Harnessing Adversarial Distances to Discover High-Confidence Errors

Walter Bennette
Information Directorate
Air Force Research Lab
Rome, NY
walter.bennette.1@us.af.mil

Karsten Maurer
Department of Statistics
Miami University
Oxford, OH
maurerkt@miamioh.edu

Sean Sisti
Information Directorate
Air Force Research Lab
Rome, NY
sean.sisti@us.af.mil

***Abstract*—Given a deep neural network image classification model that we treat as a black box, and an unlabeled evaluation dataset, we develop an efficient strategy by which the classifier can be evaluated. Randomly sampling and labeling instances from an unlabeled evaluation dataset allows traditional performance measures like accuracy, precision, and recall to be estimated. However, random sampling may miss rare errors for which the model is highly confident in its prediction, but wrong. These high-confidence errors can represent costly mistakes, and therefore should be explicitly searched for. Past works have developed search techniques to find classification errors above a specified confidence threshold, but ignore the fact that errors should be expected at confidence levels anywhere below 100%. In this work, we investigate the problem of finding errors at rates greater than expected given model confidence. Additionally, we propose a query-efficient and novel search technique that is guided by adversarial perturbations to find these mistakes in black box models. Through rigorous empirical experimentation, we demonstrate that our Adversarial Distance search discovers high-confidence errors at a rate greater than expected given model confidence.**

***Index Terms*—Deep learning, Computer vision, Classification, Evaluation strategies**

I. Introduction

Given a deep neural network image classification model that we treat as a black box, and an unlabeled evaluation dataset, it is necessary to have an efficient strategy to evaluate the classifier. For example, if a physician is teamed with some black-box diagnostic tool, it would be prudent for the physician to evaluate the tool before utilizing it in practice. A desirable evaluation procedure should be respectful of the physician's time and effort, but help reveal the strengths and weaknesses of the model.

One strategy to evaluate a black box model with an unlabeled evaluation dataset is to randomly sample and label instances from the dataset, and estimate traditional performance measures like accuracy, precision, and recall. Another strategy is to sample low confidence predictions to discover areas where the model is prone to error. However, these strategies may miss errors for which the model is highly confident in its prediction, but wrong. These high-confidence errors can represent costly mistakes (e.g misdiagnosis), and therefore should be explicitly searched for.

In this paper, we propose a novel and query-efficient approach for guiding a human, or oracle, to high-confidence classification errors made by black box image classification models. Specifically, we propose a search that leverages small perturbations to an image to help identify instances within an unlabeled evaluation dataset for which the classification model has high confidence in its prediction, but is wrong. These perturbations are similar to those of recent developments in adversarial images. Special attention is devoted to ensure that the developed technique is applicable to black box classifiers where specifics of the model's training data and architecture may be unknown.

High-confidence errors can be interpreted as blind spots to a classification model [1]. These high-confidence errors can be caused by dataset shift during use [2], dataset bias during training [3], overfitting, and other reasons for poor model performance. For example, [4] describes a classification model learned from a biased dataset of dogs with dark fur and cats with light fur. When used for inference, this model is highly confident that dogs with light fur belong to the cat class. Discovering that dogs with light fur can be misclassified with high confidence reveals a weakness of the classifier.

Previous efforts have designed search techniques to help discover high-confidence errors in an unlabeled evaluation set by searching for errors above a confidence threshold, τ (typically set to 0.65 for binary classification) [4], [5]. Unfortunately, these techniques ignore the logical expectation that some amount of error is expected to occur at a confidence level less than 100%. Meaning, 30% of the predictions made with 70% confidence should be errors, 20% of the predictions made with 80% confidence should be errors, and so on. Therefore, existing methods may simply discover errors by chance, not by some sophisticated search procedure that leverages commonalities between errors to increase the rate of error discovery. Instead, in this work we consider the problem of finding errors at rates greater than expected, to encourage search methods that discover something about a model's weaknesses to increase the rate of error discovery.

Contributions of this work are summarized as follows:

1) We define the problem of finding errors within an unlabeled evaluation dataset at rates greater than what model confidence would suggest.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

2) We design a novel error search that utilizes adversarial perturbations to improve the chance of discovering prediction errors.
3) We empirically demonstrate that our novel search procedure finds errors at a rate greater than the rate suggested by model confidence.

The remainder of this paper is organized as follows: In Section II we discuss existing methods used to search for high-confidence errors, and provide background on adversarial images. In Section III we formulate the problem of discovering errors at a rate that exceeds expectation given model confidence. Then, in Section IV, we introduce a novel method to search for errors that leverages adversarial perturbations to glean extra information about model confidence. Next, in Section V and VI, we present experimental results and provide a discussion. Finally, in Section VII, we conclude and provide thoughts for future research.

II. Related Work

In this section we discuss existing methods to search for high-confidence errors from black box classification models within an unlabeled evaluation dataset. We review adversarial images and their relation to our proposed search technique. We also briefly discuss model calibration.

A. High-Confidence Errors

Attenberg (2015) [1] introduced the concept of searching for high-confidence errors in relation to machine learning classification models. Here, high-confidence errors were defined to be predictions for which a classification model was highly confident, but wrong. Works considering the search for high-confidence errors [1], [5], [6] all follow a general structure: 1) define a utility function to describe a search's value, and 2) develop a search method to help maximize the defined utility function.

In Attenberg (2015) [1], the objective was to motivate human users to find high-confidence errors. The defined utility was a monetary value that would be paid for every high-confidence mistake that was found. The search method was to allow the human searcher to query the model when they discovered instances they felt the model may incorrectly classify. As a result, the human searcher developed their own search technique to try and "Beat the Machine". Although relevant to the initial formulation of the problem, recent papers focus on algorithmic approaches to help guide an oracle to the discovery of high-confidence errors.

The first algorithmic approach to search for high-confidence errors, within an unlabeled evaluation dataset, was introduced by Lakkaraju (2017) [4]. This human-in-the-loop search defined a utility function that gave a uniform value for each discovered high-confidence error and discounted this value by the cost of the human, or oracle, to label a sampled instance (regardless if it was a high-confidence error or not). However, the utility function is simplified for imagery as it places uniform cost for each call to the oracle. A multi-armed bandit algorithm was then used to search through clusters in a derived feature space to find high-confidence errors. The search is driven by tracking the average utility of a cluster, which can be viewed as the likelihood of finding a high-confidence error in that cluster.

Bansal and Weld (2018) [5] defined a utility function to encourage the high-confidence error search to be spread throughout a derived feature space. Given an unlabeled evaluation dataset, X, where c_x is the confidence of a model's prediction for $x \in X$, $Q \subseteq X$ is a query set of instances to evaluate for correctness, and $Cover(x|Q)$ is a function to calculate how much an instance x is covered by an error found in the query set Q, the utility function is then:

$$U = \sum_{x \in X} c_x * Cover(x|Q). \tag{1}$$

This utility function rewards the discovery of errors that "cover" the evaluation dataset. Here, coverage of an instance is a function of its distance to the nearest error found in the query set, where closer points yield larger values. Note that the utility function does not directly reward the discovery of high-confidence errors, but rather rewards finding errors that are near high-confidence points. A greedy search was then used to search through clusters of the derived feature space where the probability of each cluster containing an error was tracked. Full details can be found in [5].

Maurer and Bennette (2019) [7] present an extension to [4] and [5] that identifies the flaw of valuing error discovery at the rate expected given model confidence. Meaning, the work identifies the fact that errors should be expected for confidence levels below 100%. The Standardized Discovery Ratio is introduced as a new measure of search performance, and compares the actual number of discovered errors to the expected number of errors given the confidence of the model's predictions. This measure is discussed in much greater detail in Section III.

B. Adversarial Images

In Convolutional Neural Networks (CNNs) an adversarial image is formed by inserting small targeted perturbations to an original image such that it is confidently misclassified by the model [8]. The difference between the adversarial example and the original input is often indistinguishable to the human eye, but is still successful at fooling the classifier. Many methods exist to create adversarial images, and they can be split into two main classes: model-based and decision-based.

Model-based adversarial attacks leverage knowledge of the model's weights and architecture. For example, the fast gradient sign method [8] relies on gradient information to create targeted perturbations to be added to the original image. Although effective, model-based methods require model information that may not always be available with a black-box classifier.

Decision-based adversarial attacks require no knowledge of the model's weights and architecture. Instead, they only require access to the model to predict labels for new images [9], [10]. Of particular interest is the Boundary Attack [9]

which begins with a large adversarial perturbation and iteratively reduces the amount of perturbation while still remaining misclassified. More specifically, the attack begins with an adversarial image (perhaps created through the injection of Gaussian noise) and then performs a series of steps in random directions to reduce the size of the perturbations. Each orthogonal step is adjusted to move along the decision boundary towards the original input, with the intent to find the minimal distance between the perturbation and the original input while still being misclassified.

Most research of adversarial images has been devoted to creating adversarial attacks or defending against adversarial attacks. However, Stock and Cisse (2017) [3] leveraged adversarial images to identify model prototypes and criticisms to help expose classifier biases. For example, they discovered a bias in a classifier that confidently identified street lights set against a blue sky as traffic lights. This was done by looking at model criticisms, or, images that required the least amount of perturbation to turn the image adversarial. Additionally, Ilyas (2019) [11] showed that image classification models discriminate instances through features that are robust and through features that are non-robust. Robust features are highly predictive and related to the classification task as perceived by humans. Non-robust features can also be highly predictive, but do not necessarily pertain to the human perceived classification task (in the example above, perhaps the presence of features related to the sky). Ilyas (2019) [11] also showed that non-robust features are more susceptible to an adversarial attack. Meaning, a classification decision based on a non-robust feature may require less perturbation to change the prediction. These two works hint that there may be a discrepancy between a classifier's prediction confidence and the amount of perturbation required to turn an image adversarial, and could be leveraged to help discover prediction errors. This is explored further in Section IV.

C. Model Calibration

Guo (2017) [12] found that modern neural networks are poorly calibrated. Meaning, the maximum value of the softmax layer, often taken as the confidence of the classifier's prediction, does not represent a true probability of correctness. As stated in Guo (2017), given a model M, with $M(X) = (\hat{Y}, \hat{P})$, where X are model inputs with true labels Y, $\hat{Y}$ are model predictions, and $\hat{P}$ are model confidences, a perfectly calibrated model satisfies the following:

$$\mathbb{P}\left(\hat{Y} = Y | \hat{P} = p\right) = p, \quad \forall p \in [0, 1]. \tag{2}$$

Meaning, for a well calibrated model, the probability that a prediction made with p confidence is correct, is equivalent to the reported confidence, p. Although in practical settings perfect model calibration cannot be achieved, [12] showed that temperature scaling can be used to adequately calibrate neural network models with a validation dataset.

Therefore, all of the classifiers used in our study have been calibrated using temperature scaling on a validation dataset. The intention of this step is to ensure that the discovered overconfident errors are not an artifact of poor model calibration, but systematic errors made by the classifier for the unlabeled evaluation dataset.

III. Problem Formulation

Lakkaraju (2017) [4] defined a utility function that valued the discovery of high-confidence errors uniformly. Bansal and Weld (2018) [5] defined a utility function that weighted the discovery of a high-confidence error by the amount of the dataset it helps "explain", or the "coverage" of the error. This was done to discourage the search method from sampling a rich pocket of errors and ignoring the rest of the search space. Additionally, both of these formulations defined a high-confidence error to be a classification error above a prediction confidence threshold τ, where τ is set to 65% for binary classification.

Unfortunately, these prior formulations ignore the reasonable expectation that prediction errors should occur at confidence levels anywhere below 100%. Meaning, the existing search methods may simply discover errors by chance, not from some sophisticated search procedure that leverages commonalities between errors to increase the rate of error discovery. More specifically, we argue that discovering errors at the expected rate is no more informative about the weaknesses of the model than random search, because it could be expected to find the same number of errors by randomly sampling predictions from a defined confidence range. Instead, we should explicitly encourage the discovery of errors at rates that exceed expectations to promote search methods that uncover model weaknesses to increase their rate of discovery.

We consider the problem of discovering high-confidence errors at rates greater than what a model's confidence would suggest, which was recently introduced by [7]. Given a black-box classifier, M, with $M(x) = (\hat{y}_x, \hat{p}_x)$, where x is an instance from an unlabeled evaluation set X, $\hat{y}_x$ is the model's prediction, $\hat{p}_x$ is the model's confidence, and y_x is the true label assigned by some oracle, the task is to find a query set of data points, $Q \subseteq X$, that maximize the Standardized Discovery Ratio (SDR). Here SDR is defined as:

$$SDR = \frac{\sum_{q \in Q} \mathbb{1}(\hat{y}_q \neq y_q)}{\sum_{q \in Q} (1 - \hat{p}_q)}, \tag{3}$$

where $|Q| = B$, $\hat{p}_q > 0.65$ for $q \in Q$, and B represents the labeling budget of the oracle used to find true labels. The SDR can then be interpreted as the number of discovered misclassifications relative to what would be expected given the confidence of the predictions.

In this formulation, a query set that leads to an SDR of one indicates that errors were discovered at the rate expected given the confidence of the predictions. Values greater than one indicate that errors were discovered at a rate greater than expected. While previously developed methods do not explicitly value this type of error discovery, it is still possible that errors are discovered at rates that exceed expectation.

However, this formulation allows us to explicitly value this higher rate of discovery, and maximize the value of the oracle's search.

Additionally, Theorem 3.1 shows that a query set for a model who's SDR has an expected value greater than one is an indicator of model overconfidence. However, a query set obtained through some search procedure with an SDR greater than one does not prove the existence of model overconfidence, because the i.i.d assumption of the proof is almost certainly violated. Still, an SDR greater than one does suggest the presence of model overconfidence, and the discovered errors may provide insight to particular weaknesses.

Theorem 3.1: Suppose there exists a sufficiently large query set sampled i.i.d. from an unlabeled evaluation dataset. If the expected SDR is greater than one then there exists some level of model confidence where the probability of a correct prediction is less than the model confidence.

Proof by contradiction: Assume the probability of making a correct prediction at a specific model confidence is always greater than or equal to the model confidence.

The expected value of the SDR can be calculated as:

$$E[SDR] = \frac{E\left[\sum_{q \in Q} \mathbb{1}(\hat{y}_q \neq y_q)\right]}{E\left[\sum_{q \in Q} (1 - \hat{p}_q)\right]}.$$

The expected number of errors in the numerator can be substituted with the expectation of the true probability of error, simplifying to:

$$E[SDR] = \frac{\sum_{q \in Q} E\left[(1 - p_q)\right]}{\sum_{q \in Q} E\left[(1 - \hat{p}_q)\right]} \leq 1.$$

Because of our assumption we know the numerator is smaller than or equal to the denominator implying the expected SDR must be less than or equal to one. Therefore, if the expected SDR is greater than one there must exist some model confidence that is greater than the probability of a correct prediction at that confidence level, and the model is overconfident. ■

IV. Adversarial Distance Search

This section introduces a methodology to search for high-confidence errors that utilizes an Adversarial Distance measure to guide the search.

A. Adversarial Distance

A classifier's prediction for a specific image can be changed by strategically perturbing the pixels of that image until the classifier assigns it a different label. These perturbations result in an adversarial image if the image has been minimally changed such that a human can not identify the difference. In our work we call an image adversarial if it has been perturbed and changes the classifier's original prediction, even if the new prediction matches the image's true label. Additionally, in our work, no human check is performed to verify that the image has been minimally changed.

Given the work in [11], we believe deep neural network models may erroneously base some of their high-confidence predictions on non-robust features. Using the observation of [3] that some images are easier to turn adversarial than others, and the observation of [11] that non-robust features are easily broken by adversarial attacks, we present a measure to help identify predictions that are more susceptible to adversarial attack. Or alternatively, a measure to identify instances for which the model is potentially overly confident in its prediction due to the presence of non-robust features. If these types of predictions are indeed more prone to error than what is suggested by the confidence of the model, selecting them for a query set would result in an SDR greater than one and reveal something about model weaknesses.

We introduce the term Adversarial Distance to describe how much perturbation an image needs for the classifier to change its prediction in comparison to the expected amount of perturbation, as determined by predictions of similar confidence. To begin, Mean Absolute Error (MAE) can be used to measure the mean pixel-wise difference between an adversarial image and the original image. MAE can be calculated for two NxM images called a and b as:

$$MAE(a, b) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} |a_{(i,j)} - b_{(i,j)}| \tag{4}$$

Adversarial Distance is then defined to be the difference between an image's observed MAE and expected MAE, based on confidence.

$$AdvDist(x) = MAE\left(x, A(M, x)\right) - F\left(\hat{p}_x\right), \tag{5}$$

where x is the image for which we are calculating the Adversarial Distance, M is the classifier being evaluated, $A(M, x)$ is a mechanism to alter x such that M's prediction is changed, and $F\left(\hat{p}_x\right)$ is a function to calculate the expected MAE based on the classifier's predictive confidence, $\hat{p}_x$, for the instance, x.

In our work, the mechanism A, used to create an adversarial image is the decision-based Boundary Attack [9] discussed in Section II. As a reminder, we call an image adversarial if the perturbation of the image changes the classifier's original prediction. The intuition behind the attack, as described by Brendel (2017) [9] and repeated here, is that the algorithm begins with an image that is already adversarial (perhaps through the addition of Gaussian noise) and performs a random walk to "follow the boundary between the adversarial and the non-adversarial region such that (1) it stays in the adversarial region and (2) the distance towards the target image is reduced" [9]. The Boundary Attack was selected to create adversarial images because it finds progressively smaller perturbations to make the image adversarial, and because it is a decision-based attack that requires no model information.

F should provide an expected MAE given the confidence of a prediction. Here, we calculate the MAE for every item in the evaluation set and fit a LOESS [13] regression line to estimate F. MAE is the dependent variable for this LOESS line, and the classifier confidence score is the independent variable.

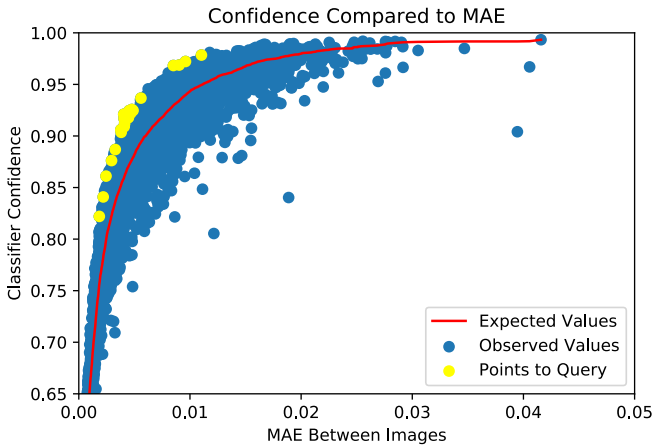

Figure 1: Example LOESS curve fit between log-MAE and classifier confidence. The horizontal distance of points from the fitted line represents the Adversarial Distance, thus the yellow points fall farthest to the left of the fitted LOESS line and have the smallest Adversarial Distances. The yellow instances would be used to query the oracle and search for errors.

Figure 1 shows an example LOESS for this application (built using the Kaggle13 dataset described later). Note that the horizontal distance of points from the fitted line represents the Adversarial Distance, thus the points falling farthest to the left of the fitted LOESS line will have the smallest Adversarial Distances. Additionally, note that calculating Adversarial Distance is completely unsupervised because the true labels of the images are not needed.

B. Search

Once the Adversarial Distance has been calculated for every instance in the evaluation set, the search for high-confidence mistakes is easily defined. Intuitively, the search queries an oracle to label the instances with the lowest Adversarial Distance. In Figure 1, these instances are colored yellow. Algorithm 1 defines the search in detail.

Algorithm 1 Adversarial Distance Search

Input: Evaluation set $\mathbb{X}$, budget B, and classifier M
$Q = \{\}$, instances that have been queried
$S = \{\}$, misclassified instances
For: $b = 1, 2, ..., B$ **do:**
  $q = \text{argmin}_{x \in \mathbb{X} \text{ and } x \notin Q} \, AdvDist(x)$
  $y_q = OracleQuery(q)$
  $Q \leftarrow Q \cup q$
  If: $y_q \neq M(q)$ **:** $S \leftarrow S \cup q$
Return: Q **and** S

Algorithm 1 operates by placing the image with the lowest Adversarial Distance not already in the query set, Q, into the query set. The oracle then labels the image and if the label does not match the model prediction the instance is added to the set S. Once the oracle has been queried B times, the search is concluded, and the set of queried points Q, and discovered errors, S, are returned for inspection.

V. Results

This section introduces the experimental datasets, classifiers, evaluation procedure, and results.

A. Datasets and Classifiers

The proposed Adversarial Distance search method is evaluated using three experimental datasets. Each dataset introduces high-confidence errors in a different way. In line with previous works, high-confidence errors are searched for over a critical class in a binary classification problem. Below is a description of each dataset and the mechanism by which high-confidence errors were introduced to the evaluation set.

Kaggle13: The Kaggle13 dataset contains 25,000 images of cats and dogs, randomly split into equal sized train and test sets, with 1/10th of the train set reserved for validation. The classification task is defined such that the classifier needs to determine animal type: “cat” or “dog”. High-confidence errors are introduced through dataset bias during training; black cats are removed from the training dataset. When searching for high-confidence errors, the “cat” class is the critical class. This dataset was originally used in [4] and made available by [5].

CelebA: The CelebA dataset contains 202,599 images of faces split into a predefined train, validation, and test datasets [14]. We restricted our test set to 10,000 images for computational considerations. The classification task is defined such that the classifier needs to determine gender, “male” or “female”. High-confidence errors are introduced by simulating a small dataset shift: the training set is made of RGB images and the test dataset has been converted to gray scale. When searching for high-confidence errors, the “male” class is the critical class.

UT-Zap50K: The UT-Zap50K dataset contains 50,025 images of footwear which is randomly split into a 2/3rds training set and a 1/3rd test set, with 1/10 of the train set reserved for validation [15], [16]. The classification task is defined such that the classifier needs to identify footwear type, “not shoe” or “shoe”. Note that boots are removed from each dataset to remove easy elements of the classification task, resulting in the removal of 12,834 images. High-confidence errors are introduced by overfitting; the classifier is trained for 75 epochs (25 epochs produces an adequate classifier). When searching for high-confidence, the “not shoe” class is the critical class.

A CNN with eight convolutional layers and three linear layers is used to build a classifier for each dataset. Models are trained until the classifier stops improving on the validation dataset (with the exception of the UT-Zap50K dataset as described in the dataset description). Furthermore, the validation datasets are used to perform temperature scaling for each classifier as recommended by [12] to rectify the naturally poor calibration of CNNs. The intent is to help ensure that any discovered high-confidence errors are not simply an artifact of poor model calibration, but a true deficiency of the model on the simulated unlabeled evaluation datasets.

Figure 2 shows a reliability diagram, as described in [12], for each test dataset and temperature scaled classifier. The reliability diagram compares expected accuracy to actual accuracy

Dataset	Validation	Test
Celeb A	98%	81%
Kaggle13	81%	81%
UT-Zap50K	98%	94%

Table I: Classifier performance split by dataset.

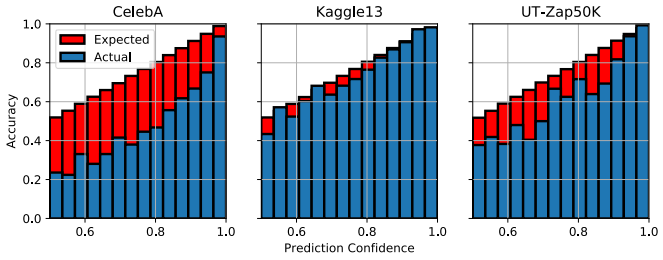

Figure 2: Reliability diagram for each dataset/classifier pair. The red in each diagram indicates overconfidence. The three datasets have differing levels of overconfidence.

for the test dataset by using a classifier's predictions, confidences, and true labels binned for different confidence levels. Visible red portions on the reliability diagrams represent model overconfidence, or confidence levels where more errors exist than are expected. The reliability diagrams shown here focus exclusively on the critical class of the dataset (as identified in the dataset description), and reveal that varying levels of overconfidence are represented by these three dataset/classifier pairs. Additionally, Table I shows the validation and test accuracy for each dataset and classifier pair. The large drop in test accuracy for the CelebA dataset is mainly attributed to the conversion of the test dataset to gray scale; much smaller drops were observed when avoiding the dataset shift.

B. Evaluation

The purpose of our evaluation is to determine if 1) a search driven by Adversarial Distance will discover diverse errors, and if 2) a search driven by Adversarial Distance can help discover a query set with an SDR greater than one, indicating that errors are discovered at a rate exceeding the rate expected given the confidence of the model's predictions. We evaluate each component separately.

As motivated by Bansal and Weld [5], it is desirable to discover diverse errors to avoid sampling a rich pocket of high-confidence errors [5]. We measure the diversity, or spread, of the discovered errors as the average minimum distance from each instance in the evaluation set to an instance selected by the search. To simulate the evaluation of a black-box classifier, and to stay consistent with previous literature, spread is calculated with a feature space derived from the principal components of the evaluation set's pixel space, as we may not have access to the features used to train the classifier. Euclidean distance is used for distance measurements. For an evaluation set, X, and a query set, Q, spread is defined as,

$$spread = \frac{\sum_{x \in \mathbf{X}} min_{q \in Q} dist(x, q)}{|\mathbf{X}|}. \quad (6)$$

As previously motivated, the SDR is used to assess the quality of the query set. SDR is defined in Equation 3 and is the ratio of discovered errors to the expected number of errors given classifier confidence. Again, this measure provides greater insight to the quality of the search rather than defining an arbitrary threshold at which a discovered mistake is deemed valuable.

C. Experiments

The proposed Adversarial Distance search is compared to the Lakkaraju search [6], the Bansal and Weld search [5], a search that samples the lowest confidence predictions, and a random search. To encourage follow on research, all of the code used to perform our experiments is available at https://github.com/afrl-ri/adversarialDistance. Code for the Lakkaraju and Bansal and Weld search was generously made available in [5] and used for this experimentation. Data analysis was done with R and the tidyverse packages [17] [18].

Due to the sensitivity of the searches to the initial conditions of the unlabeled evaluation dataset, each search is run 1,000 times using a random 2,000 instance subset of the test data. This replication simulates having 1,000 unlabeled evaluation datasets for each classifier and search method. Each evaluation set only contains instances predicted by the classifier to belong to the critical class with confidence greater than 65% (the threshold used in previous works to denote a high-confidence error). Each search selects a 50 sample query set and is compared using spread and SDR.

Figure 3 shows the mean spread of each search over 50 queries to the oracle. It is worth noting that all methods achieve similar spread, even in comparison to the Bansal and Weld search which is specifically designed to sample throughout the search space. This indicates that searches are likely not getting stuck in areas with high rates of error (as previously feared), but rather are sampling throughout the search space.

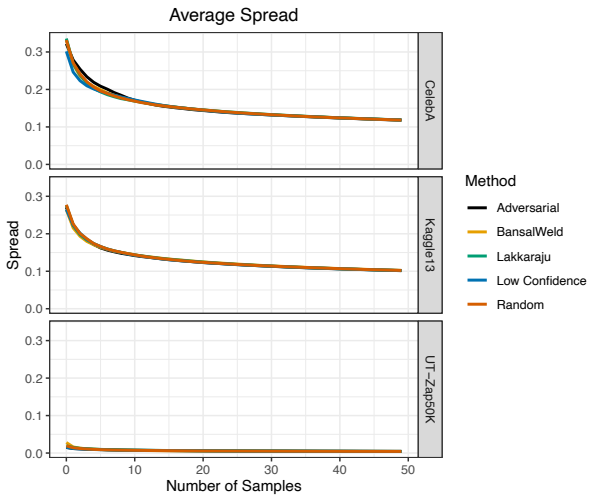

Figure 3: Mean spread across 1,000 runs of the search methods. All methods achieve a similar spread, and the spread improves as more data points are sampled.

Figure 4 shows the mean SDR of each search method over 50 queries. The average SDR achieved by the Adversarial Distance search dominates the curves of the other methods, and indicates that this search method finds errors at rates that exceed expectations. The other methods achieve an SDR near one for the Kaggle13 and UT-Zap50K datasets, which indicates that they are discovering errors at the rate indicated by model confidence. For the CelebA dataset the other methods discover errors at nearly twice the rate indicated by model confidence, but this is not surprising given the amount of overconfidence shown in Figure 2. Interestingly, the performance of the Adversarial Distance search decreases as the query size increases, indicating that the density of error prone instances lessens as the adversarial distance increases. Recommendations to alleviate this issue will be discussed in Section VII.

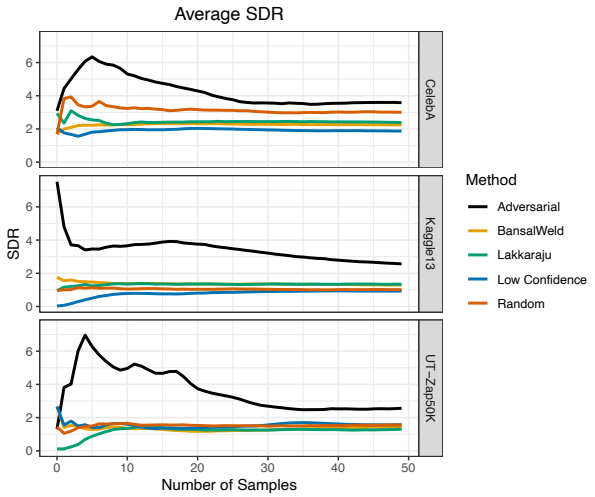

Figure 4: Mean SDR across 1,000 runs of the search methods. The Adversarial Distance method achieves the highest SDR values, meaning it has the highest rate of error discovery relative to the expected rate of error discovery.

Of particular interest, in regards to SDR, is the performance of the Adversarial Distance search for the Kaggle13 dataset. Recalling the reliability diagram in Figure 2, there is very little overconfidence for any of these search methods to discover. However, at 20 queries the Adversarial Distance search discovers errors at four times the rate that the model confidence would suggest, while the other methods are discovering errors at almost exactly the rate indicated by model confidence. Even at 50 queries, the Adversarial Distance search is finding more than twice as many errors as model confidence would suggest.

VI. Discussion

In this section, we discuss the Adversarial Distance search when considering the utility functions presented in previous works. We then show some of the high-confidence mistakes discovered by the Adversarial Distance search and discuss what they tell us about model quality. We also provide a discussion on why Adversarial Distance helps reveal these informative instances.

A. Other Utility Functions

The Bansal and Weld Utility function is defined in Equation 1, and shows that the utility function rewards the discovery of errors that occur near high-confidence points. Being *near* high-confidence points is an important distinction because it does not directly reward finding high-confidence errors. Figure 5 shows that the Bansal and Weld search achieves high values for the Bansal and Weld utility. However, by looking at the number of errors discovered (Figure 6), and the confidence of the points sampled by the Bansal and Weld search (Figure 7), it becomes obvious that high values of the Bansal and Weld utility can be achieved by finding a large number of lower confidence errors; even if these errors should be expected given the model confidence. The Bansal and Weld search achieves an SDR similar to random search, and it is not clear that the search is achieving anything other than selecting samples in the lower confidence ranges. This is further confirmed by the strong performance of the low confidence search for this utility function. The Adversarial Distance search may not perform as well for this utility measure because it discovers fewer errors, but our results from the previous section show it still samples throughout the search space and finds more errors than expected given the confidences of the sampled predictions.

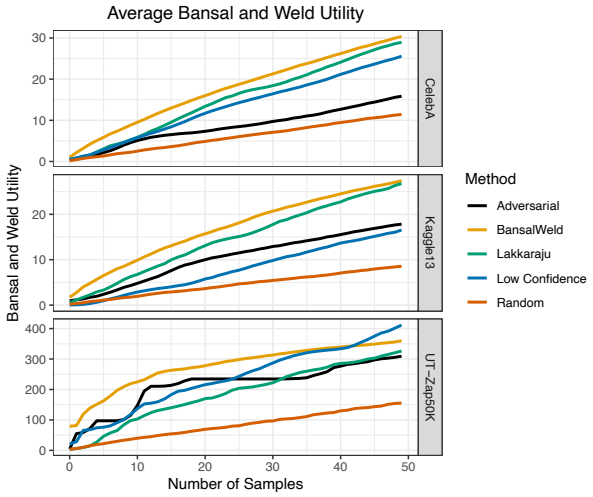

Figure 5: Mean Bansal and Weld utility across 1,000 runs of the search methods.

The Lakkaraju utility function counts the number of errors discovered by the search method. Figure 6 shows that the low confidence search and the Bansal and Weld search maximize this utility. However, as shown in Figure 7, these methods sample lower confidence points, and so we should expect them to find errors at high rates. The Adversarial Distance search samples predictions with similar confidence levels to the random search (Figure 7), but finds more errors. We believe that this is strong evidence that our method finds errors that point to model overconfidence because both methods sample predictions of similar confidence, but the Adversarial Distance search finds more errors and achieves greater SDR values.

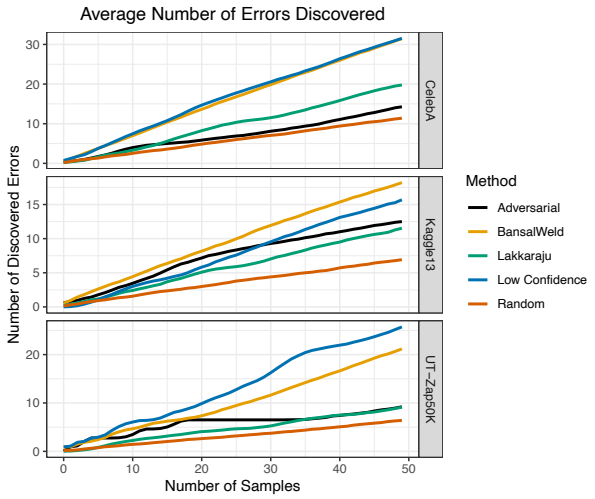

Figure 6: Mean number of errors discovered across 1,000 runs of the search methods. This is the utility function presented by Lakkaraju for imagery.

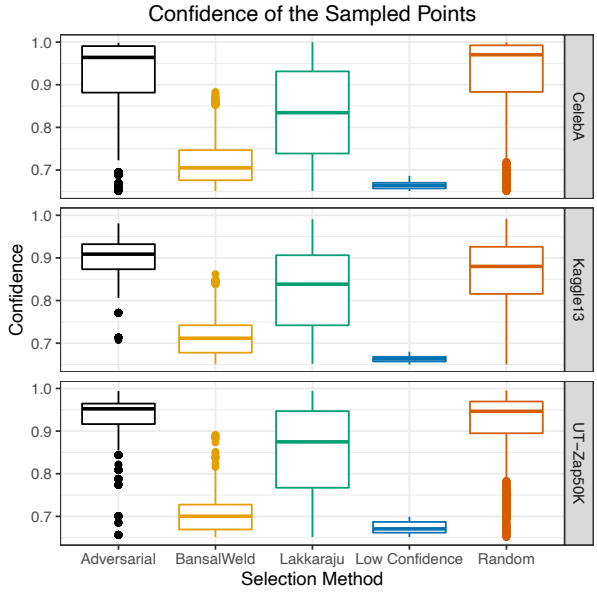

Figure 7: Box plot showing the model confidence of the sampled data points.

Interestingly, the Lakkaraju search method achieves competitive values for the Bansal and Weld utility while having a lower number of discovered errors. It is likely that the discovered errors are in close proximity to high-confidence predictions. Still, the query sets of the Lakkaraju search achieve a low SDR which indicates that errors are occurring at the expected rate (with the exception of the UT-Zap50K dataset which is revealed to have a large amount of overconfidence in Figure 2).

B. Discovered Errors

Figure 8 shows the first six errors discovered by the Adversarial Distance search for the Kaggle13 dataset. LIME [19] has been run for each image to find the superpixels that the model considers most important in classifying these images as "cat". Note that some images are missing LIME information (the green outline) because the method did not identify superpixels for that image that exceeded the default threshold of importance. In general, the errors discovered by the Adversarial Distance search were of dogs with light fur, or dogs on a light background. For the cases where LIME discovered important superpixels, the light-colored superpixels were the most important indicators of the image containing a cat. These high-confidence mistakes suggest that the model is biased to place images with light colors into the cat class. This is consistent with the training set containing only light furred cats after the dataset was biased.

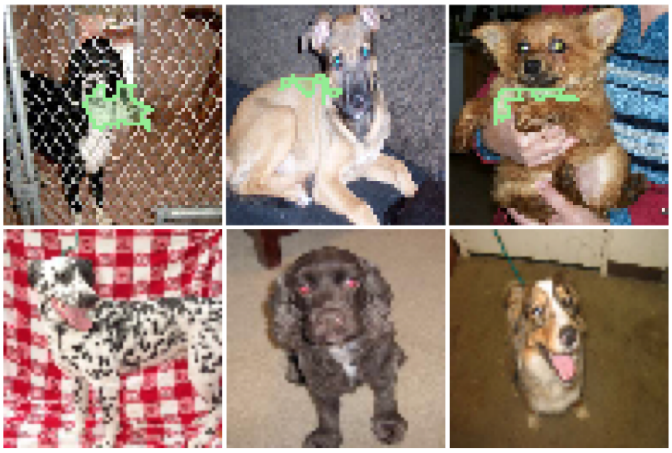

Figure 8: Dogs predicted to be cats with high confidence. Notice the dogs have light fur or are on a light background. LIME also indicates that light colored superpixels are the most important indicator of the cat prediction (LIME did not identify critical superpixels for each image).

Similar results can be found for the CelebA and UT-Zap50K datasets, but were not included for brevity.

C. Insight to Adversarial Distance

Figure 9 provides some insight as to how Adversarial Distance helps find high-confidence mistakes. The first column shows the original image and the important superpixels leading to the image's misclassification. The second column shows the adversarial image and the superpixels leading to the image's correct classification. The third column shows the image from the critical class with the highest Adversarial Distance, as a kind of prototypical instance.

The first row of Figure 9 is from the CelebA dataset. For the original image, the classifier predicts the image to be "male", and may be focused on the absence of bangs. After perturbing a very small number of pixels, the classifier predicts female and seems to be focused on the absence of sideburns (as shown in the prototypical male image). In the second row, the classifier predicts the original image to be a cat, and seems focused on the light color of the hand (similar to the light color of the prototypical cat image). In the adversarial image, the classifier predicts the image to be a dog, and is now focused on the dark nose. In the third row, the classifier predicts the image to be "not shoe", and seems to be focused on the toe, and the absence of a heal. For the adversarial image, the

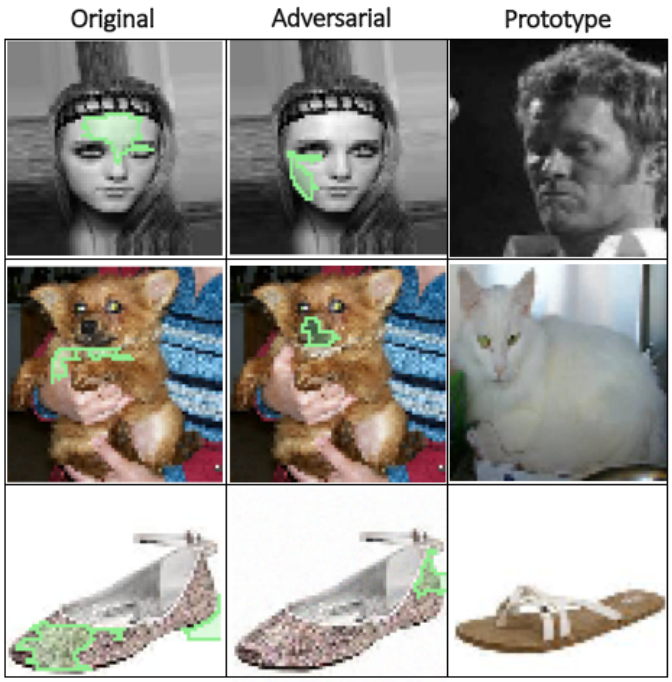

Figure 9: The first column is the original image (incorrectly labeled the critical class) with LIME activation . The second column is the adversarial image (now correctly classified) with LIME activation . The third column is the image from the critical class with the highest Adversarial Distance. It is interpreted as a prototypical instance from the critical class.

classifier predicts shoe and highlights the back of the shoe (the prototypical "not shoe" has no back).

In the cases highlighted above, the classifier is incorrect in its prediction because it seems to focus on non-robust features of the image. However, robust features that could lead to the correct prediction are also present in the image. For example: the shoe has a well defined back, the dog has a dark nose, and the woman does not have sideburns or facial hair. These images likely have low Adversarial Distances because these robust features exist in the image and only small perturbations are required to break the non-robust features leading to an incorrect prediction. Additionally, because these robust features exist in the image, the classifier should not have been as confident in its prediction as it was. A low Adversarial Distance seems to indicate the presence of contradictory robust features or model overconfidence. Sampling these types of images helps discover errors at rates exceeding expectation.

VII. Conclusions

In this work, we introduced the concept of Adversarial Distance and showed how it can be used to help discover prediction errors at rates exceeding what would be expected given the confidence of the model's predictions. That is, when the Mean Absolute Error between an image and its adversarial version is lower than expected for a given classifier confidence, the classifier may be more confident in its prediction than is appropriate.

Experimental results compared the Adversarial Distance search to existing methods designed to search for high-confidence classification errors. Results showed that all methods achieved similar values of "spread", meaning, they all searched evenly throughout the problem's derived feature space. However, the Adversarial Distance search achieved the largest Standardized Discovery Ratios, meaning, it resulted in the highest rate or error discovery relative to the expected error rate.

Future work should focus on the observation that the Adversarial Distance search seems to discover fewer errors as the number of search queries increases. This is likely because the density of mistakes decreases as Adversarial Distance increases. Therefore, when considering large searches, we believe it may be beneficial to use the Adversarial Distance search to prime methods that learn a meta-model of classifier error. Additionally, future work should focus on the uses of the discovered errors. For example, further model calibration or improved risk mitigation strategies.

References

[1] J. Attenberg, P. Ipeirotis, and F. Provost, "Beat the Machine," *Journal of Data and Information Quality*, vol. 6, no. 1, pp. 1–17, 2015. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2742852.2700832

[2] M. Sugiyama, D. Lawrence, and A. Schwaighofer, *Dataset shift in machine learning*, 2017, vol. 1291.

[3] P. Stock and M. Cisse, "ConvNets and ImageNet Beyond Accuracy: Explanations, Bias Detection, Adversarial Examples and Model Criticism," 2017. [Online]. Available: http://arxiv.org/abs/1711.11443

[4] H. Lakkaraju, E. Kamar, R. Caruana, and E. Horvitz, "Identifying Unknown Unknowns in the Open World: Representations and Policies for Guided Exploration," 2017. [Online]. Available: http://arxiv.org/abs/1610.09064

[5] G. Bansal, D. S. Weld, and P. G. Allen, "A Coverage-Based Utility Model for Identifying Unknown Unknowns," *Aaai 2018*, p. 8, 2018. [Online]. Available: http://aiweb.cs.washington.edu/ai/pubs/bansal-aaai18.pdf

[6] H. Lakkaraju, S. H. Bach, and L. Jure, "Interpretable Decision Sets: A Joint Framework for Description and Prediction," *Kdd*, vol. 2016, pp. 1675–1684, 2016. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/27853627

[7] K. Maurer and W. Bennette, "Facility Locations Utility for Uncovering Classifier Overconfidence," in *International Conference of Machine Learning Applications*, 2019. [Online]. Available: http://arxiv.org/abs/1810.05571

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *Iclr 2015*, pp. 1–11, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[9] W. Brendel, J. Rauber, and M. Bethge, "Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models," pp. 1–12, 2017. [Online]. Available: http://arxiv.org/abs/1712.04248

[10] J. Rauber, W. Brendel, and M. Bethge, "Foolbox: A python toolbox to benchmark the robustness of machine learning models," *arXiv preprint arXiv:1707.04131*, 2017. [Online]. Available: http://arxiv.org/abs/1707.04131

[11] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial Examples Are Not Bugs, They Are Features," pp. 1–36, 2019. [Online]. Available: http://arxiv.org/abs/1905.02175

[12] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *International Conference on Machine Learning*, 2017. [Online]. Available: http://arxiv.org/abs/1706.04599

[13] W. S. Cleveland and S. J. Devlin, "Locally weighted regression: an approach to regression analysis by local fitting," *Journal of the American statistical association*, vol. 83, no. 403, pp. 596–610, 1988.

[14] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[15] A. Yu and K. Grauman, “Fine-grained visual comparisons with local learning,” in *Computer Vision and Pattern Recognition (CVPR)*, Jun 2014.

[16] Yu and Grauman, “Semantic jitter: Dense supervision for visual comparisons via synthetic images,” in *International Conference on Computer Vision (ICCV)*, Oct 2017.

[17] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2017. [Online]. Available: https://www.R-project.org/

[18] H. Wickham, *tidyverse: Easily Install and Load the 'Tidyverse'*, 2017, r package version 1.2.1. [Online]. Available: https://CRAN.R-project.org/package=tidyverse

[19] M. T. Ribeiro, S. Singh, and C. Guestrin, ““Why Should I Trust You?”: Explaining the Predictions of Any Classifier,” 2016. [Online]. Available: http://arxiv.org/abs/1602.04938