Paolo Arcaini
Tao Yue
Erik M. Fredericks (Eds.)

LNCS 14415

Search-Based Software Engineering

15th International Symposium, SSBSE 2023
San Francisco, CA, USA, December 8, 2023
Proceedings

Springer

Expound: A Black-Box Approach for Generating Diversity-Driven Adversarial Examples

Kenneth H. Chan(✉) and Betty H.C. Cheng

Department of Computer Science and Engineering, Michigan State University, 428 S Shaw Ln, East Lansing, MI 48824, USA
{chanken1,chengb}@msu.edu

Abstract. Deep neural networks (DNNs) have been increasingly used in safety-critical systems for decision-making (e.g., medical applications, autonomous vehicles, etc.). Early work with adversarial examples shows that they can hinder a DNN's ability to correctly process inputs. These techniques largely focused on identifying individual causes of failure, emphasizing that minimal perturbations can cause a DNN to fail. Recent research suggests that *diverse* adversarial examples, those that cause different erroneous model behaviors, can better assess the robustness of DNNs. These techniques, however, use white-box approaches to exhaustively search for diverse DNN misbehaviors. This paper proposes a black-box, model and data-agnostic approach to generate diverse sets of adversarial examples, where each set corresponds to one type of model misbehavior (e.g., misclassification of image to a specific label), thus termed a *failure category*. Furthermore, each failure category comprises a diverse set of perturbations, all of which produce the same type of model misbehavior. As such, this work provides both breadth and depth-based information for addressing robustness with respect to multiple categories of failures due to adversarial examples. We illustrate our approach by applying it to popular image classification datasets using different image classification models.

Keywords: Adversarial Example · Novelty Search · Machine Learning

1 Introduction

Due to their ability to learn abstract patterns and representations directly from data, an increasing number of autonomous systems are using machine learning, DNNs, and related technologies for decision making [8]. However, the existence of adversarial examples [19] suggests that they are prone to unexpected failures, such as uncertainty factors and minor perturbations possibly due to adversarial attacks. The *robustness* of DNNs describes their ability to correctly operate in the face of minor noise perturbations or uncertainty factors [2,28]. DNNs used in safety-critical autonomous systems (e.g., medical imaging, autonomous

P. Arcaini et al. (Eds.): SSBSE 2023, LNCS 14415, pp. 19–34, 2024.
https://doi.org/10.1007/978-3-031-48796-5_2

vehicles [3,25], etc.) must be robust against minor data perturbations, as the failure of these systems may lead to potential injuries, financial damages, or loss of life. This paper introduces a black-box diversity-focused search framework to discover failure categories and enable the automated assessment of the severity of each failure category.

Recent state-of-the-art research has suggested the use of *diverse*[1] adversarial examples to address the inability of a DNN to generalize to previously unseen or unknown data deviations [1,21,27]. Traditional approaches to generate adversarial examples typically discover the "nearest" adversarial examples. In contrast, Rozsa *et al.* [21] promoted the use of adversarial examples that 1) have non-minimal perturbations and 2) that result in different model misbehaviors. Aghababaeyan *et al.* [1] showed that the robustness of a DNN model can be better assessed and improved using inputs that trigger different faults. Currently, state-of-the-art research identifies diverse adversarial examples by using white-box techniques to exhaustively search for perturbations, each of which triggers a unique given model misbehavior. However, white-box techniques require access to hidden model information and may not work if the gradient is obfuscated. Additionally, brute-force approaches do not scale as the number of model behaviors (e.g., number of class labels) in a dataset increases.

This paper introduces **EXP**loration and expl**O**itation **U**sing **N**ovelty search for **D**iversity (EXPOUND),[2] a black-box search-based approach to generate diverse adversarial examples to identify the failure categories for a DNN model. Our work contributes several key insights. As the space of possible perturbations for adversarial examples is large, our work leverages the *exploration and exploitation paradigm* [7] in order to strategically discover diverse adversarial examples for different failure categories. More specifically, EXPOUND first uses a coarse-grained *exploration* search to discover the categories of diverse erroneous behaviors for a DNN model, each of which is termed a *failure category*. Second, for each failure category, EXPOUND then applies a localized fine-grained *exploitation* search to identify a diverse secondary collection of adversarial examples, each of which leads to similar model misbehavior (i.e., different perturbations that lead to the same misclassification). Developers can then analyze the adversarial examples and focus their efforts to improve the model's performance against the most relevant adverse failure categories for their applications.

In order to generate diverse adversarial examples, EXPOUND uses novelty search [15] in two complementary ways. In contrast to traditional evolutionary search techniques, novelty search abandons the notion of objective fitness by promoting a diversity metric instead. Specifically, novelty search rewards individuals that are different from each other and produces an archive of diverse individuals. We developed two novelty-based techniques to support our objectives. First, KOBALOS[3] uses novelty search to *explore* the search space in order to *discover* the diverse set of possible failure categories for a DNN, where the

[1] This work uses the term *diversity* to describe different DNN model behaviors.
[2] One definition for *expound* is to explain systematically or in detail.
[3] Kobalos is a mischievous sprite in Greek mythology.

diversity metric is based on the diversity in the *output of the DNN model* (e.g., image classification label). Kobalos *explores* and reduces the search space to diverse types of faults that are likely to induce a failure in the model. Next, Tarand[4] *exploits* the failure categories identified by Kobalos. Tarand applies novelty search for each failure category, generating a *diverse set of faults* (e.g., perturbations) that cause the same type of model misbehavior. Tarand enables developers to assess the failure categories to determine which categories might be confidently misclassified. The breadth and depth-based approach to generate a diverse collection of image perturbations can better inform developers as to how a model's performance and robustness can be improved (e.g., select specific training data, robustness analysis, etc.). Figure 1 overviews the differences between image classification, existing black-box techniques for adversarial examples, and Expound, where elements with hatched shading denote adversarial artifacts. Image classification algorithms seek to correctly label an object in an image (Fig. 1a). The objective of evolutionary search-based adversarial attacks, such as GenAttack [17] or EvoAttack [5], is to generate *one* adversarial example per execution (Fig. 1b) to establish the existence of adversarial examples. In contrast, Expound identifies multiple failure categories, each of which comprises a number of distinct adversarial examples (Fig. 1c).

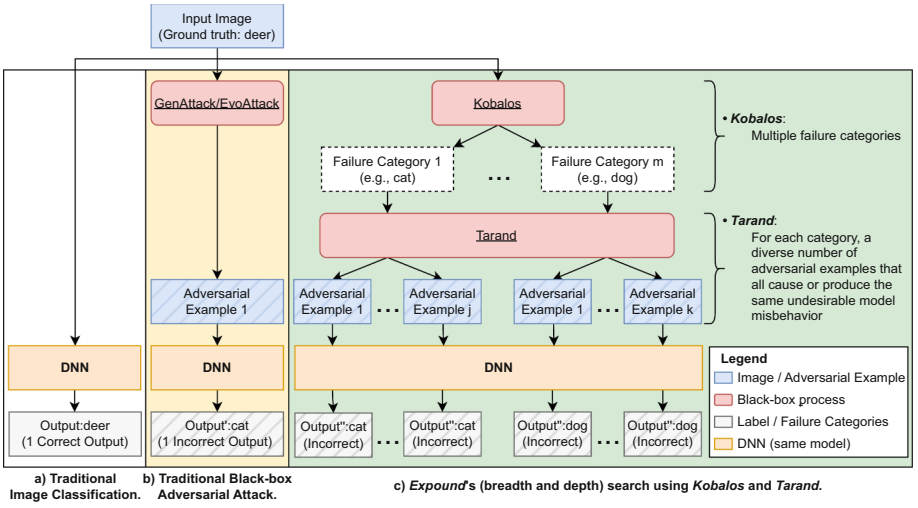

Fig. 1. Overview of image classification, adversarial example, and Expound.

Preliminary results indicate that Expound can discover adversarial examples that lead to different (undesirable) model behaviors that would otherwise not be discovered by existing evolutionary search-based approaches. We implemented the proposed two-phase framework and demonstrated the results on

[4] Tarand is a legendary creature with chameleon-like properties in Greek mythology.

existing image classification algorithms and benchmark datasets. To illustrate that our approach is model and data-agnostic, we used two different DNN architectures, ResNet-20 [10] and MobileNet-V2 [22], on two different benchmark datasets, CIFAR-10 [11] and German Traffic Sign Recognition Benchmark (GTSRB) dataset [23]. The remainder of the paper is organized as follows. Section 2 overviews background and related information for adversarial examples and novelty search. Section 3 describes the methodology for our proposed framework. Section 4 provides the implementation details and the results of a proof-of-concept validation experiment. Section 5 discusses related work. Section 6 concludes the paper and discusses future directions.

2 Background

This section overviews background information and enabling technologies used in this work. We describe adversarial examples and how they impact image classification algorithms. We also overview the key differences between traditional fitness-based evolutionary searches and novelty search.

2.1 Adversarial Examples

Adversarial examples are expected inputs for a machine learning model but modified with minor perturbations (i.e., noise). They hinder a DNN model's ability to correctly process inputs. For example, an image classification model may correctly classify an object in an image as a *deer*. However, when carefully crafted (human-imperceptible) perturbation is applied to the image, the model may incorrectly classify it as a *frog*. An *adversarial direction* [21] is defined as the class of perturbations for adversarial examples that lead to the same misclassified label. An image dataset with a total of n class labels has $n-1$ adversarial directions for each image, as each image can be incorrectly classified as one of $n-1$ labels.

Due to their computational efficiency, *white-box* attacks are popular approaches to generate adversarial examples [4,9,19]. In white-box attacks, the adversary has full access to the model's parameters, weights, and architecture. Traditional approaches for generating adversarial examples analyze a model's gradient information and introduce perturbations that are likely to induce a failure in the model. In practice, the model's hidden parameters are often proprietary (i.e., only the compiled model is available for public use). In contrast, black-box attacks assume that the adversary has no *a priori* knowledge of the model [24]. Instead, they may query the model with any valid input and obtain the model's output. The direction of perturbation must be inferred from the change in behavior caused by the perturbation during each query. Thus, black-box attacks are more realistic but more challenging to successfully generate adversarial examples when compared to white-box attacks [5,17].

2.2 Novelty Search

In contrast to traditional evolutionary techniques, whose objective is to seek solutions that optimize specific criteria, *novelty search* is an evolutionary search technique that focuses on the diversity of the evolved solutions. Traditional evolutionary algorithms focus on optimizing a fitness function (e.g., test case metrics, prediction confidence, etc.) and produce similar output(s). These algorithms are efficient and effective for simple problems, but are prone to be trapped in suboptimal regions [16]. Additionally, when there is no clear fitness metric or multiple solutions exist in different search regions/directions, fitness-based evolutionary search is no longer effective for such problem spaces. In order to explore the large search space and discover multiple interesting (unexpected) solutions, novelty search abandons the notion of fitness [15]. Specifically, novelty search promotes individuals in a population to be different than their neighbors. Example diversity criteria for novelty search include the diverse output labels of a DNN or diverse perturbation noise for adversarial examples. Finally, rather than searching for a single solution (i.e., one individual), novelty search produces and maintains an archive of the most diverse individuals by replacing similar individuals with least similar candidates in the archive during each generation.

3 Methodology

This section describes our EXPOUND framework, a two-phase search-based technique used to generate diverse archives of adversarial examples that result in different failure categories of the DNN model. Figure 2 shows a Data Flow Diagram (DFD) for EXPOUND, where circles indicate process bubbles, arrows indicate data flow between processes, and parallel lines indicate external data stores. EXPOUND takes a non-perturbed input image and the black-box image classification algorithm as inputs. First, a developer provides the evolutionary parameters (e.g., mutation rate, population size, etc.), operating context, and behavior specification for EXPOUND. The operating context defines the range of perturbations that EXPOUND can search to discover different model behaviors (i.e., the genome range). For example, a developer may define the operating context to generate adversarial examples with non-minimal perturbation, restricted to 10% of a pixel's maximum value. The behavior specification defines the possible range of behaviors exhibited by the model (i.e., the phenome range), such as image classification labels. Next, we describe the steps of EXPOUND in turn.

Step 1. Exploration. During the *exploration* phase, EXPOUND uses a coarse-grained novelty search to identify the broad adversarial directions when exposed to perturbations. For discussion purposes, we use the term *failure category* in this work to describe the diverse collection of adversarial examples that lead to the same adversarial direction. Specifically, different incorrect model outputs (e.g., misclassifying a *dog* as a *cat* versus a *dog* as a *deer*) are considered different failure categories, as the model incorrectly classified the image in different ways. For

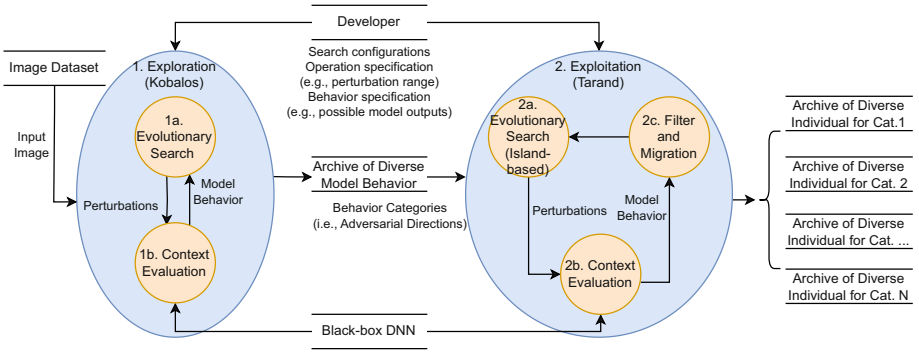

Fig. 2. EXPOUND's process to identify failure categories, each comprising diverse adversarial examples.

image classification problems, the number of potential failure categories increases based on the number of labels in the dataset. As such, it is computationally inefficient to exhaustively search every label to determine whether the model will misclassify an input image as the corresponding label when exposed to perturbations. Given an unperturbed input image and the DNN classifier, KOBALOS addresses this issue by exploring the search space and generating a collection of individuals that result in the most diverse model behaviors (i.e., most number of mutually exclusive class labels). The left portion of Fig. 3 provides a visual representation with colored regions denoting classification categories to illustrate KOBALOS's search on an input image of a *deer*. In this example, KOBALOS identifies two adversarial examples with different incorrect class labels, *dog* and *cat*, for the clean input image that has been exposed to perturbation. The bold arrows point towards incorrect class labels, indicating the failure categories identified by KOBALOS.

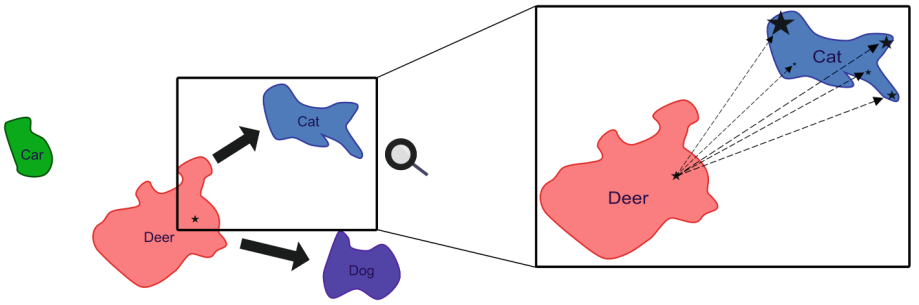

Fig. 3. High level interpretation of EXPOUND. Stars denote images. KOBALOS (left) discovers the failure categories while TARAND (right) exploits the information to discover diverse adversarial examples for each category identified.

Algorithm 1 shows the pseudo-code of KOBALOS to identify the broad failure categories using novelty search. KOBALOS largely follows a standard evolutionary algorithm approach in Step 1. First, a population of perturbation filters is randomly generated (i.e., filters are initialized with randomly sampled uniform noise). During each generation, individuals in the population are selected to reproduce, favoring those that result in phenome diversity. Specifically, the outputs of the DNN (i.e., vectors consisting of the label probabilities) are used as the phenomes. Parents create offspring using a single-point crossover operator and creep mutation operator. The creep mutation operator slightly increases or decreases the value of elements randomly sampled between a *mutation shift* variable with a *mutation rate* probability. Next, KOBALOS evaluates the individuals with the DNN to obtain the probability vectors (Step 1b in Fig. 2). Individuals are then ranked by computing a novelty score based on the *Euclidean distance* between phenomes with other individuals in the archive (line 13). Expression 1 shows the novelty score metric used in EXPOUND, computing the average distance between an individual (p) with the k-nearest neighbors in the archive (P).

$$\text{novelty}(\text{p}, \text{P}, k) = \text{mean}(\text{min-k}(\, \|\text{p} - \text{p}_i\|^2 \, \forall \text{p}_i \in \text{P} : \text{p}_i \neq \text{p}, k)) \tag{1}$$

The archive of individuals (i.e., the output of KOBALOS) is then updated to maintain the most diverse individuals (i.e., those with different labels) discovered by replacing individuals with low novelty scores with individuals with higher scores (line 15). KOBALOS uses phenome diversity (i.e., classification output labels) to strategically explore the search space and identify the probable types of faults, instead of using a brute-force approach to explore all possible labels.

Algorithm 1: Pseudo-code for Kobalos and Tarand [13,15].

```
1  function NOVELTY-SEARCH(n_generations, image):
2      archive ← ∅
3      popl ← RANDOM-POPULATION()            // Initialize population with random noise.
4      for ( n to n_generations ) {
5          popl ← SELECTION(popl)            // Select genomes via tournament selection.
6          popl ← RECOMBINATION(popl)          // Recombine via single-point crossover.
7          popl ← MUTATION(popl)                    // Mutation via creep mutation.
8          popl ← EVALUATE(popl)                  // Evaluate individuals using DNN.
9          archive, popl ← COMMIT-TO-ARCHIVE(archive, popl)
10         popl ← FILTER(popl)            // TARAND ONLY: filter and migrate individuals.
11     }
12     return archive
13 function COMMIT-TO-ARCHIVE(archive, popl):
14     scores ← RANK-INDIVIDUALS(archive, popl)        // Calc. novelty score (Eq.(1)).
15     archive ← TRIM(archive ∪ popl)          // Combine and truncate to desired size.
16     return archive, popl
```

Step 2. Exploitation. Once the failure categories have been identified in the exploration phase, EXPOUND uses TARAND for the *exploitation* phase in **Step 2**

to exploit the KOBALOS-discovered information. For each failure category identified in **Step 1**, TARAND uses novelty search to generate secondary archives of adversarial examples based on genome diversity (i.e., different combinations of noises that lead to the same misclassification). The archives of individuals (i.e., diverse types of faults) enable a more informed assessment of the types of perturbations within a given failure category. The right portion of Fig. 3 visually illustrates TARAND's search process to exploit the information identified by KOBALOS. Stars denote the original input image (i.e., deer) and adversarial examples generated by TARAND (e.g., an individual misclassified image). The relative size of individual stars contrast the respective confidence scores of the DNN for the image [21]. In this example, TARAND discovered a number of diverse adversarial examples that lead to the same *cat* misclassification. Adversarial examples discovered by TARAND in the archive have different types of perturbations, yet cause the same type of model misbehavior, denoted by thin dashed arrows. Understanding the nature and confidence scores for each failure category enables developers to focus their efforts to improve the robustness of the DNN (e.g., identify additional training data that addresses the misclassifications).

To discover a range of perturbations for each category of misclassification, TARAND maintains multiple islands of genomes. TARAND largely follows the same evolutionary process used by KOBALOS, but differs in the diversity metric. Specifically, KOBALOS optimizes a diversity metric based on the model's output (i.e., phenome defined in terms of image classification labels), while TARAND optimizes the diversity with respect to the perturbations that yield the same model output (i.e., genome). First, a population of perturbation filters are randomly generated using uniform sampling. During **Step 2a**, individuals are selected to evolve using single-point crossover and creep mutation operators based on their genome diversity. **Step 2b** evaluates the individuals in the population. The novelty score is computed using Expression 1, where P is based on the *genome* diversity of the population instead (i.e., TARAND promotes different perturbation filters). TARAND also includes an additional step, **Step 2c**, where individuals on an island that exhibit behavior different from other individuals on the same island are moved to the island with corresponding behavior (blue line 10 in Algorithm 1). For example, an adversarial example with the label *dog* on an island of *cats* will be filtered and migrated to the island containing *dogs*.

Misclassification Score. As novelty search promotes diversity, adversarial examples generated by TARAND can provide developers with insight on the boundaries of errors and confidence scores for the misclassifications in each failure category. For example, a developer may want to identify and prioritize categories that are confidently misclassified. Confidently misclassified inputs may pose a higher risk to safety-critical systems, as the system may mistakenly trust the model's output with higher weight based on the confidence score. Identifying confidently misclassified categories enables developers to include additional training data that may help the model distinguish between the two types of images. In this work, we first calculate and sort the archive of adversarial examples by confidence scores using the output of the model. We define the misclassification scores

for each category using the average confidence score of the top 50% of sorted adversarial examples (i.e., archive[0.5 : 1].getConfidenceScores()), denoted by Expression (2). Specifically, categories that contain several confidently misclassified images will be assigned a higher score, close to 1.0.

$$\text{MisclassificationScore}(\text{archive}) = \text{mean}(\text{archive}[0.5 : 1].\text{getConfidenceScores}()) \tag{2}$$

4 Validation Studies

This section demonstrates the Expound framework using an empirical evaluation. We used two image classification DNNs with different architectures on two different image benchmark datasets to conduct our experiments. In order to assess whether Expound can identify more failure categories for the DNN models, we compared Kobalos with a Monte Carlo sampling method. Then, we compared the ability of Tarand and Monte Carlo sampling to generate archives of diverse adversarial examples for each failure category. This section addresses the following research questions:

RQ1: Can Expound identify distinct failure categories using Kobalos?
RQ2: Can Expound identify more faults using Tarand for each failure category?

4.1 Experimental Setup

This section describes the experimental setup and evolutionary parameters used in our validation experiments. For our experiments, we used existing image classification algorithms implemented using the Pytorch deep learning research platform [18]. In order to illustrate that our approach is model and data-agnostic, we use two different benchmark datasets for our validation work. First, the CIFAR-10 benchmark dataset is commonly used for image classification algorithms [11], comprising ten evenly distributed categories. For the CIFAR-10 dataset, we used a ResNet-20 model with a base accuracy of 91.58% and a MobileNet-V2 model with a base accuracy of 91.52%. Second, we also use the GTSRB dataset [23]. Unlike the CIFAR-10 dataset, the GTSRB dataset consists of 43 classes of traffic signs with an uneven distribution of labels. For the GTSRB dataset, we used a ResNet-20 model with a base accuracy of 96.28% and a MobileNet-V2 model with a base accuracy of 90.77%.

In order to provide a baseline evaluation for our framework, we implemented a Monte Carlo sampling approach (i.e., randomly sample noise). Each approach generates an archive size based on the total number of labels in the dataset (i.e., 10 for CIFAR-10 and 43 for GTSRB). Table 1 shows the evolutionary parameters used in Expound, where the parameters are based on empirical studies. The maximum perturbation for each approach is set to $[-0.1, +0.1]$ of the maximum pixel value. All experiments are conducted on a NVIDIA GeForce GTX 1080

GPU with an Intel Core i7-7700K CPU on Ubuntu 22.04. Finally, each experiment is repeated ten times using different seeds. We show the average value of the experiments, their standard deviations, and provide statistical analysis to ensure the feasibility of our approach. For the results, we verify their statistical significance using the Mann-Whitney U test.

Table 1. Configuration settings for EXPOUND

	CIFAR10		GTSRB	
Parameter	Value (Kobalos)	Value (Tarand)	Value (Kobalos)	Value (Tarand)
Archive Size	10	10	43	10
Population Size	10	10	10	43
Num Generations	100	500	100	500
Mutation Rate	0.15	0.15	0.15	0.15
Mutation Shift	0.2	0.2	0.2	0.2

4.2 Coarse-Grained Search for Distinct Failure Categories (RQ1)

To address our first research question, we evaluate KOBALOS's ability to discover distinct failure categories for DNN models by comparing the results with Monte Carlo sampling. We define our null and alternate hypotheses as follows:

$RQ1 - H_0$: KOBALOS does not identify more failure categories.
$RQ1 - H_1$: KOBALOS identifies more failure categories.

For each experiment and dataset, 30 randomly-sampled input images were used. For each image, a collection of 10 individuals is generated for the CIFAR-10 dataset while a collection of 43 individuals is generated for the GTSRB dataset, based on the number of possible labels for each dataset.

In order to ensure the feasibility of our approach, results are shown as the mean over ten trials to account for statistical variance. Table 2 shows the average number of unique labels identified by each approach when used to discover failure categories. The standard deviation is denoted in parenthesis, while the best performing technique is indicated in bold. The p-value for each experiment is shown in the last column. For each combination of DNN architectures and datasets, EXPOUND consistently discovered more failure categories than Monte Carlo sampling (on average 80% more for CIFAR-10 and 270% more for GTSRB). We found strong evidence ($p \leq 0.01$) to reject H_0 and support H_1. These results indicate EXPOUND can consistently discover more distinct failure categories.

Fig. 4 shows an example output of EXPOUND's exploration phase for an image randomly sampled in the CIFAR-10 dataset against the ResNet-20 model. The top image shows the clean input image and the model's correct classification output. Next, the collection of noise filters generated by KOBALOS (left) and

Table 2. Results for Monte Carlo sampling and EXPOUND's search for failure categories in the DNN's output. EXPOUND consistently identified more failure categories.

DNN and Dataset	Number of Average Unique Labels		p-value significance
	Monte Carlo	KOBALOS	
ResNet-20 - CIFAR-10	2.02 ($\sigma = 0.10$)	**3.62** ($\sigma = 0.17$)	< 0.01
MobileNet-V2 - CIFAR-10	1.83 ($\sigma = 0.09$)	**3.33** ($\sigma = 0.13$)	< 0.01
ResNet-20 - GTSRB	1.94 ($\sigma = 0.08$)	**7.11** ($\sigma = 0.19$)	< 0.01
MobileNet-V2 - GTSRB	1.73 ($\sigma = 0.09$)	**6.45** ($\sigma = 0.09$)	< 0.01

the corresponding adversarial examples (right) are shown in the second row. As indicated by the number of misclassifications, adversarial examples generated by KOBALOS induce different distinct misbehaviors in the model's output.

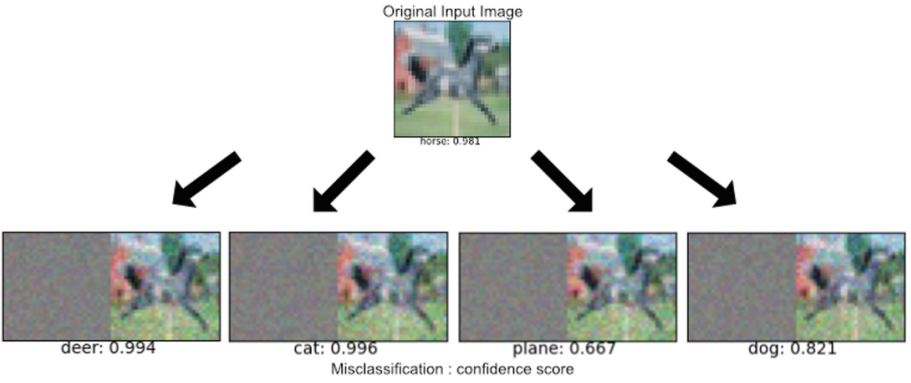

Fig. 4. Example output of KOBALOS on an input image of a *horse*. KOBALOS discovered four different incorrect model behaviors when exposed to perturbations.

4.3 Fine-Grained Search for Diverse Adversarial Examples (RQ2)

In this experiment, we demonstrate that TARAND can generate a diverse archive of adversarial examples for each failure category identified by KOBALOS. Specifically, we illustrate the ability of TARAND to exploit the information from KOBALOS and generate an archive of ten individuals for each category of misbehavior identified. For example, if six failure categories have been identified by KOBALOS, then TARAND attempts to generate six secondary archives of adversarial examples, one for each failure category. We compare the results of TARAND with Monte Carlo sampling. Monte Carlo sampling generates adversarial examples equal to the number of adversarial examples generated by TARAND. The adversarial examples generated by Monte Carlo sampling are then sorted into their corresponding failure categories. We define our null and alternate hypotheses to answer this question as follows:

$RQ2 - H_0$: TARAND does not identify more faults for each failure category.
$RQ2 - H_1$: TARAND identifies more faults for each failure category.

Table 3 shows the experimental results for the fine-grained exploitation search. The table shows how well a given technique under study is able to generate distinct and diverse adversarial examples, each resulting in similar misclassification labels. For example, a full archive (i.e., 100%) for an image from the CIFAR-10 dataset denotes that 10 adversarial examples were discovered for each failure category identified. The standard deviation is shown in parenthesis. The final column of the table shows the corresponding p-value using the Mann-Whitney U test. When applied to the same set of thirty images, TARAND is able to successfully generate a (relatively full) archive of adversarial examples for most failure categories. In contrast, Monte Carlo sampling consistently did not generate adversarial examples for the failure categories. For each experiment, we found strong evidence ($p \leq 0.01$) to reject H_0 and support H_1. The results indicate that TARAND can generate more and different perturbations for a given failure category, a new capability not offered by existing approaches.

Table 3. Comparing TARAND's generation of diverse archives of adversarial examples for the categories of misbehavior against Monte Carlo sampling.

DNN and Dataset	Percentage of Archives Filled		p-value significance
	Monte Carlo	TARAND	
ResNet-20 - CIFAR-10	26.36% ($\sigma = 2.69\%$)	**93.73%** ($\sigma = 1.35\%$)	< 0.01
MobileNet-V2 - CIFAR-10	33.43% ($\sigma = 1.09\%$)	**91.59%** ($\sigma = 1.20\%$)	< 0.01
ResNet-20 - GTSRB	14.48% ($\sigma = 1.05\%$)	**81.16%** ($\sigma = 1.09\%$)	< 0.01
MobileNet-V2 - GTSRB	11.12% ($\sigma = 0.95\%$)	**71.93%** ($\sigma = 1.43\%$)	< 0.01

The archives of adversarial examples generated by TARAND can be used to assess the model's robustness towards each of the incorrect labels identified by EXPOUND. Specifically, each archive of adversarial examples generated by TARAND promotes genome diversity that produces the same model behavior failure. Thus, we can identify relevant and confidently misclassified failure categories based on the confidence scores of the adversarial examples. We calculate the misclassification score based on Expression (2) defined in Sect. 3. Figure 5 shows a sample output of TARAND randomly sampled from the CIFAR-10 dataset. The ground truth for the clean input image is *truck*. Each row denotes a failure category identified by EXPOUND. Individual scores show the confidence scores for the diverse archive of adversarial examples generated by TARAND. Finally, the misclassification score of each category is computed and shown in parenthesis next to the class label. The failure categories are sorted based on the misclassification score, in descending order. As the results indicate, adversarial examples with the labels *bird* and *ship* are not commonly misclassified with high confidence scores (i.e., less than 0.7). However, the model confidently misclassifies adversarial examples as *frogs*. Thus, developers may use this information to include additional training samples that distinguish frogs from trucks.

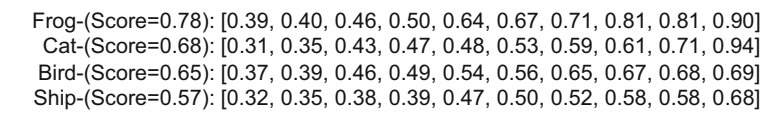

Fig. 5. An analysis of adversarial examples generated with TARAND for an image of a *truck*. The (sorted) misclassification score is shown in parenthesis. The numbers in brackets show the confidence scores of individual adversarial examples.

4.4 Threats to Validity

The results in this paper show that a diverse collection of adversarial examples can be generated against DNNs using novelty search. The results of the experiment may vary with repeated executions, as novelty search and evolutionary search-based algorithms rely on randomness. To ensure the feasibility of our approach, a wide range of sample of input images were sampled from different datasets. Additionally, each experiment was repeated ten times with different randomly-generated seeds to ensure comparable results.

5 Related Work

Early work with adversarial examples was generated using white-box approaches [24]. These approaches leverage gradient information of the model to be attacked and introduce noise to hinder the model's capability. Szegedy *et al.* [19] first introduced adversarial examples generated using the L-BFGS algorithm. Carlini and Wagner [4] proposed a similar approach with a modified objective function and constraint, known as the C&W attack. Goodfellow *et al.* [9] introduced the Fast Gradient Sign Method (FGSM) that perturbs the target image based on the signed gradient at each step. However, these techniques are all white-box, where gradient information and other model parameters are assumed to be accessible.

A number of researchers have also explored black-box evolutionary search-based approaches to generate adversarial examples. Compared to white-box approaches, black-box approaches closely resemble a real attack scenario, where the development of the model is proprietary. Papernot *et al.* [20] demonstrated that white-box adversarial attacks can transfer to other models. In contrast, Alzantot *et al.* [17], Vidnerová *et al.* [26], Chen *et al.* [6] proposed various genetic algorithm approaches to generate adversarial examples against image classification algorithms. Chan and Cheng proposed EvoAttack [5] that showed adversarial attacks also apply to object detection algorithms, an important component for perception sensing in autonomous vehicles. However, these approaches generate adversarial examples that do not reveal diverse model behaviors.

Several existing work has explored the use of diversity for DNNs. Rozsa *et al.* [21] introduced the use of a diverse set of adversarial examples to evaluate and retrain models. However, their approach generates adversarial examples using a white-box approach and does not discover failure categories. Aghababaeyan *et al.*

[1] proposed the use of the geometric diversity metric to identify a diverse set of test cases from the original input dataset based on their similarity to other samples. Langford and Cheng [13,14] proposed several novelty search techniques to explore the effect of environmental uncertainties on DNNs. However, their approach does not address the robustness of the DNN against noise perturbation or adversarial examples. In their work, Enki [13] is used to generate environmental uncertainties against a DNN model applied over a dataset. The identified environmental uncertainties demonstrate the inability of the DNNs to process the occluded inputs, including those that prevent human classification. They also proposed Enlil [14], a novelty search framework to discover operating contexts that can lead to different categories of performance degradation. To the best of our knowledge, EXPOUND is the first black-box approach to explore the use of novelty search to generate diverse adversarial examples that can discover a number of distinct failure categories, providing both breadth and depth-based information about the types of perturbations that cause model misbehaviors.

6 Conclusion

This paper proposed EXPOUND, a novelty search-based framework to generate diverse adversarial examples. We showed that our two-phase framework first identifies failure categories for image classification algorithms that would otherwise not be discovered by existing approaches. Furthermore, EXPOUND then generates secondary archives of diverse adversarial examples for each failure category identified, enabling the assessment for the misclassification score of each category. We conducted a series of experiments to show that our approach can successfully generate adversarial examples against different datasets and models without access to model weights, architectures, and gradient information.

Future work will explore additional datasets and models for validation results. Additional studies may be performed to explore whether noise generated from EXPOUND transfers to other images. Future studies may also explore whether novelty search might be used to discover if "universal" adversarial perturbations exist and can be used to improve a DNN's robustness. Additional work may explore the ability of EXPOUND to generate diverse adversarial examples against models with defense mechanisms [12]. Finally, we will investigate whether EXPOUND can be applied to other types of machine learning algorithms, such as speech analysis, natural language processing, or object detection algorithms.

Acknowledgements. We greatly appreciate Michael Austin Langford's contributions on our preliminary work. We also greatly appreciate the insightful and detailed feedback from the reviewers. This work was supported in part by funding provided by Michigan State University, the BEACON Center, the Air Force Research Laboratory, and our industrial collaborators.

References

1. Aghababaeyan, Z., Abdellatif, M., Dadkhah, M., Briand, L.: DeepGD: a multi-objective black-box test selection approach for deep neural networks. arXiv (2023)
2. Arrieta, B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion **58**, 82–115 (2020). https://doi.org/10.1016/j.inffus.2019.12.012
3. Cai, Z., Fan, Q., Feris, R.S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 354–370. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_22
4. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy (SP), pp. 39–57. IEEE (2017)
5. Chan, K., Cheng, B.H.C.: EvoAttack: an evolutionary search-based adversarial attack for object detection models. In: Papadakis, M., Vergilio, S.R. (eds.) SSBSE 2022. LNCS, vol. 13711, pp. 83–97. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-21251-2_6
6. Chen, J., et al.: POBA-GA: perturbation optimized black-box adversarial attacks via genetic algorithm. Comput. Secur. **85**, 89–106 (2019)
7. Črepinšek, M., Liu, S.H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput. Surv. (CSUR) **45**(3), 1–33 (2013)
8. Gheibi, O., Weyns, D., Quin, F.: Applying machine learning in self-adaptive systems: a systematic literature review. ACM TAAS **15**(3), 1–37 (2021)
9. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (2015)
10. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
11. Krizhevsky, A., et al.: Learning multiple layers of features from tiny images (2009)
12. Kurakin, A., et al.: Adversarial attacks and defences competition. In: Escalera, S., Weimer, M. (eds.) The NIPS '17 Competition: Building Intelligent Systems. TSSCML, pp. 195–231. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94042-7_11
13. Langford, M.A., Cheng, B.H.C.: Enki: a diversity-driven approach to test and train robust learning-enabled systems. ACM TAAS **15**(2), 1–32 (2021)
14. Langford, M.A., Cheng, B.H.C.: "Know what you know": predicting behavior for learning-enabled systems when facing uncertainty. In: 2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems, pp. 78–89. IEEE (2021)
15. Lehman, J., Stanley, K.O.: Abandoning objectives: evolution through the search for novelty alone. Evol. Comput. **19**(2), 189–223 (2011)
16. Lehman, J., Stanley, K.O.: Novelty search and the problem with objectives. In: Riolo, R., Vladislavleva, E., Moore, J. (eds.) Genetic Programming Theory and Practice IX. Genetic and Evolutionary Computation, pp. 37–56. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-1770-5_3
17. Alzantot et al.: GenAttack: practical black-box attacks with gradient-free optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1111–1119 (2019)
18. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Wallach, H., et al. (eds.) Advances in Neural Information Processing Systems vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019)

19. Szegedy et al.: Intriguing properties of neural networks. In: International Conference on Learning Representations (2014)
20. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv (2016)
21. Rozsa, A., et al.: Adversarial diversity and hard positive generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
22. Sandler, M., et al.: Mobilenetv 2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
23. Stallkamp, J., et al.: The GTSRB: a multi-class classification competition. In: The 2011 International Joint Conference on Neural Networks, pp. 1453–1460. IEEE (2011)
24. Sun, L., et al.: A survey of practical adversarial example attacks. Cybersecurity **1**, 1–9 (2018)
25. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
26. Vidnerová, P., Neruda, R.: Vulnerability of classifiers to evolutionary generated adversarial examples. Neural Netw. **127**, 168–181 (2020)
27. Wallace, E., et al.: Trick me if you can: human-in-the-loop generation of adversarial examples for question answering. TACL **7**, 387–401 (2019)
28. Yu, F., et al.: Interpreting and evaluating neural network robustness (2019). https://doi.org/10.24963/ijcai.2019/583. (IJCAI 2019)