SOM-US: A Novel Under-Sampling Technique for Handling Class Imbalance Problem

Ajay Kumar

***Abstract*—A significant research challenge in data mining and machine learning is class imbalance classification since the majority of real-world datasets are imbalanced. When the dataset is highly unbalanced, the majority of available classification techniques frequently underperform on minority-class cases. This is due to the fact that they disregard the relative distribution of each class in favor of maximizing the overall accuracy. Various techniques based on sampling methods, cost-sensitive learning, and ensemble methods have recently been employed to handle the class imbalance problem. This paper proposes a new clustering-based under-sampling (US) technique, called SOM-US, for handling the class imbalance problem using the self-organized map (SOM). To validate the proposed approach, an experimental study was conducted to improve the capability of a classifier-logistic regression for software defect prediction by applying SOM-US over a NASA software defect dataset. The proposed approach was compared with six existing under-sampling methods on two performance measures. The results demonstrate that the SOM-US significantly improves the prediction capability of logistic regression over other under-sampling techniques for software defect prediction.**

***Index terms*—Class Imbalance, Under-Sampling, Software Defect Prediction.**

I. Introduction

Analysis of classification is a thoroughly considered technique in the domains of data mining and machine learning. Because of its forecasting ability, classification has been used in a wide range of real-world applications, such as fraud detection (credit card), predicting customer churn, product categorization, image classification, medical diagnosis, software defect prediction, etc. By examining the characteristics of a dataset with classes, classification's study can create a class prediction system, also known as the classifier [1]. The classifier is capable of predicting the classes for the new examples with undefined class labels. As an example, the health prediction system can be used by a medical officer to predict whether a patient has a drug allergy or not. A training dataset is one that has data for a certain class, and a classifier needs to be trained on a training dataset in order to be able to predict classes. In brief, the following steps include the process of classification analysis:

1. Collection of samples.
2. Selection of attributes and samples for training.
3. Use training samples to train a class prediction system.
4. To predict the class of incoming samples, use the predicting system.

One of the most important aspects that impact the predictive performance of a classifier is class imbalance [2]. When there are disproportionately more instances of one class than another, the data is said to be unbalanced. Fig. 1 depicts the imbalance spreading of samples in the minority class and majority class. The majority class is represented by red asterisk symbols, whereas blue circles represent the minority class. It is obvious that majority-class areas are much denser than minority-class areas. In this scenario, classifiers frequently produce an influenced learning model that has lower predicted accuracy for minority classes than for majority classes.

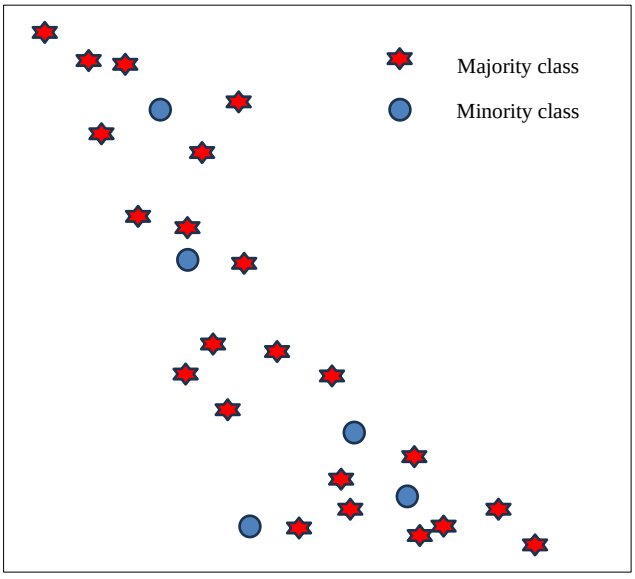

Fig. 1. The class imbalance problem

Researchers have suggested numerous approaches for addressing the problem of class imbalance, since it occurs in several real-life applications, such as medical research, risk management, intrusion prevention, and fraud detection. These approaches for addressing issues of class inequality are broadly categorized into two categories- algorithm level approach and

Manuscript received October 3, 2023; revised December 21, 2024. Date of publication January 30, 2024. Date of current version January 30, 2024. The associate editor prof. Matko Šarić has been coordinating the review of this manuscript and approved it for publication.

Author is with the Department of Information Technology, KIET Group of Institutions, Delhi-NCR, Ghaziabad-201206, India (e-mail: ajaygarg100@gmail.com).

Digital Object Identifier (DOI): 10.24138/jcomss-2023-0133

data level approach [3-4]. In the algorithm-level approach, classifiers are modified to reduce their sensitivity towards the class imbalance during learning over an imbalanced dataset. In the data-level approach, the imbalanced distribution of the dataset is handled by some kind of preprocessing on the original dataset. The data level approach includes two popular techniques- over-sampling and under-sampling. To deal with the unbalanced dataset, the oversampling strategy increases the minority class's samples. The under-sampling strategy, on the other hand, reduces majority class samples. In general, under-sampling approaches outperform the oversampling approaches [5]. As a result of their independence from the underlying classifier and ease of application to any situation, data-level techniques are the most well-liked and often used the data-level approaches are the most popular and the most frequently employed because they are independent of the underlying classifier and may be simply applied to any problem [6].

This study suggests a new cluster-based under-sampling technique (SOM-US) using the SOM for handling class imbalance problem. This study is motivated because of the following three reasons.

- First, due to the ever-increasing volume of data produced in numerous scientific and technical areas, over-sampling procedures are much more expensive in terms of memory and processing requirements for subsequent classification tasks [6].
- Second, using under-sampling approaches has also improved classifier performance and is now the standard approach to dealing with asymmetric distributions [7].
- Third, due to previous related research [8] that has shown superior performance than neighborhood-based advancements, we have chosen to apply a clustering-based strategy for under-sampling the class-imbalanced datasets.

The key contribution of this study is outlined as follows:

- This study proposes a novel under-sampling (US) technique using a self-organized map (SOM) to address the class imbalance problems in imbalanced datasets. SOM-US is the name given to the suggested method.
- This paper presents a novel application of SOM (a popular neural network clustering technique) to reduce the samples of the majority class in an imbalanced dataset.
- For the validity of (SOM-US), a classifier-logistic regression for software defect prediction was trained by applying SOM-US over a NASA software defect dataset.
- The performance of SOM-US is compared with various under-sampling techniques in terms of two performance measures, G-measure and nMCC.
- A statistical test was conducted to further demonstrate that SOM-US significantly outperformed other under-sampling techniques.

From section II to section VI, this study's remaining section is explained. The relevant research is described in Section II, Section III describes the suggested under-sampling method SOM US, and the experimental setup is presented in Section IV. In section V, the results and discussion are given. The paper is concluded in section VI.

II. Related Work

This section summarizes recent studies by a variety of authors on how to deal with the issue of class imbalance.

The authors in [9] propose a novel strategy to handle class imbalance problems in software defect prediction for both within-project and cross-project. In order to generate dynamic training datasets with the balanced dataset, they employ the idea of stratification embedded in the nearest neighbor. In [10], the authors conducted a review study on the application of under-sampling techniques for handling class imbalance problems. They present the category-wise detailed comparison of the various under-sampling techniques, including pure under-sampling, cluster-based under-sampling, and hybrid under-sampling techniques.

The authors present an empirical study in [11] to show the impact of class imbalance on classifier performance. Ten classifiers that are widely used and have been found to be effective were trained for this empirical study. Additionally, in order to optimize the performance of each classifier, thorough hyperparameter tuning was performed for every piece of data.

In [12], authors proposed an innovative ensemble technique for handling class imbalance problems using the under-sampling technique and constraint projection. Each base classifier is built using two steps: First, under-sampling the samples from the minority/majority class set to create a set of pairwise constraints, then using that set to develop a projection matrix. Second, a basic classifier is built in the new feature space using the under-sampled new training dataset. The authors propose a method in [13] for handling class imbalance problems in classifier chains using random under-sampling. The author demonstrates the effectiveness of their proposed method by an experimental study using eighteen multi-label datasets.

In [14], the authors focus on the challenge of zero-shot failure detection in rolling bearings since it represents the most extreme instance of class imbalance. A two-stage zero-shot fault recognition system is suggested as a solution to this issue. First, a new feature-generating network will produce a large number of pseudo-fault features by including an additional sequence in the condition. Second, these artificial pseudo-defect characteristics are used as the classifier to train an improved deep neural network. In order to recognize the unobserved fault samples, a condition index is specifically created to represent various fault classes. Finally, three datasets are used to demonstrate the efficacy of the suggested strategy. The results of the experimental study demonstrate that, even in the absence of fault data during training, the feature generation network can detect typical errors with reasonable accuracy.

In [15], authors proposed a hybrid method for solving the class imbalance problem in android malware detection. In the proposed hybrid approach, the authors first applied K-means clustering for the under-sampling of the dataset to keep relevant

majority class samples. Then, they created minority class samples for data balance using the synthetic minority oversampling technique.

The authors propose a method in [16] for handling the class imbalance problem by applying cluster-based under-sampling in place of random under-sampling while selecting a classifier in the classic hybrid approach redefinition strategy. The performance was evaluated on the performance measures-specificity, sensitivity, and G-measure while training classifiers on three datasets Vehicle1, Vowel, and Page-Blocks.

A hybrid score-based model is proposed in [17] for handling class imbalance problems by integrating oversampling and under-sampling approaches. Based on the significance of the samples in the feature space, the authors use the sharing technique in both rounds (oversampling and under-sampling) to choose more appropriate samples. Synthetic samples for the smaller class are created during the oversampling stage by interpolating across sparser data. Denser samples from the bigger class are then chosen and eliminated during the subsequent under-sampling stage. Oversampling and under-sampling are performed based on probabilities using the binary tournament selection operator in both phases.

A comparative study is conducted in [18] to compare various class imbalance techniques for breast cancer classification using deep learning. In order to address the class imbalance, they thoroughly assess a number of techniques, including oversampling, under-sampling, and class weighting.

In [19], the author proposed a neighborhood under-sampling approach called N-US for handling class imbalance problems for software defect prediction. N-US was compared with three common existing under-sampling methods in order to determine its applicability.

In [20], authors proposed a two-phase approach in the categories of ensemble-based approaches and under-sampling data-level approaches. In their study, it is anticipated that employing the suggested strategy will lower the likelihood of modifying the data distribution while maintaining the feature space's overall data pattern.

Following are the observations after thoroughly reviewing the related work for handling the class imbalance problem.

- In previous studies, various researchers have proposed a variety of models to tackle the class imbalance problem in a dataset to improve the performance of a classifier to predict the class of a new incoming sample.
- Most of the researchers have focused on the under-sampling approach for handling the class imbalance problem to improve the predictive capability of a classifier.
- However, no researcher has utilized the concept of neural network technique (SOM) for under-sampling the majority samples of an imbalanced dataset to handle the class imbalance problem.

This paper proposes a novel under-sampling technique using the neural network technique- SOM to deal with class imbalance problems in imbalanced datasets. The proposed approach is referred to as SOM-US. To the best of the authors' knowledge, this is the first attempt to handle the class imbalance problem using the neural network technique- SOM. The following section presents a detailed description of the proposed under-sampling approach.

III. Proposed Method

This paper proposes a novel under-sampling method, SOM-US, using SOM to handle the class imbalance problem. The proposed under-sampling approach is based on clustering majority class samples using an artificial neural network technique-SOM. SOM [21] is a two-layered unsupervised neural network based on the concept of competitive learning. Each layer has a specific function, with the first layer serving as input and the second as output. The output layer is also called a feature map. A one- or two-dimensional lattice of neurons typically makes up the output layer of a SOM. Mapping each training vector into a feature space is the key component of the SOM neural network. SOM seeks to visualize the similarity between data vectors inside a low-dimensional feature space.

In the context of this study, SOM is applied for clustering on the majority class samples of the imbalanced dataset. An overview of the proposed under-sampling approach SOM-US is given in Fig. 2, followed by a detailed stepwise description.

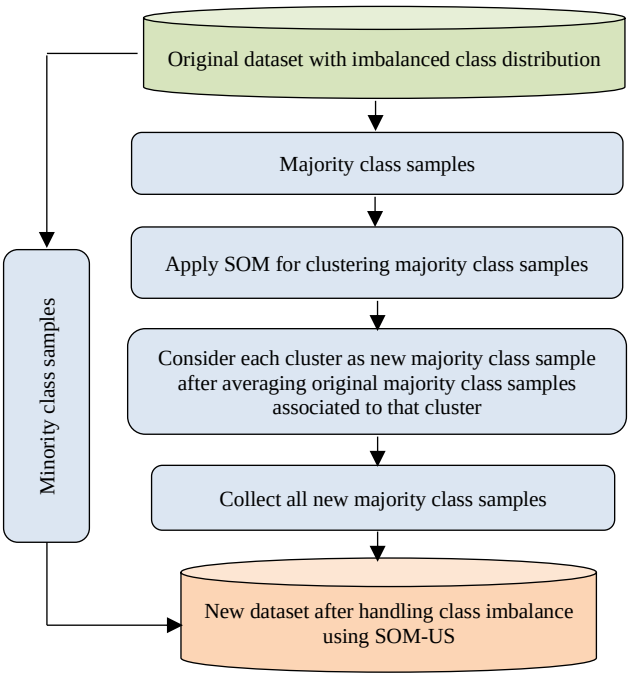

Fig. 2. An overview of the proposed under-sampling approach (SOM-US)

Step 1: Consider the imbalanced dataset, let's say $D_{n\times p}$ as shown in "(1)". In this dataset 'n' denotes the total number of samples (S) and with 'p' denotes the total number of attributes (A).

$$D_{n\times p} = \begin{matrix} S/A \\ s_1 \\ s_2 \\ . \\ s_n \end{matrix} \begin{matrix} a_1 & a_2 & . & a_p \\ d_{11} & d_{12} & . & d_{1p} \\ d_{21} & d_{23} & . & d_{2p} \\ . & . & . & . \\ d_{n1} & d_{n2} & . & d_{np} \end{matrix} \quad (1)$$

Step 2: Separate the imbalanced dataset $D_{n\times p}$ into two parts, majority class samples ($X_{q\times p}$) and minority class samples ($Y_{r\times p}$), as shown in "(2)" and "(3)", respectively.

$$X_{q\times p} = \begin{matrix} S/A & a_1 & a_2 & . & a_p \\ s_1 & x_{11} & x_{12} & . & x_{1p} \\ s_2 & x_{21} & x_{23} & . & x_{2p} \\ . & . & . & . & . \\ s_q & x_{q1} & x_{q2} & . & x_{qp} \end{matrix} \quad (2)$$

Here q represents the total number of majority samples.

$$Y_{r\times p} = \begin{matrix} S/A & a_1 & a_2 & . & a_p \\ s_1 & y_{11} & y_{12} & . & y_{1p} \\ s_2 & y_{21} & y_{23} & . & y_{2p} \\ . & . & . & . & . \\ s_r & y_{r1} & y_{r2} & . & y_{rp} \end{matrix} \quad (3)$$

Here r represents the total number of minority samples. It may be noted that (q + r) is equal to n, and the value of q is much higher than that of r.

Step 3: Apply SOM on $X_{q\times p}$

Apply SOM, a neural network-based clustering technique, on majority class samples ($X_{q\times p}$) to partition the majority class samples into k number of clusters. Convert each cluster into a new majority class sample by averaging all the attribute values of the original majority class samples corresponding to each cluster. In this way, k number of new majority class samples are generated. New majority class samples ($Z_{k\times p}$) are shown in "(4)".

$$Z_{k\times p} = \begin{matrix} S/A & a_1 & a_2 & . & a_p \\ s_1 & z_{11} & z_{12} & . & z_{1p} \\ s_2 & z_{21} & z_{23} & . & z_{2p} \\ . & . & . & . & . \\ s_k & z_{k1} & z_{k2} & . & z_{kp} \end{matrix} \quad (4)$$

Step 4: Combine new majority class samples ($Z_{k\times p}$) obtained from "(4)" (see step 3) with original minority class samples ($Y_{r\times p}$) from "(3)" (see step 2) to obtain a new under-sampled dataset ($U_{m\times p}$) with a total of m samples as shown in "(5)". Here it may be noted that m is the summation of k (new samples obtained by applying SOM on majority class samples in the original dataset) and r (minority class samples in the original dataset).

$$U_{m\times p} = \begin{matrix} S/A & a_1 & a_2 & . & a_p \\ s_1 & u_{11} & u_{12} & . & u_{1p} \\ s_2 & u_{21} & u_{23} & . & u_{2p} \\ . & . & . & . & . \\ s_m & u_{m1} & u_{m2} & . & u_{mp} \end{matrix} \quad (5)$$

IV. Experimental Setup

To check the validity of under-sampling approach SOM-US, an experimental study was carried out for predicting software defects using logistic regression as the classifier over a NASA software defect dataset. The performance of SOM-US is compared with various existing under-sampling techniques in terms of two performance measures, G-measure and nMCC.

A. Dataset

For the validation of the proposed method, this paper uses the CM1 dataset [22]. CM1 is the publicly available NASA software defect dataset written in C language. The total number of samples in CM1 is 498, of which 49 (9.8%) samples correspond to defective class samples, and 449 (90.2 %) samples correspond to non-defective class samples. Here it can be observed that the dataset CM1 is highly imbalanced. The total number of attributes in CM1 is twenty-two. In these twenty-two attributes, five attributes represent the different line of code measure, four attributes represent the base Halstead measure, three attributes represent the McCabe metrics, eight attributes represent the derived Halstead measures, one attribute represents the branch count, and the last attribute is the defects (class attribute).

B. Performance Measures

This study uses two accuracy measures, G-measure and nMCC, to measure the predictive performance of the classifier over a software defect dataset after handling class imbalance problems using the proposed under-sampling approach SOM-US. G-measure and nMCC are reported as stable performance measures [23] and are widely used in previous studies [23-25] for software defect prediction. These two measures are explained in terms of true positive (*TP*), false positive (*FP*), true negative (*TN*), and false negative (*FN*) as follows:

- nMCC (normalized Matthews Correlation Coefficient) [23] can be obtained by using the following equations.

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)}} \quad (6)$$

$$nMCC = \frac{1+MCC}{2} \quad (7)$$

- G-measure: It can be calculated by using the following equations [23].

$$TPR = \frac{TP}{TP+FN} \quad (8)$$

$$TNR = \frac{TN}{TN+FP} \quad (9)$$

$$G-measure = \frac{2*TPR*TNR}{(TPR+TNR)} \quad (10)$$

C. Experimental Design

A stepwise description of the experimental design for validating the proposed under-sampling approach SOM-US is given below followed by the pictorial representation as shown in Fig. 3.

Step 1: Take the software defect dataset CM1 as described in section IV. *A*.

Step 2: Apply the proposed under-sampling approach SOM-US (as described in detail in section III) on the software defect dataset CM1.

- Step 2.a. Use "(2)" and "(3)" to separate the original dataset into two parts-dataset with majority class samples (non-defective class) and minority class samples (defective class), respectively.
- Step 2.b. In this step, SOM, a neural network-based clustering technique is applied to partition the majority class samples (non-defective class) into clusters. SOM was implemented using MATLAB 2022a version 9.12.
- Step 2.c: Convert each cluster into a new majority class sample by averaging all the original majority class samples associated with that particular cluster.
- Step 2.d: Combine the new majority class samples with the original minority class samples to obtain the new dataset without the class imbalance problem.

Step 3: Apply other existing under-sampling techniques on the software defect dataset CM1.

For comparing the performance of the proposed under-sampling approach (SOM-US), six other under-sampling techniques were also applied to the software defect dataset CM 1. These six under-sampling techniques [26] are- Simple K-Mean (SKM)-US, Tomek Links (TL)-US, Edited Nearest Neighbours (ENN)-US, Condensed Nearest Neighbour (CNN)-US, One-Sided Selection (OSS)-US, and Random (R)-US. All six under-sampling techniques were implemented in Jupyter Notebook [27].

Step 4: Train the classifier- logistic regression [28] for software defect prediction on eight datasets. These eight datasets are:

- SOM-US: Dataset obtained by applying the proposed under-sampling method.
- SKM-US: Dataset obtained by applying the Simple K-Mean under-sampling.
- TL-US: Dataset obtained by applying the Tomek Links under-sampling.
- ENN-US: Dataset obtained by applying the Edited Nearest Neighbours under-sampling.
- CNN-US: Dataset obtained by applying the Condensed Nearest Neighbour under-sampling.
- OSS-US: Dataset obtained by applying the One-Sided Selection under-sampling.
- R-US: Dataset obtained by applying the Random under-sampling.
- W-US: Dataset without under-sampling.

Step 5: Compare the software defect prediction results for each dataset in terms of two performance measures, G-measure, and nMCC, as described in section IV. *B.*

V. Results and Discussion

This section is further divided into two parts. The first part presents the software defect prediction results for all eight datasets as described in section IV.*C*. The second part presents the statistical test that was performed to prove that the proposed SOM-US methodology significantly outperforms existing under-sampling techniques for improving the predictive performance of a classifier while being trained on the software defect dataset.

A. Software Defect Prediction Results

Software defect prediction results in terms of nMCC and G-measure for eight datasets (seven under-sampling datasets + one dataset without under-sampling) are listed in Table I. Table I demonstrates that, in terms of both the performance metrics G-measure and nMCC, the suggested SOM-US improved the capabilities of the software defect prediction model in comparison to alternative under-sampling techniques.

TABLE I
Software Defect Prediction Results in Terms of G-measure and nMCC

Dataset	G-measure	nMCC
SOM-US (Proposed)	***0.7036***	***0.7093***
Simple K- Mean (SKM)-US [26]	0.6327	0.6327
Tomek Links (TL)-US [29]	0.2490	0.5854
Condensed Nearest Neighbour (CNN)-US [26]	0.4722	0.6045
Edited Nearest Neighbours (ENN)-US [30]	0.3363	0.6076
One-Sided Selection (OSS)-US [30]	0.2489	0.5805
Random (R)-US [26]	0.6393	0.6541
Without under-sampling (W-US)	0.2174	0.5660

The following observations are inferred from the Table I:

- It can be observed that the proposed under-sampling approach SOM-US outperformed the other existing under-sampling approaches for the software defect prediction over the performance metric-G-measure. Two under-sampling approaches R-US and SKM-US also performed well. However, there is a significant difference between the performance of SOM-US and other under-sampling approaches.
- It can also be observed that the proposed under-sampling approach SOM-US performs better than the other existing under-sampling approaches for the software defect prediction over the performance measure nMCC.

To prove that the SOM-US significantly outperforms existing under-sampling techniques for improving the predictive performance of a classifier while being trained on the software defect dataset, we have conducted a statistical test as described in the following subsection (section V.B).

B. Statistical Test

The Bayesian sign test was carried out to show that the suggested SOM-US methodology significantly outperforms existing under-sampling techniques for improving the predictive performance of a classifier while being trained on the software defect dataset. The Bayesian sign test was proposed by Benavoli [31]. The results of the Bayesian Test for the proposed SOM-US approach are listed in Table II to Table III.

From Table II to Table III, it is evident that the performance of the proposed SOM-US strategy against all other under-sampling approaches is identifiable since the posterior

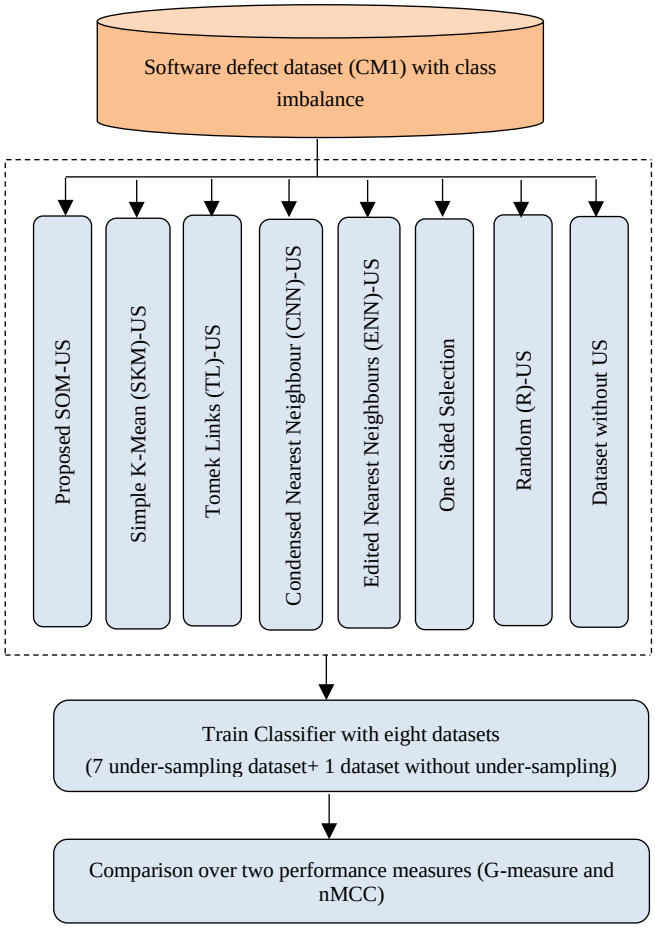

Fig. 3. Experimental design

probability p(rope) is almost equal to 0 (0.05). Furthermore, compared to alternative under-sampling techniques, the posterior probability p(left) of SOM-US for both measures (G-measure and nMCC) is significantly higher (> 95%). Hence it can be concluded that the proposed under-sampling approach SOM-US significantly outperformed other under-sampling methods.

TABLE II
BAYESIAN TEST RESULTS FOR G-MEASURE

SOM-US against	G-measure		
	p (left)	*p (rope)*	*p (right)*
Simple K- Mean (SKM)-US	0.9686	0.0314	0
Tomek Links (TL)-US	0.9688	0.0312	0
CNN-US	0.9682	0.0317	0
ENN-US	0.9676	0.0324	0
One-Sided Selection (OSS)-US	0.9687	0.0313	0
Random (R)-US	0.9686	0.0314	0
W-US	0.9688	0.0312	0

V. CONCLUSION

This study proposed a novel under-sampling approach SOM-US using SOM for handling class imbalance problems. The proposed approach is based on clustering majority samples using the SOM, a neural network technique. For the validation of the proposed approach, an experimental study was conducted for software defect prediction on a NASA software defect dataset-CM1. Firstly, SOM-US was applied to dataset CM1 to get the under-sampled dataset. Next, a classifier (in this study, logistic regression) was trained on the under-sampled dataset generated by the proposed SOM-US approach. Next, the performance of SOM-US was compared with six other under-sampling approaches considering two performance measures-G-measure and nMCC.

TABLE III
BAYESIAN TEST RESULTS FOR NMCC

SOM-US against	nMCC		
	p (left)	*p (rope)*	*p (right)*
Simple K- Mean (SKM)-US	0.9681	0.0318	0
Tomek Links (TL)-US	0.9692	0.0308	0
CNN-US	0.9689	0.0311	0
ENN-US	0.9696	0.0304	0
One-Sided Selection (OSS)-US	0.9688	0.0312	0
Random (R)-US	0.9678	0.0322	0
W-US	0.9694	0.0306	0

The experimental results demonstrate that the SOM-US greatly outperforms other under-sampling strategies in terms of G-measure as well as nMCC for the software defect prediction model. In order to demonstrate that SOM-US greatly outperformed alternative under-sampling techniques, a statistical test known as the Bayesian sign test was carried out.

In a dataset with an unequal class distribution, the class imbalance issue can be solved using the suggested under-sampling method. The proposed method SOM-US is utilized in this work to predict software defects, but it can be applied to various classification issues in many application fields of engineering, science, management, etc. to address issues with class imbalance.

This study used only one dataset for the experimental study. However, the proposed approach may be extended over a large number of datasets as the future work. Moreover, the proposed approach may be applied to handling the class imbalance problem in other application areas such as in finance-related applications (for example credit card fraud detection), medical science (for example in predicting various kinds of diseases), etc.

REFERENCES

[1] D. Dablain, B. Krawczyk, and N. V. Chawla, "DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data," IEEE Transactions on Neural Networks and Learning Systems, 2022.

[2] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert System with Applications* vol. 73, pp. 220–239, 2017.

[3] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Tackling class overlap and imbalance problems in software defect prediction," *Software Quality Journal vol.* 26, no. 1, pp. 97–125, 2018.

[4] E. A. Felix, and S. P. Lee, "Systematic literature review of preprocessing techniques for imbalanced data," *IET Software vol.* 13, no. 6, pp. 479–96, 2019.

[5] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," Expert Systems with Applications, vol. 36, no. 3, pp. 5718–5727, 2009., https://doi.org/10.1016/j.eswa.2008.06.108

[6] G. Ponce, J. S. Sánchez, R. M. Valdovinos, and J. R. Marcial-Romero, "DBIG-US: A two-stage under-sampling algorithm to face the class

imbalance problem," Expert Systems with Applications, vol. 168, no. 114301, p. 114301, 2021. https://doi.org/10.1016/j.eswa.2020.114301.

[7] A. D. Pozzolo, O. Caelen, and G. Bontempi, "When is undersampling effective in unbalanced classification tasks?," in Machine Learning and Knowledge Discovery in Databases, Cham: Springer International Publishing, 2015, pp. 200–215. https://doi.org/10.1007/978-3-319-23528-813

[8] C.-F. Tsai, W.-C. Lin, Y.-H. Hu, and G.-T. Yao, "Under-sampling class imbalanced datasets by combining clustering analysis and instance selection," Information Sciences, vol. 477, pp. 47–54, 2019. https://doi.org/10.1016/j.ins.2018.10.029

[9] L. Gong, S. Jiang, L. Bo, L. Jiang, and J. Qian, "A novel class-imbalance learning approach for both within-project and cross-project defect prediction," IEEE Transactions on Reliability, vol. 69, no. 1, pp. 40–54, 2020. https://doi.org/10.1109/TR.2019.2895462

[10] D. Devi, S. K. Biswas, B. Purkayastha, "A review on solution to class imbalance problem: Undersampling approaches," In: International Conference on Computational Performance Evaluation (ComPE), IEEE, 2020.

[11] W. Zheng and M. Jin, "The effects of class imbalance and training data size on classifier learning: An empirical study," SN Computer Science, vol. 1, no. 2, 2020. https://doi.org/10.1007/s42979-020-0074-0

[12] H. Guo, J. Zhou, and C. A. Wu, "Ensemble learning via constraint projection and undersampling technique for class-imbalance problem," *Soft Computing* vol. 24, no. 7, pp. 4711–27, 2020.

[13] B. Liu, and G. Tsoumakas, "Dealing with class imbalance in classifier chains via random undersampling," *Knowledge- Based Systems vol.* 192, no. 105292, p. 105292, 2020.

[14] T. Pan, J. Chen, J. Xie, Z. Zhou, and S. He "Deep feature generating network: A new method for intelligent fault detection of mechanical systems under class imbalance," *IEEE Transactions on Industrial Informatics* vol. 17, no. 9, pp. 6282–93, 2021.

[15] J. Guan, X. Jiang, and B. Mao, "A method for class-Imbalance Learning in Android malware detection," Electronics (Basel), vol. 10, no. 24, p. 3124, 2021. https://doi.org/10.3390/electronics10243124

[16] H. Hartono, E. Ongko, and D. Abdullah, "Hybrid approach redefinition with cluster-based instance selection in handling class imbalance problem," International Journal of Advances in Intelligent Informatics, vol. 7, no. 3, p. 345, 2021. https://doi.org/10.26555/ijain.v7i3.515

[17] B. Mirzaei, F. Rahmati, and H. Nezamabadi-pour, "A score-based preprocessing technique for class imbalance problems," Pattern Analysis and Applications, vol. 25, no. 4, pp. 913–931, 2022. https://doi.org/10.1007/s10044-022-01084-1

[18] R. Walsh and M. Tardy, "A comparison of techniques for class imbalance in deep learning classification of breast cancer," Diagnostics (Basel), vol. 13, no. 1, p. 67, 2022. https://doi.org/10.3390/diagnostics13010067

[19] S. Goyal, "Handling class-imbalance with KNN (neighbourhood) under-sampling for software defect prediction," Artificial Intelligence Review, vol. 55, no. 3, pp. 2023–2064, 2022. https://doi.org/10.1007/s10462-021-10044-w

[20] F. Hooshmand, and S. A. MirHassani, "A novel two-phase clustering-based under-sampling method for imbalanced classification problems," Expert Systems with Applications, vol. 213, no. 119003, p. 119003, 2023. https://doi.org/10.1016/j.eswa.2022.119003

[21] T. Kohonen, "Essentials of the self-organizing map," Neural Networks, vol. 37, pp. 52–65, 2013. http://dx.doi.org/10.1016/j.neunet.2012.09.018

[22] J. S. Shirabad and T. J. Menzies, The PROMISE repository of software engineering databases. School of Information Technology and Engineering, University of Ottawa, Canada, 2005. [Online]. Available: http://promise.site.uottawa.ca/SERepository

[23] K. Kaur and A. Kumar, "MCDM-EFS: A novel ensemble feature selection method for software defect prediction using multi-criteria decision making," Intelligent Decision Technologies, vol. 17, no. 4, pp. 1283-1296, 2023. https://doi.org/10.3233/IDT-230251

[24] N. A. Bhat and S. U. Farooq, "An improved method for training data selection for cross-project defect prediction," Arabian Journal for Science & Engineering, vol. 47, no. 2, pp. 1939–1954, 2022.

[25] M. Nevendra and P. Singh, "A survey of software defect prediction based on deep learning," Archives Computational Methods in Engineering, vol. 29, no. 7, pp. 5723–5748, 2022.

[26] S. Loov, "Comparison of undersampling methods for prediction of ¨ casting defects based on process parameters," Master's thesis, University of Skovde, 2021.

[27] Kluyver, T. et al., 2016. Jupyter Notebooks – a publishing format for reproducible computational workflows. In F. Loizides & B. Schmidt, eds. Positioning and Power in Academic Publishing: Players, Agents and Agendas. pp. 87–90.

[28] L. Gong, S. Jiang, and L. Jiang, "Tackling class imbalance problem in software defect prediction through cluster-based over-sampling with filtering," IEEE Access, vol. 7, pp. 145725–145737, 2019. https://doi.org/10.1109/ACCESS.2019.2945858

[29] M. Alamri and M. Ykhlef, "Survey of Credit Card Anomaly and Fraud Detection Using Sampling Techniques," Electronics, vol. 11, no. 23, pp. 4003, 2022.

[30] P. Kumar, R. Bhatnagar, K. Gaur, and A. Bhatnagar, "Classification of imbalanced data:Review of methods and applications," IOP Conference Series: Materials Science & Engineering, vol. 1099, no. 1, p. 012077, 2021.

[31] Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis", The Journal of Machine Learning Research, vol. 18 no. 1, pp.2653-2688, 2017. http://jmlr.org/papers/v18/16-305.html

Ajay Kumar is assistant professor with Department of Information Technology, KIET Group of Institutions, Delhi-NCR, Ghaziabad, India. He completed his Ph.D. in Computer Science and Engineering (CSE) from the University School of Information, Communication & Technology, Guru Gobind Singh Indraprastha University (GGSIPU), New Delhi, India. He received his master degree in CSE from National Institute of Technical Teachers Training and Research, Chandigarh. He has done his Bachelor of Engineering degree in CSE from Dr. B. R. A University Agra, Uttar Pradesh, India. His research fields include software engineering, multi-criteria decision-making, soft computing, artificial intelligence, and machine learning.