A Tale of EDA's Long Tail: Long-Tailed Distribution Learning for Electronic Design Automation

Zixuan Jiang[*]
zixuan@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

Mingjie Liu[*]
jay_liu@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

Zizheng Guo
gzz@pku.edu.cn
School of Integrated Circuits
Peking University
Beijing, China

Shuhan Zhang
shuhan.zhang@utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

Yibo Lin
yibolin@pku.edu.cn
School of Integrated Circuits
Peking University
Beijing, China

David Pan
dpan@ece.utexas.edu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas, USA

ABSTRACT

Long-tailed distribution is a common and critical issue in the field of machine learning. While prior work addressed data imbalance in several tasks in electronic design automation (EDA), insufficient attention has been paid to the long-tailed distribution in real-world EDA problems. In this paper, we argue that conventional performance metrics can be misleading, especially in EDA contexts. Through two public EDA problems using convolutional neural networks and graph neural networks, we demonstrate that simple yet effective model-agnostic methods can alleviate the issue induced by long-tailed distribution when applying machine learning algorithms in EDA.

CCS CONCEPTS

• **Hardware → Best practices for EDA**; *Placement*.

KEYWORDS

long-tailed distribution; machine learning; wafer defect classification; net delay prediction

1 INTRODUCTION

Long-tailed distribution is a ubiquitous phenomenon in real-world problems. The instance frequency is dominated by the head that accounts for a small portion of possible values, while a long tail is formed by the majority of possible values where the instance frequency is small. Eigenvalues of realistic massive graphs, ranging from internet graphs to social networks, follow the power-law degree distribution [7]. In quantitative linguistics, researchers have observed that the corpus of natural language utterances follows Zipf's law [16], where the frequency of words is inversely proportional to their rank. It is extremely challenging to use machine learning algorithms to handle these long-tailed distributions, since trained models can be easily biased towards the head, leading to poor performance on the tail with limited occurrence.

It is perhaps unsurprising that random variables follow long-tailed distributions in the area of electronic design automation (EDA). Anomaly detection tasks in EDA contexts are formulated as binary classification, and "regular" classes usually dominate abnormalities in terms of quantity, which exhibits severe imbalance. For instance, in the hotspot detection problem, the number of hotspots is much smaller than that of non-hotspots, resulting in biased classification towards non-hotspots [9]. Similarly, routing congestion predictions are less accurate when testing on small-sized designs [21]. The placement and routing algorithms can find near-optimal solutions for small circuits, thus generating fewer congestion data samples. On the contrary, we obtain more congestion data points for large circuits. The long-tailed distribution impedes the performance of the tail, which is the prediction on small designs.

While the data imbalance issue could be addressed for binary classification tasks, this issue is easily overlooked in multi-class classification and especially regression tasks in EDA problems. In post-layout parasitic estimation, the authors of [17] observed poor prediction results in high-value net capacitance, since models are biased towards low-value capacitance whose frequency is much higher in the training dataset. We target at reducing model "pessimism" with decreased mean squared error (MSE) and increased correlation between prediction and ground truth, when predicting net delay [5] and routed wirelength [20] from circuit pre-routing information in prior work. Although conventional metrics, such as accuracy and MSE, are strong indicators of model performance, these metrics can become significantly misleading for multi-class classification and regression tasks when tackling problems with long-tailed distributions. Especially in the scope of EDA, we tend to put more weight on the *abnormal* and *extreme* cases, such as the *worst* negative slack timing of circuit designs.

[*]Both authors contributed equally to this research.

MLCAD '22, September 12–13, 2022, Snowbird, UT, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9486-4/22/09…$15.00
https://doi.org/10.1145/3551901.3556485

In the machine learning community, considerable progress has recently been made to address the challenges of long-tailed distribution [6, 12, 14, 22], from the perspectives of theories, algorithms, and applications. Now that more and more machine learning algorithms are applied to EDA, it is a great opportunity to leverage these advances in machine learning.

In this paper, we address and draw attention to the common, critical, yet overlooked issue of long-tailed distribution in EDA problems. We introduce the long-tailed distributions and related machine learning methods, which can be leveraged and imported into the field of EDA. Then, through two cases, wafer map defect pattern classification and net delay prediction, we demonstrate how we address the long-tailed distribution in real problems. We leverage model-agnostic methods, specifically reweighting and resampling, and demonstrate their effectiveness on these two problems with convolutional neural networks (CNNs) and graph neural networks (GNNs), respectively.

2 LONG-TAILED DISTRIBUTION

In this section, we give a short introduction to the long-tailed distribution and related machine learning algorithms. We also discuss the performance metrics when we handle problems with long-tailed distribution.

2.1 Long-Tailed Categorical Distribution

A discrete random variable Y follows a long-tailed categorical distribution if it has a high probability of occurrence on the head classes and a low probability of occurrence on the tail classes, i.e., $p(Y = t) \ll p(Y = h)$ where t and h are labels of a tail class and a head class, respectively. In a classification problem on long-tailed categorical distribution, we predict the class label y given an input feature vector x. Namely, our target is to build a conditional probability model $p(y|x)$ where the label distribution $p(y)$ is long-tailed.

Long-tailed distributions also exist in continuous and discrete (but not categorical) random variables. However, in literature, most of the algorithms are for classification problems. Moreover, these continuous and discrete (but not categorical) random variables can be converted into categorical variables. Above all, we focus on the long-tailed categorical distributions in this paper.

2.2 Performance metrics

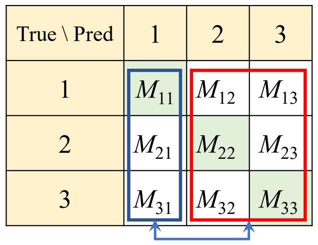
Measurement of imbalance. Let D be a dataset with long-tailed categorical distribution with C classes. Let n_i be the number of samples in the i-th class sorted by n_i in non-decreasing order. We demonstrate three widely used metrics to measure the imbalance in a long-tailed categorical distribution, whose definitions are listed below,

$$\texttt{Imbalance Ratio:}\ \frac{n_C}{n_1} \tag{1}$$

$$\texttt{Imbalance Divergence:}\ \sum_i p_i \log \frac{p_i}{q_i} \tag{2}$$

$$\texttt{Gini Coefficient:}\ \frac{\sum_i n_i(2i - C - 1)}{C \sum_i n_i} \tag{3}$$

where $p_i = \frac{n_i}{\sum_i n_i}$ is the frequency of the i-th class, and $q_i = 1/C$ is the probability in a uniform distribution. The imbalance ratio measures the gap between two ends, while the other metrics evaluate the divergence between the given long-tailed distribution and uniform distribution. These metrics are 1, 0, and 0 for uniform distributions and become larger for more imbalanced distributions.



True \ Pred	1	2	3
1	M_{11}	M_{12}	M_{13}
2	M_{21}	M_{22}	M_{23}
3	M_{31}	M_{32}	M_{33}

Figure 1: The confusion matrix with 3 classes. Its entry M_{ij} is the number of data points whose true label is i and predicted label is j. We assume that $p(y = 3) \leq p(y = 2) \ll p(y = 1)$, meaning that Class 1 forms the head while the other two are tail classes. If our prediction is always 1, the label of the head class, meaning that we decrease the numbers in the red box and increase the numbers in the blue box, we can achieve high accuracy at the cost of low precision on the head class and small recall on tail classes.

Metrics on classification performance. We then introduce the metrics for a classification problem with C classes. A confusion matrix M is a $C \times C$ square matrix that allows analysis and visualization of the final results, as shown in Figure 1. Specifically, its entry M_{ij} is the number of data points whose true label is i and predicted label is j. Therefore, all correct predictions are located in the diagonal of this matrix. The classification accuracy can be calculated as $\frac{\sum_{i=j} M_{ij}}{\sum M_{ij}}$. There are two widely used performance metrics based on the confusion matrix. For the i-th class, its *precision* p_i, *recall* r_i are defined as follows.

$$p_i = \frac{M_{ii}}{\sum_j M_{ji}}, r_i = \frac{M_{ii}}{\sum_j M_{ij}} \tag{4}$$

Namely, given a data sample whose true label is i, the r_i is the frequency (probability) that we can correctly predict its label. The harmonic mean of precision and recall is the F1 score $\frac{2}{p_i^{-1}+r_i^{-1}}$. Given a label distribution $p(y)$, the classification accuracy is the weighted average recall, as shown below.

$$\sum_{i=1}^{C} p(y = i) r_i \tag{5}$$

With a uniform label distribution, i.e., $p(y = i) = 1/C, \forall i \in [1, C]$, the weighted average recall is also called macro average recall $\frac{1}{C}\sum_{i=1}^{C} r_i$.

For a classification problem on long-tailed categorical distributions, the weighted average recall (i.e., accuracy) can only demonstrate one perspective of the performance. We may list the confusion matrix and calculate the precision and recall of each class for detailed comparisons.

We can also interpret the classification on long-tailed distribution as a multi-objective optimization problem. The vector of recall $\mathbf{r} = [r_1, r_2, ..., r_C]$ can be an alternative performance metric. If $\mathbf{r}_a < \mathbf{r}_b$

$(\forall i \in [1, C], \mathbf{r}_a[i] < \mathbf{r}_b[i])$, then the $\mathbf{r}_b$ dominates $\mathbf{r}_a$, meaning that the prediction b always achieves a better accuracy than a for any label distributions $p(y)$.

2.3 Long-Tailed distribution in Machine Learning

Classification problems on long-tailed distribution have been obtained much attention in the machine learning community [6, 12, 14]. We can only provide a short description of several methods used in the following case studies. We refer readers to [22] and related papers for details regarding these methods and other methods.

Reweighting. We typically use softmax cross entropy loss to train a classifier in machine learning algorithms. It is natural and intuitive to assign different weights to data samples in different classes.

Given an predicted logit z, we can obtain the predicted probability vector p. With the true label of c, we list several loss functions to conduct reweighting,

$$\texttt{Cross entropy} \ -\log p_c \tag{6}$$

$$\texttt{Weighted loss} \ -\frac{1}{n_c^{\gamma}} \log p_c \tag{7}$$

$$\texttt{Focal loss} \ -(1-p_c)^{\gamma} \log p_c \tag{8}$$

$$\texttt{Class-balanced loss} \ -\frac{1-\gamma}{1-\gamma^{n_c}} \log p_c \tag{9}$$

$$\texttt{Balanced softmax} \ -\log\left(\frac{n_c e^{z_c}}{\sum_i n_i e^{z_i}}\right) \tag{10}$$

where γ is a hyperparameter and n_i is the number of samples in the i-th class. Compared with the baseline of cross entropy loss, the weighted loss [11] and the class-balanced loss [8] assign a larger weight to the data samples belonging to tail classes. Based on the observation that machine learning models often exhibit high confidence in the head classes but low confidence in the tail classes, the focal loss [15] assigns larger weights to the less confident data points to improve the learning on tail classes. By leveraging the prior knowledge that we focus on the macro average recall on the test distribution, balanced softmax loss [18] is proposed to use the label frequencies to adjust predicted logits to mitigate the bias of class imbalance.

Resampling. In [13], the authors evaluate the following sampling strategies for representation learning on long-tailed categorical distributions.

- *Instance-balanced sampling.* All data points share the same sampling probability, which is the baseline method.
- *Class-balanced sampling.* All classes share the same sampling probability $1/C$. All data points in the same class share the same sampling probability.
- *Square root sampling.* The sampling probability of the i-th class is proportional to $\sqrt{n_i}$ to mitigate the imbalance issue.
- *Progressively-balanced sampling.* We start from the instance-balanced sampling, and gradually move forward to the class-balanced sampling.

Two-stage learning. We can first pre-train a model on the original long-tailed dataset using the conventional cross entropy loss, which helps the representation learning. Afterwards, we can use the methods mentioned above to further fine-tune the model, empowering the machine learning models to handle long-tailed label distributions.

Issues in the existing algorithms. Most algorithms and benchmarks in the machine learning community pay much attention to the macro average recall, which is the accuracy under a uniform label distribution. Also, most of the algorithms leverage the prior knowledge that the testing datasets have uniform label distributions, As a result, these methods usually make the trade-off between the recall of head and tail classes. They usually achieve a higher recall on tail classes at the expense of slight degradation on the recall of head classes. From the perspective of the confusion matrix shown in Figure 1, we increase the numbers in the red box and decrease the numbers in the blue box to achieve a higher macro average recall.

3 CASE STUDIES WITH OPEN-SOURCE EDA DATASETS

In this section, we cite two problems with long-tailed categorical distribution in electronic design automation. The problems are identifying the failure pattern in the wafer maps and net delay prediction.

3.1 Wafer Map Defect Classification

In the semiconductor industry, we need to check if there is any problem in every single fabricated wafer. If yes, we would like to know the exact type of failure pattern such that we can further investigate the reason for failure and fix the related issues. In the traditional routine, the wafer manufacturing process is monitored by experienced engineers, who can classify the failure patterns and are aware of the failure cause. However, it is impossible and tedious for experts to monitor every single wafer. Our objective is to identify different types of wafer map failure patterns automatically instead of manual work. Specifically, we are required to predict a categorical label based on an input wafer map, which is represented by a matrix.

It is typical that the label distribution is long-tailed. Most of the wafers do not have a failure pattern, while the wafers with failure only account for a small portion of the whole distribution. For example, we list the number of data samples in each class in the industrial WM-811K wafer map training dataset [19] in Table 1. The imbalance ratio, the imbalance divergence, and the Gini coefficient are 680, 1.086, and 0.7313, respectively. 67.57% of the wafer maps do not have any failures, which dominate the distribution. Figure 2 illustrates examples of each failure pattern.

Researchers have proposed various approaches to tackle this classification problem. Machine learning methods, especially deep learning methods, have been applied to this challenge. Specifically, this challenge is treated as an image classification problem, which is well studied in the machine learning community.

3.2 Net Delay Prediction

Fast and accurate pre-routing timing prediction is essential for timing-driven placement since repetitive routing and static timing analysis (STA) iterations are expensive and unacceptable. We follow the prior work [5] regarding the net delay prediction problem

index	failure	# samples	percentage (%)
0	none	36730	67.57
1	edge-ring	8554	15.74
2	center	3462	6.369
3	edge-loc	2417	4.447
4	loc	1620	2.980
5	random	609	1.120
6	scratch	500	0.920
7	donut	409	0.752
8	near-full	54	0.099
Total		**54355**	**100**

Table 1: The original training split of WM-811K dataset.

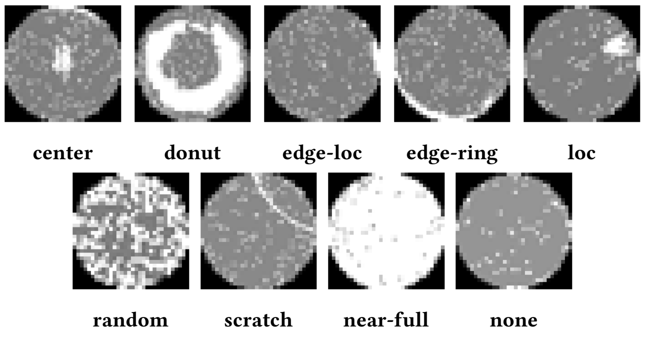
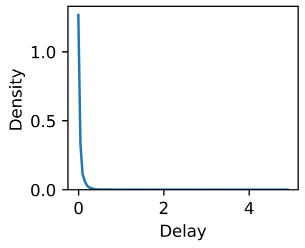
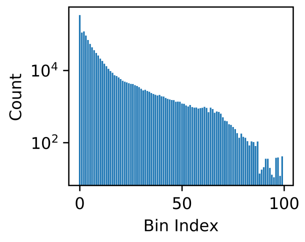


Figure 2: Eight failure types in WM-811K dataset. 'None' stands for wafer without any failures.



(a) Density of net delay.



(b) Histogram of net delay.

Figure 3: The density and histogram with 100 bins of the long-tailed net delay dataset. The vertical axis in Figure (b) is in logarithmic scale. We care more about the nets with large delay, which have small density and thus belong to the tail.

formulation. The objective is to estimate post-routing timing behavior, given the circuit placement result. In this work, we collect net delay timing reports from open-source EDA tool OpenROAD [3] on SkyWater 130nm [2] process on 21 real-world open-source designs from [1].

We slightly modify the problem setting and convert the original regression problem into a classification problem. The nets are classified into 100 equally spaced bins based on their delay logarithm value. Specifically, the class ID is as follows,

$$ID_i = \left\lfloor \frac{d_i - d_{min}}{d_{max} - d_{min}} \times 100 \right\rfloor, \tag{11}$$

where d_i is the logarithm of the delay for net i. The net delay distribution density is shown in Figure 3a, which exhibits a long-tailed distribution. While prior work [5] focuses on reducing the model pessimism with the increased correlation between prediction and labeled delay, we argue that in circuit design, we tend to care more about the worst delay cases or the tail of the distribution where the net delay is large. The training dataset distribution after processing is shown in Figure 3b. The imbalance ratio, the imbalance divergence, and the Gini coefficient are 33241.7, 0.7458, and 0.8397, respectively. 78.12% of the net delays are in the first 10 bins, while the last 50 bins only account for 2.57%.

3.3 Experimental Settings

We investigate the performance of the methods introduced in Section 2 to show how these methods handle long-tailed distribution. We work on the WM-811K and net delay prediction dataset mentioned above to verify the effectiveness of these methods. Our target is to see if and how these methods will impact long-tailed distribution classification, instead of achieving better overall classification accuracy on these benchmarks. Our generated dataset and code implementations are available at this link.

3.3.1 Wafer Defect Classification with CNNs. Following the analysis and settings in [4], we use the original training dataset only by splitting it into 0.8 : 0.2 for training and testing. In particular, there are 43,484 and 10,871 samples in the training and testing datasets. Hence, the training and testing datasets share identical long-tailed label distributions, which is different from the well-studied long-tailed classification benchmarks. The original wafer map is represented as a matrix. We treat it as an image and resize it to 32 × 32 pixels.

We use a mini-batch size of 64 and train the ResNet20 [10] model via 5×10^4 iterations for all the methods. We use the SGD optimizer with a momentum of 0.9 and weight decay of 10^{-4}. The initial learning rate is 0.1 and decays following the cosine annealing learning rate scheduler. For the hyperparameters in the loss functions, we use $\gamma = 1$ and $\gamma = 0.5$ for weighted cross entropy loss in Equation (7). γ is 2 for the focal loss in Equation (8), while the γ is 0.999 for the class-balanced loss in Equation (9). For two-stage training, we first use the standard cross entropy to train for 4×10^4 iterations and then apply the corresponding methods to train the remaining 10^4 iterations.

3.3.2 Net Delay Prediction with GNNs. Analogous to the two types of timing arcs in timing engines, we represent the circuit as a heterogeneous graph consisting of two types of edges: net edges and cell edges. The nodes in the graph denote pins in the circuit. The node and edge features are listed in Table 2. The pin capacitance at the node features corresponds to the four timing corner combinations of the early/late and rise/fall. The delay prediction task is node classification, and the nodes are randomly split into 0.8 : 0.2 for training and testing.

Our GNN model computes on the bi-direction graph consisting of net edges and reversed net edges. There are three net convolution

Type	Name	Dimension
Node Features	is primary I/O pin	1
	is fanin or fanout	1
	distance to boundaries	4
	pin capacitance	4
Edge Features	net distances (hori. and vert.)	2

Table 2: Node and edge features of net delay prediction.

layers in our model. Each layer transforms the node features by graph broadcast and graph reduction. In graph broadcast, information flows from the net driver to the net sinks through the net edges. We concatenate the features of the net driver, the net sinks, and the net edges passed through a fully connected neural network layer to obtain the next layer's net features. In graph reduction, information flows backward from the net sinks to the net driver through the reversed net edges similarly. The model learns statistics from all net sinks through two concatenated reduction channels: sum and max operations. We use the node embedding output of the final layer to predict net delay from inputs to outputs.

4 EXPERIMENTAL RESULTS AND DISCUSSIONS

We show and discuss our experimental results in this section.

Tables 3 and 4 are the confusion matrix and the classification report for the baseline method and one-stage class-balanced sampling method on the wafer defect classification problems. Compared with the baseline, the resampling method achieves a higher precision on the head classes and recall on the tail classes $3 \sim 7$. Namely, the classifier will achieve a high prediction accuracy when given a data sample whose true label belongs to tail classes. As a result, we obtain a higher macro average recall and a smaller macro standard deviation on the recall with the class-balanced sampling method.

We notice that the precision and recall of Class 8 are both 1, meaning that the data points in this tail class can be predicted perfectly. We explain that Class 8 (near-full failure pattern) is significantly different from other classes, and thus it is easy to recognize with only few training samples. We conclude that the performance of one class depends on not only the number of data points in it, but also the distance from other classes.

We also observe that except Class 0, the per-class precision decreases as we misclassify the data points whose true label is 0. For instance, we improve the recall of Class 6 from 0.5714 to 0.8878, while the precision of this class drops from 0.8358 to 0.1302. Namely, in our predictions of tail classes, there will be more misclassified data points. Further, we cannot improve the weighted average recall, i.e., accuracy, with the resampling method if the training and testing datasets follow the same label distribution $p(y)$.

Tables 5 and 6 list the results of all the discussed methods on two problems, respectively. Similar to the analysis above, we find that almost all methods tackling long-tailed categorical distribution can improve the macro average recall significantly. For example, the class-balanced sampling can achieve 0.9464 and 0.9444 macro average recall with one-stage and two-stage training, while the corresponding result in the baseline method is 0.9099 in the wafer map classification problem. The class-balanced sampling improves the macro average recall from 0.3723 to 0.5063. In the net delay problem, the macro average precision and F1 score also increase with these methods. These methods obtain degradation on the weighted average performance metrics under the condition that the training and testing datasets follow the same label distribution. Further, we do not find a consistent improvement when using two-stage training.

Above all, most existing long-tailed methods improve recall of the tail classes at the cost of recall of head classes. Namely, given a data point whose true label belongs to the tail classes, we can predict its label more accurately, which is critically important when handling the wafer failure pattern and net delay. We obtain performance degradation in other metrics, especially the precision. This means there will be more misclassified data samples in our prediction, since more data samples in the head classes will be predicted as the tail classes.

5 CONCLUSION

Long-tailed distribution is a common, critical, yet overlooked issue in EDA. Now that this issue is obtaining more attention in the machine learning community, we can leverage these advances with our insights on real-world EDA problems. Specifically, we can take advantage of the reasons and results of the long-tailed distribution to tackle related problems. In other words, we may integrate our prior knowledge regarding the specific problems. We usually tend to assign more weight to the abnormal and extreme data samples in EDA. In the two cases mentioned in this paper, the importance of data samples should not be proportional to their occurring frequency, since we are aware of the importance of the tail. We leverage model-agnostic methods, specifically reweighting and resampling, and demonstrate their effectiveness on these two examples with convolutional neural networks and graph neural networks, respectively.

Above all, we would like to draw attention to the issue of long-tailed distribution in EDA, especially when we try to solve problems using machine learning algorithms.

ACKNOWLEDGMENTS

This work is supported in part by NSF under grants 1718570 and 2112665, and NVIDIA.

REFERENCES

[1] 2021. OpenCores. https://opencores.org/
[2] 2021. SkyWater. https://github.com/google/skywater-pdk
[3] Tutu Ajayi, Vidya A. Chhabria, Mateus Fogaça, et al. 2019. Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project. In *Proceedings of the 56th Annual Design Automation Conference 2019 (DAC '19)*. Association for Computing Machinery, New York, NY, USA, Article 76, 4 pages. https://doi.org/10.1145/3316781.3326334
[4] Mohamed Baker Alawieh, Duane Boning, and David Z. Pan. 2020. Wafer Map Defect Patterns Classification using Deep Selective Learning. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1109/DAC18072.2020.9218580
[5] Erick Carvajal Barboza, Nishchal Shukla, Yiran Chen, et al. 2019. Machine Learning-Based Pre-Routing Timing Prediction with Reduced Pessimism. In *Proceedings of the 56th Annual Design Automation Conference 2019 (DAC '19)*. Association for Computing Machinery, New York, NY, USA, Article 106, 6 pages. https://doi.org/10.1145/3316781.3317857

True\Pred	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	7373, 6710	13, 40	10, 42	7, 68	7, 35	1, 7	5, 454	0, 0	0, 0
1	8, 2	1707, 1707	0, 0	9, 15	0, 0	0, 0	0, 0	0, 0	0, 0
2	7, 5	1, 1	683, 683	0, 2	2, 1	1, 1	1, 1	0, 1	0, 0
3	19, 6	11, 6	1, 1	415, 431	6, 8	1, 0	1, 2	0, 0	0, 0
4	33, 4	0, 0	1, 3	14, 10	285, 298	0, 2	4, 19	3, 4	0, 0
5	2, 0	1, 1	0, 0	1, 1	1, 1	100, 102	0, 0	0, 0	0, 0
6	38, 9	0, 1	1, 2	1, 1	2, 1	0, 0	56, 84	0, 0	0, 0
7	0, 0	0, 0	0, 0	2, 1	2, 0	1, 1	0, 0	88, 91	0, 0
8	0, 0	0, 0	0, 0	0, 0	0, 0	0, 0	0, 0	0, 0	6, 6

Table 3: The confusion matrix of testing results for the wafer defect classification problem. For each entry, the two items are for the baseline method and the class balanced resampling method.

		class 0	class 1	class 2	class 3	class 4	class 5	class 6	class 7	class 8	macro avg	macro std	weighted avg
precision	baseline	0.9856	**0.9850**	**0.9813**	**0.9243**	**0.9344**	**0.9615**	**0.8358**	**0.9670**	**1.0000**	**0.9528**	**0.0503**	**0.9793**
	cls-bal samp.	**0.9961**	0.9721	0.9343	0.8147	0.8663	0.9027	0.1500	0.9479	**1.0000**	0.8427	0.2668	0.9678
recall	baseline	**0.9942**	**0.9901**	**0.9827**	0.9141	0.8382	0.9524	0.5714	0.9462	**1.0000**	0.9099	0.1368	**0.9799**
	cls-bal samp.	0.9122	**0.9901**	**0.9827**	**0.9493**	**0.8765**	**0.9714**	**0.8571**	**0.9785**	**1.0000**	**0.9464**	**0.0522**	0.9302
F1	baseline	**0.9898**	**0.9876**	**0.9820**	**0.9192**	**0.8837**	**0.9569**	**0.6788**	0.9565	**1.0000**	**0.9283**	**0.1008**	**0.9793**
	cls-bal samp.	0.9523	0.9810	0.9579	0.8769	0.8713	0.9358	0.2553	**0.9630**	**1.0000**	0.8660	0.2330	0.9452

Table 4: The classification report for testing results with the baseline method and the class-balanced sampling method for the wafer defect classification problem.

	macro average						**weighted average**					
	precision		**recall**		**F1**		**precision**		**recall**		**F1**	
	single	**double**	**single**	**double**	**single**	**double**	**single**	**double**	**single**	**double**	**single**	**double**
baseline	0.9528	-	0.9099	-	**0.9283**	-	**0.9793**	-	**0.9799**	-	**0.9793**	-
weighted, $\gamma = 1$	0.8058	0.7869	0.9386	0.9317	0.8388	0.8240	0.9653	0.9631	0.9181	0.9122	0.9371	0.9326
weighted, $\gamma = 0.5$	0.8963	0.8827	0.9324	0.9340	0.9111	0.9044	0.9746	0.9749	0.9720	0.9721	0.9729	0.9731
focal	0.9533	0.7962	0.9027	0.9396	0.9236	0.8335	0.9786	0.9709	0.9792	0.9349	0.9785	0.9494
class balanced	0.9150	**0.9540**	0.9118	0.9038	0.9121	0.9255	0.9769	0.9788	0.9775	0.9795	0.9771	0.9788
balanced softmax	0.8059	0.8414	0.9410	0.9443	0.8391	0.8587	0.9698	0.9713	0.9232	0.9188	0.9423	0.9403
class balanced samp.	0.8427	0.8150	**0.9464**	0.9444	0.8660	0.8459	0.9678	0.9718	0.9302	0.9302	0.9452	0.9471
square root samp.	0.8839	0.8945	0.9327	0.935	0.9045	0.9083	0.9766	0.9764	0.9742	0.9719	0.9750	0.9735
progressively samp.	0.8373	0.8280	0.9435	0.9437	0.8619	0.8539	0.9735	0.9715	0.9398	0.9298	0.9536	0.9469

Table 5: Macro average and weighted average results on the wafer defect classification problem. 'samp.' is short for sampling. Single and double means one-stage and two-stage training. The largest values of each metric are highlighted.

	macro average						**weighted average**					
	precision		**recall**		**F1**		**precision**		**recall**		**F1**	
	single	**double**	**single**	**double**	**single**	**double**	**single**	**double**	**single**	**double**	**single**	**double**
baseline	0.3820	-	0.3723	-	0.3730	-	**0.6654**	-	**0.6703**	-	**0.6673**	-
weighted, $\gamma = 1$	0.3443	0.3455	0.3780	0.3833	0.3573	0.3600	0.5560	0.5600	0.5599	0.5635	0.5566	0.5606
weighted, $\gamma = 0.5$	0.4062	0.3991	0.4200	0.4137	0.4102	0.4041	0.5922	0.5923	0.5971	0.5963	0.5939	0.5937
focal	0.3543	0.3019	0.3373	0.3182	0.3405	0.3032	0.6064	0.5968	0.6618	0.5992	0.6084	0.5974
class balanced	0.3617	0.3689	0.3865	0.3923	0.3707	0.3755	0.5997	0.6002	0.6072	0.6069	0.6028	0.6029
balanced softmax	0.3415	0.3382	0.3810	0.3599	0.3516	0.3435	0.6065	0.6052	0.6074	0.6065	0.6063	0.6050
class balanced samp.	**0.4922**	0.3673	**0.5063**	0.3955	**0.4979**	0.3779	0.5758	0.6031	0.5802	0.6041	0.5767	0.6027
square root samp.	0.4835	0.3771	0.4912	0.3948	0.4863	0.3819	0.5994	0.6107	0.6039	0.6143	0.6011	0.6121
progressively samp.	0.4716	0.3731	0.4859	0.4001	0.4772	0.3837	0.6334	0.6064	0.6333	0.6082	0.6328	0.6066

Table 6: Results on the net prediction problem. The notations are the same as Table 5.

[6] Paula Branco, Luís Torgo, and Rita P Ribeiro. 2016. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)* 49, 2 (2016), 1–50.

[7] Fan Chung, Linyuan Lu, and Van Vu. 2003. Spectra of random graphs with given expected degrees. *Proceedings of the National Academy of Sciences* 100, 11 (2003), 6313–6318. https://doi.org/10.1073/pnas.0937490100 arXiv:https://www.pnas.org/content/100/11/6313.full.pdf

[8] Yin Cui, Menglin Jia, Tsung-Yi Lin, et al. 2019. Class-Balanced Loss Based on Effective Number of Samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[9] Hao Geng, Haoyu Yang, Lu Zhang, et al. 2020. Hotspot Detection via Attention-based Deep Layout Metric Learning. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1–8.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[11] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intell. Data Anal.* 6, 5 (2002), 429–449. http://content.iospress.com/articles/intelligent-data-analysis/ida00103

[12] Justin M Johnson and Taghi M Khoshgoftaar. 2019. Survey on deep learning with class imbalance. *Journal of Big Data* 6, 1 (2019), 1–54.

[13] Bingyi Kang, Saining Xie, Marcus Rohrbach, et al. 2020. Decoupling representation and classifier for long-tailed recognition. In *Eighth International Conference on Learning Representations (ICLR)*.

[14] Joffrey L Leevy, Taghi M Khoshgoftaar, Richard A Bauder, et al. 2018. A survey on addressing high-class imbalance in big data. *Journal of Big Data* 5, 1 (2018), 1–30.

[15] Tsung-Yi Lin, Priya Goyal, Ross Girshick, et al. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*. 2980–2988.

[16] David M. W. Powers. 1998. Applications and Explanations of Zipf's Law. In *New Methods in Language Processing and Computational Natural Language Learning*. https://aclanthology.org/W98-1218

[17] Haoxing Ren, George F. Kokai, Walker J. Turner, et al. 2020. ParaGraph: Layout Parasitics and Device Parameter Prediction using Graph Neural Networks. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1109/DAC18072.2020.9218515

[18] Jiawei Ren, Cunjun Yu, Shunan Sheng, et al. 2020. Balanced Meta-Softmax for Long-Tailed Visual Recognition. In *Proceedings of Neural Information Processing Systems(NeurIPS)*.

[19] Ming-Ju Wu, Jyh-Shing R. Jang, and Jui-Long Chen. 2015. Wafer Map Failure Pattern Recognition and Similarity Ranking for Large-Scale Data Sets. *IEEE Transactions on Semiconductor Manufacturing* 28, 1 (2015), 1–12. https://doi.org/10.1109/TSM.2014.2364237

[20] Zhiyao Xie, Rongjian Liang, Xiaoqing Xu, et al. 2021. Net2: A Graph Attention Network Method Customized for Pre-Placement Net Length Estimation. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference (ASPDAC '21)*. Association for Computing Machinery, New York, NY, USA, 671–677. https://doi.org/10.1145/3394885.3431562

[21] Cunxi Yu and Zhiru Zhang. 2019. Painting on Placement: Forecasting Routing Congestion Using Conditional Generative Adversarial Nets. In *Proceedings of the 56th Annual Design Automation Conference 2019 (DAC '19)*. Association for Computing Machinery, New York, NY, USA, Article 219, 6 pages. https://doi.org/10.1145/3316781.3317876

[22] Yifan Zhang, Bingyi Kang, Bryan Hooi, et al. 2021. Deep Long-Tailed Learning: A Survey. *arXiv preprint arXiv:2110.04596* (2021).