Towards Exploring the Limitations of Active Learning: An Empirical Study

Qiang Hu[1], Yuejun Guo[1], Maxime Cordy[1], Xiaofei Xie[2*], Wei Ma[1], Mike Papadakis[1], and Yves Le Traon[1]
[1]University of Luxembourg, Luxembourg [2]Nanyang Technological University, Singapore

Abstract—**Deep neural networks (DNNs) are increasingly deployed as integral parts of software systems. However, due to the complex interconnections among hidden layers and massive hyperparameters, DNNs must be trained using a large number of labeled inputs, which calls for extensive human effort for collecting and labeling data. Spontaneously, to alleviate this growing demand, multiple state-of-the-art studies have developed different metrics to select a small yet informative dataset for the model training. These research works have demonstrated that DNN models can achieve competitive performance using a carefully selected small set of data. However, the literature lacks proper investigation of the limitations of data selection metrics, which is crucial to apply them in practice. In this paper, we fill this gap and conduct an extensive empirical study to explore the limits of data selection metrics. Our study involves 15 data selection metrics evaluated over 5 datasets (2 image classification tasks and 3 text classification tasks), 10 DNN architectures, and 20 labeling budgets (ratio of training data being labeled). Our findings reveal that, while data selection metrics are usually effective in producing accurate models, they may induce a loss of model robustness (against adversarial examples) and resilience to compression. Overall, we demonstrate the existence of a trade-off between labeling effort and different model qualities. This paves the way for future research in devising data selection metrics considering multiple quality criteria.**

Index Terms—**deep learning, data selection, active learning, empirical study**

I. Introduction

Deep learning (DL) has achieved tremendous success in various cutting-edge application domains, such as image processing [1], machine translation [2], autonomous vehicles [3], and robotics [4]. Two key elements to achieve high-performing predictions are well-designed deep neural networks (DNNs) (with appropriate architecture and parameters) and a carefully chosen set of labeled training data. However, data labeling is expensive and time-consuming because it requires a large amount of human effort. For example, that it took more than 3 years to prepare the first version of the ImageNet [5] dataset. Thus, acquiring labeled data is seen as a major obstacle to the widespread adoption of DL [6].

One established solution to reduce data labeling cost is active learning [7], i.e., incremental methods to select informative subsets of training data to undergo labeling in a way that the produced model is as accurate as if it was trained on all data. With active learning, engineers can thus compromise labeling effort with model performance (e.g., classification accuracy). There has been much research on devising active learning methods, each relying on different data selection metrics. These metrics [8]–[11] typically exploit information of the DNN under training (e.g., its gradient or uncertainty) to select the most informative data to label next.

On the other hand, recent work on DL testing and debugging [12]–[19] have proposed different metrics for *test generation* and *test selection*, i.e., the problem of selecting test data that are more likely to be misclassified by the model [20]. As in active learning scenarios, these test data can then be used to improve the model (by retraining).

The proliferation of data selection metrics (coming from active learning and testing) makes it challenging for engineers to decide which one they should use. Indeed, the different metrics have been evaluated on a restricted set of problems (mostly image classification datasets) under incomparable experimental settings (different models and labeling budget) [18], [19]. There is, therefore, a need for a comprehensive study of all these data selection metrics on a common ground involving different classification tasks.

Another gap in the current body of knowledge is that most experimental studies evaluate the metric wrt. the test accuracy of the trained models.[1] Other key quality indicators have been ignored, such as the model robustness to adversarial attacks and the model performance after compression. The lack of consideration for these indicators raises practical issues when deploying DL models trained using active learning (consider, e.g., biomedical image segmentation [22] or mobile DL applications [23]). Hence, one should make sure that active learning can produce models whose quality is not limited to classification accuracy but extends to *all* quality indicators relevant for the use case.

To fill these gaps, in this paper, we conduct a comparative empirical study to explore the potential limitations of active learning. Our study involves 15 data selection metrics evaluated over 5 datasets (2 image classification tasks and 3 text classification tasks), 10 DNN architectures, and 20 labeling budgets (ratio of training data that can be labeled). Specifically, our study aims to answer the following four research questions:

- **RQ1: How effective are the different data selection metrics for producing accurate models?** We answer this question by measuring how fast (i.e., how many training data are required) the accuracy of the trained model converges to

*corresponding author.

[1]Rarely, some studies [21] measure empirically the robustness of the model to adversarial attacks.

the accuracy of the fully trained model (i.e., trained with the full training set). Besides, we employ random selection as a baseline to compare the effectiveness of each metric. Our results indicate significant differences between metrics (up to 45.9% of test accuracy), especially when less than 50% of the training data is used.

• **RQ2: How robust are models trained with active learning?** We answer this question by measuring the theoretical robustness of the trained models (using the CLEVER score [24]) as well as their empirical robustness against multiple adversarial attacks. For the image classification task, our results reveal a gap between the fully trained models and those trained with active learning (up to 1.49 CLEVER score and 23.39% of success rate), whereas there are no such differences in the text classification task.

• **RQ3: Do models trained using active learning maintain accuracy after compression?** Model compression, a well-known technique to embed DL models in resource-constrained devices, has the downside effect of reducing the accuracy due to precision loss in the computed model weights. We investigate whether models trained with active learning are more sensitive to this phenomenon than fully trained models. Our results reveal that, for the image classification task, training on 50% of data entails a loss in test accuracy up to 7.47% higher compared to training with the full dataset.

• **RQ4: What is the relationship between the amount of training data, and model robustness and accuracy after compression?** We conduct additional experiments where we increase the data budget of data selection metrics. Our results indicate that with the growth of training data, the robustness of the model will also increase. The model can achieve similar robustness with the fully trained model only using 35% data, and even outperform it when more data are used. This indicates that the training process promoted by active learning can be used as an effective way to increase robustness. As for model compression, there appears to be no relationship between accuracy decay induced by compression and data budget.

With our extensive empirical study, we provide practical guidance to engineers in balancing the benefits of data selection metrics with their potential side effects. That is, we show and quantify the existence of a trade-off between the efficiency of model training (in particular, the data labeling effort) and model properties of interest (viz. robustness and accuracy after quantization). Doing so, we also open research directions to explore this trade-off and new data selection metrics aimed towards the different quality criteria.

In summary, the main contributions of this paper are:

- We conduct the largest empirical study that investigates the effectiveness of training data selection metrics on different classification tasks (image and text).
- Beyond (test) accuracy, we explore the effects of reducing the number of training data on the adversarial robustness of the models and their accuracy after compression. We, therefore, reveal the potential effects that active learning can have on these quality indicators.
- Thereby, we reveal a potential trade-off between labeling cost and the aforementioned quality indicators. This paves the way for future research in designing multi-objective data selection metrics aiming at optimizing this trade-off under a constrained labeling budget.

The rest of this paper is organized as follows. Section II introduces some background knowledge of this work. Section III presents an overview of the study. Section IV introduces the implementation and empirical configurations. Section V details the results of our study. Section VI presents the related works, and Section VII concludes this paper.

II. Background

We briefly introduce the background related to our work, including DNNs, test selection and active learning, adversarial attacks on DNNs, and model compression.

Throughout the paper, X refers to the training set for a N-class classification DNN. $x \in X \subseteq \mathbb{R}^d$ is an input and x' is its adversarial example. y and y_x indicate the true and predicted labels, respectively. $p_i(x)$, $0 \leq i \leq N$, represents the predicted probability of x belonging to the ith class, and correspondingly, $y = \underset{i=1:N}{\arg\max}\,(p_i(x))$.

A. Deep Neural Networks

In general, a DNN consists of multiple layers, i.e., an input layer, several hidden layers, and an output layer. As shown in Figure 1, each layer comprises a number of neurons (color circles). The neuron with the parameters, also called a unit or a node, is the basic entity of computation of a DNN. It receives information from the input data or the other neurons and computes an output by an activation function. The training process of a DNN is mainly about tuning the parameters to reach a minimum prediction error concerning true labels. In this paper, we focus on the classification task where the output of a DNN classifier is the probability of belonging to each category given an input. For instance, the input data in Figure 1 is predicted to be in class *Dog* with a probability of 0.90.

Generally, there are two typical types of DNNs, i.e., Feed-Forward Neural Networks (FNNs) and Recurrent Neural Networks (RNNs). In an FNN, the information only moves in the forward direction from the input layer to the output layer. This type of DNNs is widely used in image processing applications. On the other hand, an RNN utilizes different inter-units (i.e., memory cells, control units) to propagate the input information in a backward way within an RNN layer, allowing the network to retain knowledge. RNNs usually deal with sequential data processing due to their ability to capture temporal information of the data. In this work, we study both FNNs and RNNs.

B. Active Learning

Active learning, a well-known concept in both the software engineering (SE) community and machine learning (ML) community, trains a model incrementally with several steps. In a typical active learning procedure, in the beginning, a model is randomly initialized. In each step of the training, it selects a few data from the unlabeled dataset to label and

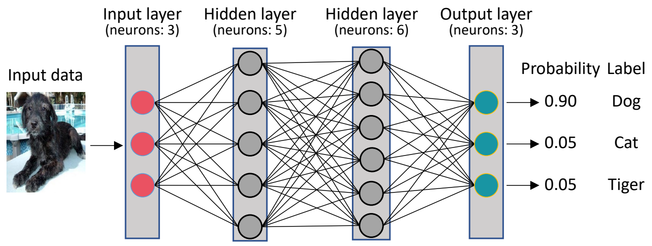

Input layer (neurons: 3)
Hidden layer (neurons: 5)
Hidden layer (neurons: 6)
Output layer (neurons: 3)
Input data
Probability
Label
0.90
Dog
0.05
Cat
0.05
Tiger

Fig. 1. An example of a DNN (feed-forward neural networks) classifier.

then retrains the model for better performance. In other words, the goal for each step is to reduce the cost of labeling as much as possible by selecting the most informative data to annotate while improving the accuracy of a pre-trained model. Therefore, active learning can help in model evolution. For instance, in reality, a model needs to be updated over time due to the rapid growth of new unlabeled data.

C. Test Selection

In the traditional field of SE, test selection has attracted extensive research attention for a long time. For instance, test selection is widely applied in regression testing [25] which aims at reducing the size of test suites since executing the entire set is remarkably costly. Depending on the purpose, the selected test suites can help to eliminate redundant test cases (test suite minimization), test relevant changed parts of the software (test case selection), and locate faults early (test case prioritization). A similar concept to test selection is feature selection [26]–[28] that is thoroughly studied as well. The difference is that feature selection focuses on seeking the optimal features in a data set to allow efficient execution, while test selection tries to reduce the size of data.

In practice, for DL-based software systems, collecting unlabelled data is easy and cheap but labeling all of them requires heavy work and specific domain knowledge. Following the same spirit of test selection in traditional SE, recent research proposed some test selection metrics for DL systems, like the neuron coverage-based test selection. These metrics select the most useful subset of unlabeled test data for both testing DNNs and improving the performance of pre-trained DNNs via retraining. We consider that these test selection metrics are promising to apply in active learning for two main reasons. First, the procedure of test selection and then retraining, generally speaking, can be regarded as one-step active learning – active learning being by nature an incremental process. Second, the test selection metrics share the same goal of determining the most useful data given a DNN. For instance, an active learning process could measure this utility as the number of new neurons that these data activate.

D. Adversarial Attacks on DNNs

DNNs have been proven to be vulnerable to adversarial examples, which causes considerable security concerns [29]. Therefore, evaluating the ability (robustness) of a DNN to deal with adversarial examples is a crucial part of DNN testing. An adversarial example is a variant of input data by introducing a small perturbation that is hardly recognized by human beings but can easily fool DNNs. The perturbation is not just random noise but carefully calculated by some adversarial attacks. In this study, we employ three powerful attacks (FGSM, JSMA, C&W) for image classification, and two (PWWS and DWB) for text classification.

• **FGSM**. Goodfellow *et al.* [30] proposed the fast gradient sign method (FGSM) to generate adversarial examples, which is the first gradient-based and one of the most used attacks. FGSM crafts x' by $x' = x + \epsilon \cdot \text{sign}\left(\nabla_x J\left(x, y\right)\right)$ where ϵ controls the perturbation size. $sign\left(\cdot\right)$ is the sign function. The sign of a real number is -1 for a negative value, 1 for a positive value, and 0 for value 0. $\nabla_x J\left(x, y\right)$ computes the gradient of the training loss J given x and its true class y.

• **JSMA**. The Jacobian-based Saliency Map Attack (JSMA) [31] first computes a saliency map by the Jacobian matrix. The map presents how influential each feature (e.g., each pixel) of the input is to predict a particular class. Through exploiting this map with targeting a class that does not match the true class of a given test sample x, JSMA modifies x at where the pixels have high-saliency values to generate an adversarial example that might be classified within a predefined threshold γ (maximum fraction of features being perturbed).

• **C&W**. Proposed by Carlini and Wagner [32], C&W is known as one of the strongest adversarial attacks. It uses a designed loss function f to replace the training loss, then generates the adversarial example x' which minimizes $dis\left(x, x'\right) + c \cdot f(x')$ where dis is a distance metric and c is a constant that controls the distance and the confidence of x'.

• **PWWS**. The probability weighted word saliency (PWWS) [33], a word-level attack, generates text adversarial examples by replacing original words with synonyms searched from a lexical database (e.g., WordNet [34]). Given a word, PWWS selects a substitution concerning two factors. 1) How does the classification change if replacing this word with a substitution? 2) How does the classification change if ignoring this word?

• **DWB**. The DeepWordBug (DWB) [35] is a black-box char-level adversarial attack, which follows two steps to generate adversarial examples. First, DWB determines the words to modify by the change of predictions before and after replacing the words with *Unknown* tokens. Second, it modifies the selected words slightly by changing at most two letters per word through a predefined manner, e.g., alter *world* to *wor1d*.

E. Model Compression

Model compression is important for efficient model deployment in software systems, especially when using large models, e.g., big DNNs. The goal of model compression is to reduce the model size while maximally maintaining the performance in terms of accuracy before practical deployment. Next, we introduce the two compression techniques considered in our work.

• **Model pruning**. Model pruning lightens the model by removing redundant and unimportant connections that have little impact on the performance. In this study, we apply two basic pruning strategies, the weight-level pruning [36] and the neuron-level pruning [37]. The weight-level method sets the weights that are smaller than a threshold to zero. The neuron-level pruning removes the neurons that have a high chance of being inactivated. Correspondingly, the related connections (weights) are also eliminated.

• **Model quantization**. In DNN, weights are stored in the 32-bit floating-point format. To compress the model, the quantization technique converts the weights from 32-bit into low-bit (e.g., 8-bit integer).

III. Overview

We first introduce the three-phase design of our empirical study, then present data selection metrics, datasets and models, and evaluation measures that are studied in our work.

The secure life cycle of deep learning is composed of some key stages [38] from the requirement analysis and data-label pair collection to the maintenance and evolution of the deep learning model. This paper studies the effect of the data collection (i.e., active learning) on the model development and deployment (i.e., the model quality). Figure 2 gives an overview of our study, which consists of three phases, effectiveness analysis of model training with different data selection metrics, adversarial robustness analysis of the trained model, and performance analysis after model deployment. Overall, we compare the models trained using the entire dataset and a subset of data selected by a specific data selection metric.

A. Study Design

More specifically, in the first phase (data collection), we compare the effectiveness of each data selection metric for training a model. Using different metrics (e.g., Entropy, Margin), we iteratively select a subset of training data and train the model, then observe the convergence trend of the test accuracy. For comparison, we also train two baseline models using entire training data and randomly selected data, respectively.

In the second phase (model development), we evaluate the robustness of the trained model. In our study, we apply multiple metrics (e.g., empirical robustness, CLEVER score) to compare the robustness of models trained with the selected and the model trained by the entire training data.

In the third phase, we focus on the model performance (test accuracy) in the model deployment phase. In general, a DNN model usually consists of a huge number of parameters, e.g., a VGG16 model requires about 258MB of hard disk memory. Before deploying such a large DNN model into the hardware (e.g., mobile devices), one has to consider the performance including the required memory and inference speed. We adopt two well-known techniques (model quantization and model pruning) to optimize a trained model. Then we evaluate the accuracy of the optimized models. Besides, we study the impact of training data size on the robustness of models and the accuracy of optimized models.

Finally, based on the results of our empirical study, we provide some practical guidelines for the usage of active learning on different tasks and summarize some potential research directions.

B. Datasets and DNN Models

We conduct experiments with two popular image datasets (MNIST [39] and CIFAR-10 [40]) and three widely used text datasets (IMDb [41], TagMyNews [42], and Yahoo! Answers [43]). MNIST includes 10-class grayscale images of handwritten digits. The dataset includes 60000 and 10000 training and test data, respectively. CIFAR-10 is a collection of 10-class color images (e.g., airplane, bird). The dataset consists of 50000 and 10000 training and test data, respectively. IMDb is a dataset including movie reviews widely used for text sentiment analysis (binary classification). Both the training and test sets include 25000 text reviews. TagMyNews provides news headlines (text) in 7 categories (e.g., Sport, Business). We randomly collect 20000 data for training and 2000 data for testing. Yahoo! Answers consists of text data of 10 topic categories (e.g., Society & Culture, and Science & Mathematics). We obtain this data from [33] directly with 3560 training data and 889 test data.

For each dataset, we employ two DNN architectures to reduce the model-dependent influence on the results. For MNIST, we use two well-known convolutional neural networks LeNet-1 and LeNet-5 [44], and for CIFAR-10, we select two models NiN [45] and VGG16 [46], both of which achieve high accuracy. For IMDb, TagMyNews, and Yahoo! Answers, we use two types of RNNs, LSTM, and GRU, derived from a base model [47]. Our companion website presents all details about the models and training parameters [48].

C. Data Selection Metrics

Various data selection metrics have been proposed and verified to reduce the labeling effort. Note that the data selection metric is also known as the acquisition function in the ML community. We include 14 data selection metrics from both the ML community (8 metrics) and the SE community (6 metrics). We first introduce the ones from the ML community.

• **Entropy** [8] considers the uncertainty of data using the prediction output. This metric is based on the Shannon entropy of the prediction probability:

$$\underset{x\in X}{\arg\max}\left(\sum_{i=1}^{N} p_i\left(x\right)\log p_i\left(x\right)\right) \tag{1}$$

• **Margin** [8] computes a score for each data by the difference between its top-2 prediction probabilities:

$$Margin\left(x\right) = p_k\left(x\right) - p_j\left(x\right) \tag{2}$$

where $k = \underset{i=1:N}{\arg\max}\left(p_i\left(x\right)\right)$ and $j = \underset{i=\{1:N\}/k}{\arg\max}\left(p_i\left(x\right)\right)$. The training data with low scores will be selected for training.

• **K-center** [10] firstly divides data into K groups via some unsupervised machine learning methods (e.g., K-means clustering), then selects the center of each group (if not enough, consider the data that are close to the center). These selected data are regarded as the representative of the entire group.

• **Expected Gradient Length (EGL)** [49] assumes that the model has no knowledge of the true label of data in advance.

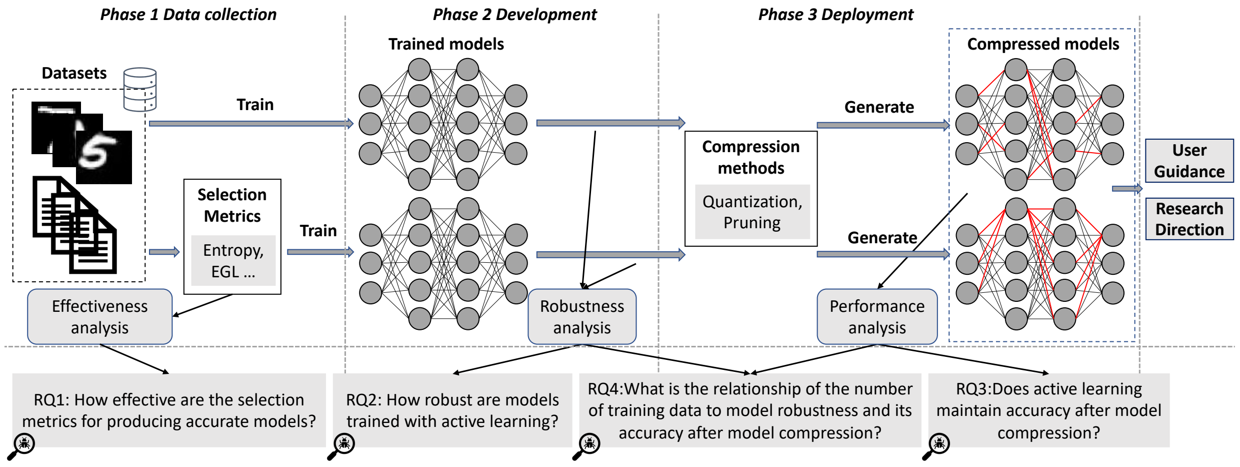

Phase 1 Data collection
Phase 2 Development
Phase 3 Deployment
Datasets
Train
Trained models
Selection Metrics
Entropy, EGL ...
Train
Compression methods
Quantization, Pruning
Generate
Generate
Compressed models
User Guidance
Research Direction
Effectiveness analysis
Robustness analysis
Performance analysis
RQ1: How effective are the selection metrics for producing accurate models?
RQ2: How robust are models trained with active learning?
RQ4:What is the relationship of the number of training data to model robustness and its accuracy after model compression?
RQ3:Does active learning maintain accuracy after model compression?

Fig. 2. Overview of experiment design

For each data, it computes the expectation of the gradients by assigning all the labels to x. The data that have great expectations are selected. EGL can be presented as follows:

$$EGL\left(x\right)=\sum_{i=1}^{N}p_i\left(x\right)\left|\left|\nabla_x J\left(x,i\right)\right|\right| \tag{3}$$

where $||.||$ is the Euclidean norm. $\nabla_x J\left(x,i\right)$ is the gradient of the loss J given x and label i.

• **Bayesian Active Learning by Disagreement (BALD)** [50] applies dropout to select the most uncertain data:

$$\underset{x\in X}{\arg\max}\left(1-\frac{count\left(mode\left(y_x^1,...,y_x^T\right)\right)}{T}\right) \tag{4}$$

where T is the number of applying dropout to the model.

• **Entropy-dropout** and **Margin-dropout** [9] apply dropout and calculate the average score of Entropy and Margin over all dropout models, e.g., Entropy-dropout is defined as

$$\underset{x\in X}{\arg\max}\frac{\sum_{i=1}^{T}Entropy\left(x^i\right)}{T} \tag{5}$$

where $Entropy\left(x^i\right)$ is the entropy of x at the i-th time.

• **Adversarial active learning (Adversarial_al)** [11] leverages an adversarial attack, DeepFool, to facilitate selecting data. Concretely, the model predicts labels for each data, then DeepFool introduces perturbations to each data until reaches an adversarial example. Finally, the data requiring smaller perturbations will be selected. Intuitively, such data are close to the decision boundary of the model.

Next, we introduce the metrics from the SE community.

• **Neuron Coverage (NC)** [16] selects data that have the largest neuron coverage:

$$\underset{x\in X}{\arg\max}\frac{\left|\left\{neu\right|neu\in Neurons\wedge activate\left(neu,x\right)\right\}\left|\right.}{\left|Neurons\right|} \tag{6}$$

where $activate\left(neu,x\right)$ indicates that the neuron neu is activated by x, namely, the output of neu is greater than a predefined threshold. We highlight that this is a variant of the original metric to fit active learning. The original NC aims at selecting data to cover all the neurons, however, in practice, just a few data are enough to reach 100% coverage [51]. Namely, after a few selection steps, all the neurons have been activated at least once and the marginal increase of coverage will be 0. Thus, we apply this variant to fit active learning.

• **K-Multisection Neuron Coverage (KMNC)** [15] improves NC by splits the lower and upper bounds of a neuron's output into k sections. Instead of using the coverage of neurons, it considers the coverage of sections. Similar to NC, the data with high coverage will be selected.

• **Multiple-Boundary Clustering and Prioritization (MCP)** [18] is an extension of Margin. First, it divides data into various "boundary areas" based on the top-2 predicted classes, e.g., for data with 10 classes, there are $P\left(10,2\right)=90$ permutations of pairwise classes. Next, MCP selects data from each area based on Margin.

• **DeepGini** [19] selects the most uncertainty data by:

$$\underset{x\in X}{\arg\max}\left(1-\sum_{i=1}^{N}\left(p_i\left(x\right)\right)^2\right) \tag{7}$$

• **Likelihood-based Surprise Adequacy (LSA)** and **Distance-based Surprise Adequacy (DSA)** [17] measure the surprise adequacy by the dissimilarity between a test data and the training set. The difference between LSA and DSA is that LSA uses kernel density estimation to estimate the surprise adequacy, while DSA uses Euclidean distance. Both select data with the largest surprise adequacy.

We remark that although the initial purpose of these SE metrics is not for active learning, their underlying selection criteria share similarities with the criteria that active learning metrics rely on. For example, both Entropy (active learning metric) and DeepGini (SE metric) metrics select the uncertain data based on the output probabilities. Therefore, we believe that it is necessary to consider them in our study. Besides, no existing research has revealed that whether these SE metrics fit in active learning or how they perform.

D. Evaluation Metrics

Effectiveness. To evaluate the effectiveness of a selection metric, we use random selection as the baseline and calculate the difference of accuracy between the models trained using random selection and this metric. The difference is defined by:

$$diff = \sum_{i=1}^{steps} \left(Acc^{i}_{TS} - Acc^{i}_{TR} \right) \tag{8}$$

where $steps$ is the total number of training steps. Acc^{i}_{TS} is the test accuracy of the model trained by a selection metric, and Acc^{i}_{TR} is by random selection. $diff$ shows the degree of a metric outperforming random selection.

Adversarial robustness. The robustness of DNNs refers to the ability to cope with adversarial examples. The robustness of DNNs can be evaluated in multiple ways. We introduce two popular methods of robustness estimation, Empirical Robustness [52] and CLEVER score [24]. **1) Empirical Robustness** This method quantifies the robustness of a DNN model through the success rate of crafting adversarial examples by an attack. In practice, given a DNN and a test set S, an attack attempts to craft an adversarial example based on each test data. The attack success rate is the ratio of adversarial examples successfully generated:

$$ASR = \frac{|\{x \mid x \in S \wedge y_{x'} \neq y\}|}{|S|} \tag{9}$$

Recall that $y_{x'}$ and y are the predicted label of x' and true label of x, respectively. A small ASR indicates a robust DNN. **2) CLEVER Score** The Cross-Lipschitz Extreme Value fornEtwork Robustness (CLEVER) score calculates the lower bound for crafting an adversarial example given an input, which is the least amount of perturbation required to fool a DNN model. A great CLEVER score indicates a robust DNN.

IV. Implementation and Configuration

Experimental environment. This project is implemented based on Keras [53] and TensorFlow [54] frameworks. We run all experiments on a high-performance computer cluster except the model pruning and quantization. Each cluster node runs a 2.6 GHz Intel Xeon Gold 6132 CPU with an NVIDIA Tesla V100 16G SXM2 GPU. For the model pruning and quantization, we conduct the experiments on a MacBook Pro laptop with macOS Big Sur 11.0.1 with a 2GHz GHz Quad-Core Intel Core i5 CPU with 16GB RAM.

Active learning. We initialize an empty labeled pool and an unlabeled pool with all the unlabeled data. Besides, we initialize the with random weights. Next, in each training step, a fixed number (step size) of data are selected from the unlabeled pool by a specific metric. Then the selected data are merged into to labeled pool after annotation. The DNN is updated by retraining using all the data in the labeled pool. The

TABLE I
Configurations of active learning

Dataset	Model	Step size	Stop point	Entire training size
MNIST	LeNet-1, LeNet-5	500	10000	60000
CIFAR-10	NiN, VGG16	2500	25000	50000
IMDb	LSTM, GRU	500	12500	25000
TagMyNews	LSTM, GRU	500	12500	25000
Yahoo!Answers	LSTM, GRU	500	12500	25000

TABLE II
Configurations of adversarial attacks. For the definition of the parameters (ϵ, γ, c, dis), please refer to Section II.

Dataset	FGSM	JSMA	C&W	CLEVER
	ϵ	γ	c	dis
MNIST	0.1, 0.2, 0.3	0.09, 0.1, 0.11	9, 10, 11	L-1, L-2, L-inf
CIFAR-10	0.01, 0.02, 0.03	0.01, 0.02, 0.03	0.1, 0.2, 0.3	

procedure terminates when the size of the labeled pool reaches a threshold (stop point). Table I lists the detailed settings. Note that previous works have different parameter settings [8], [9], [20], [55], we balance these settings to set up our experiments. We implement data selection metrics based on [20].

Robustness. We use two public libraries, Foolbox [56] for empirical robustness evaluation and ART [57] for CLEVER score calculation. In empirical robustness, we apply three attack methods, i.e., FGSM, JSMA, and C&W, for the image classification task, and conduct three groups of experiments with different parameters as in [58]. For the text classification task, we use PWWS and DWB with the default setting in [33], [35] to attack the text-related models. By default, only the correctly classified data undertake the attacks. In CLEVER score, the setting of c and dis follows the configuration in [58]. One difference is that we use 500 test data to calculate the CLEVER score, while [58] used 50. The detailed information is shown in Table II. Note that the setting of CLEVER works both for the image and text classification tasks.

Model compression In model quantization, we apply two lightweight frameworks, CoreML [59] and TensorFlowLite [60], to transform DNNs into different bit-level versions. For CoreML, we use three levels, 2-bit, 4-bit, and 8-bit. For TensorflowLite, we apply 8-bit and 16-bit level quantizations in our models since it only supports these two levels. In model pruning, for both the weight- and neuron-level, we prune a DNN into six compress versions with different degrees, from 10% to 60% at 10% intervals, using the implementations by [36], [37]. As a reminder, model pruning is conducted after the model is well-trained.

Last but not least, to reduce the influence of randomness, we repeat each experiment three times and compute the average results. In total, we trained and evaluated more than 2000 models in this study. The source code can be found on [2].

V. Experimental results

In this section, we present the results and answer each research question mentioned in Section I. In the remaining parts, we remark that "TE", "TS", and "Random" represent

[2]https://github.com/code4papers/ALempirical

TABLE III
EFFECTIVENESS WITH RESPECT TO RANDOM SELECTION (BASELINE). THE RESULTS ABOVE THE BASELINE ARE HIGHLIGHTED IN GRAY.

Metric	MNIST		CIFAR-10		IMDb		TagMyNews		Yahoo!Answers		Average
	LeNet-1	LeNet-5	NiN	VGG16	LSTM	GRU	LSTM	GRU	LSTM	GRU	
Entropy	-131.88	-183.84	26.14	3.91	49.41	44.13	29.63	37.72	3.19	-20.06	-14.16
Margin	69.82	28.78	14.44	6.29	-	-	29.68	36.23	26.10	17.44	28.59
K-center	-23.28	28.63	21.00	-7.67	62.94	39.15	25.60	36.63	9.52	15.97	20.84
EGL	-11.71	-14.13	-30.46	-17.65	3.62	-31.02	-14.55	-14.85	-155.46	-82.90	-36.91
BALD	27.84	-25.22	21.98	9.04	57.62	53.44	28.42	27.43	20.02	6.41	22.69
Entropy-dropout	-81.71	-165.96	22.88	3.18	64.81	48.91	30.23	34.43	2.29	17.51	-2.34
Margin-dropout	43.57	14.67	1.19	6.56	-	-	26.83	41.00	25.68	29.92	23.67
Adversarial _al	33.23	22.69	-77.73	29.15	-	-	-	-	-	-	1.835
NC	-26.95	-182.01	-2.76	-10.95	36.28	12.69	30.77	25.27	-72.29	-93.89	-28.38
KMNC	3.80	-153.02	0.40	12.69	34.02	39.39	-4.35	0.83	-392.01	-375.40	-83.36
MCP	25.66	8.39	14.28	13.17	-	-	24.53	30.70	11.59	17.02	18.16
DeepGini	-74.07	-213.23	21.33	12.98	-	-	30.62	31.98	20.47	-2.59	-21.56
LSA	19.83	22.37	3.41	8.97	-4.53	5.10	-1.22	-1.32	-385.71	-359.77	-69.28
DSA	9.26	23.62	1.37	10.49	5.54	5.34	-7.87	-0.78	-392.39	-359.47	-70.48

training using the entire dataset, the subset selected by data selection metrics, and randomly selected data, respectively. Note that due to the page limitation, we only illustrate a part of the results in this paper, and the complete results are provided as supplementary material [48]. The conclusions we draw below generalize to all studied datasets/subjects and DNNs.

A. *RQ1: Effectiveness of Data Selection Metrics*

First, we show, in Figure 3, the performance of using different data selection metrics to train a DNN that achieves the same test accuracy as the fully trained model. For comparison, a horizontal dashed line in each sub-figure represents the accuracy of the fully trained model. Overall, most metrics manage to produce models with the same accuracy as the fully trained model by using only 7% to 50% of training data.

Next, Table III offers a more detailed comparison of these metrics, showing their effectiveness taking random selection as a baseline (measured using Equation 8). Surprisingly, only three metrics (Margin, Margin-dropout, MCP) significantly outperform the baseline in all cases. On the contrary, in most cases (9 out of 10), EGL is worse than random selection. There are some metrics like LSA and DSA, which outperform the baseline on the image classification task but underperform the baseline on the text classification task, e.g., on Yahoo-LSTM and Yahoo-GRU, and the difference reaches up to 392.39. We conjecture that LSA and DSA tend to select data based on similarity or distance. However, different from image data, the two texts data (sentence or document) might have a big difference in the hidden space even if they are in the same category. In this case, the inter-output of the model against these two data can be completely different which makes LSA and DSA select the wrong data.

Specifically, considering the image classification task, we observe that in addition to Margin, Margin-dropout, and MCP, two other metrics, LSA and DSA, can always outperform the baseline. What's more, on LeNet-1 and LeNet-5, half of the metrics are worse than the random baseline. Especially, Entropy, Entropy-dropout, NC, and DeepGini get high negative differences, e.g., -131.88 for Entropy on LeNet-1, which means these four metrics are much worse than random selection. Turn to sub-figures 3(a) and 3(b), in the first few training steps (where the big difference comes from), these four metrics achieve much less test accuracy than the others (also random selection). One explanation is that Entropy, Entropy-dropout, and DeepGini try to find the most uncertain data. However, such uncertain data are hard to be learned by models that are not well-trained.

On the other hand, for the text classification task, the output probability-based metrics (e.g., Margin and BALD) perform better than the coverage and surprise adequacy-based metrics. Extremely, LSA performs worse than the baseline on 5 (out of 6) models. Besides, on model Yahoo_LSTM, the most efficient metric (Margin) is much better than the worst one (KMNC) with a 418.49% test accuracy gap. These results reflect that both coverage and surprise adequacy based-metrics are not suitable for the text classification task.

Answer to RQ1: The nature of the classification task significantly affects the effectiveness of multiple data selection metrics (e.g., LSA and DSA). Therefore, the limitation of active learning experiments to a single target task – even with multiple datasets – constitutes a critical threat to external validity. Some metrics, like Margin, Margin-dropout, and MCP, consistently perform well across all labeling budgets, models, and tasks.

B. *RQ2: Adversarial Robustness*

We then study the robustness of models trained by different data selection metrics against adversarial attacks. Figure 4 illustrates the empirical robustness of the models against various attacks. Once again, we have to distinguish the two types of tasks since the results of different metrics are greatly biased on the tasks. For the image classification task, the TE models are usually more robust than the TS models. However, for the text classification task, the TS models are in some cases more robust than the TE models (ad vice-versa). We conjecture that two factors may affect the robustness of the models: (i) the number of data used for training and (ii) the training process. In active learning, the early selected data are trained more times during incremental learning. Thus, the most informative data (according to the data selection metrics) have a larger influence on the model weights and, in turn, impact its robustness. On the other hand, since the selected data can be representative of the entire dataset to some extent, the difference between the TS and TE models is small. Taking VGG16 as an example, the difference varies from 1.67% to 17.21% in FGSM, from 1.42% to 3.21% in C&W, and from 0.32% to 10.54% in JSMA. Another observation that reinforces our hypothesis regarding the importance of the training process is that none of the metrics performs consistently better than random selection. We investigate this hypothesis in the RQ4 experiments, where we consider different labeling budgets.

Table IV lists the CLEVER score of different models. As for the text classification task, models trained with active learning either yield a small improvement (less than 0.58 CLEVER score) or offer inconsistent benefit (either increasing or decreasing the CLEVER score, depending on the considered metric and norm distance). Besides, none of the data selection metrics improves over the random selection, even the three metrics which performed better in terms of effectiveness (viz.

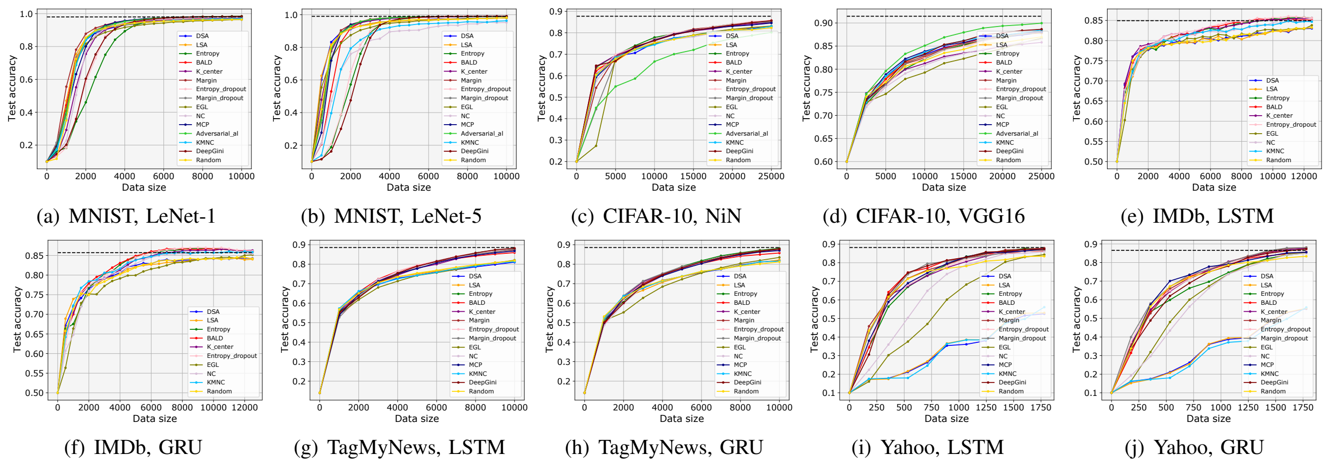
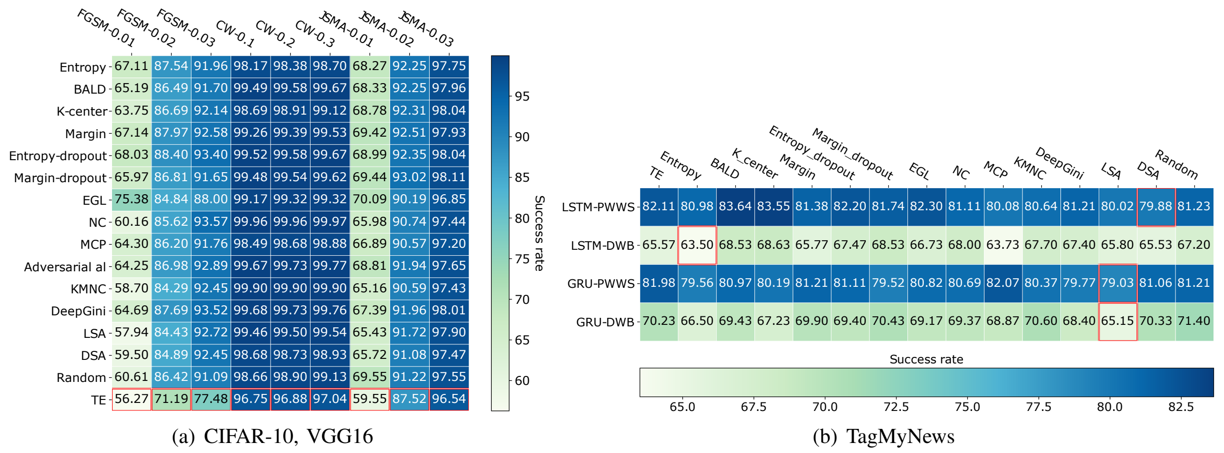


Fig. 3. Evolution of the test accuracy (y-axis) achieved by different data selection metrics given the number (x-axis) of training data.



	FGSM-0.01	FGSM-0.02	FGSM-0.03	CW-0.1	CW-0.2	CW-0.3	JSMA-0.01	JSMA-0.02	JSMA-0.03
Entropy	67.11	87.54	91.96	98.17	98.38	98.70	68.27	92.25	97.75
BALD	65.19	86.49	91.70	99.49	99.58	99.67	68.33	92.25	97.96
K-center	63.75	86.69	92.14	98.69	98.91	99.12	68.78	92.31	98.04
Margin	67.14	87.97	92.58	99.26	99.39	99.53	69.42	92.51	97.93
Entropy-dropout	68.03	88.40	93.40	99.52	99.58	99.67	68.99	92.35	98.04
Margin-dropout	65.97	86.81	91.65	99.48	99.54	99.62	69.44	93.02	98.11
EGL	75.38	84.84	88.00	99.17	99.32	99.32	70.09	90.19	96.85
NC	60.16	85.62	93.57	99.96	99.96	99.97	65.98	90.74	97.44
MCP	64.30	86.20	91.76	98.49	98.68	98.88	66.89	90.57	97.20
Adversarial al	64.25	86.98	92.89	99.67	99.73	99.77	68.81	91.94	97.65
KMNC	58.70	84.29	92.45	99.90	99.90	99.90	65.16	90.59	97.43
DeepGini	64.69	87.69	93.52	99.68	99.73	99.76	67.39	91.96	98.01
LSA	57.94	84.43	92.72	99.46	99.50	99.54	65.43	91.72	97.90
DSA	59.50	84.89	92.45	98.68	98.73	98.93	65.72	91.08	97.47
Random	60.61	86.42	91.09	98.66	98.90	99.13	69.55	91.22	97.55
TE	56.27	71.19	77.48	96.75	96.88	97.04	59.55	87.52	96.54

	TE	Entropy	BALD	K_center	Margin	Entropy_dropout	Margin_dropout	EGL	NC	MCP	KMNC	DeepGini	LSA	DSA	Random
LSTM-PWWS	82.11	80.98	83.64	83.55	81.38	82.20	81.74	82.30	81.11	80.08	80.64	81.21	80.02	79.88	81.23
LSTM-DWB	65.57	63.50	68.53	68.63	65.77	67.47	68.53	66.73	68.00	63.73	67.70	67.40	65.80	65.53	67.20
GRU-PWWS	81.98	79.56	80.97	80.19	81.21	81.11	79.52	80.82	80.69	82.07	80.37	79.77	79.03	81.06	81.21
GRU-DWB	70.23	66.50	69.43	67.23	69.90	69.40	70.43	69.17	69.37	68.87	70.60	68.40	65.15	70.33	71.40

Fig. 4. Adversarial attack success rate (%) of VGG16 and TagMyNews. The number in each cell represents the success rate and the color gives a straightforward visual comparison of different values. The most robust model is framed by a red rectangle. The lower the success rate, the better robustness.

Margin, Margin-dropout, and MCP). Overall, these results corroborate our previous findings. That is, for the image classification task, active learning yields less robust models.

TABLE IV
THE CLEVER SCORE OF CIFAR-10 (VGG16) AND TAGMYNEWS (LSTM). THE RESULTS THAT ARE BETTER THAN THE TE MODEL ARE HIGHLIGHTED IN GRAY. THE HIGHER SCORE, THE BETTER ROBUSTNESS.

Meric	**CIFAR-10 (VGG16)**			**TagMyNews (LSTM)**		
	L-1	L-2	L-inf	L-1	L-2	L-inf
Etropy	3.7143	0.3289	0.0055	3.6466	1.1559	0.0531
Margin	3.6607	0.2136	0.0051	3.5496	2.0390	1.6712
K-center	3.6475	0.3132	0.0057	3.7970	0.9840	0.0132
EGL	4.2577	0.6996	0.0037	3.6881	0.7235	0.0276
BALD	3.8096	0.3176	0.0048	3.7424	1.3236	2.9914
Etropy-dropout	3.6881	0.1495	0.0075	3.4033	0.7465	0.4027
Margin-dropout	3.8213	0.3103	0.0047	3.5453	0.8562	0.0059
Adversarial_al	4.2606	0.5242	0.0032	-	-	-
NC	4.2036	0.7017	0.0031	3.9315	0.9729	0.1879
KMNC	4.2628	0.6442	0.0340	3.1824	0.4198	0.0041
MCP	4.2955	0.4984	0.0041	3.7244	1.1289	0.8011
DeepGini	4.2616	0.5116	0.0031	3.4139	2.6286	1.6805
LSA	4.2449	0.7477	0.0041	3.8102	0.7150	0.0077
DSA	4.2567	0.6576	0.0035	3.7198	1.0144	0.0102
Random	3.9665	0.6499	0.0057	3.6295	1.6243	0.0065
TE	4.3636	0.7399	0.8053	3.3600	0.9910	1.6684

Answer to RQ2: For the image classification task, models trained with active learning can have lower robustness than models trained with the entire dataset, and the difference can be up to 1.49 CLEVER score and 23.39% attack success rate. Therefore, experimental studies should involve evaluation metrics beyond clean accuracy. For the text classification task, the results are inconsistent across attacks (empirical robustness) and distance norm (CLEVER score), indicating that other factors are at play when it comes to robustness, e.g., the training process.

C. RQ3: Test Accuracy After Model Compression

We compare the test accuracy of a model produced by different data selection metrics before and after compression. Table V shows the results of VGG16 (image classification) and TagMyNews-LSTM (text classification) by quantization.

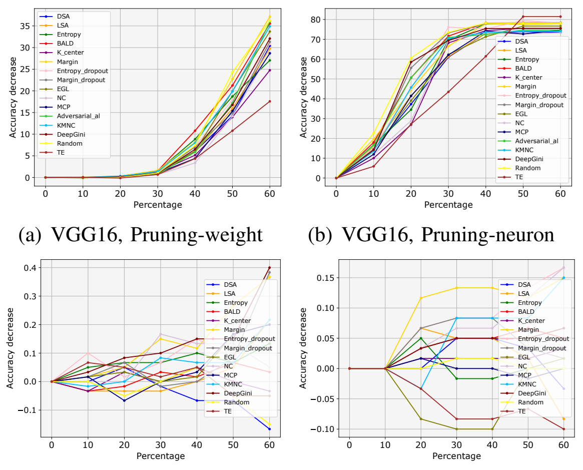
On VGG16, the accuracy of the 2-bit compressed models drops significantly by at least 76.32%, no matter it was fully trained or with active learning. By contrast, on TagMyNews-LSTM, the accuracy decay using 2-bit is relatively small (less than 9.42%). The reason could be that, *1)* the quantization process has less impact on the LSTM layer, or *2)* the text data is less sensitive to the precision of weights. In both cases, the compressed models achieve the same accuracy as the original models with 16-bit (expect Margin and Entropy-dropout On TagMyNews-LSTM). Specifically, compared with random selection, on both VGG16 and TagMyNews-LSTM, no metric always outperforms the baseline. Comparing with the TE model, we found that for the image classification task, the compressed TE model always maintains higher (with a gap up to 7.47%) test accuracy. However, for the text classification task, the TS model sometimes loses more test accuracy than the TS models after quantization. Surprisingly, with 2-bit quantization, the TE model gets the largest accuracy decay (-9.42%).

TABLE V
THE CHANGE OF TEST ACCURACY OF CIFAR-10 (VGG16) AND TAGMYNEWS (LSTM) BEFORE AND AFTER MODEL QUANTIZATION. THE BEST AND WORST RESULTS ARE HIGHLIGHTED IN GRAY AND ORANGE, RESPECTIVELY.

	CIFAR-10 (VGG16)					**TagMyNews (LSTM)**			
	CoreML			TFLite		CoreML			TFLite
	2-bit	4-bit	8-bit	8-bit	16-bit	2-bit	4-bit	8-bit	16-bit
Entropy	-78.13	-4.77	-0.09	-0.53	0	-2.13	-0.2	0	0
Margin	-78.59	-3.06	-0.14	-0.76	0	-2.5	-0.28	-0.03	-0.02
K-center	-76.32	-2.28	-0.81	-5.18	0	-1.72	-0.1	+0.02	0
EGL	-76.72	-2.35	-0.02	-0.63	0	-0.22	-0.18	+0.12	0
BALD	-77.99	-2.34	-0.01	-0.2	0	-3.9	-0.03	+0.02	0
Etropy-dropout	-77.92	-4.53	-0.76	-0.48	0	-2.82	-0.1	-0.03	-0.02
Margin-dropout	-78.02	-1.56	-0.01	-0.39	0	-2.57	-0.22	-0.02	0
Adversarial_al	-79.77	-2.05	-0.05	-3.2	0	-	-	-	-
NC	-77.17	-8.85	-0.22	-2.39	0	-2.57	-0.12	+0.02	0
KMNC	-78.92	-1.65	-0.06	-2.67	0	-2.88	-0.22	+0.03	0
MCP	-79.72	-1.56	-0.02	-3.64	0	-4.17	-0.07	0	0
DeepGini	-80.2	-3.47	0	-3.02	0	-8.73	-0.13	-0.03	0
LSA	-78.16	-1.74	-0.01	-2.53	0	-4.1	-0.12	-0.02	0
DSA	-78.28	-1.99	-0.04	-2.88	0	-2.88	0	+0.02	0
Random	-77.82	-1.74	-0.24	-1.77	0	-2.32	-0.42	-0.13	0
TE	-81.36	-1.38	0	-0.03	0	-9.42	-0.07	0	0

Figure 5 depicts the result by weight- and neuron-level pruning. In general, with pruning more weights and neurons, the test accuracy decreases gradually up to 36.20% and 81.46% on VGG16, respectively. However, on TagMyNews-LSTM, the accuracy changes negligibly by increasing or decreasing up to 0.4%. The reason might be that these two pruning methods can only affect the Convolutional layer and the Dense layer, while our LSTM models only contain one Dense layer to output the final prediction probability. Looking into VGG16, the neuron-level pruning affects the accuracy more than the weight-level for all the data selection metrics, which suggests that in practical applications, the engineers should consider more the weight-level pruning than the neuron-level to minimize the accuracy loss. Among these data selection metrics, DeepGini and NC are always better than random selection. Besides, for the image classification task, the TE model outperforms all TS models with a gap up to 33.4%. However, for the text classification task, the TE model has no advantage of maintaining test accuracy over TS models after pruning.

Answer to RQ3: Model compression inconsistently affects the performance of the models trained with active learning and no data selection metric provides satisfactory results across all tasks. For the image classification task, after compression, the fully trained models hold higher test accuracy than the models trained with active learning, and the gap can be up to



(a) VGG16, Pruning-weight (b) VGG16, Pruning-neuron

(c) TagMyNews-LSTM, Pruning-weight (d) TagMyNews-LSTM, Pruning-neuron

Fig. 5. The change of test accuracy (y-axis) after model pruning with different degrees (x-axis), i.e. the proportion of weights and neurons being pruned.

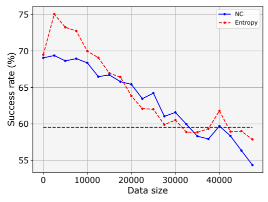
7.47% (quantization) and 33.4% (pruning). On the contrary, for the text classification task, the fully trained models have no advantage of test accuracy over the models trained with active learning after model compression.

D. RQ4: Impact of Training Data Size

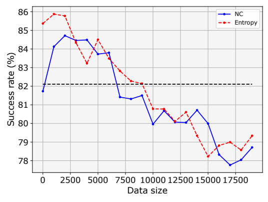
From Sections V-B and V-C, we found that the data selection metrics tend to produce less robust models and introduce greater changes of accuracy after model compression. We further extend our experiments to investigate the impact of the data size on the quality (e.g., adversarial robustness and test accuracy after compression) of models. Figure 6 shows the results on VGG16 and TagMyNews-LSTM. For adversarial attacks, we use JSMA-0.01 for VGG16 and PWWS for TagMyNews-LSTM, respectively. For model compression, we employ the 4-bit quantization by CoreML. On both models, we show the results by two data selection metrics: the prediction probability-based Entropy and neuron coverage (NC).

Figure 6(a) and Figure 6(b) show that TS models become more robust with more training data added, especially, the models could be more robust than the fully trained models in the end. According to the results, we can see that to achieve similar robustness with fully trained models, more than 60% of training data are required for the image classification task, while only 35% of training data is enough for the text classification task. This corroborates our hypothesis (see RQ2) that the training process used by active learning (which makes data selected earlier go through more training iterations than data selected later) can yield more robust models compared to a traditional training process involving all data from the beginning. This finding also opens the perspective of using such an active learning process in conjunction with common methods to improve robustness, such as adversarial training.

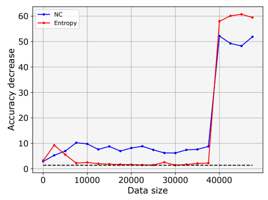
Figure 6(c) and Figure 6(d) show the test accuracy decay after model compression. For the text classification task, the difference is negligible throughout the increase in data size.



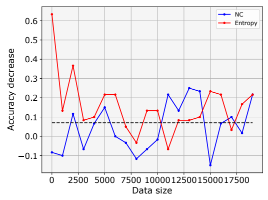
(a) VGG16, Robustness



(b) TagMyNews-LSTM, Robustness



(c) VGG16, Quantization



(d) TagMyNews-LSTM, Quantization

Fig. 6. Impact of training data size (x-axis) on the performance (y-axis) (success rate or accuracy decrease) of model. The horizontal dashed line shows the attack success rate or change of test accuracy of a fully trained model.

For the image classification task, the decay of accuracy keeps stably lower than 10% until the data size reaches 37500, where the accuracy decreases significantly by up to 60.73%. These results reveal that the data size is not a key factor in the test accuracy decay on model compression.

Answer to RQ4: Increasing the labeling budget of active learning mitigates the loss in robustness compared to a fully trained model. For example, using at least 60% training data for the image classification task (respectively, 35% for the text classification task) yields models with the same robustness as the fully trained model. However, training with more data does not change our previous conclusions regarding the inconsistent effect of model compression.

E. Discussion

We highlight our novel findings first, then discuss some practical guidance and research directions accordingly.

Novel findings and user guidance. 1) Finding: previous studies were limited to test accuracy, the image classification task, and did not compare metrics from both the ML and SE communities. We reveal that the benefits of active learning are highly task-dependent. Some data selection metrics (LSA and DSA) are highly affected by the nature of classification tasks, while some (Margin, Margin-dropout, and MCP) can achieve consistently high performance. Guidance: engineers need to choose data selection metrics according to specific tasks. **2)** Finding: the limitation of active learning also exists in the adversarial robustness and model compression, however, the evaluation was missing in the literature. We found that the active learning process has some potential but notable impact on these indicators, e.g., for the image classification task, the fully trained model is more robust than the model trained by active learning. Guidance: engineers are recommended to consider all these indicators when using active learning.

Research directions. 1) Since no existing data selection metric can perform well on all the considered objectives, an interesting research direction is to design data selection metrics that can optimize multiple qualities such as accuracy, robustness, and accuracy after compression. Regarding robustness, an adequate metric should also effectively integrate with an adversarial training process, minimizing the amount of data to form which adversarial examples are generated to increase the model robustness. **2)** Our study focuses on two types of classification tasks (image and text). Recently, DL systems have been applied in some SE-related tasks, such as source code function prediction and automatic program repair. Exploring how existing data selection metrics perform on these tasks is a potential research direction. Besides, proposing a source code-oriented data selection metric for this kind of system could be another contribution.

F. Threats to Validity

First, threats to validity may lie in the selected datasets and DNN models. Regarding the datasets, we employ five popular datasets across both image and text classification tasks in our study. As for the DNNs, we consider, in total, ten architectures (two for each dataset) to alleviate the model-dependent issue. Though the models we use perform well on the chosen tasks, an interesting direction of future work is to repeat the study on more complex model architectures, such as transformer-based models for text classification [61], and observe whether the trends remain.

Second, the parameter configuration may induce a threat to validity. Regarding the parameters in active learning, there is no universal rule for choosing the best settings. For instance, previous studies [8], [9], [20], [55] utilize different settings of labeling budget and stop strategy. We mitigate this threat to validity in two ways: (1) we took the best and most common practices from the literature to design our active learning process and settings, and (2) our research questions concern the specific impact of some parameters (e.g. RQ4 studies the impact of the stop criteria). For robustness evaluation, we follow a recent study [58] to set a range of perturbation sizes for different adversarial attacks. As for model compression, our study involves multiple settings to reduce the potential bias of the results.

Third, due to the space limitation, we only report the results of two datasets covering the image and text classification tasks. Nonetheless, we remark that the reported conclusion is generalized to all the datasets and DNNs. For example, for RQ2, in total, 31 out of 36 settings, the fully trained image-related models are more robust than the actively trained models, which is consistent with our finding. Besides, we report all our complete results on our project site [48].

VI. RELATED WORK

We review related works in three aspects: data selection in DL systems, empirical study on active learning, and empirical study for DL systems.

Data selection in DL systems. We have already witnessed the success of data selection in SE problems, such as systematic literature review [62], defect prediction based on static code attributes [28], software cost modeling [63], and software fault prediction [64]. Meanwhile, in the ML community, many data selection metrics have been proposed mainly for active learning. Recently, Ren *et al.* [65] surveyed the active learning metrics and categorized them into uncertainty-based [8], [66], deep Bayesian active learning [9], density-based [10], [67], and other methods [11], [68], [69]. On the other hand, the SE community has proposed some DL testing criteria as well as data selection metrics. Inspired by the program coverage, Pei *et al.* [16] proposed the basic neuron coverage criterion, which can be used as a data selection metric. Then, DeepGauge [15] introduced other DL test criteria, e.g., KMNC, Neuron Boundary Coverage. Kim *et al.* [17] proposed surprise guided testing metrics based on the similarity between the training data and test data. Moreover, some prediction probability based data selection metrics [18], [19], [70] are also proposed. Most recently, Wang *et al.* [71] proposed a robustness-oriented data selection metric, however, their metric can only select data that are generated by adversarial attacks, it is out of our consideration. In our work, we comprehensively studied almost all these metrics not only on their effectiveness but also the potential limitations.

Empirical study on active learning. Active learning has been widely studied over recent years, and some empirical studies of active learning have also been conducted from different domains. Ramirez-Loaiza *et al.* [72] utilized several performance measures, e.g., accuracy, F1 score, and AUC, to evaluate active learning baselines. However, all these measures are based on the correctness of classification, while our study includes more evaluation methods, such as the adversarial robustness and the performance change by model compression. Pereira-Santos *et al.* [73] conducted a large-scale empirical study of active learning for three machine learning algorithms, C4.5, SVM, and 5NN. Similarly, the evaluation is limited to the classification correctness quantified by the Area under the Learning Curve (ALC). In addition, some works target specific tasks by active learning. Settles *et al.* [49] analyzed active learning for sequence labeling tasks such as information extraction and document segmentation. For these specific problems, the evaluation mainly lies in the learning curve and runtime. Yu *et al.* [62] empirically studied existing active learning techniques for literature reviews, then proposed a novel one that outperforms the existing metrics concerning the recall vs. studies reviewed curve. Chen *et al.* [74] studied the behavior of active learning for the word sense disambiguation task. However, they only considered two basic data selection metrics, entropy and margin, and the evaluation only involves accuracy. Heilbron1 *et al.* [75] explored active learning for the action localization task, which also only considered a very limited number of (3) data selection metrics and compared the performance by ALC based on the correctness. Manabu [76] studied active learning with support vector machines (SVMs) for natural language processing. In their study, they only compared their proposed metrics with random selection based on accuracy. Besides, the finding is mainly that SVM active learning is suitable for Japanese word segmentation.

To sum up, compared with previous empirical studies, our study focuses on the application of active learning in deep learning systems and is the first one that studied both image classification and text classification tasks. More importantly, our work is the only one that analyzes the impacts of active learning on two important testing aspects during DNN deployment, the adversarial robustness and the performance of DNNs after model compression. Moreover, to the best of our knowledge, our study is the first that evaluates the data selection metrics proposed by the SE community for active learning.

Empirical study for DL systems. Recently, multiple empirical research studies focus on exploring the DL issues that are hard to be solved in theory. Guo *et al.* [77] studied the performance difference between different DL frameworks as well as the model changes after model migration. Zhang *et al.* [78] and Chen *et al.* [79] studied the challenges in the deployment phase of DL systems. Both of them revealed that developing a DL system is harder than developing software systems. Ma *et al.* [20] performed a comparison study on different data selection metrics. They investigated the ability of each metric to identify misclassified input and improve the test accuracy by retraining. What's more, Zhang *et al.* [80] conducted a comparative study about the capability of different uncertainty metrics in distinguishing adversarial examples and benign examples. Different from these works, our empirical study focuses on exploring the limitations of active learning, especially the potential limitations of the model trained by active learning, compared to the fully trained model.

VII. Conclusion

In this paper, we conducted a comprehensive empirical study to explore the limitations of active learning. In total, more than 2000 models for image classification and text classification tasks have been trained and systematically evaluated. The results reveal that, when using active learning to train a model, different data selection metrics yield models of significantly different quality (in accuracy and robustness). For the image classification task, a model trained with active learning can achieve competitive test accuracy but suffers from robustness loss and are less to compression. However, these downsides rarely occur in text classification models. Also, we further studied the relationship between the data budget and the quality of a trained model. We found that the robustness of the model increases with the amount of training data, and ultimately reaches the robustness of the fully trained model. Based on these findings, we provided some practical guidance as well as research directions. We believe that our work could give engineers and researchers some valuable insights into the whole secure life cycle of deep learning, especially in the data collection and model evolution steps.

Acknowledgments

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

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[2] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado *et al.*, "Google's multilingual neural machine translation system: enabling zero-shot translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017.

[3] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th International Conference on Software Engineering*. New York, NY, USA: Association for Computing Machinery, 2018, pp. 303–314.

[4] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: a survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: a large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, 2009, pp. 248–255.

[6] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: a big data-ai integration perspective," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2019.

[7] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[8] D. Wang and Y. Shang, "A new active labeling method for deep learning," in *International Joint Conference on Neural Networks (IJCNN)*. Beijing, China: IEEE, 2014, pp. 112–119.

[9] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1183–1192.

[10] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *International Conference on Learning Representations*. Vancouver, BC, Canada: OpenReview.net, 2018.

[11] M. Ducoffe and F. Precioso, "Adversarial active learning for deep networks: a margin based approach," *arXiv preprint arXiv:1802.09841*, 2018.

[12] X. Xie, L. Ma, F. Juefei-Xu, M. Xue, H. Chen, Y. Liu, J. Zhao, B. Li, J. Yin, and S. See, "Deephunter: a coverage-guided fuzz testing framework for deep neural networks," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2019, pp. 146–157.

[13] X. Xie, L. Ma, H. Wang, Y. Li, Y. Liu, and X. Li, "Diffchaser: Detecting disagreements for deep neural networks." in *IJCAI*, 2019, pp. 5772–5778.

[14] X. Du, X. Xie, Y. Li, L. Ma, Y. Liu, and J. Zhao, "Deepstellar: Model-based quantitative analysis of stateful deep learning systems," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 477–487.

[15] L. Ma, F. Juefei-Xu, F. Zhang, J. Sun, M. Xue, B. Li, C. Chen, T. Su, L. Li, Y. Liu, J. Zhao, and Y. Wang, "Deepgauge: multi-granularity testing criteria for deep learning systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018, pp. 120–131.

[16] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: automated whitebox testing of deep learning systems," *Commun. ACM*, vol. 62, no. 11, pp. 137—-145, 2019. [Online]. Available: https://doi.org/10.1145/3361566

[17] J. Kim, R. Feldt, and S. Yoo, "Guiding deep learning system testing using surprise adequacy," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 1039–1049.

[18] W. Shen, Y. Li, L. Chen, Y. Han, Y. Zhou, and B. Xu, "Multiple-boundary clustering and prioritization to promote neural network retraining," in *IEEE/ACM International Conference on Automated Software Engineering (ASE)*. Melbourne, Australia: IEEE, 2020, pp. 410–422.

[19] Y. Feng, Q. Shi, X. Gao, J. Wan, C. Fang, and Z. Chen, "Deepgini: prioritizing massive tests to enhance the robustness of deep neural betworks," in *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 177–188. [Online]. Available: https://doi.org/10.1145/3395363.3397357

[20] W. Ma, M. Papadakis, A. Tsakmalis, M. Cordy, and Y. L. Traon, "Test selection for deep learning systems," *ACM Trans. Softw. Eng. Methodol.*, vol. 30, no. 2, Jan. 2021. [Online]. Available: https://doi.org/10.1145/3417330

[21] X. Gao, R. K. Saha, M. R. Prasad, and A. Roychoudhury, "Fuzz testing based data augmentation to improve robustness of deep neural networks," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. IEEE, 2020, pp. 1147–1158.

[22] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Z. Chen, "Suggestive annotation: a deep active learning framework for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer. Cham: Springer International Publishing, 2017, pp. 399–407.

[23] M. Xu, J. Liu, Y. Liu, F. X. Lin, Y. Liu, and X. Liu, "A first look at deep learning apps on smartphones," in *The World Wide Web Conference*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 2125–2136.

[24] T.-W. Weng, H. Zhang, P.-Y. Chen, J. Yi, D. Su, Y. Gao, C.-J. Hsieh, and L. Daniel, "Evaluating the robustness of neural networks: an extreme value theory approach," in *International Conference on Learning Representations*. Vancouver, BC, Canada: OpenReview.net, 2018.

[25] S. Yoo and M. Harman, "Regression testing minimization, selection and prioritization: a survey," *Software Testing, Verification and Reliability*, vol. 22, no. 2, pp. 67–120, 2012.

[26] W. Afzal and R. Torkar, "Towards benchmarking feature subset selection methods for software fault prediction," in *Computational Intelligence and Quantitative Software Engineering*. Springer, 2016, pp. 33–58.

[27] T. Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener, "Defect prediction from static code features: current results, limitations, new approaches," *Automated Software Engineering*, vol. 17, no. 4, pp. 375–407, 2010.

[28] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 2–13, 2006.

[29] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, 2020.

[30] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[31] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.

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

[33] S. Ren, Y. Deng, K. He, and W. Che, "Generating natural language adversarial examples through probability weighted word saliency," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, july 2019, pp. 1085–1097.

[34] C. Fellbaum, "Wordnet," in *Theory and Applications of Ontology: Computer Applications*. Springer, 2010, pp. 231–243.

[35] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in *2018 IEEE Security and Privacy Workshops (SPW)*, May 2018, pp. 50–56.

[36] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in Neural Information Processing Systems*, vol. 28, pp. 1135–1143, 2015.

[37] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang, "Network trimming: a data-driven neuron pruning approach towards efficient deep architectures," *arXiv preprint arXiv:1607.03250*, 2016.

[38] L. Ma, F. Juefei-Xu, M. Xue, Q. Hu, S. Chen, B. Li, Y. Liu, J. Zhao, J. Yin, and S. See, "Secure deep learning engineering: A software quality assurance perspective," *arXiv preprint arXiv:1810.04538*, 2018.

[39] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278 – 2324, November 1998.

[40] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, Tech. Rep., 2009.

[41] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 142–150.

[42] D. Vitale, P. Ferragina, and U. Scaiella, "Classification of short texts by deploying topical annotations," in *European Conference on Information Retrieval*. Berlin, Heidelberg: Springer, 2012, pp. 376–387.

[43] L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman, "Knowledge sharing and yahoo answers: everyone knows something," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 665–674.

[44] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, vol. 20, no. 5, p. 14, 2015.

[45] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, San Diego, CA, USA, 2015.

[47] F. Chollet, "Keras code examples," 2020. [Online]. Available: https://keras.io/examples/nlp/bidirectional_lstm_imdb/

[48] "Active learning empirical study homepage," 2021. [Online]. Available: https://sites.google.com/view/alempirical

[49] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. USA: Association for Computational Linguistics, 2008, pp. 1070–1079.

[50] A. Siddhant and Z. C. Lipton, "Deep bayesian active learning for natural language processing: results of a large-scale empirical study," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, October 2018, pp. 2904–2909.

[51] A. Odena, C. Olsson, D. Andersen, and I. Goodfellow, "Tensorfuzz: debugging neural networks with coverage-guided fuzzing," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, June 2019, pp. 4901–4911.

[52] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[53] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[54] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, ser. OSDI'16. USA: USENIX Association, 2016, pp. 265–283.

[55] W. H. Beluch, T. Genewein, A. Nürnberger, and J. M. Köhler, "The power of ensembles for active learning in image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9368–9377.

[56] 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.

[57] M.-I. Nicolae, M. Sinn, M. N. Tran, B. Buesser, A. Rawat, M. Wistuba, V. Zantedeschi, N. Baracaldo, B. Chen, H. Ludwig, I. Molloy, and B. Edwards, "Art: adversarial robustness toolbox v1.2.0," *CoRR*, vol. 1807.01069, 2018. [Online]. Available: https://arxiv.org/pdf/1807.01069

[58] Y. Dong, P. Zhang, J. Wang, S. Liu, J. Sun, J. Hao, X. Wang, L. Wang, J. S. Dong, and D. Ting, "There is limited correlation between coverage and robustness for deep neural networks," *arXiv preprint arXiv:1911.05904*, 2019.

[59] M. Thakkar, *Beginning machine learning in ios: CoreML framework*, 1st ed. APress, 2019.

[60] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[61] P. Li, P. Zhong, K. Mao, D. Wang, X. Yang, Y. Liu, J.-x. Yin, and S. See, "Act: an attentive convolutional transformer for efficient text classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 15, 2021, pp. 13 261–13 269.

[62] Z. Yu, N. A. Kraft, and T. Menzies, "Finding better active learners for faster literature reviews," *Empirical Software Engineering*, vol. 23, no. 6, pp. 3161–3186, 2018.

[63] Z. Chen, T. Menzies, D. Port, and D. Boehm, "Finding the right data for software cost modeling," *IEEE Software*, vol. 22, no. 6, pp. 38–46, 2005.

[64] H. Lu and B. Cukic, "An adaptive approach with active learning in software fault prediction," in *Proceedings of the 8th International Conference on Predictive Models in Software Engineering*, ser. PROMISE '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 79–88.

[65] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, X. Chen, and X. Wang, "A survey of deep active learning," *arXiv preprint arXiv:2009.00236*, 2020.

[66] N. Asghar, P. Poupart, X. Jiang, and H. Li, "Deep active learning for dialogue generation," *arXiv preprint arXiv:1612.03929*, 2016.

[67] Y. Geifman and R. El-Yaniv, "Deep active learning over the long tail," *arXiv preprint arXiv:1711.00941*, 2017.

[68] M. Fang, Y. Li, and T. Cohn, "Learning how to active learn: A deep reinforcement learning approach," *arXiv preprint arXiv:1708.02383*, 2017.

[69] B. Yang, J.-T. Sun, T. Wang, and Z. Chen, "Effective multi-label active learning for text classification," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 917–926.

[70] Z. Wang, H. You, J. Chen, Y. Zhang, X. Dong, and W. Zhang, "Prioritizing test inputs for deep neural networks via mutation analysis," in *IEEE/ACM 43nd International Conference on Software Engineering (ICSE)*, 2021.

[71] J. Wang, J. Chen, Y. Sun, X. Ma, D. Wang, J. Sun, and P. Cheng, "Robot: robustness-oriented testing for deep learning systems," *arXiv preprint arXiv:2102.05913*, 2021.

[72] M. E. Ramirez-Loaiza, M. Sharma, G. Kumar, and M. Bilgic, "Active learning: an empirical study of common baselines," *Data Mining and knowledge Discovery*, vol. 31, no. 2, pp. 287–313, 2017.

[73] D. Pereira-Santos, R. B. C. Prudêncio, and A. C. de Carvalho, "Empirical investigation of active learning strategies," *Neurocomputing*, vol. 326, pp. 15–27, 2019.

[74] J. Chen, A. Schein, L. Ungar, and M. Palmer, "An empirical study of the behavior of active learning for word sense disambiguation," in *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, 2006, pp. 120–127.

[75] F. C. Heilbron, J.-Y. Lee, H. Jin, and B. Ghanem, "What do i annotate next? an empirical study of active learning for action localization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 199–216.

[76] M. Sassano, "An empirical study of active learning with support vector machines forjapanese word segmentation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 505–512.

[77] Q. Guo, S. Chen, X. Xie, L. Ma, Q. Hu, H. Liu, Y. Liu, J. Zhao, and X. Li, "An empirical study towards characterizing deep learning development and deployment across different frameworks and platforms," in *34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2019, pp. 810–822.

[78] T. Zhang, C. Gao, L. Ma, M. Lyu, and M. Kim, "An empirical study of common challenges in developing deep learning applications," in *International Symposium on Software Reliability Engineering (ISSRE)*. Berlin, Germany: IEEE, 2019, pp. 104–115.

[79] Z. Chen, Y. Cao, Y. Liu, H. Wang, T. Xie, and X. Liu, "A comprehensive study on challenges in deploying deep learning based software," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 750–762.

[80] X. Zhang, X. Xie, L. Ma, X. Du, Q. Hu, Y. Liu, J. Zhao, and M. Sun, "Towards characterizing adversarial defects of deep learning software from the lens of uncertainty," in *IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, IEEE. New York, NY, USA: Association for Computing Machinery, 2020, pp. 739–751.