DeepDTS: DIVERSE TEST SELECTION METHOD FOR DEEP NEURAL NETWORKS

by

Mert Dedeköy

B.S., Electrical and Electronics Engineering, Özyeğin University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2024

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Prof. Alper Şen, for guiding me with his experience and for his efforts. I also want to thank you to the jury members, Assist. Prof. Doğan Ulus and Assoc. Prof. Reyhan Aydoğan, for their valuable feedback.

I would especially like to thank my wife, Beste Dedeköy, for always supporting and motivating me until the very end. Without her, it would not have been possible for me to complete this thesis and graduate.

Additionally, I want to express my gratitude to all my supportive family members.

Finally, I would like to thank Prof. Fatih Uğurdağ, who contributed significantly to my academic career during my undergraduate studies.

ABSTRACT

DeepDTS: DIVERSE TEST SELECTION METHOD FOR DEEP NEURAL NETWORKS

Deep Neural Networks (DNNs) are extensively applied in data-driven modeling tasks. Testing these networks is critical due to their involvement in safety-critical applications. However, their black-box nature poses challenges in explaining their decisions and identifying incorrect behaviors. Detecting faults in a DNN requires selecting a diverse and error-revealing test set from a vast, unlabeled dataset. Diversity in the test set expands the coverage of the decision space of DNN, aiding in the identification of distinct faults. In this paper, we introduce Singular Entropy Density (SED) metric to evaluate the diversity of a test set on a DNN model. We demonstrate its effectiveness against other diversity measures. Additionally, we present DeepDTS, a diverse test selection method designed to identify misclassified inputs with higher diversity scores according to SED. To evaluate the effectiveness of DeepDTS relative to other coverage and uncertainty-based test selection methods, we conducted experiments with widely employed datasets and DNN models. Our results demonstrate that DeepDTS outperforms other selection methods in terms of selecting a diverse test set.

ÖZET

DeepDTS: DERİN SİNİR AĞLARI İÇİN ÇEŞİTLİ TEST SEÇİM YÖNTEMİ

Derin Sinir Ağları (DNN), veriye dayalı modelleme görevlerinde yaygın olarak kullanılmaktadır. Bu ağların test edilmesi, güvenlik kritik uygulamalarda yer almasından dolayı hayati öneme sahiptir. Ancak, onların siyah kutu doğası, kararlarını açıklamak ve yanlış davranışları tanımlamak konusunda zorluklar yaratır. Bir DNN'de hata tespiti, geniş, etiketlenmemiş bir veri kümesinden çeşitli ve hata ortaya çıkaran bir test setinin seçilmesini gerektirir. Test setindeki çeşitlilik, DNN'nin karar alanının kapsamını genişletir ve farklı hataların tespitine yardımcı olur. Bu makalede, bir DNN modelinde bir test setinin çeşitliliğini değerlendirmek için Tekil Entropi Yoğunluğu (SED) ölçümünü tanıtıyoruz. Diğer çeşitlilik ölçümlerine karşı etkinliğini gösteriyoruz. Ayrıca, SED'ye göre daha yüksek çeşitlilik puanlarına sahip yanlış sınıflandırılmış girişleri belirlemek için tasarlanmış çeşitli bir test seçim yöntemi olan DeepDTS'yi sunuyoruz. DeepDTS'nin diğer kapsama ve belirsizlik temelli test seçim yöntemlerine göre etkinliğini değerlendirmek için yaygın olarak kullanılan veri kümeleri ve DNN modelleri ile deneyler gerçekleştirdik. Sonuçlarımız, DeepDTS'nin çeşitli bir test seti seçme konusunda diğer seçim yöntemlerini geride bıraktığını göstermektedir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES viii
LIST OF TABLES ix
LIST OF SYMBOLS x
LIST OF ACRONYMS/ABBREVIATIONS xi
1. INTRODUCTION 1
2. BACKGROUND 4
2.1. Diversity Metrics 4
2.2. Preliminary Concepts 5
2.3. Test Selection Methods 6
3. METHODOLOGY 8
3.1. DNN Test Selection 8
3.2. Singular Entropy Diversity (SED) 8
3.3. DeepDTS 9
4. EVALUATION 12
4.1. Experimental Setup 12
4.1.1. Datasets and DNN models 12
4.1.2. Implementation 13
4.1.3. Research Questions 13
4.2. Experimental Results 14
4.2.1. RQ1: Can SED measure diversity in a given dataset 14
4.2.2. RQ2: How effective is SED for measuring dataset diversity from a given DNN model compared to other diversity metrics? 16
4.2.3. RQ3: Can DeepDTS select more diverse inputs than existing test selection methods? 18
4.2.4. RQ4: Can DeepDTS find more diverse faults? 20

4.3. Threats to Validity 21
5. RELATED WORK 23
5.1. DNN Test Selection 23
5.2. Diversity in DNN Testing 24
5.3. Coverage in DNN Testing 25
6. CONCLUSION 26
REFERENCES 28
APPENDIX A: KULLANILAN FİGÜRLER 33

LIST OF FIGURES

Figure 3.1. Overview of DNN Test Selection. 8

Figure 3.2. Overview of SED. 9

Figure 3.3. Overview of DeepDTS. 10

Figure 4.1. MNIST diversity scores measured from VGG-16 features. 16

Figure 4.2. CIFAR-10 diversity scores measured from VGG-16 features. 16

Figure 4.3. MNIST diversity scores measured from LeNet5 model. 17

Figure 4.4. CIFAR-10 diversity scores measured from Resnet20 model. 17

Figure 4.5. CIFAR-10 diversity scores measured from ConvNet model. 17

Figure 4.6. CIFAR-10 SED scores measured from Resnet20 model for different input sizes, from left to right 100, 500, and 1000. 18

LIST OF TABLES

Table 4.1. Datasets and models used in experiments. 12

Table 4.2. SED scores of selected subsets by each method. 19

Table 4.3. SED scores of misclassified inputs selected by each method. 21

LIST OF SYMBOLS

A	Matrix that applied SVD
c	Class of input
$\widetilde{c}$	Closest assigned cluster of input
D	DNN Dataset
$h_{test}^{(L-1)}$	Penultimate layer outputs of DNN test set
$\widetilde{h}_{train}^{(L-1)}$	Penultimate layer outputs of correctly predicted training set
$H(X)$	Entropy of input set X
M	DNN Model
n	Testing budget
$p(x_i)$	Probability of i-th input
p_{y_i}	Output probability of class i
U	Left singular vector
S	Singular values
V^T	Right singular vector
V_X	Feature matrix of input set X
V_X^T	Transpose of feature matrix of input set X
x_i	DNN model input
X	Input set
X_{test}	DNN test set
$\widetilde{X}_{train}$	Correctly predicted DNN training set
μ	Mean
σ	Standard Deviation

LIST OF ACRONYMS/ABBREVIATIONS

DeepDTS	Deep Neural Network Diverse Test Selection
DNN	Deep Neural Network
GD	Geometric Diversity
MAXP	Maxiumum Probability
NC	Neuron Coverage
NDS	Neuron Deflection Score
SD	Standard Deviation
SED	Singular Entropy Diversity
SVD	Singular Value Decomposition
VGG-16	Visual Geometry Group 16

1. INTRODUCTION

Deep Neural Networks (DNNs) are used in various domains, including safety-critical applications such as face recognition [1], autonomous driving [2], and medical diagnostics [3]. Their potential for erroneous behavior poses significant risks to human life and the environment. Consequently, rigorous testing is imperative prior to deployment to ensure that these models function as intended across diverse conditions. While traditional software testing often relies on code coverage techniques, the complex architecture of DNNs makes these methods inadequate.

The primary method for assessing DNNs involves evaluating prediction scores on test sets. Therefore, test selection methods play a critical role in DNN assessment by efficiently allocating labeling resources within predefined budgets. These methods optimize resource allocation by identifying potential misclassifications within unlabeled datasets, ensuring that labeling efforts are focused on the most informative data points. This strategic approach not only enhances the efficiency of the labeling process but also mitigates the financial burden associated with it.

In previous studies, both coverage [4, 5] and uncertainty-based methods [6–8] have been employed for test selection. While some of these methods exhibit high fault detection rates, they often overlook the diversity of misclassified inputs, focusing solely on their quantity. This narrow focus can be misleading, as it may result in high fault detection rates due to the same underlying fault being repeatedly detected [9]. A more critical consideration is the inclusion of distinct faults in selected tests, as research [9] has shown that this is closely linked to the diversity of test sets.

Our objective, therefore, is to identify a diverse set of misclassified inputs. Selecting subsets with greater diversity allows for the exploration of different behaviors of DNNs, thereby enhancing testing capabilities. Given the high-dimensional input space of DNNs, assessing the diversity of image inputs presents a significant challenge. To

address this challenge, [9] utilized features extracted from VGG-16 [10] model to measure the diversity of the input set. However, relying solely on features extracted from a separate model may not capture the true diversity of the input data presented to the tested DNN. It is important to note that diversity needs to be measured from DNN itself as we analyze the behavior of the inputs in the model. Motivated by this notion, we propose Singular Entropy Diversity (SED) as a white-box approach for assessing subset diversity directly from tested DNN itself.

In the subsequent phase, we introduce DeepDTS for diverse subset and fault selection. The concept behind DeepDTS is to identify outlier test inputs that deviate from the typical paths followed by trained images. Specifically, a test input is considered significant if it exhibits different neuron output values within the inner layers of a DNN compared to its respective predicted class. We focus particularly on the penultimate layer to extract behaviors containing more generalized information about the input. By computing the average behavior of each class from the training inputs, we then measure the distance of each test input to every class. Subsequently, we select inputs with the largest distances to their closest class. Larger distances indicate differing outputs within DNN, suggesting that the input is more likely to be misclassified and therefore should be subjected to additional examination.

The primary contributions of this thesis include:

- Introduction of a diversity metric SED, demonstrating its superior capability to measure diversity compared to the other two existing metrics.
- Proposal of DeepDTS, a test selection method capable of selecting a diverse subset of misclassified inputs.
- Conducting experiments across three datasets and five models, we demonstrate DeepDTS has superior performance compared to other test selection methods.

The remainder of the thesis is structured as follows. In Chapter 2, we provide background information on diversity metrics, preliminary concepts for our methods,

and other test selection methods that we compare. Chapter 3 presents our methods, SED and DeepDTS. In Chapter 4, we describe how we evaluate our methods and the experiment results. Chapter 5 reviews related works. Finally, in Chapter 6 we conclude the work.

2. BACKGROUND

In this chapter, we provide preliminary work related to our research, including diversity metrics, base concepts for our methods, and other test selection methods that we implemented for comparison.

2.1. Diversity Metrics

Diversity metrics measure how data is spread out and provide insights into the distribution and variability within a dataset. They offer information about the range and distribution of the data points, helping to assess the representativeness and coverage of the dataset. In this section, we define Geometric Diversity (GD) and Standard Deviation (SD), two other diversity metrics utilized in [9]. These metrics have been applied to input features extracted from VGG-16 to assess the diversity of the input set.

GD metric measures diversity in terms of the volume spanned by the input feature matrix. It is defined as

$$GD(X) = det(V_X * V_X^T), \tag{2.1}$$

where X is the input set and V_X denotes the feature matrix and V_X^T donates the transpose of the feature matrix. To calculate GD value, features are first normalized to fall between 0 and 1. Then, the scaled feature matrix is multiplied by its transpose, and the determinant of the resulting product is computed. A higher GD value indicates greater diversity among the inputs, suggesting that they are more spread out in the feature space.

SD is a measure used to express the dispersion of data around the mean. A high SD indicates diverse data. It is defined as

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}, \tag{2.2}$$

where x_i represents each input, μ is the mean of the data and N is the number of elements in the input set. To calculate SD of a dataset, its features are normalized as in GD calculation to fall between 0 and 1. Then, for each feature vector, Equation (2.2) is applied to calculate SD value of each feature. Finally, the norm of this SD vector is computed to obtain a single comparable diversity value.

2.2. Preliminary Concepts

In this section, we define Singular Value Decomposition (SVD) [11] and Entropy [12], concepts that form the basis of our diversity metric.

SVD is a fundamental matrix factorization technique in linear algebra widely employed in data analysis. It decomposes a matrix into three distinct components as follows

$$A = USV^T, \tag{2.3}$$

where A represents the matrix, U and V are orthogonal matrices representing the left and right singular vectors respectively, and S is a diagonal matrix containing singular values. These singular values are crucial as they indicate the variance captured by each principal component. SVD is commonly used in tasks such as dimensionality reduction [13], data compression [14], and noise reduction [15].

Entropy quantifies the degree of unpredictability or randomness within a system. It is calculated by

$$H(X) = -\sum_{i=1}^{N} p(x_i) \log p(x_i), \tag{2.4}$$

where $p(x_i)$ represents the probability of element i. To compute the entropy value, the probabilities of each input are multiplied by the logarithm of its probability. These values are then summed and negated. A higher entropy indicates a more even distribution of probabilities, implying greater uncertainty or randomness in the system. In situations where information is defined using probability distributions, higher uncertainty tends to yield more information.

2.3. Test Selection Methods

Now we describe the test selection methods employed in this paper. The listed methods involve prioritizing inputs based on specific criteria and selecting the top n elements where n represents the testing budget.

Neuron Coverage (NC) [4] is the first coverage metric for DNNs, inspired by line coverage in traditional software testing. It considers a DNN neuron as activated if its output exceeds a predefined threshold value, which we set to 0.75 in our experiments, following [6, 16]. NC calculates the ratio of activated neurons to all neurons in a DNN for each input passed through the network. Test selection using NC prioritizes inputs based on their coverage rates and picks top n. This method assumes that inputs with higher coverage better test DNN.

Maximum Probability (MaxP) [8] selects inputs based on the uncertainty of DNN output probabilities, prioritizing those with the smallest prediction probability. It operates under the assumption that when DNN is uncertain about its prediction, meaning the top probability is close to the second top probability, the input is more prone to be misclassified, thus making it a prioritized test input.

DeepGini [6] is another metric that prioritizes inputs based on output probabilities. It assesses the likelihood of inputs being misclassified by

$$DeepGini(x) = 1 - \sum_{i=1}^{N} p_{y_i}^2(x), \tag{2.5}$$

where p_{y_i} represents DNN output probability of class i and N is the number of classes. A test input is likely to be misclassified if DNN output probabilities are similar for each class, which yields a higher DeepGini value. Consequently, test inputs are prioritized based on descending values of DeepGini metric.

Entropy is also utilized as a test selection metric [17], sorting inputs based on their DNN output probability entropy. Entropy values are calculated from output

probabilities as described in Equation (2.4). Higher entropy indicates a more even distribution of output probabilities, suggesting that inputs with higher DNN output entropy are more likely to be misclassified. Therefore, these inputs are selected first during test selection.

3. METHODOLOGY

In this chapter, we introduce the methodologies proposed in this thesis. We begin by outlining the general structure of DNN test selection strategy. Next, we introduce SED, a white-box diversity metric for DNNs. Finally, we present DeepDTS, a diverse test selection methodology.

3.1. DNN Test Selection

DNN test selection aims to identify the most effective subset from an unlabeled test set pool, considering the costly nature of labeling data. As displayed in Figure 3.1, the test selection process starts with an unlabeled test set pool. DeepDTS and other test selection methods are employed to select subsets from this pool. The selected test sets are then evaluated using SED diversity metric introduced in this thesis. The test set with the highest SED score is chosen, indicating its potential to select distinct images in the model. This selected set is labeled for evaluating the model.

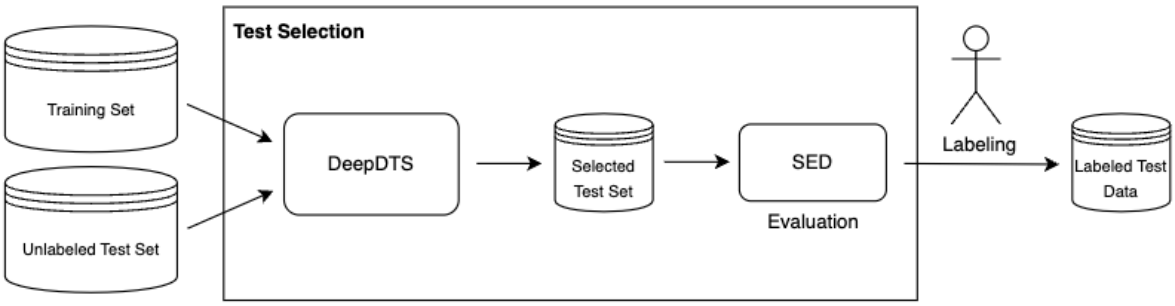

Figure 3.1. Overview of DNN Test Selection.

3.2. Singular Entropy Diversity (SED)

SED is designed to quantify the diversity of any input set exercised by a DNN. As mentioned in the introduction, the main goal is to analyze the diverse behaviors of the inputs in DNN model. Figure 3.2 illustrates an overview of SED.

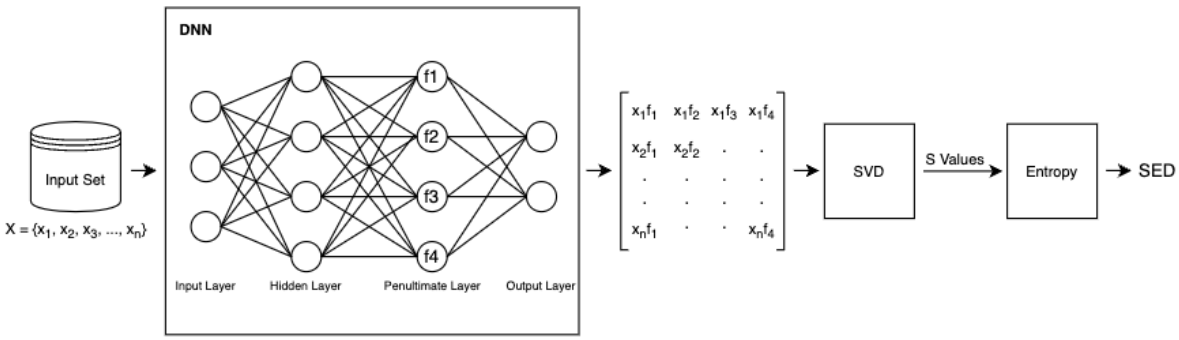

Figure 3.2. Overview of SED.

We first extract features from the input set, derived from the penultimate layer of DNN model. These outputs are termed as features because they encapsulate semantic information about the inputs. Subsequently, we construct a feature matrix where each row represents an input and columns correspond to features. Then, we decompose the feature matrix using SVD, as described in Equation (2.3), to obtain singular values. These singular values serve as further compressed versions of the feature values. Finally, we compute the entropy of the singular values to assess uncertainty and randomness, resulting in a diversity measure. This final entropy value is denoted as SED.

The detailed implementation is as follows: given a dataset D and a trained model M, the matrix $h_D^{(L-1)}$ represents the penultimate layer outputs of dataset D on model M, where L denotes the total number of layers. Subsequently, $h_D^{(L-1)}$ undergoes SVD, resulting in a matrix S of singular values. To highlight dominant features, we square S values to increase the sensitivity of SED to higher variances. Finally, we compute SED value by normalizing S and calculating its entropy by using Equation (2.4).

3.3. DeepDTS

The goal of DeepDTS is to construct a test set that includes misclassified inputs with a high diversity score on a trained DNN model. Figure 3.3 provides an overview of DeepDTS.

Initially, we utilize correctly predicted training data to collect statistics from the model, ensuring that only accurately classified instances are selected to maintain the integrity of class statistics. We operate under the assumption that inputs belonging to the same class should follow a similar trajectory within the model, and their intermediate layer outputs, especially the penultimate layer, should demonstrate similarities as they represent input features.

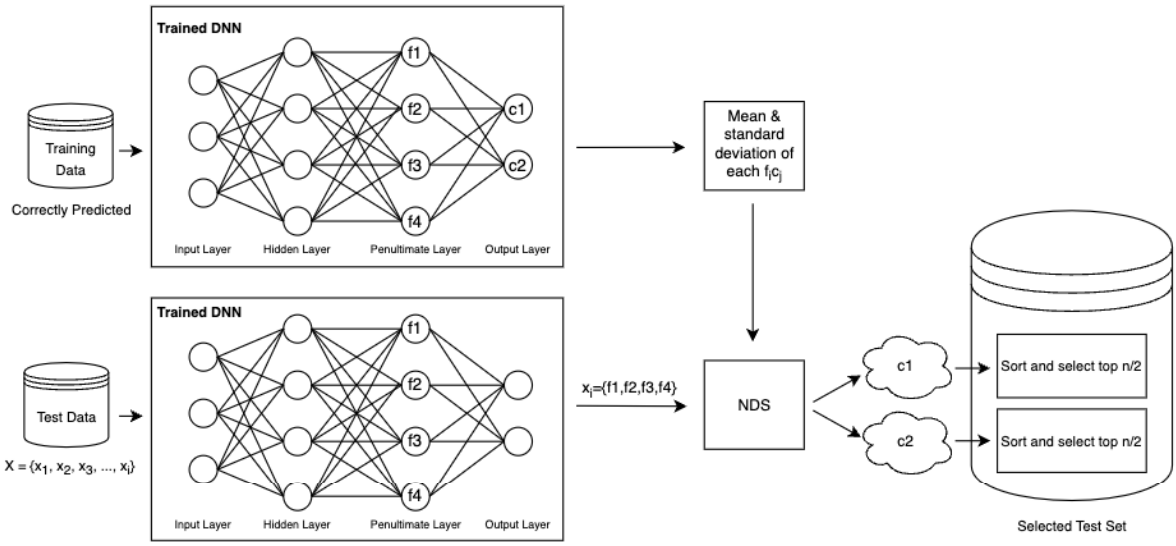

Figure 3.3. Overview of DeepDTS.

For each class, we compute the mean and standard deviation of each feature corresponding to inputs belonging to that class. Subsequently, we extract the features of each test input. Using the test features, we calculate the distance of each feature of a test input to the mean features of every class, resulting in a distance metric that we proposed Neuron Deflection Score (NDS). Subsequently, we assign each input to the class with the smallest NDS, indicating the closest match between the input and the respective class mean features.

Finally, we sort the inputs within each assigned class based on the descending NDS. This sorting prioritizes inputs that exhibit the greatest deviation from their respective class means. This prioritization ensures the selection of inputs that are most likely to be misclassified due to their significant deviations from their respective class means. Subsequently, we choose an equal number of such inputs from each assigned

class until the testing budget is met.

The implementation details are as follows: let $\widetilde{X}_{train}$ represent the set of indices of correctly predicted training samples, denoted as $(x_1, x_2, ..., x_k)$, obtained from a DNN model M. The matrix $\widetilde{h}_{train}^{(L-1)}$ contains the penultimate layer outputs (features) from M, where L is the total number of layers. Using $\widetilde{h}_{train}^{(L-1)}$, we calculate the mean μ_{jc} and standard deviation σ_{jc} for each feature j and class c.

Given the test set $X_{test} = (x_1, x_2, ..., x_i)$, we define NDS as

$$NDS(x_i, c) = \sum_{j=1}^{N} \frac{|\mu_{jc} - h_{x_{ij}}^{(L-1)}|}{\sigma_{jc}}, \tag{3.1}$$

where $h_{x_{ij}}^{(L-1)}$ represents the output of the j-th feature neuron of input x_i. The distance between neuron outputs and class means is scaled by the standard deviation to account for distribution spread. A low standard deviation σ implies that training outputs from the selected class cluster closely around the mean, thus penalizing the distance more. Conversely, a high standard deviation indicates that training outputs are more widely distributed around the mean, resulting in a lesser penalty for the distance. By summing the deflection of each feature to the given class, we obtain NDS. A lower NDS value indicates a closeness to the class, allowing us to pick the closest cluster for each input.

Next, each input x_i is assigned to the closest class cluster $\widetilde{c}$ with the minimum NDS, expressed as

$$\widetilde{c} = \arg\min_{c} NDS(x_i, c). \tag{3.2}$$

Then, within each class cluster $\widetilde{c}$, inputs are sorted in descending order of their NDS to prioritize them according to their distances from the class mean. Finally, an equal number of elements are selected from each cluster until the testing budget is reached.

4. EVALUATION

In this chapter, we begin by outlining our experimental setup, which includes information about the datasets and DNN models employed in our experiments, as well as implementation details and the research questions guiding our investigation. We then proceed to present the results of our experiments, followed by a discussion on potential threats to the validity of our findings.

4.1. Experimental Setup

4.1.1. Datasets and DNN models

We present the datasets and DNN models utilized in our experiments in Table 4.1. Additionally, we include the sizes of their penultimate layers as they are used in SED and DeepDTS to extract the features of inputs. We evaluate the proposed metrics on three well-known image recognition datasets: MNIST [18], CIFAR-10 [19], and Fashion-MNIST [20]. These datasets were chosen due to their popularity in evaluating test selection methods.

Table 4.1. Datasets and models used in experiments.

Dataset	**DNN Model**	**Penultimate Layer Size**
MNIST	LeNet-1	588
	LeNet-5	120
CIFAR-10	ResNet-20	64
	12-layer ConvNet	512
Fashion	LeNet-4	84

MNIST is a handwritten digit dataset that contains 60000 training and 10000 test examples. CIFAR-10 dataset contains colored images categorized into 10 different classes. It consists of 50000 training images and 10000 test images. Fashion-MNIST

comprises 60000 training images and 10000 test images of fashion and clothing items, with 10 distinct classes.

We employ five state-of-the-art DNN models in our experiments: LeNet-1, LeNet-5, ResNet-20, a 12-layer ConvNet, and LeNet-4, all of which are pre-trained models obtained from [9]. The choice of models is tailored to the characteristics of the datasets used in our experiments.

- LeNet-1 and LeNet-5: Originally designed for handwritten digit recognition tasks, these models are particularly suitable for datasets with grayscale images. Hence, they are employed with the MNIST dataset.
- ResNet-20 and ConvNet: These models are selected for their ability to handle complex image features such as CIFAR-10.
- LeNet-4: This model is a more complex variant of LeNet-1 and is well-suited for tasks involving grayscale images. Therefore, it is paired with the Fashion-MNIST dataset.

4.1.2. Implementation

We implemented SED and DeepDTS using the open-source machine learning frameworks Tensorflow [21] and Keras [22]. The experiments were conducted on a MacBook Pro with an M2 chip and 16GB of memory.

4.1.3. Research Questions

- RQ1: Can SED measure diversity in a given dataset?
- RQ2: How effective is SED for measuring dataset diversity from a given DNN model compared to other diversity metrics?
- RQ3: Can DeepDTS select more diverse inputs than existing test selection methods?
- RQ4: Can DeepDTS find more diverse faults?

RQ1 assesses whether SED can effectively serve as a diversity measurement for a dataset. Prior studies have shown that diversity scores like GD and SD increase as dataset diversity based on VGG-16 features increases. Our objective is to demonstrate that SED can similarly measure diversity from VGG-16 features before conducting further evaluations of SED.

RQ2 evaluates diversity metrics in terms of their effectiveness in measuring dataset diversity based on the penultimate layer neuron outputs of a given DNN model. We argue that for a dataset diversity measure to be meaningful, it should be closely tied to the characteristics of the specific DNN model being used. Therefore, diversity measurement should be conducted directly from DNN model itself. Our goal is to establish whether SED metric outperforms GD and SD in accurately measuring dataset diversity from the penultimate layer.

RQ3 assesses the effectiveness of DeepDTS and other test selection methods in selecting a diverse dataset. Our objective is to demonstrate that the test set selected by DeepDTS has a higher diversity score compared to other methods.

RQ4 evaluates the effectiveness of test selection methods in finding a subset that includes misclassified images with higher diversity scores. This higher diversity score indicates the presence of more distinct faults in the selected set.

4.2. Experimental Results

4.2.1. RQ1: Can SED measure diversity in a given dataset

To validate the effectiveness of SED as a diversity metric, we conducted an experiment following the methodology outlined in [9]. Previous research demonstrated that both GD and SD effectively measured diversity within a dataset using VGG-16 features of inputs. Additionally, GD score exhibited a notable correlation with the number of distinct faults in a DNN. Therefore, in this research question, we employed

our proposed metric SED to accurately measure data diversity in the black-box context, similar to GD and SD.

We adapted SED to a black-box setting by applying it to VGG-16 features of MNIST and CIFAR-10 datasets instead of the penultimate layer outputs. In this experiment, we gradually increased dataset diversity by adding inputs from different classes while maintaining a constant input size. Initially, 500 inputs were randomly selected from a single class. Subsequently, in each step, half of the inputs were replaced with inputs from another class. This process continued until all classes were represented with an equal number of inputs.

It is worth noting that in the original paper, the input size was set to 100 instead of 500. However, due to the randomness effect, we did not observe a clear increase in diversity scores for any of the metrics. Therefore, to mitigate the impact of randomness, we repeated the experiment 20 times, and the input size was chosen as 500.

As we modified the inputs and increased the number of included classes, we measured the values of SED, GD, and SD. We expected an increase in diversity scores with each additional class, reflecting the augmentation of dataset diversity by incorporating samples from various classes. The diversity scores, illustrated in Figure 4.1 for MNIST dataset and Figure 4.2 for CIFAR-10 dataset, exhibited a similar increasing trend for GD and SD as reported in [9]. Notably, SED scores also showed a positive correlation with the number of included classes from 1 to 10.

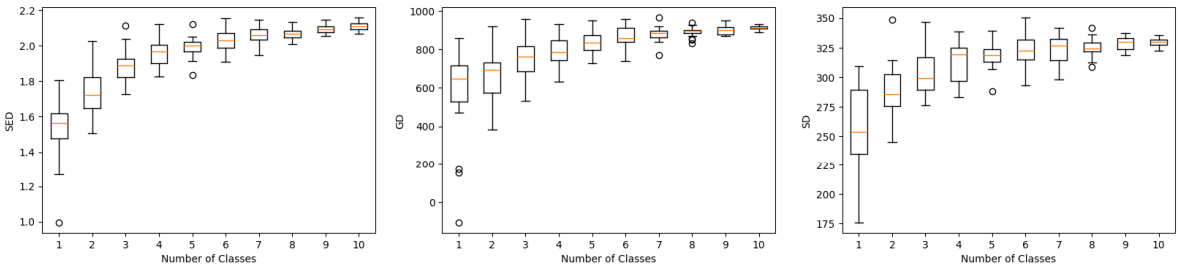

Figure 4.1. MNIST diversity scores measured from VGG-16 features.

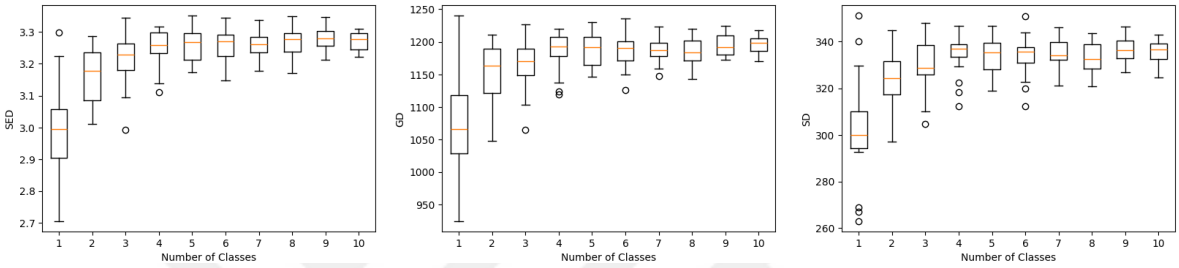

Figure 4.2. CIFAR-10 diversity scores measured from VGG-16 features.

Our analysis of the graphs indicates that SED performs comparably to GD and SD for both the MNIST and CIFAR-10 datasets. Additionally, we observed that the diversity scores (SED and GD) measured from CIFAR-10 are higher than those from MNIST. We interpret this as an indication that CIFAR-10 is a more complex dataset than MNIST, leading to higher diversity scores.

We conclude that SED effectively measures dataset diversity based on VGG-16 features and can be used for diversity assessment.

4.2.2. RQ2: How effective is SED for measuring dataset diversity from a given DNN model compared to other diversity metrics?

In this research question, our objective was to evaluate diversity metrics within a white-box context. We conducted an experiment similar to RQ1, where we randomly selected 100 inputs from a single class and progressively increased the included classes in each step. However, instead of computing diversity scores from VGG-16 features at each step, we utilized the penultimate layer outputs of selected models. To mitigate the effect of randomness, we repeated the experiment 20 times.

We select three different dataset-model combinations to evaluate the diversity metrics: MNIST with LeNet-5, CIFAR-10 with ResNet-20, and CIFAR-10 with ConvNet. The results are presented in Figures 4.3, 4.4, and 4.5, respectively.

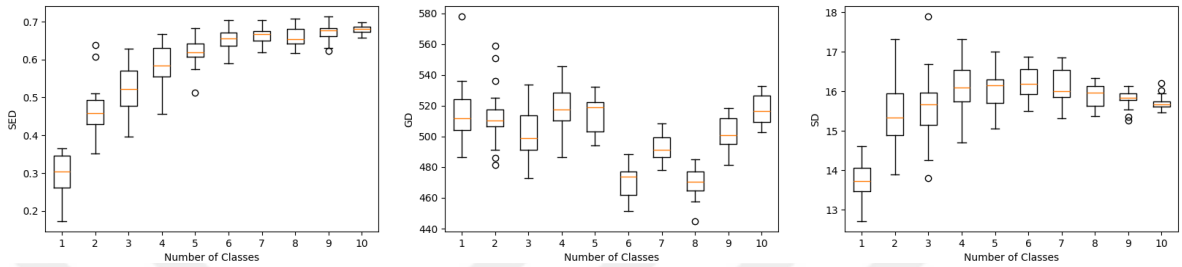

Figure 4.3. MNIST diversity scores measured from LeNet5 model.

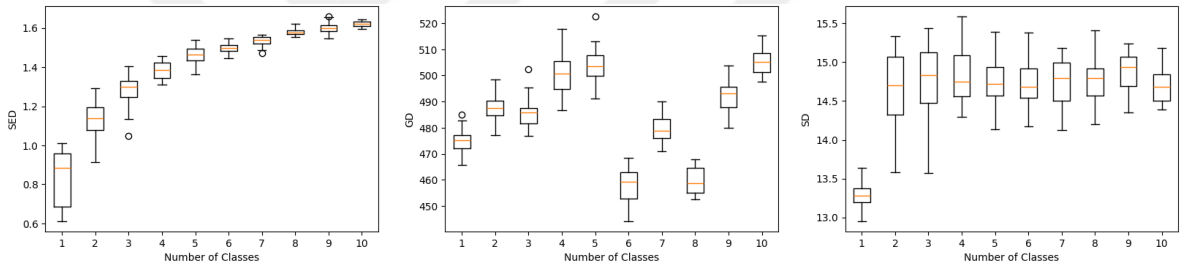

Figure 4.4. CIFAR-10 diversity scores measured from Resnet20 model.

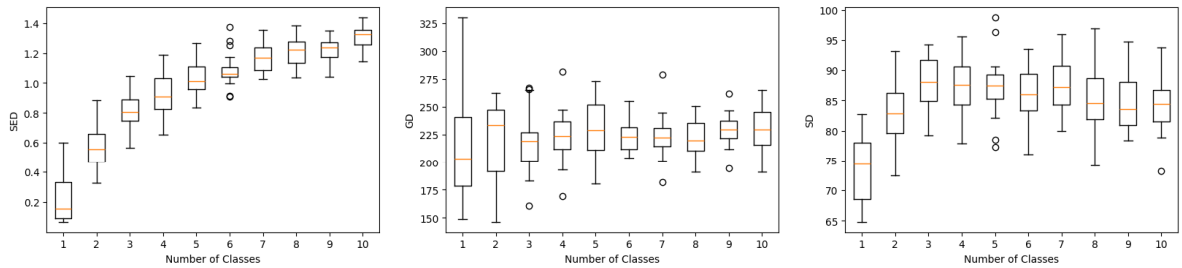

Figure 4.5. CIFAR-10 diversity scores measured from ConvNet model.

While the expected behavior is for diversity scores to increase as we include more classes, the figures reveal that GD and SD scores fail to exhibit such an increase at each step for each of the three model dataset configurations.

For SD scores, subsets show an increase of up to 3 class inclusions, but then the values seem stable after including 3 classes in each of the figures. GD scores appear to be unrelated to the number of included classes in Figures 4.3 and 4.4, and

they remain unchanged in Figure 4.5. On the other hand, SED scores demonstrate a gradual increase and show a strong correlation with data diversity measured from the given models in each of the figures.

Additionally, we investigate whether the number of inputs affects SED scores, as adding similar inputs should not increase diversity score. For CIFAR-10 with ResNet-20 setup, we conducted the same experiment with input sizes of 100, 500, and 1000. Figure 4.6 demonstrates that SED values do not depend on the number of inputs in the dataset. Each SED score in Figure 4.6 varies between 0.6 and 1.6, enabling a fair diversity comparison even when input sets have different sizes. This indicates a focus on real diversity rather than an increased score due to population size.

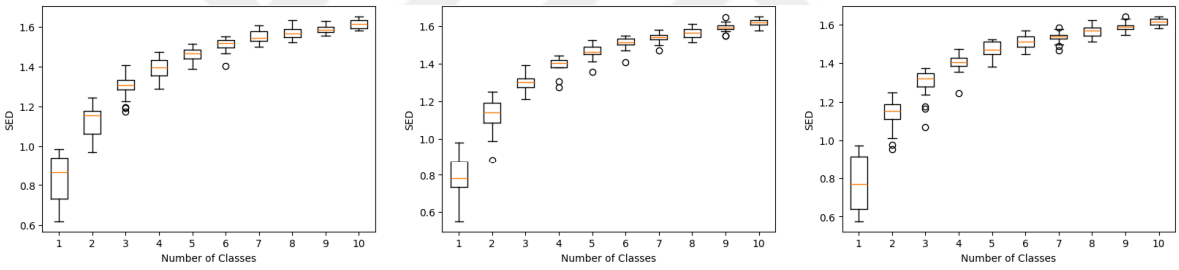

Figure 4.6. CIFAR-10 SED scores measured from Resnet20 model for different input sizes, from left to right 100, 500, and 1000.

After evaluating diversity metrics for measuring dataset diversity from given models, we find that SED scores demonstrate a more robust positive correlation with the increasing diversity of the selected subset. Notably, we observe that SED scores remain consistent irrespective of the number of inputs, indicating real diversity. Therefore, we select SED as the diversity metric to use in subsequent research questions.

4.2.3. RQ3: Can DeepDTS select more diverse inputs than existing test selection methods?

We compare the test selection methods based on the diversity of the selected subsets, assuming that a higher SED score indicates greater diversity. We hypothesize that subsets with higher SED scores contain a more diverse range of error-revealing inputs.

We selected subsets using DeepDTS, NC, MaxP, DeepGini, and Entropy methods. For each dataset, model, and method, we selected subsets with sizes of 100, 500, and 1000. Table 4.2 presents SED scores of each selected subset, with the highest SED score for each row highlighted in bold text.

Table 4.2. SED scores of selected subsets by each method.

Dataset	**Model**	**Size**	**DeepDTS**	**NC**	**MaxP**	**DeepGini**	**Entropy**
MNIST	LeNet1	100	**0.89**	0.70	0.70	0.62	0.58
		500	**0.83**	0.82	0.75	0.73	0.71
		1000	**0.94**	0.85	0.77	0.76	0.74
	LeNet5	100	**0.63**	0.21	0.43	0.36	0.34
		500	**0.69**	0.38	0.49	0.46	0.43
		1000	**0.68**	0.48	0.52	0.49	0.48
CIFAR10	ResNet20	100	**1.62**	1.41	1.21	1.19	1.16
		500	**1.71**	1.46	1.36	1.28	1.27
		1000	**1.71**	1.49	1.40	1.39	1.34
	ConvNet	100	**1.29**	0.72	1.17	1.16	1.15
		500	1.37	0.75	**1.39**	1.33	1.30
		1000	1.34	0.86	**1.44**	**1.44**	1.41
Fashion	LeNet4	100	**1.96**	0.45	1.45	1.42	1.35
		500	**2.06**	0.84	1.55	1.49	1.51
		1000	**2.06**	1.31	1.62	1.56	1.51

From the results, we observe that subsets selected by DeepDTS have significantly higher SED values than those selected by other methods, particularly noticeable in subset size 100 for each configuration. As the subset sizes increase, the difference between DeepDTS and other methods decreases, indicating that when there is a limited budget for labeling test inputs, DeepDTS selects more diverse data than other selection methods.

Additionally, we notice that uncertainty metrics MaxP, DeepGini, and Entropy yield similar SED values. Although for CIFAR-10 with ConvNet configuration, MaxP has the highest score for subset size 500, and for size 1000, MaxP and DeepGini have the highest scores, SED values obtained by DeepDTS are very close to theirs. A possible reason why DeepDTS is not the best method for the ConvNet model could be insufficient training of the model. When DNN models are well-trained, the feature values of the data become more representative, leading to a selection that aligns better with the aims of our test selection method, DeepDTS.

We conclude that DeepDTS is the best metric among the compared methods for selecting diverse subsets.

4.2.4. RQ4: Can DeepDTS find more diverse faults?

In existing literature, comparisons of test selection methods have primarily focused on identifying the highest number of misclassified inputs within a fixed test budget. However, as mentioned in this paper, some misclassifications may originate from redundant faults within the model, emphasizing the need to enhance the detection of distinct faults to improve overall model robustness. This, highlights the importance of diversity, as it implies a broader spectrum of distinct faults.

To address this, we investigate SED scores of misclassified inputs identified by various test selection methods across different dataset-model configurations and subset sizes (100, 500, and 1000). The resulting SED scores, outlined in Table 4.3, highlight the maximum score achieved in each row for each subset size.

Our analysis indicates that DeepDTS consistently achieves notably high SED scores, indicating its effectiveness in identifying a diverse range of misclassified inputs across MNIST, CIFAR-10, and Fashion datasets. While MaxP shows competitive performance for CIFAR-10 with ConvNet at subset sizes of 500 and 1000, its scores closely align with those of DeepDTS.

We conclude that DeepDTS stands out among the compared methods in uncovering diverse faults across various dataset-model configurations and subset sizes. Its consistently high SED scores demonstrate its ability to detect a wide spectrum of misclassified inputs. This supports the hypothesis that DeepDTS is adept at uncovering a broader range of faults compared to alternative methods.

Table 4.3. SED scores of misclassified inputs selected by each method.

Dataset	**Model**	**Size**	**DeepDTS**	**NC**	**MaxP**	**DeepGini**	**Entropy**
MNIST	LeNet1	100	**0.74**	0.47	0.70	0.62	0.58
		500	**0.87**	0.76	0.76	0.73	0.72
		1000	**0.90**	0.81	0.78	0.77	0.75
	LeNet5	100	**0.63**	0.25	0.43	0.36	0.34
		500	**0.63**	0.40	0.49	0.47	0.44
		1000	**0.63**	0.45	0.53	0.50	0.49
CIFAR10	ResNet20	100	**1.47**	1.00	1.20	1.19	1.18
		500	**1.56**	1.32	1.36	1.29	1.27
		1000	**1.54**	1.40	1.39	1.39	1.34
	ConvNet	100	**1.31**	0.49	1.16	1.13	1.11
		500	1.33	0.85	**1.37**	1.32	1.29
		1000	1.37	1.00	**1.43**	1.41	1.39
Fashion	LeNet4	100	**1.62**	0.00	1.43	1.40	1.31
		500	**1.92**	1.34	1.53	1.46	1.49
		1000	**1.93**	1.56	1.60	1.56	1.50

4.3. Threats to Validity

We ensure internal validity by conducting rigorous testing and validation of SVD and Entropy calculations employed in the implementation of SED diversity metric. To mitigate the risk of obtaining random results, we ran experiments independently 20 times and compared the consistency of the outcomes. Additionally, by testing

SED across two distinct datasets, three different models, and varying input sizes, we aim to minimize internal validity threats arising from potential biases or errors in the experimental setup.

A potential internal validity concern arises from our reliance on penultimate layer outputs when measuring the diversity of test data. While penultimate layers are commonly used for feature extraction in DNNs, their representation of inputs may not be comprehensive enough. Therefore, further investigation is necessary to ascertain the effectiveness of this approach in capturing the true diversity of inputs.

Another limitation may arise from using unbalanced datasets with DeepDTS. Since we select an equal number of inputs from each cluster that represents different input classes, having an unbalanced dataset may negatively impact the ability of DeepDTS to effectively select diverse and misclassified inputs.

The external validity of our study may be limited by the reliance on DNN models trained by [9]. While these datasets and models are widely used in the literature, the generalizability of our findings to other datasets and DNN architectures may be constrained. To address this limitation, future work should involve experiments with additional datasets and DNN models to validate the effectiveness of SED and DeepDTS across diverse contexts. By expanding the scope of our analysis, we can enhance the external validity of our results and ensure that they are applicable beyond the specific conditions of our study.

Another limitation arises from the fact that we only tested our SED metric on image inputs. Future research should also involve testing these methods with regression tasks and different contextual tasks, such as text classification, natural language processing, and speech recognition. This broader exploration will provide a more comprehensive understanding of the applicability and effectiveness of our proposed diversity metric across various domains and tasks.

5. RELATED WORK

In this section, we provide various DNN testing methods that have been employed in the literature. We analyze them in three categories DNN test selection, diversity in DNN testing, and coverage in DNN testing.

5.1. DNN Test Selection

Extensive research has been conducted on DNN test selection, aiming to address labeling budget constraints and determine the optimal subset for testing purposes.

Earlier research [23] proposes likelihood-based surprise adequacy and distance-based surprise adequacy, which extract features from both training and test data in DNNs to select test inputs with more surprise features compared to training features. These two methods also introduce surprise coverage metrics. An adaptive test selection method [7] uses DNN output vectors to select test inputs based on the fault pattern coverage score of the currently selected test set. TestRank [24] introduces a test prioritization technique leveraging intrinsic and contextual attributes of inputs, with contextual attributes extracted from graph neural networks and intrinsic features obtained from DNN models. Additionally, in the study by [25], three sentiment measures confidence, uncertainty, and surprise are examined for their effectiveness in revealing faults in DNNs. Multiple-boundary Clustering and Prioritization [26] detect boundary areas between classes based on the first and second highest class output probabilities, and then select test inputs equally from each boundary area.

DeepGini [6] prioritizes test inputs based on output uncertainty, demonstrating superior performance in identifying the highest number of misclassified inputs within a given testing budget compared to coverage-based test selection. MaxP [8] and Entropy [17] are other metrics that utilize output probabilities to rank test inputs. These metrics exploit the principle that higher uncertainty is associated with a higher probability of

misclassification.

While the test selection methods reported promising results in terms of finding misclassified inputs, [27] shows that existing test selection metrics fail to detect faults when there are correctly classified but uncertain data and misclassified but confident data.

5.2. Diversity in DNN Testing

The studies presented here focus on incorporating test diversity into their methodologies to comprehensively test DNNs and uncover various faults in models.

Aghababaeyan et al. (2023) [9] utilize diversity metrics on input features extracted from VGG-16 model. They examine the correlation between diversity scores and the count of distinct faults, finding that their chosen black-box diversity metric outperforms existing coverage criteria in identifying distinct faults. In a separate work [16], they proposed a black-box test selection approach leveraging genetic algorithm and geometric diversity to search for uncertain and diverse test inputs. Zhao et al. (2022) [28] conduct experiments with five state-of-the-art test input selection methods and show that test diversity is not satisfactory for these methods. Yuan et al. (2023) [29] propose a new coverage criterion called Neural Coverage. They demonstrate its correlation with the diversity of a test set and highlight the promising results of test input mutation guided by their criterion in identifying erroneous behaviors. Chen et al. (2020) [30] introduce Practical Accuracy Estimation to select a small set of test inputs that represent all test inputs. Their approach involves clustering inputs with different testing capabilities and subsequently selecting inputs from data that do not belong to any cluster to enhance diversity. The work by [7] defines fault diversity for misclassified inputs as the number of mappings from the ground-truth label of the input to the predicted output.

5.3. Coverage in DNN Testing

Some of the previous studies have explored the application of test coverage strategies from traditional software testing to testing of DNNs. These studies propose coverage metrics to guide selecting and generating test inputs that maximize coverage of different aspects of DNN behavior.

DeepXplore [4] introduces neuron coverage, which quantifies the number of activated neurons surpassing a predetermined threshold in a DNN model. Test inputs are generated to increase neuron coverage on DNN model. DeepGauge [5] presents a suite of coverage metrics that consider neuron output ranges. It shows that inputs that are created by adversarial techniques increase the coverage of different sections of neuron output ranges. DeepCT [31] adapts the combinatorial testing approach to deep learning systems and proposes coverage-guided test selection criteria. Another approach [32] applies Modified Condition/Decision Coverage criterion to structural features of DNNs and proposes four coverage metrics. Neuron Path Coverage [33] offers two path coverage methods based on control and data flow in DNNs. Importance-driven Coverage [34] identifies important neurons that influence the decision of DNN models using layer-wise relevance propagation [35] and aims to increase cluster coverage of these neurons to enhance test quality.

Several coverage-guided test generation frameworks have been proposed in the literature. DeepHunter [36] introduces a mutation-based test generation approach guided by coverage metrics. ADAPT [37] is a white-box testing methodology that presents a neuron selection strategy and learns the required parameters online, leading to increased coverage and the discovery of adversarial inputs. Additionally, Zhang et al. (2021) [38] propose a Coverage-guided Adversarial Generative Fuzzing testing method. This method incorporates an adversarial example generator and evaluates the generated inputs based on the similarity of features to the original inputs, resulting in enhanced neuron coverage.

6. CONCLUSION

DNN test selection is crucial for reducing labeling efforts and identifying the most informative test inputs. However, previous methods have often focused solely on maximizing fault detection rates without considering the diversity of misclassified inputs. This narrow approach may result in incomplete assessments of model behavior since including a maximum number of misclassified inputs in the test set may not be sufficient for uncovering distinct faults.

To address this limitation, we introduced the Singular Entropy Diversity (SED) metric. SED directly measures subset diversity from the penultimate layer of tested DNNs, providing a more comprehensive understanding of model behavior. Our experiments showed that SED outperforms existing diversity metrics in accurately capturing the diversity of input sets from DNN models.

Furthermore, we presented DeepDTS, a novel test selection method tailored to identifying diverse subsets of misclassified inputs. DeepDTS utilizes information from accurately classified training inputs to understand the characteristics of the dataset. It identifies test inputs with a higher potential for misclassification by measuring the distance of their features to clusters within the dataset. By selecting inputs from different clusters, DeepDTS enhances diversity in the test subset. Extensive experiments across multiple datasets and models provided empirical evidence of the superior performance of DeepDTS compared to other test selection methods. Specifically, DeepDTS demonstrated its effectiveness in uncovering a broader range of diverse faults, as measured by SED metric.

In our future work, we plan to conduct more experiments with different datasets and models to enhance the generalizability of our methods. Additionally, we aim to introduce a new test generation method based on SED, focusing on creating a more diverse set of test cases for evaluating DNNs. Furthermore, we intend to improve

DNN performance and robustness by retraining it with selected test inputs identified by DeepDTS. Through iterative refinement of the model with these inputs, we aim to enhance its ability to generalize to new data and improve overall performance.

REFERENCES

1. Minaee, S., A. Abdolrashidi, H. Su, M. Bennamoun and D. Zhang, "Biometrics Recognition Using Deep Learning: A Survey", *Artificial Intelligence Review*, Vol. 56, No. 8, pp. 8647–8695, 2023.

2. Teng, S., X. Hu, P. Deng, B. Li, Y. Li, Y. Ai, D. Yang, L. Li, Z. Xuanyuan, F. Zhu and L. Chen, "Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives", *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 6, pp. 3692–3711, 2023.

3. Mall, P. K., P. K. Singh, S. Srivastav, V. Narayan, M. Paprzycki, T. Jaworska and M. Ganzha, "A Comprehensive Review of Deep Neural Networks for Medical Image Processing: Recent Developments and Future Opportunities", *Healthcare Analytics*, Vol. 4, p. 100216, 2023.

4. Pei, K., Y. Cao, J. Yang and S. Jana, "DeepXplore: Automated Whitebox Testing of Deep Learning Systems", *Proceedings of the 26th Symposium on Operating Systems Principles*, p. 1–18, Shanghai, China, 2017.

5. Ma, L., 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", *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, p. 120–131, Montpellier, France, 2018.

6. Feng, Y., Q. Shi, X. Gao, J. Wan, C. Fang and Z. Chen, "DeepGini: Prioritizing Massive Tests to Enhance the Robustness of Deep Neural Networks", *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, p. 177–188, Virtual Event, USA, 2020.

7. Gao, X., Y. Feng, Y. Yin, Z. Liu, Z. Chen and B. Xu, "Adaptive Test Selection for Deep Neural Networks", *Proceedings of the 44th International Conference on Software Engineering*, p. 73–85, Pittsburgh, Pennsylvania, USA, 2022.

8. Ma, W., M. Papadakis, A. Tsakmalis, M. Cordy and Y. L. Traon, "Test Selection for Deep Learning Systems", *ACM Transactions on Software Engineering and Methodology*, Vol. 30, No. 2, pp. 1–22, 2021.

9. Aghababaeyan, Z., M. Abdellatif, L. Briand, R. S and M. Bagherzadeh, "Black-Box Testing of Deep Neural Networks through Test Case Diversity", *IEEE Transactions on Software Engineering*, Vol. 49, No. 5, pp. 3182–3204, 2023.

10. Simonyan, K. and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *3rd International Conference on Learning Representations*, pp. 1–14, San Diego, CA, USA, 2015.

11. Berrar, D. P., W. Dubitzky and M. Granzow, *A Practical Approach to Microarray Data Analysis*, Springer, New York, 2003.

12. Masisi, L., V. Nelwamondo and T. Marwala, "The Use of Entropy to Measure Structural Diversity", *IEEE International Conference on Computational Cybernetics*, pp. 41–45, Stara Lesna, Slovakia, 2008.

13. Salem, N. and S. Hussein, "Data Dimensional Reduction and Principal Components Analysis", *Procedia Computer Science*, Vol. 163, pp. 292–299, 2019.

14. de Souza, J. C. S., T. M. L. Assis and B. C. Pal, "Data Compression in Smart Distribution Systems via Singular Value Decomposition", *IEEE Transactions on Smart Grid*, Vol. 8, No. 1, pp. 275–284, 2017.

15. Li, H., T. Liu, X. Wu and S. Li, "Correlated SVD and Its Application in Bearing Fault Diagnosis", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 34, No. 1, pp. 355–365, 2023.

16. Aghababaeyan, Z., M. Abdellatif, M. Dadkhah and L. Briand, "DeepGD: A Multi-Objective Black-Box Test Selection Approach for Deep Neural Networks", ArXiv:2303.04878, 2023.

17. Wang, D. and Y. Shang, "A New Active Labeling Method for Deep Learning", *International Joint Conference on Neural Networks*, pp. 112–119, Beijing, China, 2014.

18. Deng, L., "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]", *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp. 141–142, 2012.

19. Krizhevsky, A., "Learning Multiple Layers of Features from Tiny Images", 2009, `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`, accessed on May 22, 2023.

20. Xiao, H., K. Rasul and R. Vollgraf, "Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms", ArXiv:1708.07747, 2017.

21. Abadi, M., P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu and X. Zheng, "TensorFlow: A System for Large-Scale Machine Learning", *12th USENIX Symposium on Operating Systems Design and Implementation*, pp. 265–283, Savannah, GA, 2016.

22. Ketkar, N., *Deep Learning with Python*, Apress, Berkeley, CA, 2017.

23. Kim, J., R. Feldt and S. Yoo, "Guiding Deep Learning System Testing Using Surprise Adequacy", *IEEE/ACM 41st International Conference on Software Engineering*, pp. 1039–1049, Montreal, QC, Canada, 2019.

24. Li, Y., M. Li, Q. Lai, Y. Liu and Q. Xu, "Testrank: Bringing Order Into Unla-

beled Test Instances for Deep Learning Tasks", *Advances in Neural Information Processing Systems*, Vol. 34, pp. 20874–20886, 2021.

25. Byun, T., V. Sharma, A. Vijayakumar, S. Rayadurgam and D. Cofer, "Input Prioritization for Testing Neural Networks", *IEEE International Conference On Artificial Intelligence Testing*, pp. 63–70, Newark, CA, USA, 2019.

26. Shen, W., Y. Li, L. Chen, Y. Han, Y. Zhou and B. Xu, "Multiple-Boundary Clustering and Prioritization to Promote Neural Network retraining", *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, p. 410–422, Virtual Event, Australia, 2021.

27. Hu, Q., Y. Guo, X. Xie, M. Cordy, W. Ma, M. Papadakis and Y. L. Traon, "Evaluating the Robustness of Test Selection Methods for Deep Neural Networks", ArXiv:2308.01314, 2023.

28. Zhao, C., Y. Mu, X. Chen, J. Zhao, X. Ju and G. Wang, "Can Test Input Selection Methods for Deep Neural Network Guarantee Test Diversity? A Large-Scale Empirical Study", *Information and Software Technology*, Vol. 150, No. C, p. 106982, 2022.

29. Yuan, Y., Q. Pang and S. Wang, "Revisiting Neuron Coverage for DNN Testing: A Layer-Wise and Distribution-Aware Criterion", *IEEE/ACM 45th International Conference on Software Engineering*, pp. 1200–1212, Melbourne, Australia, 2023.

30. Chen, J., Z. Wu, Z. Wang, H. You, L. Zhang and M. Yan, "Practical Accuracy Estimation for Efficient Deep Neural Network Testing", *ACM Transactions on Software Engineering and Methodology*, Vol. 29, No. 4, pp. 1–35, 2020.

31. Ma, L., F. Juefei-Xu, M. Xue, B. Li, L. Li, Y. Liu and J. Zhao, "DeepCT: Tomographic Combinatorial Testing for Deep Learning Systems", *IEEE 26th International Conference on Software Analysis, Evolution and Reengineering*, pp. 614–618,

Hangzhou, China, 2019.

32. Sun, Y., X. Huang, D. Kroening, J. Sharp, M. Hill and R. Ashmore, "Structural Test Coverage Criteria for Deep Neural Networks", *ACM Transactions on Embedded Computing Systems*, Vol. 18, No. 5s, pp. 1–23, 2019.

33. Xie, X., T. Li, J. Wang, L. Ma, Q. Guo, F. Juefei-Xu and Y. Liu, "NPC: Neuron Path Coverage via Characterizing Decision Logic of Deep Neural Networks", *ACM Transactions on Software Engineering and Methodology*, Vol. 31, No. 3, pp. 1–27, 2022.

34. Gerasimou, S., H. F. Eniser, A. Sen and A. Cakan, "Importance-Driven Deep Learning System Testing", *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, p. 702–713, Seoul, South Korea, 2020.

35. Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller and W. Samek, "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation", *PLOS One*, Vol. 10, No. 7, pp. 1–46, 2015.

36. Xie, X., 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", *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, p. 146–157, Beijing, China, 2019.

37. Lee, S., S. Cha, D. Lee and H. Oh, "Effective White-Box Testing of Deep Neural Networks with Adaptive Neuron-Selection Strategy", *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, p. 165–176, Virtual Event, USA, 2020.

38. Zhang, P., B. Ren, H. Dong and Q. Dai, "CAGFuzz: Coverage-Guided Adversarial Generative Fuzzing Testing for Image-Based Deep Learning Systems", *IEEE Transactions on Software Engineering*, Vol. 48, No. 11, pp. 4630–4646, 2022.

APPENDIX A: KULLANILAN FİGÜRLER

Bu tez çalışması kapsamında ortaya çıkan ve telif hakkı yayınevine devredilen görseller, yayınevinin kendi ağ sayfasında bulunan "yazarın kendi ürettiği yazı ve grafiklerin tekrar kullanımı hakkında geçerli olan yayın politikası"na uygun olarak tez kitabında kullanılmıştır.