regardless of the method used. This is seen by comparing the l_∞ values. Additionally, it is evident that the AEs generated using our approach are visually more similar to the original examples than those generated using Algorithm 2.

VI. Conclusion and Future Work

This paper introduced a new approach to generating adversarial examples that aim to induce coarse misclassification. A non-targeted, bound-restricted optimization problem is formulated based on a grouping function of labels. Introducing a probabilistic score for each super class set allowed us to extend the solution for the standard attack formulation.

The required gradient computations are theoretically specified using the cross entropy loss. Our experimental results show improved efficiency in terms of required perturbations and runtime compared to current state-of-the-art.

Future plans include validation with larger Neural Network structures and datasets, and integration with Adversarial Training for the design of stronger ML classifiers with respect to coarse predictions.

ACKNOWLEDGMENT

This work was supported by NSF CAREER Award CCF1552497, NSF grant CCF2106339, DOE Award Number DE-EE0009152, the Air Force Research Laboratory through Contract Number FA8750-20-3-1003, and the Air Force Office of Scientific Research through Award 20RICOR012.

REFERENCES

[1] Asif Iqbal Khan, Junaid Latief Shah, and Mohammad Mudasir Bhat, "Coronet: A deep neural network for detection and diagnosis of covid-19 from chest x-ray images," *Computer Methods and Programs in Biomedicine*, p. 105581, 2020.

[2] Samet Akcay, Mikolaj E. Kundegorski, Chris G. Willcocks, and Toby P. Breckon, "Using deep convolutional neural network architectures for object classification and detection within x-ray baggage security imagery," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2203–2215, 2018.

[3] Weize Quan, Kai Wang, Dong-Ming Yan, and Xiaopeng Zhang, "Distinguishing between natural and computer-generated images using convolutional neural networks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2772–2787, 2018.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[5] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[6] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[7] Ismail R. Alkhouri, Alvaro Velasquez, and George K. Atia, "Boss: Bidirectional one-shot synthesis of adversarial examples," in *2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP)*, 2022, pp. 1–6.

[8] Eric Wong, Leslie Rice, and J Zico Kolter, "Fast is better than free: Revisiting adversarial training," *arXiv preprint arXiv:2001.03994*, 2020.

[9] Ismail R Alkhouri, Alvaro Velasquez, Stanley Bak, and George K Atia, "On the coarse robustness of classifiers," in *Asilomar*, 2022, pp. 1–6.

[10] Ismail Alkhouri, George Atia, and Wasfy Mikhael, "Fooling the big picture in classification tasks," *Circuits, Systems, and Signal Processing*, pp. 1–31, 2022.

[11] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57.

[12] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

[13] Francesco Croce and Matthias Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International conference on machine learning*. PMLR, 2020, pp. 2206–2216.

[14] Emilio Rafael Balda, Arash Behboodi, and Rudolf Mathar, "On generation of adversarial examples using convex programming," in *52nd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2018, pp. 60–65.

[15] Ismail Alkhouri, Alvaro Velasquez, and George Atia, "Boss: Bidirectional one-shot synthesis of adversarial examples," *arXiv preprint arXiv:2108.02756*, 2021.

[16] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein, "Adversarial training for free!," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] Zhichao Huang, Yanbo Fan, Chen Liu, Weizhong Zhang, Yong Zhang, Mathieu Salzmann, Sabine Süsstrunk, and Jue Wang, "Fast adversarial training with adaptive step size," *arXiv preprint arXiv:2206.02417*, 2022.

[18] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay, "Adversarial attacks and defences: A survey," *arXiv preprint arXiv:1810.00069*, 2018.

[19] Martin Riedmiller and Heinrich Braun, "A direct adaptive method for faster backpropagation learning: The rprop algorithm," in *IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[20] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[21] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9185–9193.

[22] Alex Krizhevsky et al., "Learning multiple layers of features from tiny images," 2009, https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[24] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[25] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song, "Spatially transformed adversarial examples," *arXiv preprint arXiv:1801.02612*, 2018.