Callisto: Entropy-based Test Generation and Data Quality Assessment for Machine Learning Systems

Sakshi Udeshi*, Xingbin Jiang† and Sudipta Chattopadhyay‡
Singapore University of Technology and Design
Email: *sakshi_udeshi@mymail.sutd.edu.sg, †xingbin_jiang@sutd.edu.sg, ‡sudipta_chattopadhyay@sutd.edu.sg

***Abstract*—Machine Learning (ML) has seen massive progress in the last decade and as a result, there is a pressing need for validating ML-based systems. To this end, we propose, design and evaluate CALLISTO– a novel test generation and data quality assessment framework. To the best of our knowledge, CALLISTO is the first blackbox framework to leverage the uncertainty in the prediction and systematically generate new test cases for ML classifiers. Our evaluation of CALLISTO on four real world data sets reveals thousands of errors. We also show that leveraging the uncertainty in prediction can increase the number of erroneous test cases up to a factor of 20, as compared to when no such knowledge is used for testing.**

CALLISTO has the capability to detect low quality data in the datasets that may contain mislabelled data. We conduct and present an extensive user study to validate the results of CALLISTO on identifying low quality data from four state-of-the-art real world datasets.

I. INTRODUCTION

Due to the massive progress in Machine Learning (ML) in the last decade, its popularity now has reached a variety of application domains, including sensitive and safety critical domains, such as automotive, finance, education and employment. One of the key reasons to use ML is to automate mundane and error-prone manual tasks in decision making. This often results in defective machine-learning systems that slip into production run [19]. To alleviate this issue, we have developed CALLISTO, a test generation and data quality analysis tool. CALLISTO leverages the entropy of the outputs of the ML classifiers to quantify the uncertainty in the prediction of these classifiers.

To further elucidate our motivation to research this technique, we sketch two situations that are likely to occur in the future. We also show these problems that may crop up because of the widespread proliferation of ML and then we sketch a sample solution for these problems that uses our CALLISTO technique.

Test Generation: It is easy to see that the success of ML is critically dependent on our ability to collect and annotate data. As we move towards more and more sophisticated techniques and tools for data collection, we will need to update our toolkit for data management as well. A critical part of this ML pipeline remains the testing of classifiers produced.

Consider a testing framework which generates tests by applying simple transformations such as rotation, zooming and panning to the data in the datasets (training and testing datasets). These transformations are metamorphic transformations. Let a dataset have n data points. Assuming the test framework only runs three metamorphic tests, a naïve approach will have to run $3n$ tests. We believe a better approach is to use CALLISTO to identify inputs which are prone to errors. This set is usually much smaller than the full dataset and will exhibit highly erroneous behaviours. Test set minimisation is a powerful technique to reduce the effort to find errors without loss in effectiveness [24].

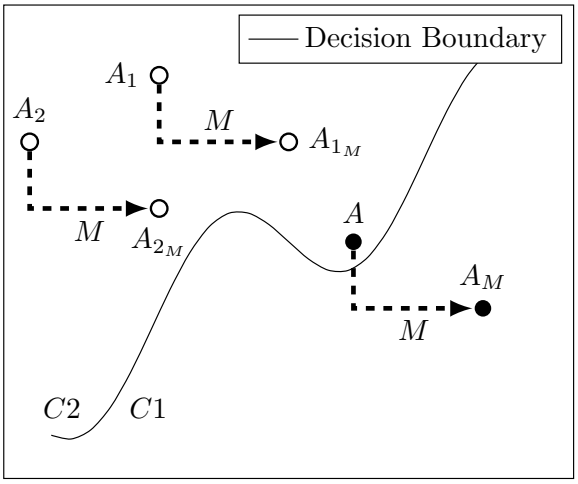


Fig. 1: Test CALLISTO intuition

CALLISTO employs techniques to quickly and efficiently identify the inputs in the datasets for test generation. This aids the user to discover erroneous behaviours without extensive testing of the entire dataset. We illustrate the intuition behind CALLISTO's test generation approach in Figure 1. Consider a metamorphic transformation M (e.g. rotating a picture by a small amount) and inputs A_1, A_2 and A. A rudimentary approach would be to apply M to all the data points leading to large computational overheads. CALLISTO aims to discover points like A which will allow users to selectively apply a metamorphic relation to inputs which are likely to cause errors. CALLISTO will avoid points such as A_1 and A_2.

Data Quality: With the ever-increasing volume of data in training ML systems, it is critical to identify the cases where the automatic data generation and/or labelling failed and are likely of low quality. A naïve approach is to engage humans to verify the data. The cost of such an approach is very high and recurring.

Our CALLISTO approach aims to identify the mislabelled and/or low quality inputs for further intervention. The size of

these identified sets should be significantly smaller in comparison to the full dataset and the cost of human intervention is minimal. This approach is first seen in identifying wrong labels [16]. Additionally, we conduct an extensive user study to validate our experiments. This strategy allows the data to be sanitised efficiently and will lead to building of high quality datasets with low computational overhead. Consequently, the quality of the classifiers will also improve.

Solution Sketch: In a typical ML classifier, there is usually a softmax layer [5]. This layer normalises the second last layer's output to a probability distribution of the number of output classes. The class with the highest probability is then identified as the prediction. The higher the probability, the more confident the DNN is in that particular prediction.

We aim to leverage the output layer of classifiers to make predictions about the data point. To illustrate this, consider data points A and B with label l_2 and output vectors, respectively as follows:

$$\begin{aligned}[l_0,\ l_1,\ l_2,\ l_3]_A &\approx [0.033, 0.033, 0.9, 0.034] \\ [l_0,\ l_1,\ l_2,\ l_3]_B &\approx [0.25, 0.2, 0.3, 0.25]\end{aligned} \tag{1}$$

Intuitively, the quality of the prediction for input B, despite being correct, is worse. This is because of the lack of high confidence. To quantify the confidence of a classifier prediction, we use Shannon Entropy [3]. The Shannon Entropy quantifies the diversity of the prediction. It is important to note that the Shannon Entropy has been used as a popular measure of diversity in ecological literature to quantify the diversity of a population [15]. It is defined as follows:

$$H' = -\sum_{j=1}^{N} l_j \ln l_j \tag{2}$$

In Equation (2), N is the number of labels and l_j is the probability that the prediction class belongs to the j^{th} label. For example, the H' value for input A is ≈ 0.44 and the same for input B is ≈ 1.38. Thus, the higher the Shannon index for an output the lower is the quality of the output.

We use this to build test sets for ML classifiers. The data points which have a high Shannon index are the ones likely to be affected most by metamorphic transformations. For Test Generation, we capture the inputs like datapoint B and employ metamorphic relations/perturbations to generate test examples efficiently.

Consider another data point C with label l_0. Let the prediction output is as follows:

$$[l_0,\ l_1,\ l_2,\ l_3] \approx [0.0033, 0.0033, 0.99, 0.0034] \tag{3}$$

The Shannon Entropy for the output is $\approx$0.066. Thus, the ML model is highly confident in the prediction. It is likely that such inputs are few in numbers, but it is nevertheless important to find these inputs. This is because these data points are the ones which are most likely to be mislabelled [16].

The remainder of the paper is organised as follows. After providing a brief background and related work (Section II), we make the following contributions:

1) We present CALLISTO, a novel approach to generate the tests for Machine Learning classifiers. We present the first technique, to the best of our knowledge which leverages the entropy of the output of the classifiers for effective test generation. CALLISTO is also completely blackbox and can be used easily for ML services. (Section III)
2) We re-implement the technique seen in [16] and replicate the results seen before for discovering mislabelled data [16]. We extend the results to include four datasets. (Section IV)
3) We conduct an extensive user study to validate the results of our experimentation. Additionally, we publicly release the results of the user study. (Section IV)
4) We provide the implementation and data of CALLISTO based on Python which is publicly available. (Section IV)

We finally discuss lessons learned from building CALLISTO, threats to its validity (Section V) and conclude (Section VI)

II. BACKGROUND AND RELATED WORK

In recent works, most of the state of the art image classifiers have been Deep Neural Networks (DNN). As a result of this trend, we present some background for DNNs and then move on to testing of ML systems.

Deep Neural Networks: We can think of a DNN as a function F with numerous parameters $F_\Theta : \mathbb{R}^N \rightarrow \mathbb{R}^M$. The function F with parameters Θ maps an input $x \in \mathbb{R}^N$ to an output $y \in \mathbb{R}^M$. Illustrating this with an example, consider an image x (reshaped as a vector) that has to be classified into one of m different classes. The last layer of the DNN is usually a softmax layer [5] that outputs y, which is a vector of the probabilities of m classes. The predicted label $\hat{y}$ is the class with the highest probability: $arg\ max_{i \in [1,M]}\ y_i$. In CALLISTO we use this layer's output to quantify the entropy of a particular input as seen in Equation (2).

Internally, the DNN can be understood as a feed forward network. The network has L hidden layers which consist of N_i neurons where $i \in [1, L]$. These neurons perform computations and the outputs of these computations are usually called *activations*. For the i^{th} layer the vector of activations can be written as follows

$$a_i = \Delta(w_i \cdot a_{i-1} + b_i) \quad \forall i \in [1, L] \tag{4}$$

where $a_i \in \mathbb{R}^{N_i}$ and $\Delta : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is an non-linear function. We can see that $a_0 = x$ and $N_0 = N$. This can be interpreted as the inputs to the first layer and the input to the network is the same.

The parameters $w_i \in \mathbb{R}^{N_{i-1}} \times N_i$ and $b_i \in \mathbb{R}^{N_i}$ of Equation (4) are fixed weights and biases respectively and these are learnt during the training phase. The output of the network is a function of these activations, $\gamma(w_{L+1} \cdot a_L + b_{L+1})$, where $\gamma : \mathbb{R}^N \rightarrow \mathbb{R}^M$ is usually the softmax layer [5].

Testing DNN Systems: One of the first works to test DNNs [14] presents a whitebox differential testing algorithm for systematically finding errors in DNNs. Another early work [18] uses a metamorphic testing approach to find bugs in DNNs. A

feature-guided black-box approach [23] was also proposed to validate the safety of DNNs. A set of testing criteria based on multi level and granularity coverage for testing DNNs is proposed by DeepGauge [11]. Aequitas [20] aims to uncover fairness violations in machine learning models. DeepConcolic [17] designs a framework to perform concolic testing for discovering robustness violations. A recent work [21] uses model mutation methods to detect adversarial attacks. PMV [26] proposes a new technique to validate model relevance and detect underfitting or overfitting in ML models.

The goal of CALLISTO is to aid the existing testing systems by minimising the test set. Specifically, adding tests using all available data is computationally expensive. CALLISTO seeks to make this problem more tractable by constructing a smaller set of inputs for testing. Additionally, the inputs should be as effective in uncovering low quality data [16].

Verification of DNN Systems: In contrast to works that attempt verification of DNN systems [4], [6], [12], [22]CALLISTO aims to preserve the flavour of testing in contrast to these approaches. In addition to test generation, CALLISTO also flags data which may be of low quality.

Guiding Tests for DNN Systems: As testing of ML systems becomes more mainstream, the need to guide these massively data intensive systems is apparent. To this end, a recent work [7] proposes a new test coverage metric, called *Surprise Adequacy*. This is based on the behaviour of models with respect to their training data and develops adequacy criteria for tests. Another work [16] uses entropy to identify mislabelled data.

CALLISTO is the first work, to the best of our knowledge, to use entropy for generating tests and minimise test sets. CALLISTO is also fully blackbox in contrast to other test coverage metrics [7]. Finally, CALLISTO reimplements an earlier approach [16], extends the approach to real word datasets and conducts a user study to validate the approach, showing that the metric targeted by CALLISTO can be leveraged for multiple use cases in the ML domain.

III. METHODOLOGY

In this section, we elucidate the methodologies behind CALLISTO in detail. CALLISTO consists of two base algorithms. The first one is the test generation framework and another one evaluates data quality. Both of these algorithms leverage the Shannon Diversity index [15] to automatically select data points for test generation and to evaluate data quality respectively.

We now introduce some notations that help us to illustrate our CALLISTO approach. These notations are outlined in Table I.

Test generation in CALLISTO: Algorithm 1 and Figure 3 outline the overall test generation process that CALLISTO implements. The input is a large corpus of data points (usually the training data and the testing data) X and the labels of the training data Y. The training data set is generally very large and it is computationally expensive to generate and evaluate metamorphic tests for all $x_i \in X$.

To aid efficient test generation, CALLISTO generates tests only for those data points where it is likely for the metamorphic transformations to cause an error. Such data points are the ones whose output $f(x_i)$ has a Shannon index $s_i \in \mathbb{S}_X$ such that $s_i > \tau_{high}$. Intuitively, these data points indicate scenarios where the model f had low confidence in the prediction. We also impose the additional condition that the output $f(x_i)$ is equal to the label $y_i \in Y$. We construct a set $\mathbb{G}$ that contains data points satisfying these conditions. It is important to note that these are clean inputs and not adversarially crafted inputs where the confidence is artificially high. As a result, we expect a well trained model to be confident of the output and low confidence is an indication of a gap in learning [16].

Once the set $\mathbb{G}$ is constructed, CALLISTO iterates over each datapoint in this set and recursively applies a pre-selected metamorphic transformation. We choose transformations similar to those seen in [18]. For example, consider a data point $x_G \in \mathbb{G}$ where we pick a transformation $T \in \delta_{meta}$ and apply it to x_G to produce x_G^T. Once we have this input x_G^T, we add x_G^T to the error set if and only if $f(x_G^T) \neq f(x_G)$. CALLISTO then returns the error set for the user to evaluate. We summarise this approach in Figure 3.

Identifying low quality data in CALLISTO: The aim of this part of CALLISTO is to create a set $\mathbb{F}$ which flags and identifies potential low quality data. In an annotated dataset which is used for supervised learning, it is possible that some data is mislabelled and/or ambiguous. Manually going through this

TABLE I: Notations used in CALLISTO approach

f	The machine learning model under test.
X	The vector of data points.
Y	The vector of the ground truth labels of the data points.
$\hat{Y}$	The vector of the predictions of the data points.
$\mathbb{S}_X$	The Shannon indices [3] for all $x \in X$
τ_{low}	The lower threshold of the Shannon index. Elements with Shannon index lower than τ_{low} are considered to have very high confidence in their prediction.
τ_{high}	The higher threshold of the Shannon index. Elements with Shannon index higher than τ_{high} are considered to have very little confidence in their prediction.
δ_{meta}	The set of valid metamorphic transformations

Algorithm 1 CALLISTO for Test generation

```
1: procedure GENERATE(X, Y, S_X, δ_meta, τ_high)
2:     G ← ∅
3:     errors ← ∅
4:     for s_{x_i} ∈ S_X do
5:         if s_{x_i} > τ_high and y_i == ŷ_i then z
6:             G ← G ∪ {x_i}
7:         end if
8:     end for
9:     for x_i ∈ G do
10:        ▷ Choose a transformation to apply
11:        T ← Choice (δ_meta)
12:        x_i^T ← Apply_Transform (x_i, T)
13:        if f(x_i^T) ≠ y_i then
14:            errors ← errors ∪ {x_i^T}
15:        end if
16:    end for
17:    return errors
18: end procedure
```

Algorithm 2 CALLISTO for Low Data Quality Detection

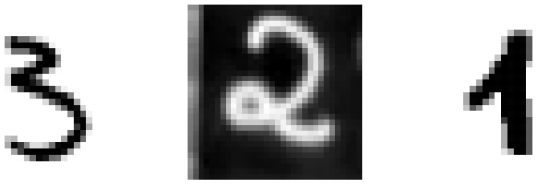
```
1: procedure DETECT(X, Y, 𝕊_X, τ_low)
2:     𝔽 ← ϕ
3:     for s_{x_i} ∈ 𝕊_X do
4:         if s_{x_i} < τ_low and y_i ≠ ŷ_i then
5:             𝔽 ← 𝔽 ∪ {x_i}
6:         end if
7:     end for
8:     return 𝔽
9: end procedure
```

data incurs high cost. We formalise the notion below.

Definition 1. ***(Low Quality Data)*** *Consider a data point $x \in X$ with label $y \in Y$ and prediction $\hat{y} \in \hat{Y}$. We consider the data point x to be of low quality if y is not the correct label and instead $\hat{y}$ is the*

Algorithm 2 illustrates the CALLISTO approach to discover low quality data. For each input $x_i \in X$, we check the Shannon index $s_{x_i} \in \mathbb{S}_X$. A Shannon index lower than the threshold τ_{low} can be interpreted as the model f having high confidence in the prediction. CALLISTO also requires that the predicted value $\hat{y}_i$ to be different than the label y_i. Intuitively, this means that the prediction for data point x_i has high confidence, yet the prediction does not match with the label. Thus, it is likely that the label y_i is incorrect and therefore, x_i is considered to be of low quality. As in the previous section, it is important to note that x_i is a clean input and not an adversarially crafted input where the confidence is artificially high.

Some of the examples of low quality data found by CALLISTO can be seen in Figure 2.



Label - 5
Prediction - 3
(a)

Label - 6
Prediction - 2
(b)

Label - 4
Prediction - 1
(c)

Fig. 2: Some examples of low quality data found in the MNIST [9] and SVHN [9] datasets

To further validate the efficacy of CALLISTO, we have designed a user study. The objective of this user study is to check whether the low quality data points discovered by CALLISTO are indeed of low quality perceived by users. To design the user study, we constructed a set of representative examples of the low quality data found in the MNIST Digit [9], Fashion-MNIST [25], CIFAR-10 [8] and SVHN [13] datasets. These examples were presented to users and they were asked to choose the correct output class of an example x between two options. The first option was the respective prediction $\hat{y}$ or the second option was the label y. Of course, the sequence

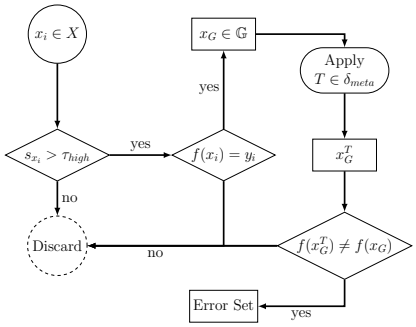


Fig. 3: Test generation with CALLISTO

TABLE II: Test Generation Effectiveness

	Ratio of erroneous inputs (#error/#test)			
Shannon Threshold	Panning	2D rotation	Affine	Perspective
Fashion MNIST				
$s_x < 0.001$	0.19	0.58	0.41	0.07
$s_x > 0.4$	0.61	0.78	0.80	0.58
MNIST-Digit				
$s_x < 0.001$	0.20	0.12	0.07	0.03
$s_x > 0.4$	0.65	0.51	0.45	0.51
CIFAR-10				
$s_x < 0.001$	0.09	0.55	0.53	0.27
$s_x > 0.4$	0.40	0.62	0.63	0.51
SVHN				
$s_x < 0.001$	0.03	0.46	0.59	0.14
$s_x > 0.4$	0.54	0.79	0.84	0.58

in which these options appear was randomized for each user. We also asked each user to rate the confidence of her choice on a five point Likert Scale [10]. Examples of the questions that we asked for Figure 2(a) are as follows:

```
What is the number displayed above?
        Option a - 5
        Option b - 3

How confident are you in your answer?
          1  2  3  4  5
 Low Confidence      High Confidence
```

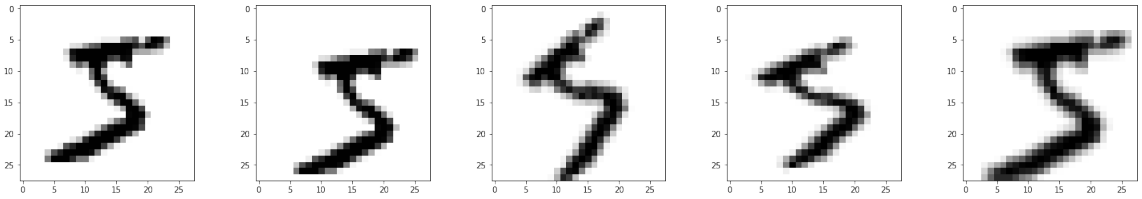
We elaborate on the results of this user study in Section IV.

IV. RESULTS

To evaluate the efficacy of CALLISTO, we answer the following research questions.

RQ1.1: Is the test generation effective?

To evaluate the effectiveness of this research question, we compute the ratio of erroneous inputs obtained via metamorphic transformations. We say a transformed input is an error when the prediction of the model does not match the corresponding label. In our evaluation, we picked four transformations, namely panning, 2D rotation, affine and perspective. These transformations can be seen in Figure 4 and are similar to the transformations seen in state-of-the-art work [18]. It has been shown that Machine Learning models are not robust to



Original (a) | Panning (b) | 2D Rotation (b) | Affine (d) | Perspective (e)

Fig. 4: Transformations of an image

even simple transformations [2]. The objective for Callisto is to minimise the number of transformations and maximise the number of erroneous inputs. We show that the Shannon index is an effective measure to maximise the error rates. Inputs with a high Shannon index (more than 0.4), as seen in Table II, consistently show higher error rates in comparison to the entire dataset (Threshold - 0.0). Moreover, the effectiveness of the test generation does not depend on the type of transformation. All four of the transformations show an increase in the ratio of erroneous inputs with a higher Shannon index.

RQ1.2: How do the error rates vary with Shannon index?

To answer this research question, we evaluated the error rates for the Fashion MNIST as seen in Figure 5. Intuitively, the error rates should increase with an increase in the Shannon Threshold. This is because the model outputs that have a high Shannon index can be understood as being less confident in their prediction and being more brittle. Thus the respective inputs are more prone to errors due to transformations.

As we can see in Figure 5, our intuition holds. An increase in the Shannon threshold causes an increase in the error rates across all the transformations. For example, the error rate increased up to almost ten times in the case of the perspective transformation. This trend holds for all the datasets, please refer to the supplementary materials (Section VI).

RQ2: Is low quality data effectively identified?

The usage of the Shannon diversity to identify mislabelled data was first proposed to detect mislabelled training instances [16]. Our solution to detect low quality data (i.e. Algorithm 2) is similar, but we conduct a thorough user study to validate our results.

We conducted a survey with representative examples from the MNIST-Digit, Fashion-MNIST, SVHN and CIFAR-10 datasets. These examples had incorrect predictions and the prediction outputs had low Shannon index. Our intuition is that these are the images which are likely of low quality. To evaluate this, we conducted a survey on the Amazon's mTurk [1] with 197 users. We asked the users to choose between the label and

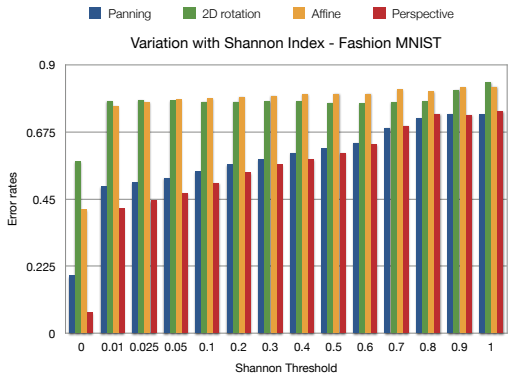


Fig. 5: Test generation with CALLISTO

TABLE III: Identifying low quality data

	Image #1	Image #2	Image #3
Fashion MNIST			
%users choosing prediction	95.5	94.5	83.90
Confidence (1 to 5)	4.17	3.71	4.40
MNIST-Digit			
%users choosing prediction	98	100	90.50
Confidence (1 to 5)	4.54	4.76	4.39
CIFAR-10			
%users choosing prediction	32.7	27.1	85.9
Confidence (1 to 5)	3.77	3.34	3.32
SVHN			
%users choosing prediction	100	100	100
Confidence (1 to 5)	4.74	4.82	4.92

the predicted output for three representative examples (from each dataset) which were predicted incorrectly, but had a low Shannon index (i.e. high output confidence).

As seen in Table III, for the Fashion MNIST, MNIST-Digit and SVHN the prediction were largely chosen over the label and the confidence is generally high. For CIFAR-10, in two out of the three examples, the label was chosen as a majority. We believe that the low confidence across CIFAR indicates that these images are generally hard to label and understand. Nevertheless, we acknowledge that further investigations are required to detect low quality data in datasets like CIFAR.

V. THREATS TO VALIDITY

Testing Vision Systems: The test subjects for CALLISTO are exclusively vision systems. We have not validated CALLISTO

on other classifiers, such as text classifiers. For other types of ML systems, CALLISTO might need to involve additional metamorphic transformations. Nonetheless, we believe that Shannon Entropy, as leveraged by CALLISTO, is a general property that can easily be used by other types of classifiers.

CIFAR-10 Data Quality: For CALLISTO we have reimplemented the approach first seen for mislabelling datasets [16]. This approach does not perform well for the CIFAR-10 dataset in two out of the three representative images as seen in Table III. Further investigation is necessary along this line of research.

VI. CONCLUSION

In this paper, we present CALLISTO, a novel entropy-based test generation framework. We show that using existing information about the output uncertainty, we can effectively generate erroneous inputs. We also reimplement and validate an approach first seen for finding mislabelled data [16]. Additionally, we have conducted extensive user studies to try and validate the results seen in CALLISTO.

CALLISTO is completely blackbox and does not require any information about the structure of classifiers and can easily be deployed for testing ML services. To promote research in this area and reproduce our results, we have made our implementation and all experimental data publicly available: https://github.com/sakshiudeshi/Callisto

REFERENCES

[1] Amazon Mechanical Turk, 2109. URL: https://www.mturk.com/.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 284–293, 2018. URL: http://proceedings.mlr.press/v80/athalye18b.html.

[3] Monica Borda. *Fundamentals in information theory and coding*. Springer Science & Business Media, 2011.

[4] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin T. Vechev. AI2: safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 3–18, 2018.

[5] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[6] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, pages 97–117, 2017.

[7] Jinhan Kim, Robert Feldt, and Shin Yoo. Guiding deep learning system testing using surprise adequacy. In *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*, pages 1039–1049, 2019. URL: https://doi.org/10.1109/ICSE.2019.00108, doi:10.1109/ICSE.2019.00108.

[8] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[9] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL: http://yann.lecun.com/exdb/mnist/ [cited 2016-01-14 14:24:11].

[10] Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[11] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. Deepgauge: multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, Montpellier, France, September 3-7, 2018*, pages 120–131, 2018.

[12] Ramon E. Moore, R. Baker Kearfott, and Michael J. Cloud. *Introduction to Interval Analysis*. SIAM, 2009. URL: https://doi.org/10.1137/1.9780898717716, doi:10.1137/1.9780898717716.

[13] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

[14] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 1–18, 2017.

[15] Ian F Spellerberg and Peter J Fedor. A tribute to claude shannon (1916–2001) and a plea for more rigorous use of species richness, species diversity and the 'shannon–wiener'index. *Global ecology and biogeography*, 12(3):177–179, 2003.

[16] Jiangwen Sun, Feng-ying Zhao, Chong-Jun Wang, and Shifu Chen. Identifying and correcting mislabeled training instances. In *Future Generation Communication and Networking, FGCN 2007, Ramada Plaza Jeju, Jeju-Island, Korea, December 6-8, 2007, Proceedings*, pages 244–250, 2007. URL: https://doi.org/10.1109/FGCN.2007.146, doi:10.1109/FGCN.2007.146.

[17] Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. Concolic testing for deep neural networks. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, Montpellier, France, September 3-7, 2018*, pages 109–119, 2018.

[18] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*, pages 303–314, 2018.

[19] Songül Tolan, Marius Miron, Emilia Gómez, and Carlos Castillo. Why machine learning may lead to unfairness: Evidence from risk assessment for juvenile justice in catalonia. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law, ICAIL 2019, Montreal, QC, Canada, June 17-21, 2019*, pages 83–92, 2019. URL: https://doi.org/10.1145/3322640.3326705, doi:10.1145/3322640.3326705.

[20] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. Automated directed fairness testing. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, Montpellier, France, September 3-7, 2018*, pages 98–108, 2018.

[21] Jingyi Wang, Guoliang Dong, Jun Sun, Xinyu Wang, and Peixin Zhang. Adversarial sample detection for deep neural network through model mutation testing. In *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montréal, Canada, May 25 - May 31, 2019*, 2019.

[22] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.*, pages 1599–1614, 2018.

[23] Matthew Wicker, Xiaowei Huang, and Marta Kwiatkowska. Feature-guided black-box safety testing of deep neural networks. In *Tools and Algorithms for the Construction and Analysis of Systems - 24th International Conference, TACAS 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, Part I*, pages 408–426, 2018.

[24] W. Eric Wong, Joseph Robert Horgan, Saul London, and Aditya P. Mathur. Effect of test set minimization on fault detection effectiveness. In *17th International Conference on Software Engineering, Seattle, Washington, USA, April 23-30, 1995, Proceedings*, pages 41–50, 1995.

[25] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. arXiv:cs.LG/1708.07747.

[26] Jie M. Zhang, Earl T. Barr, Benjamin Guedj, Mark Harman, and John Shawe-Taylor. Perturbed model validation: A new framework to validate model relevance. *CoRR*, abs/1905.10201, 2019. URL: http://arxiv.org/abs/1905.10201.