Neural Computing and Applications (2023) 35:4009–4026
https://doi.org/10.1007/s00521-022-07812-2

ORIGINAL ARTICLE

DRE: density-based data selection with entropy for adversarial-robust deep learning models

Yuejun Guo[1] · Qiang Hu[1] · Maxime Cordy[1] · Michail Papadakis[1] · Yves Le Traon[1]

Received: 25 May 2022 / Accepted: 6 September 2022 / Published online: 19 October 2022

Abstract
Active learning helps software developers reduce the labeling cost when building high-quality machine learning models. A core component of active learning is the acquisition function that determines which data should be selected to annotate.State-of-the-art (SOTA) acquisition functions focus on clean performance (e.g. accuracy) but disregard robustness (an important quality property), leading to fragile models with negligible robustness (less than 0.20%). In this paper, we first propose to integrate adversarial training into active learning (adversarial-robust active learning, *ARAL*) to produce robust models. Our empirical study on 11 acquisition functions and 15105 trained deep neural networks (DNNs) shows that ARAL can produce models with robustness ranging from 2.35% to 63.85%. Our study also reveals, however, that the acquisition functions that perform well on accuracy are worse than random sampling when it comes to robustness. Via examining the reasons behind this, we devise the density-based robust sampling with entropy (DRE) to target both clean performance and robustness. The core idea of DRE is to maintain a balance between selected data and the entire set based on the entropy density distribution. DRE outperforms SOTA functions in terms of robustness by up to 24.40%, while remaining competitive on accuracy. Additionally, the in-depth evaluation shows that DRE is applicable as a test selection metric for model retraining and stands out from all compared functions by up to 8.21% robustness.

Keywords Active learning · Adversarial robustness · Image classification · Deep learning testing

1 Introduction

Aware of the great potential of deep learning (DL) systems, big companies such as Google, Microsoft, and Facebook take huge engineering efforts to build and maintain DL-based tools and contribute to the development of the DL field. DL techniques are also useful to support software engineering tasks, e.g. image recognition for software security [1, 2] and image classification for bugs detection in mobile apps [3].

However, one of the most important hurdles in developing a DL model (i.e. a deep neural network (DNN) with competitive performance is the demand of a large amount of labeled data required for training. Although massive data can come at little to no cost, labeling all of them is time-consuming and almost impossible, especially when expert knowledge is indispensable. To overcome this problem, researchers have proposed approaches to maximize learning capabilities in situations where unlabeled data are abundant while labeling capacities are limited.

Active learning is such an approach and has been widely applied in various domains, such as image classification [4], information extraction [5], software defect prediction [6], and acoustical signal classification [7]. Active learning aims to train a DL model using a small number of but carefully selected labeled data given a limited budget. This model can achieve a similar performance compared to the

✉ Yuejun Guo
yuejun.guo@uni.lu

Qiang Hu
qiang.hu@uni.lu

Maxime Cordy
maxime.cordy@uni.lu

Michail Papadakis
michail.papadakis@uni.lu

Yves Le Traon
yves.letraon@uni.lu

[1] SnT, University of Luxembourg, Esch-sur-Alzette, Luxembourg

model that would be trained on all (labeled) data. This process is usually iterative: at each iteration (stage), a so-called *acquisition function* determines the set of unlabeled data to use and an oracle (for instance, a human annotator) is queried to provide the labels of these data. Past studies have shown that active learning has the potential to reach the same performance as a model trained with all data labeled using only 20% of these data (for the street view house numbers(SVHN) dataset) [8].

An important limitation of active learning so far is that it focuses only on *clean performance* (most often, test accuracy [4, 8, 9]) and ignores other important quality indicators. In particular, active learning does not produce models that are robust to *adversarial examples*, therefore, raises major security concerns [10, 11]. These examples are produced by introducing a small perturbation to benign inputs in a way that it changes the decision of the model although it should not [12]. It is challenging to address that successive methods improve models' robustness merely by a small percentage [13].

In this paper, we bootstrap an endeavor towards improving the adversarial robustness of DNNs trained through active learning. We propose the *adversarial-robust active learning*, *ARAL*, training process that merges active learning with adversarial training [14], the most effective defense against adversarial examples. The key idea of ARAL is to iteratively select the data to label (just like in "standard active learning") and, then, to train the model not on the original data but on their adversarial counterparts. An essential question is whether existing acquisition functions remain effective in ARAL. We, therefore, conduct the first comprehensive study that evaluates whether these acquisition functions can produce models that are both accurate *and* robust. The study undertakes experiments on four datasets, six DNN architectures, eleven acquisition functions, and dozens of labeling budgets. Our investigations reveal that while state-of-the-art (SOTA) acquisition functions remain effective when it comes to clean performance, when it comes to robustness, all of them are outperformed by random sampling.

Following our findings, we explore what key factors explain the lack of robustness of models trained using existing acquisition functions. We empirically demonstrate that, while acquisition functions are inherently biased towards selecting data with specific characteristics (e.g. data with the highest entropy), this bias strongly negatively correlates with robustness. This implies that, in ARAL, effective acquisition functions should select a subset of representative data of the whole dataset, with respect to the aforementioned characteristics) of the whole dataset. To this end, we propose a new acquisition function—named *density-based robust sampling with entropy (DRE)*—that selects data while minimizing the difference between the entropy distributions of the selected set and full dataset. Compared with all existing acquisition functions, the results demonstrate that DRE is the sole acquisition function that achieves higher robustness than random while being competitive in terms of accuracy.

While ARAL aims at training robust models from scratch, an adjacent problem is DL testing, i.e. the problem of finding (adversarial) examples that a trained (high accuracy) DL model misclassifies. Such examples can then be used to retrain the model, thereby improving its adversarial robustness. A crucial part of DL testing is the selection of data from which to generate the adversarial examples. The same acquisition functions that are used in active learning can also serve this purpose. We, therefore, investigate the effectiveness of the acquisition functions in driving the DL testing and retraining. Our evaluation demonstrates that DRE achieves the highest gains in robustness.

To sum up, the main contributions of this paper are:

1. We are the first to formulate and investigate ARAL to solve the problem of training accurate and robust DL models given a limited budget of labeled data.
2. We conduct the first empirical study on the effectiveness of existing acquisition functions in terms of accuracy and robustness. Our results demonstrate that, though some acquisition functions yield higher accuracy than random sampling (by up to 8.08%), none of the functions outperforms random sampling in terms of robustness (by up to 22.50%).
3. We demonstrate that the inherent bias of acquisition functions towards the "informative data" is the cause for their lower robustness (compared to random). Accordingly, we propose DRE, a new acquisition function that yields better robustness (by up to 24.40%) than existing functions while achieving competitive accuracy. DRE, therefore, forms a new baseline for future research on ARAL.
4. We investigate the use of DRE in the adjacent problem of test selection for model retraining. Experimental results show that DRE achieves competitive accuracy and outperforms other acquisition functions by up to 8.21% robustness.

The rest of this paper is organized as follows. Section 2 introduces the background and related work. Section 3 describes the ARAL. Section 4 presents the experimental design. Sections 5 shows the preliminary results of our empirical study. Section 6 describes DRE and its evaluation. The last section concludes this paper.

2 Background and related work

As the focus of this paper is considering adversarial robustness in active learning, this section introduces the relevant related work on active learning, adversarial robustness, and test selection. Readers can refer to several surveys [15–17] for comprehensive surveys of these topics.

2.1 Active learning

Active learning is a set of techniques that aims at building high-performing models using only a small set of labeled data. The main hypothesis is that if a model learns from the most informative data, it can obtain a similar performance using substantially fewer data than using the entire training set.

2.1.1 Problem scenario and active learning in deep learning

Typically, in active learning, there are three types of problem scenario [15, 17], membership query synthesis, selective sampling, and pool-based sampling. In the case of membership query synthesis, the model generates data to query for the labels instead of choosing data from the available unlabeled set [18]. Selective sampling also refers to stream-based or sequential active learning [19]. In this scenario, the model receives unlabeled data one at a time and decides whether or not to request the label based on the informativeness. Compared with selective sampling, in pool-based sampling, the model requests labels for a collection of unlabeled data at once in each query [20]. Since in deep learning, data are divided into batches to train DNNs and a single data would not impart significant change in the model [12], the pool-based sampling is the most widely applied active learning scenario.

2.1.2 Empirical study

As many approaches (acquisition functions) exist for active learning, several empirical studies have compared their effectiveness. Arguing that most previous empirical studies were limited to a single model and performance metric, Ramirez-Loaiza *et al.* [21] rather employ two models and five measures in their study. However, all the used metrics (i.e. precision, recall, F1, accuracy, and under the curve-AUC) evaluate the correctness of the model on clean data only. [22] studies more datasets, classification models, and selection approaches, but the comparison is still based on correctness only. On the other hand, some studies focus on specific applications, such as segmenting Japanese word segmentation [23], text classification [24], learning English verb senses [25], labeling sequence [26], and locating temporal activation in video data [27].

In this paper, we overcome the two main limitations of existing studies. First, we go beyond prediction correctness and consider adversarial robustness as an important success metric of active learning. Second, we compare 11 competitive active learning approaches on common ground configuration.

2.1.3 Active learning for SE

Many tasks in software engineering (SE) benefit from active learning. For instance, Bowring *et al.* [28] apply active learning in the automatic classification of program behavior. The model is incrementally trained using selected executions that represent unknown behaviors for the model. When performing on the software effort estimation data, active learning is proved to largely prune the training data and quickly find the essential content [29]. Yu *et al.* [30] show that active learning can help with conducting literature reviews. Recently, Cambronero *et al.* [31] proposed the use of active learning to automatically infer programs, which can significantly alleviate the difficulty of manual annotations. Based on the Mozilla Firefox vulnerability data, Yu *et al.* [32] prove that active learning is useful in building a prediction model which learns from the historical source code data and predicts the candidates to inspect. Similarly, Yang *et al.* [33] study active learning for static code analysis. Active learning also facilitates building defect prediction models [6, 34, 35]. For instance, Tu *et al.* [35] developed an active learning tool, EMBLEM, to label the most problematic commits and they claim the first use of active learning in commit defect prediction.

2.2 Adversarial robustness

The adversarial robustness of DNNs relates to the ability of the model to distinguish adversarial examples. In other words, similar to the accuracy on clean data, the robustness of a DNN is measured by its accuracy on adversarial examples crafted from the clean data.

2.2.1 Adversarial example

Considering the image classification task, given an input image, the corresponding adversarial example is crafted by adding a carefully calculated perturbation into this image to mislead DNNs [12]. Since this perturbation is hardly perceptible for human beings, an adversarial example is regarded as following the same data distribution and having the same label as the original input. The cause of adversarial examples is still under exploration. Some speculative explanations are the extreme nonlinearity of

DNNs [12], the high-dimensional linearity of DNNs [36], the presence of non-robust features [37], insufficient regularisation [38], and insufficient model averaging [36].

2.2.2 Adversarial attack

The approaches towards crafting adversarial examples are called adversarial attacks (also known as adversaries, threat models). In general, adversarial attacks are divided into three types: black-box, gray-box, and white-box. The black-box attacks have no access to the model, and the perturbation is calculated by using the predicted probabilities or logits (score-based), e.g. square attack [39], by simply relying on the max class label (decision-based) [40], or by transferring from another DNN model trained with the same training data (transfer-based) [41]. The gray-box attacks only have knowledge of DNN architectures. In contrast, the white-box attacks have full access to the data, parameters, and DNN architectures. The gradient of training loss is mostly utilized in various white-box methods, e.g. projected gradient descent (PGD) [14]. Besides, there is the Auto attack [42], which combines the black-box and white-box attacks and is commonly used as a strong baseline in robustness evaluation. In this paper, we comprehensively consider the square attack (black-box), PGD (white-box) attack, and Auto attack (adaptive).

2.2.3 Adversarial defense

Mitigating the threat of adversarial examples and improving the robustness of DNNs is of great importance and has attracted considerable attention [15]. Adversarial defense refers to a defense mechanism that secures DNNs against adversarial attacks. Many defense approaches have been proposed, such as adversarial training [14], input denoising [43], input transformation [44], randomization [45], and defensive distillation [46, 47]. Among all, adversarial training with a PGD adversary [14] has been proven to be one of the most effective methods. The difference is that, in standard training, the original training data are fed into the DNN models to tune all the parameters. However, in each epoch of adversarial training, a new set of data is generated by applying the PGD attack to the original training data and then is used to train the model.

2.3 Test selection for retraining

Software engineering researchers have devoted substantial effort towards evaluating and improving the quality of DL models. DL testing techniques aim to expose flaws in DL models by selecting test data (from an existing data pool) or the generation of new test data (i.e. adversarial examples). DL engineers can then use those data to retrain the model and improve it. Multiple test selection metrics have been proposed to solve various DL testing problem, including test accuracy estimation [48, 49], test data prioritization [50–52], DNN comparison [53], and DNN retraining [3, 51, 54].

In particular, past research on DNN retraining [51, 54] have shown that, given a trained DNN and a set of unlabeled test data, test selection metrics can select data to retrain the DNN (either on the selected data or on adversarial examples produced from the data) and improve its quality (clean performance or robustness). This testing and retraining process shares the similarity with ARAL as defined in this paper. Indeed, both processes aim to improve the quality of DNNs through the selection of the most appropriate data. The difference is that active learning applies during training and targets the training data, whereas DL testing applies after training and on a set of data not used during training. Therefore, the retraining process can be seen as one-stage active learning after training and on a new set of data. The acquisition functions used in active learning—including the one proposed in this paper—can thus be used for retraining.

3 ARAL: adversarial-robust active learning

Consider a C-class classification problem over a sample space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$, and a collection of data $\mathcal{D} = \{x_i, y_i\}_{i\in[n]} \sim p_{\mathcal{Z}}$, where C, $x_i \in \mathcal{X}$, and $y_i \in \mathcal{Y}$ are the number of classes, the data, and label, respectively. $p_{\mathcal{Z}}$ is the data distribution of $\mathcal{D}$. Let $\mathcal{H}$ be the hypothesis space and f_θ be a neural network architecture parameterized by a parameter vector $\theta \in \mathcal{H}$. The loss function is $J : \mathcal{H} \times \mathcal{Z} \rightarrow \mathcal{R}$. In deep learning, the object of training f_θ is to minimize the expected loss (also called as the expected risk):

$$\mathbb{E}_{x,y\sim p_{\mathcal{Z}}}[J(\theta_{\mathcal{D}}, x, y)] \tag{1}$$

where $\theta_{\mathcal{D}}$ indicates that the parameters are tuned using $\mathcal{D}$. Note that the value of θ depends on the data used to train; thus, to put this in evidence, we use the notation $\theta_{\mathcal{D}}$ indicating that this parameter vector corresponds to the data $\mathcal{D}$.

Figure 1 illustrates our proposed process of ARAL and its difference to standard active learning. The data are initially unlabeled and stored in an unlabeled pool, $\mathcal{U} = \{x_i\}_{i\in[n]} \sim p_{\mathcal{Z}}$. Given a labeling budget b, an acquisition function calculates the priority of each data to be labeled and the process requests some oracle to label the highest priority data. These newly labeled data are then moved to a labeled pool $\mathcal{L} = \{x_i, y_i\}_{i\in[m]} \sim p_{\mathcal{Z}}$ $(m \leq b)$. In standard active learning, at each stage, the set of model parameters θ is updated using all data from the labeled

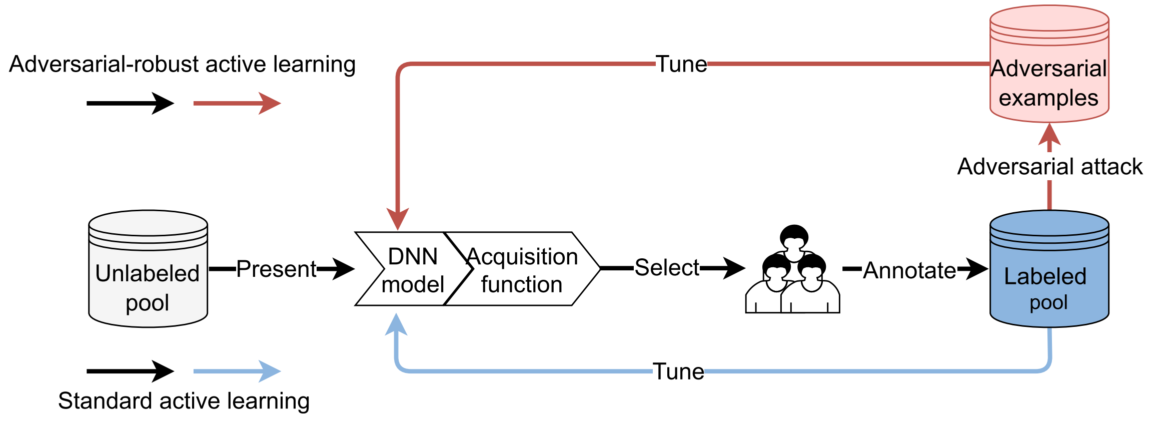


Fig. 1 Overview of pool-based active learning

pool—so as to minimize the average model loss on these labeled data, that is,

$$\mathbb{E}_{x,y\sim p_{\mathcal{Z}}}[J(\theta_{\mathcal{L}}, x, y)] \tag{2}$$

Just like DNNs trained on clean data are well known to be vulnerable to adversarial examples [12, 36–38], DNNs trained with standard active learning should have low robustness as well. To confirm this, we conducted experiments on six subjects to measure the robustness of models that result from standard active learning using different acquisition functions (following the experimental protocol described in Sect. 4). Table 1 summarizes the results. In all cases, the achieved robustness is extremely low (below 0.20%), while the highest accuracy approaches 100%.

To facilitate the trained DNN to be resilient to adversarial examples, we propose to incorporate the adversarial training into active learning. Compared to standard active learning, for each labeled data we craft an adversarial example—using the PGD attack [14]—and train the DNN with the produced examples (instead of the original labeled data). Hence, the clean labeled data $\{(x_i, y_i)\}$ are replaced by perturbed examples $\mathcal{L}_{adv} = \{x_i + \delta, y_i\}_{i\in[m]}$ $(m \le b)$ during training and the training objective becomes to minimize the risk concerning the adversarial examples:

$$\mathbb{E}_{x,y\sim p_{\mathcal{Z}}}[J(\theta_{\mathcal{L}_{adv}}, x+\delta, y)] \tag{3}$$

where δ is the upper bound of the perturbation. Remark that in adversarial training, the perturbed data produced are

Table 1 Standard active learning: result of accuracy (Acc, %) and adversarial robustness (Rob, %) against the Auto attack by 11 acquisition functions

Acquisition function	MNIST Lenet-5		Fashion-MNIST Lenet-5		SVHN VGG8		CIFAR-10 VGG16		CIFAR-10 ResNet18		CIFAR-10 PreActResNet18	
	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob
BALD	99.61	0.00	87.65	0.00	90.85	0.08	93.04	0.11	94.81	0.16	94.29	0.19
DFAL	99.45	0.00	87.73	0.00	90.61	0.07	93.25	0.08	94.36	0.20	94.38	0.09
EGL	99.61	0.00	84.73	0.00	88.69	0.05	92.65	0.08	94.19	0.15	94.40	0.11
MaxEntropy	99.69	0.00	87.57	0.00	91.05	0.05	93.21	0.08	94.53	0.11	94.63	0.09
DropOut-Entropy	99.59	0.00	87.28	0.00	91.03	0.05	93.27	0.07	94.60	0.11	94.45	0.13
DeepGini	99.63	0.00	87.87	0.00	90.77	0.04	93.45	0.11	94.51	0.19	94.65	0.14
Core-set	99.67	0.00	87.64	0.00	90.81	0.06	93.17	0.08	94.29	0.17	94.55	0.20
LC	99.59	0.00	87.98	0.00	90.91	0.06	93.20	0.16	94.57	0.17	94.47	0.12
Margin	99.63	0.00	87.86	0.00	90.95	0.08	93.09	0.10	94.51	0.15	94.53	0.17
MCP	99.54	0.00	87.55	0.00	90.45	0.07	92.76	0.08	94.33	0.16	94.05	0.14
Random	98.84	0.00	85.73	0.00	87.53	0.11	91.09	0.16	92.23	0.19	92.27	0.14
Maximum	**99.69**	**0.00**	**87.98**	**0.00**	**91.05**	**0.08**	**93.45**	**0.20**	**94.81**	**0.20**	**94.65**	**0.20**

Maximum the maximum (in bold) accuracy/robustness in a dataset over all functions used even if the DNN correctly classifies them (i.e. even if the adversarial attack "fails").

The main research question that we study in this paper is given a few labeled data, whether ARAL can produce robust models. We, therefore, conduct a comprehensive empirical study involving a large set of acquisition functions.

4 Experimental design

4.1 Implementation

All experiments were conducted on a high-performance computer cluster consisting of GPU nodes. To reduce the influence of randomness, each experiment is repeated three times and, in total, 15105 DNNs are trained in this study. The whole results corroborate our findings are also available on our companion project website [55].

4.2 Datasets and models

Four popular publicly available datasets, modified national institute of standards and technology (MNIST) [56], Fashion-MNIST [57], SVHN [58], and Canadian institute for advanced research (CIFAR)-10 [59] are selected for evaluation. These datasets have previously been used for robustness assessment in the context of DL testing [3]. Both MNIST and Fashion-MNIST include a collection of 28 x 28 grayscale images. MNIST comprises handwritten digits $0 \sim 9$ and Fashion-MNIST presents fashion products. SVHN and CIFAR-10 consist of 32 x 32 RGB images. SVHN shows street view house numbers and CIFAR-10 contains more complex entities, such as vehicles and animals. Table 2 summarizes the detailed information of the datasets and DNN models. In all our experiments, we use the training set to craft the adversarial examples during the adversarial training. The test data are evenly split into a validation and test sets. The validation set is used in the training process to tune the parameters of DNNs, and the test set is independent of training for evaluating the accuracy (directly on the test data) and the robustness (by crafting adversarial examples from the test data).

Table 2 Summary of datasets and DNNs

Dataset	DNN	#Train	#Test	Acc
MNIST	Lenet-5	60k	10k	99.47
Fashion-MNIST	Lenet-5	60k	10k	90.78
SVHN	VGG8	50k	10k	92.69
CIFAR-10	VGG16	50k	10k	89.75
	ResNet18	50k	10k	90.65
	PreActResNet18	50k	10k	90.68

Acc test accuracy (%) of DNNs by standard training using the entire set

4.3 ARAL process and test selection

Table 3 summarizes the active learning setups. Concretely, we follow [60] to initialize the labeled pool with a small amount of data uniformly sampled from the unlabeled pool. The initial DNN is tuned using these initial sets via standard training for all the acquisition functions. In test selection for model retraining, 20 epochs are used for SVHN and 10 for the others to ensure the training process converges.

4.4 Evaluation measures

We measure the accuracy and adversarial robustness of each DNN. Given a test set and a DNN, the accuracy is the percentage of correctly classified data over the entire test set. Adversarial robustness is calculated by the percentage of correctly classified adversarial examples over the entire adversarial test set. The adversarial test set consists of adversarial examples crafted on the clean test set by a specific adversarial attack. Table 4 lists the parameters of adversarial attacks for ARAL. Both white-box and black-box attacks are employed for robustness evaluation. The white-box attack PGD is commonly used to evaluate DNNs' robustness. Considering that in this case, the defense DNN knows this attack in advance, the SOTA black-box attack named square attack [39] is also applied. Finally, we use the Auto attack [42], an adaptive attack that is widely utilized as a strong baseline thanks to its ability to overcome common defense mechanisms such as gradient masking [61]. All these attacks are implemented using a public library, Torchattacks [62]. The default setting in the library for other related parameters is applied for each attack.

4.5 Acquisition functions

Given a DNN f_θ where θ is randomly initialized, unlabeled pool $\mathcal{U}$, an acquisition function helps the AL system in each step to query the most informative and useful data to solve the optimization object in Eq. (2). Various acquisition functions have been proposed for data selection. This section reviews 11 widely used acquisition functions.

As Wald showed [63], a solution to solve the optimization object in Eq. (2) is to minimize the maximum of the risk. Based on this, several acquisition functions have been proposed to select the most informative data by assigning and ranking the importance of data. Let

Table 3 Configuration setting of ARAL

Dataset	DNN	Budget	Initial	New	Stage	Initial accuracy	Initial robustness			Full accuracy	Full robustness		
							PGD	Square	Auto		PGD	Square	Auto
MNIST	Lenet-5	5k	200	200	24	29.65	0.00	0.04	0.00	98.69	89.52	76.27	76.17
Fashion-MNIST	Lenet-5	6k	200	200	29	74.22	0.00	0.04	0.00	72.55	64.18	34.01	34.06
SVHN	VGG8	10k	1k	500	18	51.21	0.34	1.92	0.25	83.95	40.77	37.50	35.94
CIFAR-10	VGG16	25k	1k	500	48	37.01	1.71	4.27	1.54	69.54	42.52	43.41	40.71
	ResNet18	25k	1k	500	48	31.69	1.40	3.73	1.21	73.11	42.52	43.41	42.65
	PreActResNet18	25k	1k	500	48	33.88	2.51	4.76	2.23	73.24	44.29	45.59	42.65

Budget maximum number of data to label; *Initial* number of labeled data when launching active learning; *New* number of data to annotate in each stage; *Stage* number of stages in active learning; *Initial accuracy/robustness* (%) performance of the initial model. *Full accuracy/robustness* (%) performance of the model adversarially trained using the entire set

Table 4 Configurations of adversarial training and evaluation

Operation	Attack	MNIST Fashion-MNIST	SVHN CIFAR-10
Adversarial Training	PGD	ϵ = 0.3, α = 0.01, I = 40	ϵ = 0.3, α = 0.01, I = 40
Robustness Evaluation	PGD	ϵ = 0.3, α = 0.01, I = 50	ϵ = 8/255, α = 2/255, I = 50
	Square	ϵ = 0.3	ϵ = 8/255
	Auto	ϵ = 0.3	ϵ = 8/255

ϵ, α, and I denote the perturbation size, step size of perturbation, and the number of iterative steps, respectively

$\{p(y=i \mid x;\theta)\}_{i\in[C]}$ be the softmax output of f_θ (recall that C is the number of classes).

4.5.1 MaxEntropy

In information theory, the entropy (also known as Shannon entropy) quantifies the uncertainty of prediction [64]:

$$H(y \mid x;\theta) = -\sum_{i\in[C]} p(y=i \mid x;\theta)\log(p(y=i \mid x;\theta)) \tag{4}$$

MaxEntropy ranks the uncertainty of data based on the predictive entropy and selects the most uncertain ones.

4.5.2 DeepGini*

The Gini impurity is a loss metric in decision tree learning to decide the optimal split from a root node and subsequent splits. It measures the likelihood of misclassification of a new instance, which reflects the uncertainty of a model to this instance. Borrowing the idea of Gini impurity to deep learning, DeepGini [51] is proposed to select the most informative data:

$$\arg\max_{\mathbf{x}\in\mathcal{U}} \left(1 - \sum_{i\in[C]} p^2(y=i \mid x;\theta)\right) \tag{5}$$

Similar to MaxEntropy, DeepGini utilizes the output of f_θ to assign the informativeness to data. However, as mentioned by the authors of DeepGini, computing the quadratic sum is easier and simpler than performing the logarithmic computation, which is supposed to give better performance.

4.5.3 BALD

Taking the concept of Bayesian neural network where a DNN is defined by a set of parameters drawn from a posterior distribution, the Bayesian active learning by disagreement (BALD) [65] seeks data for which the parameters under the posterior disagree about the prediction the most.

$$\arg\max_{\mathbf{x}\in\mathcal{U}} \left(H(y \mid x;\mathcal{U}) - \mathbb{E}_{\theta\sim p_{\theta,\mathcal{U}}}[H(y \mid x;\theta)]\right) \tag{6}$$

In other words, BALD can be explained as identifying on which the DNN is on average most uncertain about the prediction (big $H(y \mid x;\mathcal{U})$) but existing model parameters are confident (small $\mathbb{E}_{\theta\sim p_{\theta,\mathcal{U}}}[H(y \mid x;\theta)]$). In practice, to approximate the inferences, the Monte Carlo dropout is widely applied due to its great capacity and low cost. We set $T = 10$ with probability 0.1 at test time to sample different DNNs as [4].

4.5.4 DropOut-Entropy

Instead of computing the entropy over one DNN with fixed parameters, this acquisition [66] function calculates the uncertainty over multiple Bayesian DNNs inferred by the Monte Carlo dropout. Thus, the object changes to find data that maximize the uncertainty as follows:

$$H(y \mid x; \mathcal{H}) = -\sum_{i \in [C]} p(y = i \mid x; \mathcal{H}) \log p(y = i \mid x; \mathcal{H}) \tag{7}$$

where $p(y = i \mid x; \mathcal{H}) = \frac{1}{T}\sum_{t \in [T]} p(y = i \mid x; \theta^t)$ is the average predicted output over T times of applying dropout to the model. θ^t denotes the parameters of the t-th dropout. We set $T = 10$ with probability 0.1 at test time to sample different DNNs as [4].

4.5.5 LC

Least confidence uses the most probable label of an instance to capture the uncertainty. It ranks data based on the highest posterior probability and selects data with the least confidence on the prediction:

$$\arg\max_{\mathbf{x} \in \mathcal{U}} (1 - p(y = y' \mid x; \theta)) \tag{8}$$

where $y' = \arg\max_{i \in [C]}(p(y = i|\mathrm{x}; \theta))$ indicating the predicted class label by f_θ.

4.5.6 Margin

Since LC only utilizes the most confident class label, the information about the remaining labels is discarded but can also be useful. To solve this issue, the margin sampling [67] ranks data based on the difference between the most confident and second most confident labels and chooses the data with a small difference:

$$\arg\min_{\mathbf{x} \in \mathcal{U}} \left(\arg\max_{\mathbf{i} \in [C]} (p(y = i \mid x; \theta)) - \arg\max_{\mathbf{i} \in [C]/y'} (p(y = i \mid x; \theta)) \right) \tag{9}$$

The hypothesis is that if a DNN predicts a similar probability on the top two labels for an instance, then this instance is not well learned and remains close to the decision boundary.

4.5.7 MCP*

Noticing that the data selected by margin sampling might be unbalanced distributed concerning the decision boundary areas, the Multiple-Boundary Clustering and Prioritization (MCP) [54] improves the margin sampling by uniformly selecting data from different areas. Besides, it computes the priority by the ratio of probabilities of the top two labels instead of using difference. Each decision boundary area is a cluster defined by the top two classes. Data are selected from each cluster with high priorities.

4.5.8 DFAL

To approximate how close an instance is to the decision boundary, the DeepFool active learning (DFAL) [4] utilizes the magnitude of the minimum perturbation to successfully craft an adversarial example of this instance by the DeepFool attack. Indeed, adversarial attacks are designed to push the clean data to cross the decision boundary. The object of DFAL is

$$\arg\min_{\mathbf{x} \in \mathcal{U}} \left(DeepFool\left(x, f_\theta, L_p\right)\right) \tag{10}$$

where $DeepFool\left(x, f_\theta, L_p\right)$ is a function that attacks f_θ given x using the L_p norm ($p = 2$) and outputs the perturbation between x and its adversarial example.

4.5.9 EGL

Given that the DNN training generally uses gradient-based optimizations, the expected gradient length (EGL) [9] ranks an instance with high importance if it would induce the greatest change in the gradient. A challenge is that in active learning, the true labels are not available, thus, EGL assumes all the possible labels to data and computes the expected gradient. The data are selected by

$$\arg\max_{\mathbf{x} \in \mathcal{U}} \left(\sum_{i \in [C]} p(y = i \mid x; \theta) \parallel \nabla J(\theta, x, y = i) \parallel \right) \tag{11}$$

where $\| \cdot \|$ is the L_2 norm (Euclidean distance), $\nabla J(\theta, x, y = i\cdot)$ denotes the gradient of the loss function at x given θ, and $y = i$.

Different from focusing on minimizing the maximum of the risk, some acquisition functions solve the optimization object in Eq. (2) via adding data that are far from the labeled data, such as Core-set.

4.5.10 Core-set

The Core-set selection [8] converts the optimization object in Eq. (2) to the k-center problem by setting an upper bound. It solves the k-center problem by selecting the data that are far from data in the labeled pool $\mathcal{L}$. Given that the selection of data is time-consuming, the k-Center-Greedy is utilized instead of robust k-Center since they exhibit similar behavior [8].

4.5.11 Random

Random sampling is the simplest and model-free method that each data has an equal probability of selection. This method, strictly speaking, belongs to passive learning but is commonly taken as a baseline in active learning.

Note that the functions with $*$ are originally designed for test selection, but they all select the most informative data. Thus, they are suitable for active learning.

5 Preliminary results of the empirical study

5.1 Effectiveness of ARAL

The effectiveness of ARAL is investigated in creating models that are both accurate and robust.

Results Figures 2 and 3 visualize the accuracy and robustness curves of 11 acquisition functions, respectively, for each dataset and model architecture. For brevity, the results of CIFAR-10 are plotted by every 5 stages. First, we observe that most of the acquisition functions keep the ability to achieve the same level of test accuracy as the model adversarially trained with the full dataset. In general, 3%, 2%, 14%, and 22% of the labeled data are enough to reach this level of accuracy for MNIST, Fashion-MNIST, SVHN, and CIFAR-10, respectively. However, some functions (e.g. MaxEntropy, Entropy, DeepGini, LC, and EGL) perform worse than the others, including random sampling. MCP, Core-set, BALD, and (occasionally) DFAL are the best performing metrics and outperform random sampling.

Second, all acquisition functions allow a substantial increase of robustness compared to models trained with original data only. However, the obtained models are less robust than the model adversarially trained with all data (between 2% and 10% robustness difference compared to the best performing acquisition function). Interestingly, random sampling stands out in all the datasets and DNNs, and often by a significant margin. Just like for accuracy, MaxEntropy, Entropy, DeepGini, LC, and EGL, achieve the lowest levels of robustness. The other acquisition functions perform irregularly across different datasets and DNNs, yet remain inferior to random sampling. This indicates that the criteria used by existing acquisition functions have a negative effect on robustness, to the extent that random sampling—a simple, random, model-free, and data-free method—performs much better.

To confirm this conclusion, we conduct statistical analysis and assess whether the difference of robustness between random sampling and the other acquisition functions is statistically significant. The analysis is based on the Wilcoxon signed-rank test [68], which is a nonparametric statistical hypothesis test commonly used for comparing two independent paired samples. In our case, each sample is the robustness obtained based on a given acquisition function. Random sampling is compared with each of the other 10 functions on all datasets/models and three attacks. This yields 180 ($10 \times 6 \times 3$) statistical tests. The significance level α is set to $5.00E-02$. All these tests have rejected the null hypothesis that there is no difference between random sampling and the other acquisition function, with a p-value ranging from 8.15E-10 to 9.49E-03. Hence, we conclude that random sampling significantly outperforms the other functions in terms of robustness.

Conclusions Using only a limited set of labeled data, ARAL can produce models as accurate as the model adversarially trained with all data. When it comes to robustness, random sampling performs consistently better than the other acquisition functions, though there remains a substantial margin for improvement compared to using all data.

5.2 Data selection biases

As an attempt to explain the better effectiveness of random sampling and devise an improved acquisition function, we investigate the characteristics of the data selected by each acquisition function at all stages of the active learning process. We hypothesize that random sampling performs better than existing acquisition functions because the latter are *biased* towards the "most" informative data—informative being defined by each function in terms of the metric it uses to prioritize the data. By contrast, random sampling is naturally unbiased as it associates each data with the same probability to be selected. Therefore, on average it selects a set of data that are representative of the whole unlabeled pool.

We aim at establishing a relationship between the biases that the acquisition functions introduce and their inability to produce robust models. From the corresponding acquisition functions, we study four data characteristics, entropy, Gini impurity, least confidence (lc), and margin. The characteristics of the others are not applicable due to the use of dropouts (BALD, DropOut-Entropy) and model independence (MCP, DFAL, EGL, Core-set). Additionally, we measure the bias in the true class label, which is available at first hand and has been studied in the literature [24].

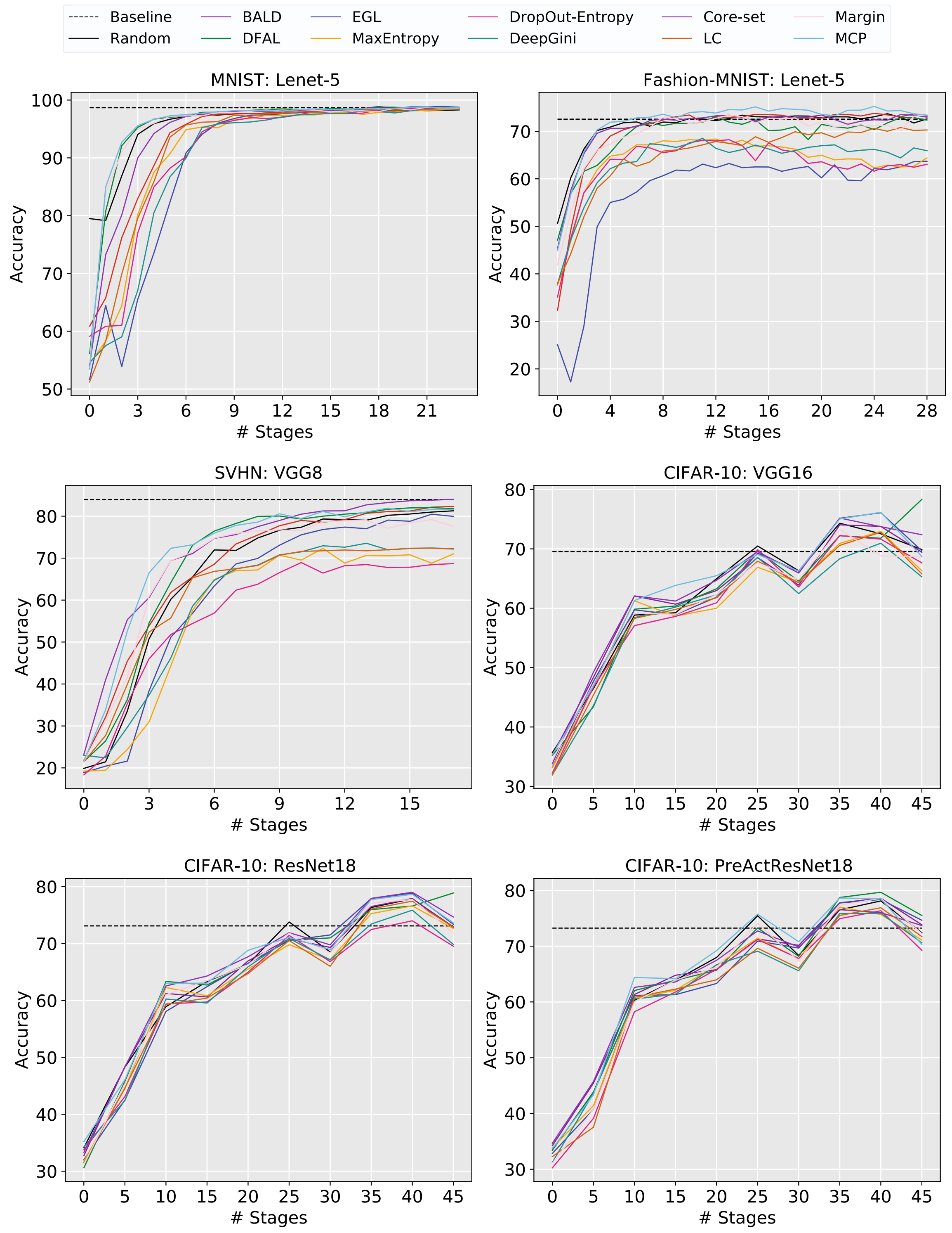


Fig. 2 Test accuracy (%) of 11 acquisition functions over different stages of ARAL. Baseline is the model adversarially trained using the entire data

First, we measure the bias of each acquisition function towards each of these characteristics. Given a dataset and DNN, let $\mathcal{U}_{i,j}$ be the unlabeled pool in the i-th stage of active learning using the j-th acquisition function and $\mathcal{S}_{i,j} \subseteq \mathcal{U}_{i,j}$ be the set of data selected by j at this i-th stage. Given a characteristic function ϕ that, given an input data x returns the value of the characteristic under study for x, we generate two sets of variables $\phi(\mathcal{S}_{i,j})$ and $\phi(\mathcal{U}_{i,j})$ that

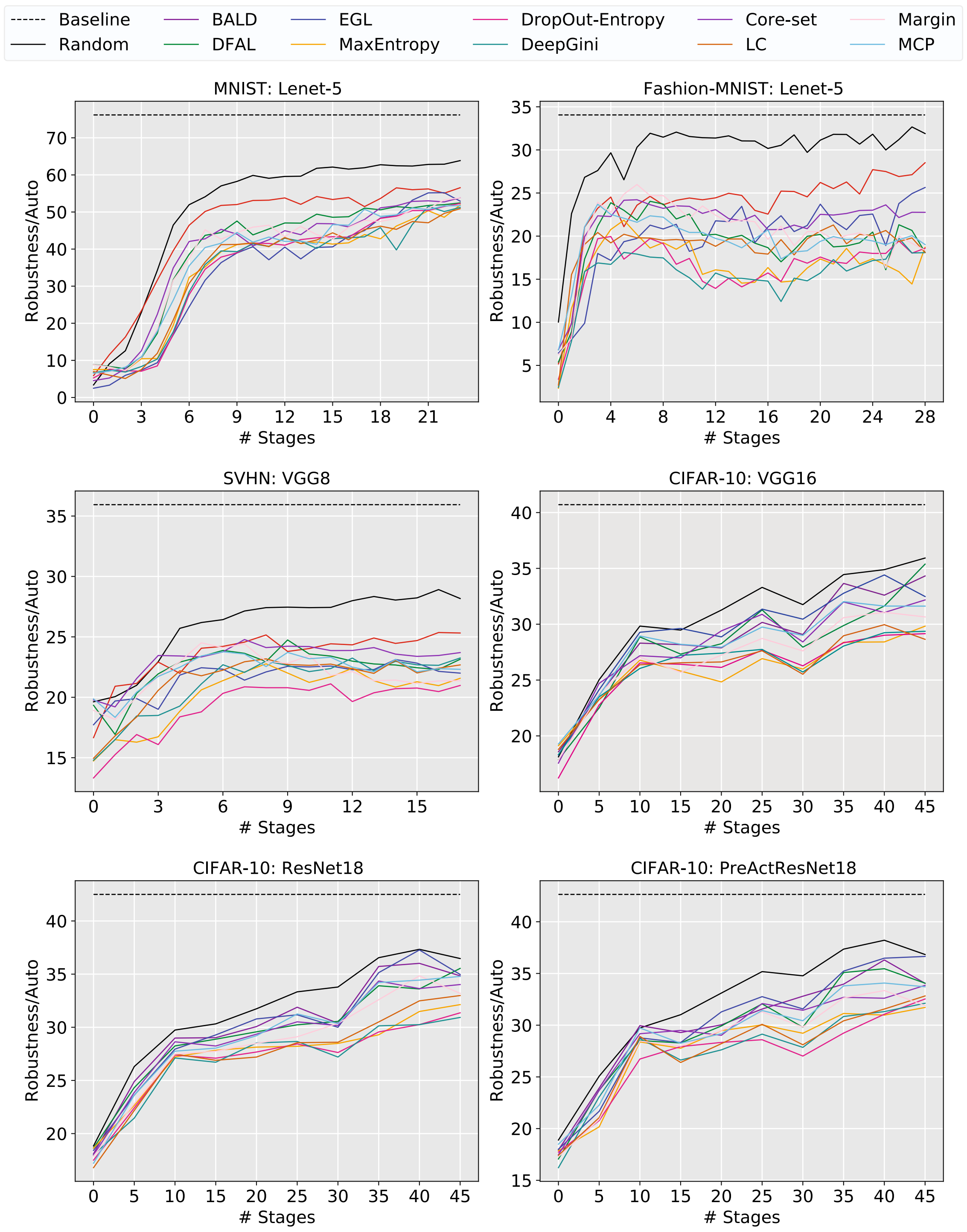


Fig. 3 Robustness (%) against Auto attack of 11 acquisition functions over different stages of ARAL. Baseline is the model adversarially trained using the entire data

contain the characteristic value of all data in $\mathcal{S}_{i,j}$ and $\mathcal{U}_{i,j}$, respectively. We, then, estimate the probability density functions (PDFs) of $\phi(\mathcal{S}_{i,j})$ and $\phi(\mathcal{U}_{i,j})$ using the histogram method with 50 bins [69], yielding $PDF_{\phi(\mathcal{S}_{i,j})}$ and $PDF_{\phi(\mathcal{U}_{i,j})}$, respectively. We calculate the difference between two PDFs based on the Jensen-Shannon divergence (JSD)—an established method to measure the divergence between two distributions [70]. This difference is, therefore, given by

$$d_{i,j} = \text{JSD}\left(PDF_{\phi(\mathcal{S}_{i,j})} \parallel PDF_{\phi(\mathcal{U}_{i,j})}\right) \tag{12}$$

Second, we measure the correlation at each stage for every characteristic between (a) the robustness of models produced by different acquisition functions and (b) the characteristic bias of data selected by corresponding functions. The Pearson correlation is used to measure this correlation, which captures linear relationships between two variables. Hence, for each stage i and each characteristic, a correlation coefficient r_i is calculated by [71]:

$$r_i = \text{Pearson}(d_i, a_i), d_i = \{d_{i,j}\}_{j \leq 11}, a_i = \{a_{i,j}\}_{j \leq 11} \tag{13}$$

where $a_{i,j}$ denotes the robustness of the model produced at the i-th stage by the j-th acquisition function.

Results Figure 4 shows a heat map of the Pearson correlation coefficients $\{r_i\}$ obtained from SVHN/VGG8, and the Auto attack. All correlations are negative as the colors are always red, which supports our hypothesis that characteristic biases have a negative effect on robustness. In other words, selecting data that are more representative of the entire unlabeled pool is more likely to produce a robust DNN. Among all the characteristics, entropy exhibits the strongest negative correlations. Over the 18 stages, the maximum and average correlations obtained by entropy are -0.84 and -0.65 (strong negative correlations), whereas the lowest correlations are achieved by margin (maximum -0.76 and -0.56 on average). Entropy, therefore, seems the most capable metric to drive the selection of a representative set of data that will be used to produce robust models.

Conclusions The inherent bias of acquisition functions towards the "most informative" data has strong negative correlations with robustness. To improve robustness, an ideal function should rather select data that have a representative level of informativeness. Among the informative characteristics, entropy bias has the strongest negative correlations.

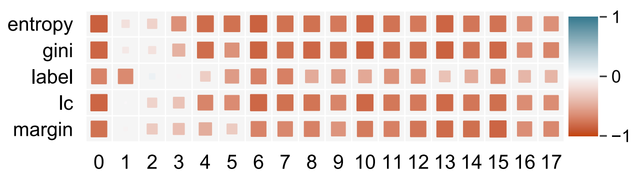


Fig. 4 Heat map of Pearson correlation between bias of characteristics adversarial robustness against the Auto attack. x-axis: stages in active learning; y-axis: characteristics. The size of each square corresponds to the magnitude of the correlation it represents. Dataset: SVHN; DNN: VGG8

6 Density-based robust sampling with entropy

6.1 Definition and algorithm

Inspired by our previous results, we propose a new acquisition function for ARAL: the density-based robust sampling with entropy (DRE). The key principle of DRE is to maintain a balance between selected data and unlabeled pool in terms of entropy distribution (represented by entropy PDF as defined in the previous section).

Concretely, in the i-th stage, let $\mathcal{L}_i$ and $\mathcal{U}_i$ be the labeled pool and unlabeled pool, respectively. A DNN parameterized by $\theta_{\mathcal{L}_i}$ is trained on $\mathcal{L}_i$. First, for each data $x \in \mathcal{U}_i$, we calculate the entropy (by Eq. (4)) of the prediction by this DNN and obtain a set of entropy scores:

$$\mathcal{A}_i = \{H(y, x; \theta_{\mathcal{L}_i}), x \in \mathcal{U}_i\} \tag{14}$$

Afterwards, we estimate the entropy distribution, $PDF_{\mathcal{A}_i}$, of $\mathcal{U}_i$ via the histogram method [69]. Correspondingly, each data is divided into a certain density interval based on its entropy score. Finally, we randomly select data from each density interval and add to $\mathcal{L}_i$. The number of data to be selected from the j-th density interval is determined by:

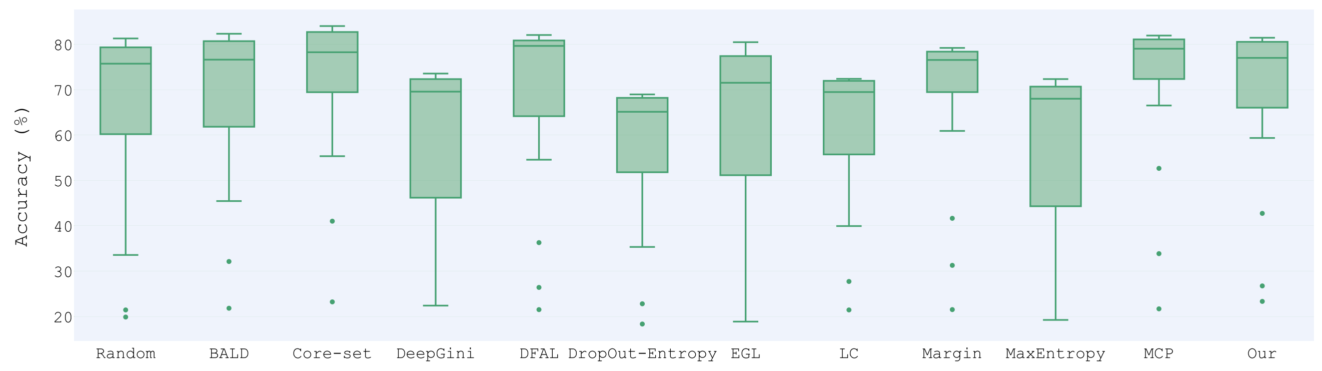
$$n_j = n * PDF_{\mathcal{A}_i}(j) \tag{15}$$

n is the number of labeling in each query (see *New* in Table 3). In this stage, θ_i will be updated by adversarial training on $\mathcal{L}_i$ and the selected data.

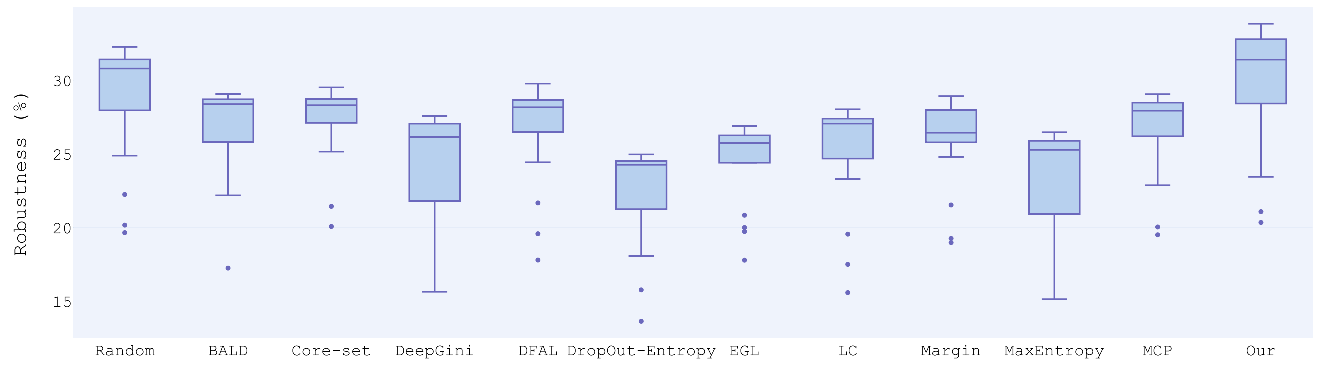
6.2 Effectiveness of DRE in active learning

DRE is evaluated using the same datasets, DNNs, and experimental protocol that are previously used in Section 5.

Results Figure 5 shows the box plots of accuracy and adversarial robustness by each acquisition of SHVN. Table 5 shows the accuracy and the robustness of the models produced by each acquisition function at one-third (1/3) and at the end ("last stage") of the ARAL process of all datasets and DNNs. DRE is competitive in terms of accuracy. At the end of the process, our acquisition function achieves the best accuracy for Fashion-MNIST and CIFAR-10/PreActResNet18, which is close to the best for MNIST and CIFAR-10/ResNet18, and ranks at about the middle for SVHN and CIFAR-10/VGG16. When it comes to robustness, DRE stands out and outperforms all the other acquisition functions for all models and datasets except CIFAR-10/VGG16 where it yields the robustness slightly lower than random sampling (− 0.39%). On the other datasets and models, DRE improves the robustness over random sampling on the other datasets and models by 0.75% up to 3.84%.



(a) Accuracy



(b) Robustness (Auto attack)

Fig. 5 Box plots of test accuracy (%) and robustness (%) against Auto attack of 12 acquisition functions in ARAL. The higher, the better. Dataset: SVHN. DNN: VGG8

We performed statistical tests to assess the statistical significance of the robustness improvement that DRE brings on all datasets, models, and learning stages. A Wilcoxon signed-rank test rejects the null hypothesis at the significance level of 5.00E-02 that there is no difference between DRE and any other acquisition function, with a p-value ranging from 8.15E-10 to 2.25E-03.

Conclusions DRE succeeds in consistently achieving better robustness than SOTA acquisition functions—and does so while remaining competitive in terms of accuracy. DRE, therefore, forms the baseline for ARAL.

6.3 DRE for test selection

While we previously showed that ARAL can train robust models *from scratch*, we study the adjacent problem of DL testing and retraining (DL T &R). DL T &R starts from a trained model and attempts to generate new test data that the model misclassifies. In turn, it uses these data to improve the model (through retraining) [3]. Test generation methods generally proceed in two steps: 1) selecting clean data to start from and 2) generating test data from the clean data. For the second step, one can rely on dedicated methods that research has proposed [72] or simply on adversarial attacks as [51].

The acquisition functions in active learning can also serve for test data selection (the first step mentioned above) and combine adversarial attacks to generate data for retraining. Hence, we investigate the capability of these functions to select data with which the robustness of the model will improve after retraining.

We utilize the same datasets and DNNs as previously. We pre-train the DNNs using standard training on the entire training set. Then, following established experimental protocols [3], we use a given acquisition function to select a budget test data (1% to 10% of the test set). We then generate adversarial examples from the selected data and retrain the DNNs using both the training data and the generated adversarial examples. Finally, we evaluate both the test accuracy and robustness of retrained DNNs.

Results Table 6 presents the accuracy and robustness of each model, before and after retraining, against the Auto attack. We draw the first conclusion that all acquisition functions are applicable as metrics in test selection. Compared with the baseline where no test data are added for training, they all manage to improve the robustness against

Table 5 Comparison of accuracy (Acc, %) and robustness (Rob, %) against the Auto attack between DRE and 11 acquisition functions

Acquisition Function	MNIST Lenet-5		Fashion-MNIST Lenet-5		SVHN VGG8		CIFAR-10 VGG16		CIFAR-10 ResNet18		CIFAR-10 PreActResNet18	
	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob	Acc	Rob
	1/3 stage											
BALD	97.15	50.13	73.09	24.15	65.51	24.06	60.68	28.17	60.61	29.02	64.79	29.29
DFAL	97.88	43.71	71.71	21.99	73.07	23.37	60.37	27.33	62.72	28.88	63.89	28.28
EGL	94.08	31.60	61.85	21.37	56.84	22.45	58.93	29.62	62.43	29.28	61.31	28.32
MaxEntropy	95.34	35.18	67.85	18.48	57.55	20.61	58.67	25.81	60.82	27.84	62.11	27.77
DropOut-Entropy	94.52	34.56	65.92	16.73	54.41	18.79	58.63	26.41	59.74	27.09	61.79	27.93
DeepGini	95.25	35.83	66.61	16.10	58.55	21.11	60.19	27.21	59.55	26.71	61.41	26.63
Core-set	97.46	42.77	72.33	23.51	71.10	23.34	61.21	26.99	64.30	28.24	63.59	29.49
LC	96.17	36.49	66.07	19.61	65.31	21.78	59.81	26.55	60.42	26.89	62.31	26.39
Margin	97.29	45.19	72.57	23.43	71.42	24.51	58.99	25.69	63.47	27.75	63.93	27.97
MCP	97.73	40.42	72.85	21.03	73.19	23.42	63.85	28.20	63.04	28.04	64.16	28.25
Random	97.51	54.10	71.84	32.07	65.48	26.19	59.24	29.47	63.33	30.32	63.90	31.01
DRE	**97.61**	**56.00**	**72.51**	**31.75**	**68.27**	**25.95**	**61.35**	**30.83**	**63.43**	**31.38**	**65.17**	**31.76**
	Last stage											
BALD	98.56	56.55	73.19	28.51	82.31	25.32	71.02	33.17	74.55	35.27	74.28	34.83
DFAL	98.43	52.45	72.37	18.11	81.79	23.15	78.11	35.48	81.07	36.91	74.47	33.74
EGL	98.75	52.92	63.71	25.64	79.95	22.00	70.81	33.19	73.99	35.86	73.57	34.81
MaxEntropy	98.47	51.88	64.41	18.71	70.98	21.57	66.71	29.37	71.74	32.01	71.82	32.18
DropOut-Entropy	98.62	52.29	63.07	18.59	68.70	20.98	64.67	30.15	70.08	31.65	70.69	31.47
DeepGini	98.41	51.23	65.93	18.08	72.27	23.25	65.89	28.19	72.50	32.67	73.22	31.83
Core-set	98.63	53.82	72.99	22.76	84.01	23.70	70.09	32.20	72.74	34.09	74.53	34.49
LC	98.48	50.89	70.29	18.11	72.17	22.68	66.39	29.91	72.60	32.21	73.57	31.64
Margin	98.65	53.80	73.43	18.36	77.55	21.37	68.85	31.40	70.91	33.65	73.22	33.24
MCP	98.71	51.71	73.41	19.02	81.59	22.31	68.25	30.73	74.19	33.35	74.39	34.14
Random	98.30	63.85	72.61	31.90	81.28	28.17	70.03	35.70	75.29	37.38	73.28	36.75
DRE	**98.59**	**67.69**	**73.79**	**33.79**	**81.41**	**29.81**	**69.35**	**35.31**	**74.99**	**38.66**	**74.69**	**37.86**

1/3 stage: the 8th, 9th, 6th, 16th stages for MNIST, Fashion-MNIST, SVHN, and CIFAR-10, respectively. *Last*: the last stage. Highlight in gray: DRE is better than a function

the Auto attack by up to 3.23% for SVHN and the three CIFAR-10 models. However, for MNIST and Fashion-MNIST, only DRE can increase the robustness of retrained DNNs by up to 2.58%. Under different sizes of budget, DRE achieves competitive accuracy (the difference is less than 2.03%) compared with the other 11 acquisition functions. Besides, concerning the robustness, DRE outperforms the others in most (116 of 144) cases by up to 8.21%.

Conclusions Although all acquisition functions are viable for DL testing and retraining, DRE yields higher robustness (by up to 8.21%) than any of the other functions.

7 Threats to validity

The internal threat of our work mainly comes from the implementation of compared acquisition functions and evaluation metrics. Regarding the 10 functions (except random sampling), we searched for the original implementations provided by the corresponding authors or the implementation by the other researchers, then carefully modified them into our framework based on PyTorch. We follow the suggestions in their implementations or publications for the involved parameters. In the robustness evaluation and statistical analysis, we adopt popular libraries, such as Torchattacks and SciPy.

The external threats to validity are related to the selection of datasets, DNNs, and compared acquisition functions. For the datasets and DNNs, we employ four publicly available datasets and six DNN architectures that are widely studied in the experiments of different works about active learning. Regarding the compared acquisition functions, we include random sampling and ten well-designed functions in the literature, which cover the basic and recently proposed ones. Besides, these ten functions exist in both the machine learning and software engineering (e.g. MCP and DeepGini) communities.

Table 6 Comparison of accuracy (Acc, %)and robustness (Rob, %) against the Auto attack with different budgets (1%, 4%) in test selection

Acquisition Function	**MNIST Lenet-5**				**Fashion-MNIST Lenet-5**				**SVHN VGG8**				**CIFAR-10 VGG16**				**CIFAR-10 ResNet18**				**CIFAR-10 PreActResNet18**			
	Acc		**Rob**		**Acc**		**Rob**		**Acc**		**Rob**		**Acc**		**Rob**		**Acc**		**Rob**		**Acc**		**Rob**	
	1%	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**	**1%**	**4%**
Baseline	*98.69*		*76.17*		*72.55*		*34.06*		*83.95*		*35.94*		*69.54*		*40.71*		*73.11*		*42.65*		*73.24*		*42.65*	
BALD	98.38	98.42	74.80	75.17	66.79	66.02	32.12	33.77	86.23	87.49	37.25	37.69	76.04	76.01	43.11	43.39	79.97	79.92	45.42	45.49	79.97	78.91	45.42	45.49
DFAL	98.23	98.53	74.22	75.73	65.64	65.53	32.28	33.61	86.34	86.64	36.61	35.78	75.27	75.73	43.01	42.88	80.39	80.31	44.86	45.01	80.39	78.89	44.86	45.01
EGL	98.26	98.30	73.96	73.60	64.93	65.16	30.54	31.91	86.13	88.43	35.51	35.12	75.77	76.47	42.81	42.60	80.37	79.09	44.53	44.49	80.37	79.12	44.53	44.49
MaxEntropy	98.25	98.42	74.99	75.68	63.77	64.07	32.11	30.76	86.35	87.42	36.34	35.96	76.14	76.27	42.67	42.87	80.53	80.29	45.09	44.79	80.53	79.01	45.09	44.79
DropOut-Entropy	98.39	98.40	75.49	75.37	65.33	66.19	31.17	32.05	85.64	87.69	36.65	36.36	75.69	76.03	43.28	42.87	80.15	80.17	45.13	45.11	80.15	79.29	45.13	45.11
DeepGini	98.32	98.19	74.65	74.43	67.48	63.65	31.37	33.43	86.71	87.01	35.60	36.14	76.09	75.77	42.97	43.07	80.32	80.09	45.04	44.78	80.32	79.01	45.04	44.78
Core-set	98.31	98.13	75.27	74.59	64.53	65.77	30.53	31.51	85.59	86.65	36.83	37.15	75.71	75.70	43.19	43.51	78.83	78.94	45.53	45.58	78.83	78.75	45.53	45.58
LC	98.21	98.33	74.03	75.29	64.86	65.71	31.83	32.03	86.71	87.67	35.78	35.99	75.64	75.57	42.91	42.89	79.97	79.99	45.17	45.00	79.97	78.85	45.17	45.00
Margin	98.31	98.40	75.07	75.15	65.27	65.19	32.17	31.35	86.71	86.45	36.27	36.38	75.74	75.84	42.87	43.19	80.06	78.68	45.01	44.98	80.06	78.89	45.01	44.98
MCP	98.35	98.50	75.35	75.79	67.71	67.37	32.71	32.03	87.09	87.69	35.77	36.00	75.19	76.55	43.13	43.13	79.97	80.29	45.20	45.15	79.97	78.90	45.20	45.15
Random	98.21	98.18	74.28	75.08	65.72	68.14	29.55	28.43	85.37	85.93	37.29	37.09	75.65	75.61	43.45	43.30	80.29	80.03	45.03	45.29	80.29	78.69	45.03	45.29
DRE	**98.32**	**98.54**	**76.52**	**76.88**	**69.46**	**69.64**	**34.56**	**36.64**	**86.58**	**86.40**	**37.36**	**38.46**	**76.34**	**75.76**	**43.62**	**43.94**	**82.54**	**82.60**	**45.58**	**45.40**	**82.54**	**78.72**	**45.58**	**45.40**

Baseline adversarially trained DNN using all training data. Highlight in gray: DRE is better than a function

The construct threats might be from the randomness and the robustness evaluation measures. To reduce the threat by randomness, each experiment is repeated three times and we present the average. To gauge the adversarial robustness, we adopt two commonly used attacks (PGD and Auto attack, which are the current standard for robustness evaluation in the literature) and added the black-box gradient-free square attack. Though we show the results for the Auto attack only, the results for the other attacks (available on our companion website) corroborate our findings. Indeed, as the Auto attack is stronger than both the PGD attack and square attack, the robustness obtained by an acquisition function is lower when using the Auto attack to evaluate. Correspondingly, the relative difference will be slightly larger. For example, DRE achieves better robustness than random sampling by up to 6.89% against the PGD attack (versus better by up to 3.84% robustness against the Auto attack). All detailed results are provided on our companion website [55]. Using other adversarial attacks, such as the fast gradient sign method (FGSM) [36] and iterative FGSM (i-FGSM) [73], will give the same conclusion.

8 Conclusion

We investigated the use of active learning for building robust deep learning systems. We conducted a comprehensive study with 11 existing acquisition functions and 15105 trained DNNs and revealed that, in adversarial-robust active learning (**ARAL**), random sampling achieves better robustness than existing functions but fails on accuracy. Via our analyses, we demonstrated that the selected data for training a robust model should be representative of the entire set and proposed DRE, the first acquisition function dedicated to RAL. Extensive experiments have demonstrated that DRE achieves better robustness than the 11 acquisition functions and achieves competitive accuracy. Besides, the DL testing and retraining experiments have demonstrated that DRE is suitable for this task and still outperforms the other acquisition functions. We hope that our formulation of the ARAL process, the experimental protocol we put in place, and the baseline we propose (DRE) will altogether inspire future research on developing robust DL models.

The main limitation of this work is the selection of datasets and DNNs and we mainly focused on the image domain. In the future work, we will investigate DRE for other domains, such as text classification and source code analysis. In addition, DRE is developed based on our experimental observations. We would like to seek theoretical strategies to perform adversarial-robust active learning.

9 Supplementary information

We release all our source code publicly available at https://github.com/testing-cs/robustAL.git. The full results are available at our project site https://sites.google.com/view/robust-al/.

Author Contributions All authors contributed to the targeting problem and results discussion. Experiments were conducted by Yuejun Guo. The first draft was written by Yuejun Guo and all authors provided comments.

Funding This work is supported by the Luxembourg National Research Funds (FNR) through CORE project C18/IS/12669767/STELLAR/LeTraon.

Declarations

Competing interests The authors declare that they have no conflict of interest.

Consent to participate All authors gave explicit consent to participate in this work.

Consent for publication All authors gave explicit consent to submit this paper and obtained the responsible authorities at SnT where the work has been carried out.

Data availability All the benchmark datasets used in this paper are publicly available and can be loaded in PyTorch. All the source code are publicly available at GitHub.

References

1. Yu S, Fang C, Yun Y, Feng Y (2021) Layout and image recognition driving cross-platform automated mobile testing. In: 43rd International Conference on Software Engineering, pp 1561–1571. IEEE
2. Alahmadi M, Khormi A, Parajuli B, Hassel J, Haiduc S, Kumar P (2020) Code localization in programming screencasts. Empir Softw Eng 25(2):1536–1572
3. Wang J, Chen J, Sun Y, Ma X, Wang D, Sun J, Cheng P (2021) Robot: robustness-oriented testing for deep learning systems, pp 300– 311
4. Ducoffe M, Precioso F (2018) Adversarial active learning for deep networks: a margin based approach (2018)
5. Settles B, Craven M, Friedland L (2008) Active learning with real annotation costs. In: NIPS Workshop on Cost-Sensitive Learning, 1
6. Lu H, Kocaguneli E, Cukic B (2014) Defect prediction between software versions with active learning and dimensionality reduction. In: 25th International Symposium on Software Reliability Engineering, pp 312– 322
7. Karlos S, Aridas C, Kanas VG, Kotsiantis S (2021) Classification of acoustical signals by combining active learning strategies with semi-supervised learning schemes. Neural Computing and Applications, pp 1–18
8. Sener O, Savarese S (2018) Active learning for convolutional neural networks: a core-set approach. In: International Conference on Learning Representations
9. Settles B, Craven M (2008) An analysis of active learning strategies for sequence labeling tasks. In: Conference on Empirical Methods in Natural Language Processing, pp 1070– 1079. Association for Computational Linguistics, USA
10. Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, Jackel LD, Monfort M, Muller U, Zhang J et al (2016) End to end learning for self-driving cars
11. Yuan Z, Lu Y, Wang Z, Xue Y (2014) Droid-sec: deep learning in android malware detection. In: ACM Conference on SIGCOMM, pp 371– 372. Association for Computing Machinery, New York, USA
12. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow IJ, Fergus R (2014) Intriguing properties of neural networks. In: International Conference on Learning Representations
13. Croce F, Andriushchenko M, Sehwag V, Debenedetti E, Flammarion N, Chiang M, Mittal P, Hein M (2020) RobustBench: a standardized adversarial robustness benchmark
14. Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A (2018) Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations, Vancouver, Canada
15. Ren K, Zheng T, Qin Z, Liu X (2020) Adversarial attacks and defenses in deep learning. Engineering 6(3):346–360
16. Tong S (2001) Active learning: theory and applications. In: PhD thesis, Stanford University
17. Settles B (2010) Active learning literature survey. Technical Report 1648, University of Wisconsin, Madison
18. Angluin D (1988) Queries and concept learning. Mach Learn 2(4):319–342
19. Atlas LE, Cohn DA, Ladner RE (1990) Training connectionist networks with queries and selective sampling. In: Advances in Neural Information Processing Systems, pp 566– 573. Citeseer
20. Lewis DD, Gale WA (1994) A sequential algorithm for training text classifiers. In: 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp 3– 12. Springer
21. Ramirez-Loaiza ME, Sharma M, Kumar G, Bilgic M (2017) Active learning: an empirical study of common baselines. Data Min Knowl Discov 31(2):287–313
22. Pereira-Santos D, Prudêncio RBC, de Carvalho AC (2019) Empirical investigation of active learning strategies. Neurocomputing 326:15–27
23. Sassano M (2002) An empirical study of active learning with support vector machines for japanese word segmentation. In: 40th Annual Meeting of the Association for Computational Linguistics, pp 505– 512
24. Prabhu A, Dognin C, Singh M (2019) Sampling bias in deep active classification: an empirical study. In: Conference on Empirical Methods in Natural Language Processing, pp 4049–4059
25. Chen J, Schein A, Ungar L, Palmer M (2006) An empirical study of the behavior of active learning for word sense disambiguation. In: Human Language Technology Conference of the NAACL, Main Conference, pp 120– 127
26. Settles B, Craven M (2008) An analysis of active learning strategies for sequence labeling tasks. In: Conference on Empirical Methods in Natural Language Processing, pp 1070– 1079. Association for Computational Linguistics, USA
27. Heilbron FC, Lee J-Y, Jin H, Ghanem B (2018) What do i annotate next? an empirical study of active learning for action localization. In: European Conference on Computer Vision, pp. 199– 216. Springer, Germany
28. Bowring JF, Rehg JM, Harrold MJ (2004) Active learning for automatic classification of software behavior. In: ACM SIGSOFT International Symposium on Software Testing and Analysis,

pp 195– 205. Association for Computing Machinery, New York, USA
29. Kocaguneli E, Menzies T, Keung J, Cok D, Madachy R (2013) Active learning and effort estimation: finding the essential content of software effort estimation data. IEEE Trans Softw Eng 39(8):1040–1053
30. Yu Z, Kraft NA, Menzies T (2018) Finding better active learners for faster literature reviews. Empir Softw Eng 23(6):3161–3186
31. Cambronero JP, Dang THY, Vasilakis N, Shen J, Wu J, Rinard MC (2019) Active learning for software engineering. In: ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software. Association for Computing Machinery, New York, USA
32. Yu Z, Theisen C, Williams L, Menzies T (2019) Improving vulnerability inspection efficiency using active learning. In: IEEE Transactions on Software Engineering (Early Access), 1–1
33. Yang X, Yu Z, Wang J, Menzies T (2021) Understanding static code warnings: an incremental ai approach. Expert Syst Appl 167:114134
34. Lu H, Cukic B (2012) An adaptive approach with active learning in software fault prediction. In: 8th International Conference on Predictive Models in Software Engineering, pp 79– 88. Association for Computing Machinery, New York, USA
35. Tu H, Yu Z, Menzies T (2020) Better data labelling with emblem (and how that impacts defect prediction). In: IEEE Transactions on Software Engineering, 1–1
36. Goodfellow IJ, Shlens J, Szegedy C (2015) Explaining and harnessing adversarial examples. In: International Conference on Learning Representations
37. Ilyas A, Santurkar S, Tsipras D, Engstrom L, Tran B, Madry A (2019) Adversarial examples are not bugs, they are features. In: Advances in Neural Information Processing Systems, Vancouver, BC, Canada, pp 125– 136
38. Schmidt L, Santurkar S, Tsipras D, Talwar K, Madry A (2018) Adversarially robust generalization requires more data. In: 32nd International Conference on Neural Information Processing Systems, pp 5019– 5031. Curran Associates Inc., Red Hook, USA
39. Andriushchenko M, Croce F, Flammarion N, Hein M (2020) Square attack: a query-efficient black-box adversarial attack via random search. In: European Conference on Computer Vision, pp 484– 501. Springer, Germany
40. Brendel W, Rauber J, Bethge M (2018) Decision-based adversarial attacks: reliable attacks against black-box machine learning models. In: International Conference on Learning Representations
41. Bhagoji,AN, He W, Li B, Song D (2018) Practical black-box attacks on deep neural networks using efficient query mechanisms. In: European Conference on Computer Vision. Springer, Germany
42. Croce F, Hein M (2020) Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: International Conference on Machine Learning, pp 2206–2216. PMLR
43. Samangouei P, Kabkab M, Chellappa R (2018) Defense-gan: protecting classifiers against adversarial attacks using generative models. In: International Conference on Learning Representations
44. Guo C, Rana M, Cisse M, van der Maaten L (2018) Countering adversarial images using input transformations. In: International Conference on Learning Representations
45. Xie C, Wang J, Zhang Z, Ren Z, Yuille A (2018) Mitigating adversarial effects through randomization. In: International Conference on Learning Representations
46. Papernot N, McDaniel P, Wu X, Jha S, Swami A (2016) Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE Symposium on Security and Privacy, pp 582– 597. Institute of Electrical and Electronics Engineers Inc., San Jose, United States
47. Warde-Farley D, Goodfellow I (2016) 11 adversarial perturbations of deep neural networks. 311
48. Chen J, Wu Z, Wang Z, You H, Zhang L, Yan M (2020) Practical accuracy estimation for efficient deep neural network testing. ACM Trans Softw Eng Method, 29(4)
49. Li Z, Ma X, Xu C, Cao C, Xu J, Lü J (2019) Boosting operational dnn testing efficiency through conditioning, pp 499– 509. Assoc Comput Mach, New York, USA
50. Wang Z, You H, Chen J, Zhang Y, Dong X, Zhang W (2021) Prioritizing test inputs for deep neural networks via mutation analysis. In: 43rd International Conference on Software Engineering, pp 397– 409
51. Feng Y, Shi Q, Gao X, Wan J, Fang C, Chen Z (2020) Deepgini: prioritizing massive tests to enhance the robustness of deep neural networks. In: 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp 177– 188. Association for Computing Machinery, New York, USA
52. Ma W, Papadakis M, Tsakmalis A, Cordy M, Traon YL (2021) Test selection for deep learning systems. ACM Trans Softw Eng Method 30(2):13–11322
53. Meng L, Li Y, Chen L, Wang Z, Wu D, Zhou Y, Xu B (2021) Measuring discrimination to boost comparative testing for multiple deep learning models. In: 43rd International Conference on Software Engineering, pp 385– 396
54. Shen W, Li Y, Chen L, Han Y, Zhou Y, Xu B (2020) Multiple-boundary clustering and prioritization to promote neural network retraining. In: International Conference on Automated Software Engineering, pp 410– 422. Association for Computing Machinery, New York, United States
55. Guo Y (2021) Project website of robust active learning. https://sites.google.com/view/robust-al/
56. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
57. Xiao H, Rasul K, Vollgraf R (2017) Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms
58. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng A (2011) Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning
59. Krizhevsky A (2009) Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto
60. Mayer C, Timofte R (2020) Adversarial sampling for active learning. In: IEEE Winter Conference on Applications of Computer Vision, pp 3060– 3068
61. Athalye A, Carlini N, Wagner D (2018) Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples. ICML 80:274–283
62. Kim H (2020) Torchattacks: a pytorch repository for adversarial attacks
63. Wald A (1945) Statistical decision functions which minimize the maximum risk. Ann Math 46:265–280
64. Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27(3):379–423
65. Houlsby N, Huszár F, Ghahramani Z, Lengyel M (2011) Bayesian active learning for classification and preference learning
66. Gal Y, Islam R, Ghahramani Z (2017) Deep bayesian active learning with image data. In: 34th International Conference on Machine Learning, pp 1183– 1192. JMLR.org, Sydney, NSW, Australia
67. Scheffer T, Decomain C, Wrobel S (2001) Active hidden markov models for information extraction. Adv Intell Data Anal, pp 309–318. Springer, Berlin, Heidelberg

68. Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1(6):80–83
69. Silverman BW (1998) Density estimation for statistics and data analysis, 1st edn. Routledge, New York
70. Lin J (1991) Divergence measures based on the Shannon entropy. IEEE Trans Inf Theory 37(1):145–151
71. Student (1908) Probable error of a correlation coefficient. Biometrika 6(2/3):302–310
72. Pei K, Cao Y, Yang J, Jana S (2017) Deepxplore: automated whitebox testing of deep learning systems. In: 26th Symposium on Operating Systems Principles, pp 1– 18. Association for Computing Machinery, New York, USA
73. Ren H, Huang T (2020) Adversarial example attacks in the physical world. In: International Conference on Machine Learning for Cyber Security, pp 572– 582. Springer, Cham