Artificial Intelligence Review (2025) 58:62
https://doi.org/10.1007/s10462-024-10971-4

Graph reduction techniques for instance selection: comparative and empirical study

Zahiriddin Rustamov[1] · Nazar Zaki[1] · Jaloliddin Rustamov[1] · Ayham Zaitouny[2] · Rafat Damseh[1]

Accepted: 24 September 2024 / Published online: 20 December 2024

Abstract

The surge in data generation has prompted a shift to big data, challenging the notion that "more data equals better performance" due to processing and time constraints. In this evolving artificial intelligence and machine learning landscape, instance selection (IS) has become crucial for data reduction without compromising model quality. Traditional IS methods, though efficient, often struggle with large, complex datasets in data mining. This study evaluates graph reduction techniques, grounded in graph theory, as a novel approach for instance selection. The objective is to leverage the inherent structures of data represented as graphs to enhance the effectiveness of instance selection. We evaluated 35 graph reduction techniques across 29 classification datasets. These techniques were assessed based on various metrics, including accuracy, F1 score, reduction rate, and computational times. Graph reduction methods showed significant potential in maintaining data integrity while achieving substantial reductions. Top techniques achieved up to 99% reduction while maintaining or improving accuracy. For instance, the Multilevel sampling achieved an accuracy effectiveness score of 0.8555 with 99.16% reduction on large datasets, while Leiden sampling showed high effectiveness on smaller datasets (0.8034 accuracy, 97.87% reduction). Computational efficiency varied widely, with reduction times ranging from milliseconds to minutes. This research advances the theory of graph-based instance selection and offers practical application guidelines. Our findings indicate graph reduction methods effectively preserve data quality and boost processing efficiency in large, complex datasets, with some techniques achieving up to 160-fold speedups in model training at high reduction rates.

Keywords Graph reduction · Instance selection · Data reduction · Machine learning · Big data · Data mining

Abbreviations

AC	Affinity clustering
AUC	Area under the curve
BFS	Breadth-first search
BRW	Biased random walk sampling

Extended author information available on the last page of the article

BRWE	Biased random walk edge sampling
BWT	Biased walktrap sampling
CBE	Community bridge edge sampling
CCIS	Class conditional instance selection
CDIS	Central density-based IS
CIS	Curious IS
CM	Community-based sampling
CNN	Condensed nearest neighbour
ConvNets	Deep convolutional neural networks
CSP	Community structure preserving sparsification
DG	Degree-based sampling
DM	Demon sampling
EGDIS	Enhanced global density-based IS
ENN	Edited nearest neighbor
FG	Fastgreedy sampling
FF	Forest fire sampling
FFE	Forest fire edge sampling
FN	False negatives
FP	False positives
FR	Frontier sampling
GLVQ	Generalized learning vector quantization
HMNE	Hit-miss network editing
HMNEI	Hit-miss network interactive editing
IB	Instance-based
ICF	Iterative case filtering
IM	Infomap sampling
IR	Instance reduction
IS	Instance selection
KC	k-core sparsification
KNN	k-nearest neighbors
LD	Leiden sampling
LDIS	Local density-based IS
LE	Leading eigenvector sampling
LP	Label propagation sampling
LR	Logistic regression
LSBo	Local set border selector
LSSm	Local set-based smoother
LV	Louvain coarsening
LVQ	Learning vector quantization
MHRW	Metropolis hastings random walk sampling
ML	Multilevel sampling
MSS	Modified selective subset selection
N2V	Node2Vec sampling
NB	Naive bayes
NBRW	Non-backtracking random walk sampling
NN	Nearest neighbours

PE	Preferential edge sampling
PR	PageRank-based sampling
PSDSP	Prototype selection based on dense spatial partitions
RE	Random edge sampling
RENN	Repeated edited nearest neighbor
RF	Random forest
RMHC	Random mutation hill climbing
RN	Random node sampling
RNGE	Relative neighbor graph editing
ROC	Receiver operating characteristic
RWE	Random walk edge sampling
SB	Snowball sampling
SC	Spectral clustering coarsening
SG	Spinglass sampling
SRW	Simple random walk sampling
TG	Triangle based sampling
TN	True negatives
TP	True positives
VC	Vertex collapsing
XGB	Extreme gradient boosting
XLDIS	EXtended local density-based IS

1 Introduction

In recent years, there has been an undeniable spurt in big data, which has transformed the landscape of every field, coining data as the new oil Taffel (2023). This shift of significant importance to data has caused organizations and institutions around the globe to generate and accumulate vast amounts of data in hopes of harnessing its potential to drive innovation and gain a competitive edge Mohr and Hürtgen (2018). Data collection has become ubiquitous thanks to advancements in technology and the increasing digitization of every aspect of our lives Sestino et al. (2020). Hence, the challenge is no longer the scarcity of data but rather the effective processing and leveraging of this abundant resource.

Machine learning emerges as the avenue for parsing through these extensive datasets, enabling the extraction of valuable insights and patterns that were previously unattainable as easily. However, the blessing of an abundance of data is also its curse Malekipirbazari et al. (2021). Processing large datasets is inherently costly and time-consuming, which can sometimes challenge the notion that "more data is better". This is often due to the presence of low-quality data, whether it's caused by outliers or noise. This realization prompts the shift from valuing the quality of data over quantity, exemplified by the advent and focus on tinyML Warden and Situnayake (2020). TinyML represents the shift towards optimizing machine learning algorithms to run effectively on limited resources, highlighting the need for a more representative selection of data, what we might term "small data."

1.1 Main challenges

In this context, instance selection (IS) or instance reduction (IR) makes it possible to identify and select the most relevant and representative samples from a dataset. The overall aim is to reduce the volume (i.e., instances) of data that machine learning algorithms need to process without significantly compromising the performance of the models Olvera-López et al. (2010). IS can help mitigate the impact of noisy, redundant, and irrelevant data on model performance, leading to faster training times, improved generalization, and reduced computational resources. However, despite its importance, existing methods of IS often rely heavily on nearest neighbours algorithms, which, while effective in producing the desired results, suffer from computational inefficiencies and sub-optimal selection criteria in handling large and complex datasets, potentially overlooking important but less obvious instances Garcia et al. (2012).

1.2 Research motivation

In response to these challenges, there has been a growing interest in exploring alternative approaches to tackle the IS problem from a different lens. One prominent approach involves representing the data as a graph, where data points are nodes (i.e., vertices) connected by edges that reflect their interconnected relationships. This graph-based representation paves the way to applying graph reduction techniques, which focus on simplifying the graph structure without altering the fundamental properties.

The graph representation captures the interconnected relationships between data points, providing insights into the underlying patterns and dependencies within the dataset that may not be explicitly represented in the original data format. This approach aligns with the broader field of graph mining, which has shown significant potential in various domains such as social network analysis, bioinformatics, and network security Rehman et al. (2012). Graph reduction techniques can leverage this graph structure to identify and retain the most informative nodes and edges, potentially leading to a more effective IS process compared to working with tabulated data. Graph reduction holds the potential to not only enhance computational efficiency by reducing the number of nodes but also by preserving critical relationships that are essential for maintaining the integrity of the dataset Zaki et al. (2024).

Historically, graph reduction techniques have been utilized in various domains, including social networks, computational biology, and compiler optimization. However, their application in the context of IS for machine learning is relatively an unexplored territory. This raises a compelling question: could graph reduction present a more effective alternative for IS, leveraging its strengths to overcome the limitations of existing methods?

1.3 Main contributions

This study is motivated by the hypothesis that graph reduction techniques could represent a viable and potentially more effective solution for the IS problem. To date, there has been limited exploration of how existing graph reduction methods perform when applied to IS, leaving a gap in our understanding and application of these techniques. Therefore, this study aims to bridge this gap by conducting a comprehensive review and evaluation of existing graph reduction techniques within the context of IS. By applying these techniques to diverse datasets, this study seeks to assess their performance and potential as a viable alternative to traditional IS methods. The main objectives of this study are:

- To review, describe, and implement common graph reduction techniques for IS.
- To conduct a comprehensive comparison of graph reduction techniques for IS in terms of effectiveness, performance, and timing. This comparison will help us determine which techniques have potential for IS.
- To illustrate the application of graph reduction techniques through case studies, enabling a visual understanding of how each technique operates in practice.The structure of the manuscript is organized as follows: Sect. 2 discusses related works. Section 3 details the graph reduction techniques implemented in this study. Section 4 outlines the experimental design. Experimental results are presented and discussed in Sects. 5 and 6, respectively. The study concludes with Sect. 7.

2 Related works

Instance selection (IS) is a process that reduces dataset size by selecting a subset of the most representative or informative instances for the training of machine learning models. This practice is known by various terms, including instance reduction Yang et al. (2019), prototype selection Garcia et al. (2012), and data reduction Cano et al. (2003). While these terms emphasize slightly different aspects, they share the common goal of enhancing model efficiency and effectiveness by minimizing redundancy without losing critical information.

Many IS methods leverage nearest neighbours (NNs) to reduce the number of instances. However, NNs can struggle with complex relationships, non-linear distributions, and intricate class boundaries. They are also sensitive to noise and outliers, which can skew distance calculations and lead to poor instance selection D Randall and Tony R (2000), Garcia et al. (2012), Ozturk Kiyak et al. (2023). Graph reduction offers an alternative approach to IS, addressing these limitations.

Several established strategies fall into graph reduction, such as graph sparsification, which involves thinning out the edges of a graph to reduce complexity while preserving key structural characteristics, such as connectivity and shortest paths, and graph coarsening, which merges closely connected vertices, simplifying the graph into a coarser representation that maintains the overall topology Eppstein et al. (1997), Spielman and Teng (2011), Chen et al. (2022). Graph reduction is defined as the process of simplifying a graph $G = (V, E)$ by selectively reducing its vertices V and edges E to form a reduced graph $G' = (V', E')$, where $V' \subseteq V$ and $E' \subseteq E$. The primary objective of graph reduction is to minimize the size of G' by reducing the counts of $|V'|$ and $|E'|$, while maintaining or approximating essential properties of the original graph, such as connectivity or distance metrics. This can be expressed as an optimization problem where the goal is to:

$$\min_{V' \subseteq V, E' \subseteq E} (|V'| + |E'|)$$

subject to:

$$f(G') \geq f(G) - \varepsilon$$

here f represents a function quantifying the relevant properties of the graph, and ε is a tolerance level that allows for a slight deviation in these properties due to reduction.

Henceforth, we will discuss related papers that perform comparative studies of IS methods. Numerous studies have reviewed graph reduction techniques; however, most of these studies have not evaluated graph reduction techniques in relation to the specific problem of IS-namely, identifying the most representative instances. Instead, they typically compare these techniques theoretically, analyze them based on their impact on graph properties or assess their performance in applications like node classification using graph models.

Yang et al. (2016) and Wagenseller et al. (2018) examine the performance of community detection algorithms with different focal points. The former evaluates eight state-of-the-art algorithms using the Lancichinetti-Fortunato-Radicchi benchmark graph to provide guidelines based on accuracy and computing time for selecting the most suitable algorithm depending on network properties. The latter introduces community size as a critical factor in his evaluations, proposing a new dimension to traditional metrics that helps understand community dynamics in social media networks. Liu et al. (2019) offers a structured overview of graph summarization methods, categorizing them by input type and methodology and highlighting their applicability in real-world scenarios. Shabani et al. (2024) extend this by focusing on deep learning techniques, including graph neural networks, which are becoming essential for handling high-dimensional graph data. Both Xu et al. (2024) and Gao et al. (2024) discuss graph condensation, with the former providing a taxonomy and a formal definition to guide future research, while the latter emphasizes the application of graph condensation in training graph neural networks efficiently. The role of graph sparsification is critically analyzed by Batson et al. (2013), who discusses spectral sparsification and its applications in computational optimization, and Chen et al. (2023) focusing on demystifying graph sparsification algorithms and their ability to preserve necessary graph properties while enhancing computational performance. Chen et al. (2022) connects graph coarsening from scientific computing applications to machine learning, exploring how principles proven in scientific computing are being adapted for machine learning applications to address similar challenges of efficiency and effectiveness. Hashemi et al. (2024) provides a comprehensive survey that unifies these various strands of research-sparsification, coarsening, and condensation-under the broader umbrella of graph reduction techniques, identifying practical applications and critical future research directions in the field.

The following studies evaluate various IS methods for various domains and contexts. Cunha et al. (2023) thoroughly examine IS methods applied to non-neural and Transformer-based text classification across 19 datasets, covering topic and sentiment classification tasks. Their study integrates various classification models, including traditional TF-IDF weighting and SVMs methods, and advanced end-to-end neural network approaches such as BERT and XLNet. They explore 13 different IS methods, including Condensed Nearest Neighbor (CNN), Edited Nearest Neighbor (ENN), Instance-Based 3 (IB3), Drop3, Iterative Case Filtering (ICF), Local Set-based Smoother (LSSm), Local Set Border Selector (LSBo), Local Density-based IS (LDIS), Central Density-based IS (CDIS), eXtended Local Density-based IS (XLDIS), Prototype Selection based on Dense Spatial Partitions (PSDSP), Enhanced Global Density-based IS (EGDIS), and Curious IS (CIS), to assess their impact on reducing the training dataset size while maintaining or enhancing model accuracy. Key evaluation metrics used in their analysis include the reduction (R) mean, macro-averaged F1, and speedup of training times. This comprehensive approach highlights the significant potential of IS in modern text classification tasks and provides empirical evidence that specific IS methods can effectively streamline the training process without compromising the effectiveness of complex machine learning models.

Albelwi and Mahmood (2016) investigate the application of IS methods to optimize training processes for Deep Convolutional Neural Networks (ConvNets) using the CIFAR-10 dataset. Their research evaluates several IS strategies, including CNN, ENN, All k-NN, Random Mutation Hill Climbing (RMHC), random subset, and clustering using k-means, for their ability to reduce redundant or noisy instances in training sets, thereby enhancing computational efficiency. The study measures the effectiveness of these methods through metrics such as reduction percentage, accuracy (*AC*), and the running time of one iteration for RMHC. This approach highlights the practical benefits of IS in deep learning contexts, demonstrating that strategic instance removal can accelerate training times significantly without degrading performance, thus addressing the computational demands of training sophisticated image processing models.

Mazurowski et al. (2011) examine IS methods within medical decision support systems, specifically employing k-nearest neighbour classifiers for breast cancer detection and diagnosis. Their study conducts a comprehensive comparative analysis across three classification scenarios, utilizing simulated Gaussian data and two clinical databases. They assess several IS methods, including CNN, ENN, All k-NN, Drop2, Drop3, EXPLORE, RMHC, and random selection, focusing on the Area Under the Curve (AUC) metric. Their findings demonstrate that IS methods can significantly reduce the size of the development dataset while maintaining or even improving classification performance. This highlights the potential of IS to enhance the efficiency and effectiveness of medical diagnostic models, making them quicker and less resource-intensive without compromising diagnostic accuracy.

Blachnik (2019) explores the effectiveness of various IS methods, including CNN, IB3, relative neighbour graph editing (RNGE), ENN, All *k*NN, Drop3, ICF, hit-miss network editing (HMNE), and class conditional instance selection (CCIS), on 43 datasets featuring only numerical attributes. This detailed analysis includes preprocessing steps like normalizing attributes and employing 10-fold cross-validation to ensure rigorous testing. This study uses multiple classifiers, including 1NN, kNN, and Gaussian SVM, to evaluate the performance of these IS methods. Metrics such as accuracy and compression are assessed, and statistical significance is tested using the Friedman and Wilcoxon rank tests. This research highlights the complex impact of different IS techniques on classifier performance, setting a foundational comparison for further studies.

Blachnik and Kordos (2022) expand the investigation to a broader set of classifiers and IS methods, utilizing 40 datasets from the KEEL repository. Their comparative analysis incorporates a diverse range of IS methods, including CNN, ENN, repeated-edited nearest neighbor (RENN), All *k*NN, instance-based 2 (IB2), RNGE, Drop family, ICF, modified selective subset selection (MSS), HMNE, hit-miss network interactive editing (HMNEI), CCIS, as well as various prototype generation methods like learning vector quantization version 1,2.1 (respectively LVQ1, LVQ2.1), generalized learning vector quantization (GLVQ), and k-Means. This study assesses IS methods' impact on classifiers ranging from 1NN to Random Forest, providing insights into the suitability of IS methods across different classifiers. They also explore the effects of initial prototype set sizes, offering comprehensive insights into how IS methods can optimize training set efficiency while maintaining or enhancing classification accuracy. Their findings particularly highlight the importance of selecting appropriate IS techniques based on specific classifier requirements and dataset characteristics.

Garcia et al. (2012) provide a detailed survey of more than 50 prototype selection methods for nearest neighbour classification, drawing from a diverse collection of 58 data sets that range from small to large sizes, sourced from the UCI Machine Learning Database Repository and KEEL dataset repository. They introduce a taxonomy to categorize these methods based on operational characteristics and evaluate their effectiveness using metrics such as accuracy, Cohen's kappa, and a combined measure of accuracy and reduction rate. Their empirical analysis involves rigorous non-parametric statistical tests, including the Wilcoxon Test, to validate the performance distinctions among the methods. This study clarifies the varied impacts of prototype selection across different dataset scales and guides the selection of methods that optimally balance computational efficiency and classification performance, offering valuable insights for enhancing pattern recognition systems.

The comparison of related studies presented in Table 1 covers various aspects, including objectives, datasets used, dataset types, IS methods evaluated, machine learning models utilized, and evaluation metrics. Most studies assess IS techniques based on accuracy (AC) and reduction (or compression) rates (R). However, there is a notable scarcity of studies that evaluate these methods through the product of performance metrics, such as accuracy combined with reduction rate. Such an approach would provide insights into IS methods' effectiveness (E) by considering a trade-off between model reduction and success rate.

While existing literature has extensively evaluated various IS methods, a significant gap exists in exploring graph-based IS techniques. Specifically, the application of graph reduction techniques to solve the IS problem has been largely unexplored. As evident from Table 1, previous studies have primarily focused on conventional IS methods, with minimal consideration given to how graph reduction can be leveraged in this context. To the best of our knowledge, this study represents the first systematic review and evaluation of graph reduction techniques for their effectiveness in IS, assessing their feasibility and suitability across various types and sizes of data. By exploring this novel approach, our research aims to expand the field of IS and foster new directions in data reduction for machine learning tasks, potentially offering more efficient and effective solutions than traditional methods.

3 Graph reduction techniques

This section outlines various graph reduction techniques, categorized by their methods for simplifying complex graph structures. The seven primary categories are node sampling, edge sampling, graph traversal and exploration, community sampling, graph sparsification, graph coarsening, and node representative sampling. Figure 1 provides visual examples for each category, illustrating how they work in general.

3.1 General framework for graph reduction

While the graph reduction techniques we explore vary in their specific approaches, they all follow a general framework for reducing a dataset through graph representation. Algorithm 1 presents a high-level overview of this process, encompassing the common steps shared across most of the reduction techniques studied in this work. This algorithm out-

Table 1 Comparison of Related Studies

Ref	Objective	Datasets	Type of Data	IS methods	Models	Evaluation metrics
Cunha et al. (2023)	Compare IS for text classification	DBLP, Books, ACM, 20 Newsgroups, OHSUMED, Reuters90, Web of Science (WOS-5736, WOS-11967), WebKB, TREC, Stanford Sentiment Treebank, Pang Movie Reviews, Movie Reviews, Vader Movie Reviews, MPQA, Subjectivity, Stanford Sentiment Treebank, Yelp Reviews, Vader NYT Reviews	Text	CNN, ENN, ICF, IB3, Drop3, LSSm, LSBo, LDIS, CDIS XLDIS, PSDSP, EGDIS, CIS	RoBERTa, BART, XLNet, BERT, DistilBERT, ALBERT, MF+SVM, GPT2, TFIDF+SVM	Reduction mean, Macro F1, Speedup
Albelwi and Mahmood (2016)	Optimize training for ConvNets	CIFAR-10	Image	Random, CNN, ENN, All *k*NN, RMHC, Clustering	ConvNet	*R*, *AC*, Running time
Mazurowski et al. (2011)	Compare IS for Medical	Simulated, Breast Cancer Diagnosis, Breast Cancer Detection	Clinical Database	CNN, ENN, All *k*NN, DROP2-3, EXPLORE, RMHC, Random	*k*NN	AUC

Table 1 (continued)

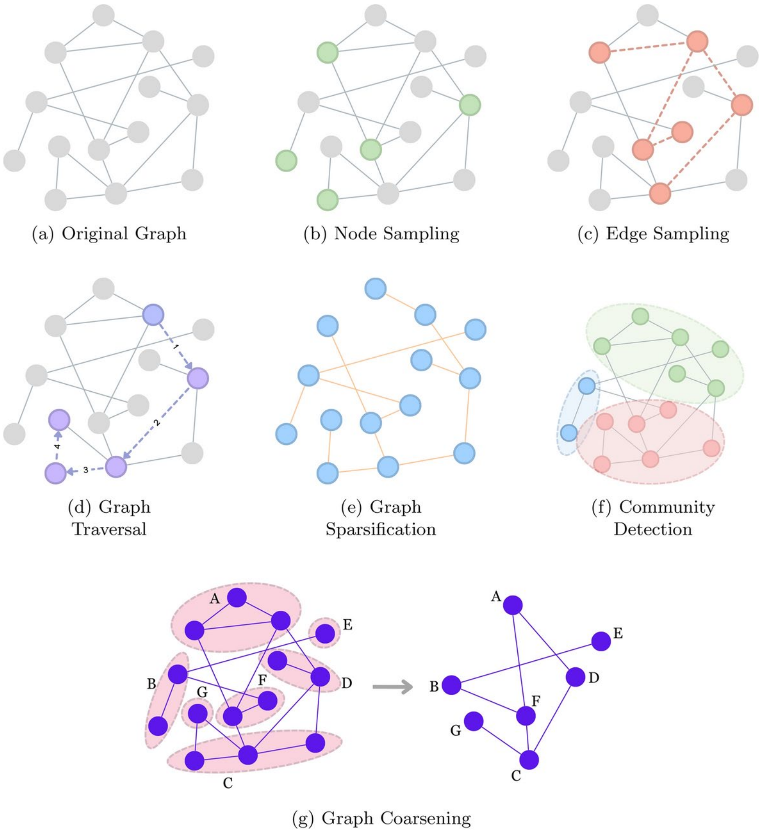
Ref	Objective	Datasets	Type of Data	IS methods	Models	Evaluation metrics
Garcia et al. (2012)	Survey and empirical study of prototype selection methods	Abalone, appendicitis, australian, automobile, balance, banana, bands, breast, bupa, car, chess, cleveland, coil2000, contraceptive, crx, dermatology, ecoli, flare-solar, german, glass, haberman, hayes-roth, heart, hepatitis, housevotes, iris, led7digit, lymphography, magic, mammographic, marketing, monk-2, newthyroid, nursery, pageblocks, penbased, phoneme, pima, ring, saheart, satimage, segment, sonar, spambase, spectheart, splice, tae, texture, thyroid, tic-tac-toe, titanic, twonorm, vehicle, vowel, wine, wisconsin, yeast, zoo	Tubular (Varied)	CNN, RNN, ENN, Ullmann, SNN, RENN, AIKNN, TCNN, MNV, MultiEdit, Shrink, IB2-3, MC1, RMHC, MCS, ELH, ELGrow, Explore, MoCS, VSM, GGE, RNGE, PF, GGA, MENN, DROP1-5, DEL, EDA, CerveronTS, ICF, MCNN, IGA, PSRCG, IKNN, ZhangTS, Iterative MaxNCN, Reconsistent, CPruner, SSGA, PBIL, CHC, POP, NCNEdit, ENRBF, ENRBF2, ENNProb, ENNTh, SVBPS, BSE, MSS, GCNN, FCNN, FCNN2, FCNN3, FCNN4, NRMCS, GA-MSE-CC-FSM, SSMA, HMNC, HMNE, HMNEI, TRKNN, PSC, CCIS, CoCoIS	1NN	AC, Cohen's kappa, $AC{\times}R$, Cohen's kappa $\times R$

Table 1 (continued)

Ref	Objective	Datasets	Type of Data	IS methods	Models	Evaluation metrics
Blachnik (2019)	Compare ensembles adapted to IS	Appendicitis, balance, banana, bands, bupa, cleveland, coil2000, glass, haberman, hayes-roth, heart, hepatitis, ionosphere, iris, led-7digit, letter, magic, mammographic, marketing, monk-2, movement_libras, newthyroid, optdigits, page-blocks, penbased, phoneme, pima, ring, satimage, segment, sonar, spambase, spectheart, tae, texture, thyroid, titanic, twonorm, vehicle, wdbc, wine, wisconsin, yeast	Tubular (Varied)	CNN, IB3, RNGE, ENN, All *k*NN, DROP3, ICF, HMNE, CCIS	1NN, *k*NN, Gaussian SVM	*AC*, *R*
Blachnik and Kordos (2022)	Compare IS methods across classifiers	Appendicitis, balance, banana, bands, bupa, cleveland, glass, haberman, hayes-roth, heart, hepatitis, ionosphere, iris, led7digit, mammographic, marketing, monk-2, movement_libras, newthyroid, optdigits, page-blocks, phoneme, pima, ring, satimage, segment, sonar, spambase, spectheart, tae, texture, thyroid, titanic, twonorm, vehicle, vowel, wdbc, wine, wisconsin, yeast	Tubular (Varied)	ENN, RENN, All *k*NN, HMNE, RNG, HMNEI, CNN, MSS, IB2, ICF, CCIS, DROP2, DROP5, DROP1, DROP3, GLVQ, LVQ1, LVQ2.1, k-Means, Random	1NN, *k*NN, NB, GLM, SVM, C4.5, RF	*AC*, *R*

Table 1 (continued)

Ref	Objective	Datasets	Type of Data	IS methods	Models	Evaluation metrics
Ours	Compare graph reduction techniques for IS	Abalone, banana, car, census-income, chess, coil2000, contraceptive, diabetes, fars, german, hcc, heart, ldpa, nursery, optdigits, page-blocks, phoneme, ringnorm, satellite, segment, skin, spambase, splice, stroke, texture, thyroid, titanic, twonorm, yeast	Tubular (Varied)	AC, KC, BRW, LP, BRWE, LE, BWT, LD, BFS, LV, CM, CBE, CSP, MHRW, DG, ML, DM, N2V, FG, NBRW, FF, FFE, FR, PR, IM, PE, RE, RN, RWE, SRW, SB, SC, SG, TG, VC	LR, RF, *k*NN, XGB, NB	AC, F1, AUC, E, R, T_R, ES



(a) Original Graph (b) Node Sampling (c) Edge Sampling

(d) Graph Traversal (e) Graph Sparsification (f) Community Detection

(g) Graph Coarsening

Fig. 1 Visual Examples of Graph Reduction Techniques. The original graph (**a**), sampling methods by node (**b**), edge (**c**), traversing (**d**), sparsification (**e**), community (**f**) and coarsening (**g**)

lines our general approach to applying graph reduction techniques for instance selection. It begins by constructing a graph representation of the dataset, then applies the chosen reduction technique to select a subset of nodes or directly produce a reduced graph. Finally, it maps the nodes of the reduced graph back to instances in the dataset. The algorithm accommodates various categories of reduction techniques, including node sampling, edge sampling, graph traversal, community sampling, graph sparsification, graph coarsening, and node representative methods. Each of these categories is implemented as a specific function call within the algorithm, allowing for a unified framework while maintaining the unique characteristics of each approach. Detailed explanations of these techniques are provided in the supplementary material.

```
Require: Dataset $D$, reduction rate $r$, reduction technique $T$
Ensure: Reduced dataset $D_{reduced}$
function GraphReduction($D$, $r$, $T$)
    $G \leftarrow$ ConstructGraph($D$)
    $n \leftarrow |V(G)| \times (1 - r)$                    ▷ Number of nodes to keep
    $G_{reduced} \leftarrow \emptyset$
    if $T =$ NodeSampling then
        $V_{selected} \leftarrow$ SelectNodes($G, n$)
    else if $T =$ EdgeSampling then
        $E_{selected} \leftarrow$ SelectEdges($G, n$)
        $V_{selected} \leftarrow$ GetNodesFromEdges($E_{selected}$)
    else if $T =$ GraphTraversal then
        $V_{selected} \leftarrow$ TraverseAndSelect($G, n$)
    else if $T =$ CommunitySampling then
        $C \leftarrow$ DetectCommunities($G$)
        $V_{selected} \leftarrow$ SelectNodesFromCommunities($C, n$)
    else if $T =$ GraphSparsification then
        $G_{reduced} \leftarrow$ SparsifyGraph($G, n$)
    else if $T =$ GraphCoarsening then
        $G_{reduced} \leftarrow$ CoarsenGraph($G, n$)
    else if $T =$ NodeRepresentative then
        $E \leftarrow$ ComputeNodeEmbeddings($G$)
        $V_{selected} \leftarrow$ ClusterAndSelectNodes($E, n$)
    end if
    if $G_{reduced} = \emptyset$ then
        $G_{reduced} \leftarrow$ ConstructSubgraph($G, V_{selected}$)
    end if
    $D_{reduced} \leftarrow$ MapNodesToInstances($G_{reduced}, D$)
    return $D_{reduced}$
end function
```

Algorithm 1 Graph reduction for instance selection

3.2 Node sampling

Node sampling techniques reduce the size of a graph by selecting a subset of representative or critical nodes based on specific criteria such as degree or PageRank Page et al. (1999). This category is distinct from edge sampling, primarily focusing on the nodes rather than their connections. The goal is to maintain a smaller graph that still encapsulates the essential characteristics of the original. This includes random node (RN) sampling, degree-based sampling, PageRank-based sampling, snowball sampling, triangle-based sampling, and community-based sampling.

3.3 Edge sampling

Edge Sampling focuses on reducing the number of edges in a graph while attempting to preserve the graph's structural properties. Techniques in this category select edges based on random strategies or specific characteristics (like edges in triangles or connecting different communities). Unlike node sampling, edge sampling directly influences the connectivity and flow within the graph, impacting how information is distributed and processed within the network. This includes random edge sampling, random walk edge sampling, biased random walk edge sampling, triangle edge sampling, forest fire edge sampling, preferential edge sampling, and community bridge edge sampling.

3.4 Graph traversal and exploration

Graph traversal and exploration techniques involve traversing or exploring the graph, often used for understanding or extracting graph features and sometimes for reducing graph size. These methods use random walks, breadth-first or depth-first search strategies to sample or map the graph. This category differs from others, such as node or edge sampling, in that it focuses on the process of graph traversal itself, which can inform or dictate subsequent reduction techniques. This includes forest fire (FF) sampling Leskovec and Faloutsos (2006), simple random walk sampling, biased random walk sampling, metropolis-hastings random walk sampling Metropolis et al. (1953), Hastings (1970), breadth-first search (BFS) sampling, non-backtracking random walk sampling, and frontier sampling.

3.5 Graph sparsification

Graph sparsification techniques aim to reduce the complexity of a graph by removing edges in such a way that the graph's fundamental properties, like path lengths and connectivity, are preserved as much as possible. This category is distinct from edge sampling because sparsification explicitly seeks to maintain the utility and integrity of the graph for analytical tasks, whereas edge sampling might be employed for more general size reduction. This includes K-core sparsification Batagelj and Zaversnik (2003), and community structure preserving sparsification.

3.6 Graph coarsening

Graph coarsening involves merging multiple nodes or edges to form a simplified version of the original graph. This category directly reduces the graph's granularity, typically by combining nodes based on certain features such as community membership or spectral properties. Coarsening is closely related to community sampling but directly emphasises restructuring the graph to reduce its complexity significantly. This includes vertex collapsing, affinity clustering coarsening Frey and Dueck (2007), Louvain coarsening Blondel et al. (2008), and spectral clustering coarsening.

3.7 Community sampling

Community sampling techniques are dedicated to detecting and leveraging community structures within graphs. These methods focus on grouping nodes that share similar connectivity patterns and are more densely connected amongst themselves than the rest of the graph. This category is closely related to coarsening, as community detection can precede merging nodes within communities to simplify the graph. This includes walktrap sampling Pons and Latapy (2005), infomap sampling Rosvall and Bergstrom (2008); Rosvall et al. (2009), leading eigenvector sampling Newman (2006), spinglass sampling Reichardt and Bornholdt (2006), Traag and Bruggeman (2009), multilevel sampling Blondel et al. (2008), fastGreedy sampling Clauset et al. (2004), label propagation sampling Raghavan et al. (2007), Leiden sampling Traag et al. (2019), and demon sampling Coscia et al. (2014, 2024),

3.8 Node representative sampling

Node representative sampling is a technique that combines node embedding methods (like Node2Vec) with clustering to select representative nodes that best capture the structural and feature-based information of the graph. This category is unique as it uses sophisticated machine learning models to generate embeddings that reflect the nodes' roles and positions within the graph, which are then used to select key nodes for retaining in the reduced graph.

To conclude, Fig. 2 illustrates the implemented graph reduction approaches along with their techniques.

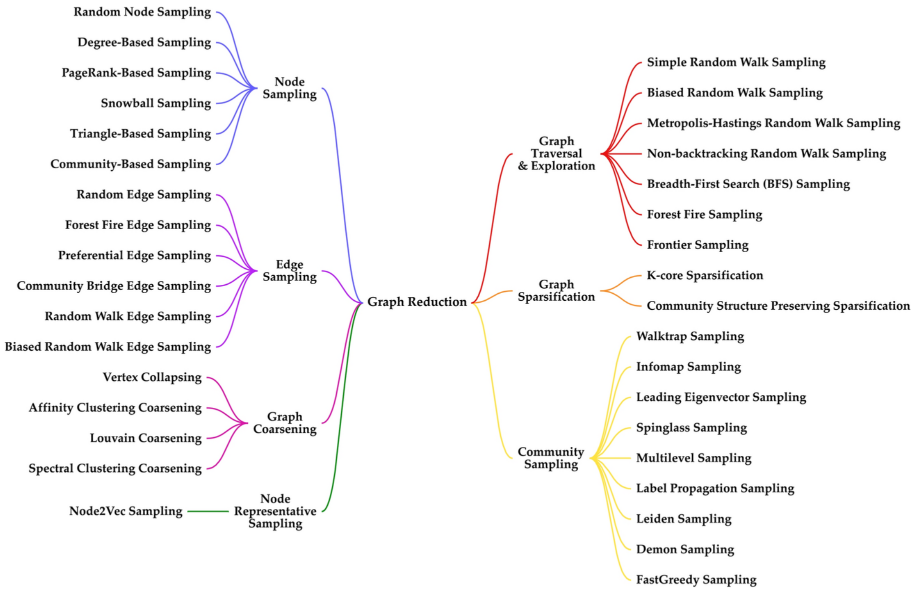

Random Node Sampling
Degree-Based Sampling
PageRank-Based Sampling
Snowball Sampling
Triangle-Based Sampling
Community-Based Sampling
Node Sampling
Random Edge Sampling
Forest Fire Edge Sampling
Preferential Edge Sampling
Community Bridge Edge Sampling
Random Walk Edge Sampling
Biased Random Walk Edge Sampling
Edge Sampling
Vertex Collapsing
Affinity Clustering Coarsening
Louvain Coarsening
Spectral Clustering Coarsening
Graph Coarsening
Node2Vec Sampling
Node Representative Sampling
Graph Reduction
Graph Traversal & Exploration
Simple Random Walk Sampling
Biased Random Walk Sampling
Metropolis-Hastings Random Walk Sampling
Non-backtracking Random Walk Sampling
Breadth-First Search (BFS) Sampling
Forest Fire Sampling
Frontier Sampling
Graph Sparsification
K-core Sparsification
Community Structure Preserving Sparsification
Community Sampling
Walktrap Sampling
Infomap Sampling
Leading Eigenvector Sampling
Spinglass Sampling
Multilevel Sampling
Label Propagation Sampling
Leiden Sampling
Demon Sampling
FastGreedy Sampling

Fig. 2 List of implemented graph reduction techniques

4 Experimental setup

This section discusses the experimental setup to outline the datasets utilised, methods for graph construction, graph reduction methods, machine learning algorithms, evaluation metrics and evaluation protocol. Figure 3 presents the high-level framework of our experimental methodology for evaluating graph reduction techniques as a novel approach to instance selection. This comprehensive workflow illustrates the process from data collection to result comparison, encompassing key stages such as preprocessing, graph construction, applying various graph reduction methods and evaluating machine learning models.

4.1 Datasets

Table 2 presents the datasets used to evaluate the graph reduction techniques for IS. It includes the names of the datasets, the total number of instances, features, classes, and imbalance ratios. We have selected 29 datasets from diverse domains such as healthcare, finance, social science, and synthetic data, sourced from the KEEL repository Derrac et al. (2015) and UCI Machine Learning Database Repository Asuncion et al. (2007). These datasets vary significantly, with sizes ranging from 303 to 250,000 instances and up to 21 classes featuring different levels of class imbalance and number of features.

The datasets are split into three subsets: training (80%), validation (10%), and testing (10%). The training subset is used to apply the graph reduction techniques to derive a reduced training set. The validation subset helps optimise the hyperparameters of these techniques, while the testing subset is employed to evaluate the machine learning models trained on the reduced datasets. To ensure reproducibility, we control randomness in the splitting process wherever possible. In terms of preprocessing, all features are retained. Categorical features are converted into numerical values, and all numerical features are scaled to a range of 0 to 1. Additionally, target columns are standardised across all datasets.

4.2 Graph conversion

Given that the datasets utilised in this study are tabular-comprising observations and attributes-direct application of graph reduction techniques is not feasible without first convert-

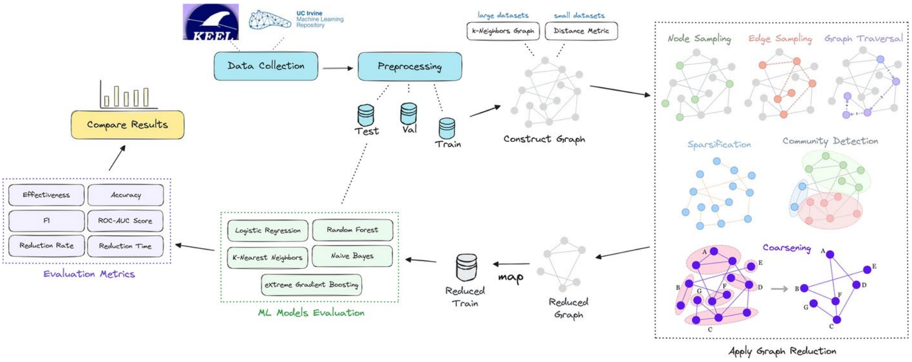


Fig. 3 High-level framework of the methodology

Table 2 Datasets overview. *N*, *P*, and *C* represent the number of instances, features, and classes

Dataset Name	Objective	Description	*N*	*P*	*C*
Heart disease	Healthcare	Predict heart disease presence	303	13	2
HCC	Oncology	Identify Hepatocellular Carcinoma cases	615	12	2
Stroke prediction	Healthcare	Predict stroke likelihood	749	10	2
Diabetes	Healthcare	Predict diabetes onset	768	8	2
German credit	Financial risk	Classify credit risks	1000	20	2
Contraceptive	Healthcare	Predict contraceptive method choice among married women	1473	9	3
Yeast	Bioinformatics	Determine the localization site of yeast cells	1484	8	10
Car evaluation	Automotive	Evaluate cars for acceptability classification	1728	6	4
Titanic	Social science	Analyze survival patterns of Titanic passengers	2201	3	2
Segment	Image recognition	Classify regions of an image from seven types of outdoor scenes	2310	19	7
Splice	Bioinformatics	Recognize boundaries between exons and introns in DNA sequences	3190	61	2
Chess end-game	Game theory	Predict game outcome	3196	36	2
Abalone	Marine biology	Predict the age of abalone from physical measurements	4168	8	21
Spam-base	Email classification	Classify emails as spam or non-spam	4601	57	2
Banana	Synthetic data	Classify instances into two clusters shaped like bananas	5300	2	2
Phoneme	Speech recognition	Distinguish between nasal and oral sounds	5404	5	2
Page-blocks	Computer science	Classify blocks of a page layout from documents	5473	10	5
Texture	Image recognition	Distinguish between 11 different textures	5500	40	11
Opt-digits	Computer science	Recognize handwritten digits	5620	64	10
Satellite	Climate	Classify the type of land cover	6435	36	6
Thyroid	Healthcare	Diagnose thyroid disease	7200	21	3
Two-norm	Synthetic data	Distinguish between two classes	7400	20	2
Ring-norm	Synthetic data	Classify into one of the rings	7400	20	2
Coil-20	Image recognition	Object recognition and classification	9822	85	2
Nursery	Social science	Rank applications for nursery schools	12960	8	5
Fars	Traffic safety	Classify level of injury suffered in car accidents	100968	29	8
LDPA	Healthcare	Classify person activity from sensor data	164860	7	11
Census-income	Social science	Predict whether income exceeds $50K/year	199523	41	2
Skin segmentation	Computer science	Classify pixels in images as skin or non-skin	245057	3	2

ing the datasets into graph formats. To address this, we employ two primary methods for graph conversion:

- Distance Metrics: We calculate pairwise distances between instances using cosine, Euclidean, and Manhattan metrics to generate a similarity matrix, *S*, of dimensions $N \times N$, where N is the number of instances. Each element S_{ij} in the matrix represents the similarity or distance between the i^{th} and j^{th} instances. This similarity matrix *S* is then used to construct a graph *G* using the NetworkX library, where distances form the weights of the edges.

- k-Neighbors Graph: For datasets exceeding 10,000 instances, we use the k-neighbors graph approach. This method involves connecting each instance to its k nearest neighbours, significantly reducing memory usage compared to a full pairwise distance matrix. This approach is favoured for large datasets due to its computational efficiency, although it might capture less complex relationships between instances than the distance-based method. The choice of k and the neighbour selection criteria directly influence the structure and quality of the resulting graph.The selection between these two methods depends on the size and characteristics of the dataset. While the distance metrics provide a comprehensive representation of all pairwise relationships, they are computationally and memory-intensive, making them less suitable for larger datasets. On the other hand, the k-neighbors method offers a scalable alternative that aligns well with the constraints of large datasets despite its limitations in capturing intricate instance relationships.

4.3 Graph reduction methods

In this study, we evaluate graph reduction techniques, as detailed in Sect. 3. These techniques were implemented in Python, leveraging existing libraries and packages wherever available. Using well-established implementations from libraries ensures that the methods are optimised for performance and robust in handling different datasets.

Each graph reduction technique has its own set of hyperparameters that can significantly affect their performance and effectiveness. To find the optimal set of hyperparameters for each method, we utilise a Bayesian optimisation approach via the Optuna framework Akiba et al. (2024). This framework is chosen for its efficiency and ability to handle complex optimisation landscapes.

The hyperparameter tuning process is configured to perform 20 trials with a maximum timeout of 30 min per trial. This limit is set to ensure that each technique is evaluated within a reasonable time frame, making the experimental process both efficient and manageable. The optimisation objective for each trial is to maximise the validation effectiveness, primarily measured by accuracy.

4.4 Machine learning models

To assess the effectiveness of the reduced training sets produced by the graph reduction techniques, we employ five architecturally diverse algorithms, including Logistic Regression (LR), Random Forest (RF), k-nearest Neighbors (KNN), Extreme Gradient Boosting (XGB) and Naive Bayes (NB). Each is chosen for its unique methodological characteristics. We used the default hyperparameters set by the respective libraries for each algorithm to maintain consistency and impartiality in our evaluations. This approach ensures that any observed differences in performance are attributable to the effectiveness of the graph reduction techniques rather than variations in model tuning.

4.5 Evaluation metrics

Our study employs a comprehensive set of evaluation metrics to assess the effectiveness of graph reduction techniques applied to IS in machine learning tasks. Table 3 presents

Table 3 Evaluation metrics for assessing graph reduction techniques for IS

Metric	Description/equation
Accuracy	$\mathrm{AC} = \frac{TP+TN}{TP+TN+FP+FN}$
Precision	$\mathrm{PR} = \frac{TP}{TP+FP}$
Recall	$\mathrm{RE} = \frac{TP}{TP+FN}$
F1 Score	$F1 = 2 \times \frac{\mathrm{PR} \times \mathrm{RE}}{\mathrm{PR}+\mathrm{RE}}$
ROC AUC Score[1]	Area under the ROC Curve
Accuracy effectiveness	$E_{AC} = \mathrm{AC} \times R$
F1 effectiveness	$E_{F1} = F1 \times R$
Efficiency score	$\mathrm{ES} = 2/\left(\frac{1}{E_{AC}} + \frac{T_R}{\text{Total Instances}}\right)$
Reduction rate	$R = \left(1 - \frac{\text{Size of Reduced Set}}{\text{Size of Original Set}}\right)$
Graph build time	$T_G = \text{Time taken to construct the graph}$
Reduction time	$T_R = \text{Time taken to reduce the dataset}$
Training time	$T_{MT} = \text{Time taken to train the model}$

TP True positives, *TN* true negatives, *FP* false positives, *FN* false negatives

[1] *ROC* Receiver operating characteristic, *AUC* area under the curve, measures the area under the ROC curve, a plot of the true positive rate against the false positive rate at various threshold settings

a detailed overview of these metrics, which include traditional measures such as accuracy, precision, recall, and F1 score Rustamov et al. (2023), Shah et al. (2020), alongside more specialised metrics such as ROC AUC score Bradley (1997), Fawcett (2006), which assesses the model's discriminative capacity. Additionally, we have introduced effectiveness metrics that combine traditional performance scores with the reduction percentage to gauge the efficiency of data reduction alongside model accuracy. These metrics-accuracy and F1 effectiveness-better illustrate trade-offs between model complexity and performance Malhat et al. (2020), which will be one of our primary metrics when evaluating the different techniques. The table also lists operational metrics such as training, reduction, and graph build time in seconds. These are crucial for assessing the practical implications of graph reduction methods in real-world scenarios.

We also introduce an efficiency score metric that balances test effectiveness with the average reduction time per instance. The efficiency score is calculated using the harmonic mean of the test effectiveness and the normalised reduction time, which is obtained by dividing the reduction time by the total number of instances. This metric provides a comprehensive assessment of the efficiency of graph reduction techniques, considering both the effectiveness of the reduced dataset in the machine learning task and the computational efficiency of the reduction process itself.

All experiments were conducted on a Windows 10 machine equipped with Python 3.10. The hardware specifications include an Intel Core i7-12700 CPU with 12 cores, 20 threads, and 128GB of memory. This robust configuration ensures that the machine can efficiently handle the computationally intensive tasks associated with our experiments. The scope of this study is explicitly focused on reducing the size of tabular data for supervised learning classification tasks.

5 Results

To present our findings effectively, we have organized the results into two primary sections based on the size of the datasets: (1) small datasets, which include datasets with fewer than 10,000 instances and are constructed using distance metrics, and (2) large datasets, which comprise datasets with more than 10,000 instances and are constructed using the k-neighbors graph approach. Table 4 provides the abbreviations used to refer to the graph reduction techniques.

Our results are initially presented in terms of the average performance for each metric across datasets for each technique, which offers a straightforward measure of effectiveness. These average metrics provide a basis for comparing the absolute performance of the techniques across varied testing environments. In addition to these average metrics, we have included the ranking of each technique, with the best achievable rank being 1 and the worst 35. The rank represents the relative standing of each graph reduction technique within the context of each dataset and is determined by comparing the performance of techniques across all experiments. This ordinal assessment complements the average metrics and is especially insightful when there is significant variability in dataset characteristics. The inclusion of rank aids in discerning the robustness and consistency of each technique, emphasizing their comparative effectiveness in a way that average values alone cannot capture. Thus, ranks alongside averages afford a multi-dimensional view of performance, high-

Table 4 Graph reduction techniques and their abbreviations

Technique	Abbr	Technique	Abbr
Affinity Clustering	AC	K Core	KC
Biased RW	BRW	Label Propagation	LP
Biased RW Edge	BRWE	Leading Eigenvector	LE
Biased Walktrap	BWT	Leiden	LD
Breadth-First Search	BFS	Louvain	LV
Community	CM	Community Bridge Edge	CBE
Community Structure Preserving	CSP	Metropolis-Hastings RW	MHRW
Degree	DG	Multilevel	ML
Demon	DM	Node2Vec	N2V
Fastgreedy	FG	Non Backtracking RW	NBRW
Forest Fire	FF	Forest Fire Edge	FFE
Frontier	FR	PageRank	PR
Infomap	IM	Preferential Edge	PE
Random Edge	RE	Random Node	RN
RW Edge	RWE	Simple RW	SRW
Snowball	SB	Spectral Clustering	SC
Spinglass	SG	Triangle Based	TG
Vertex Collapsing	VC		

lighting both absolute and relative strengths and weaknesses of the graph reduction methods under investigation.

5.1 Small datasets

Table 5 outlines the evaluation results for small datasets in terms of accuracy effectiveness (E_{AC}), F1 effectiveness (E_{F1}), reduction rate (R) and reduction time (T_R), alongside their respective rankings. Observing the table, we can point out the following insights:

- LD and ML show robust accuracy and F1 effectiveness with a relatively high rank, indicating consistent performance across datasets. Although LE is among the highest ranks in terms of accuracy effectiveness, its F1 effectiveness is slightly lower in the rankings. All three are community detection-based sampling techniques.
- LV achieves the highest rank, denoting high reduction capabilities. However, the effectiveness rankings are not as good, with its F1 effectiveness among the worst-performing techniques. This means that although it has high reduction capabilities, it fails to achieve good performance.
- Some techniques with higher average effectiveness scores, such as FR and LP, do not hold the top rankings, suggesting other techniques perform better across multiple datasets.
- Conversely, techniques like IM, PR and LP with lower average effectiveness scores maintain competitive ranks, highlighting their stable performance across diverse datasets.
- It is evident that the community detection-based sampling techniques, such as LD, LE, ML, FG and IM, are among the top performing in terms of effectiveness scores, indicating that these methods are adept in preserving essential community structures crucial to

Table 5 Average test effectiveness, reduction rate and reduction time obtained by the graph reduction techniques over small datasets

Method	$E_{AC} \uparrow$	Rank	Method	$E_{F1} \uparrow$	Rank	Method	$R \uparrow$	Rank	Method	$T_R \downarrow$	Rank
LD	0.8034	10.08	ML	0.7169	10.38	LV	0.9120	8.58	BFS	0.0017	2.92
LE	0.8030	11.33	LD	0.7158	10.54	AC	0.9785	9.96	RN	0.0008	3.29
ML	0.7991	11.42	CM	0.7127	10.71	VC	0.8693	12.54	DG	0.0050	5.12
NBRW	0.7993	11.50	SRW	0.7103	11.21	NBRW	0.9814	12.88	MHRW	4.1059	5.29
FG	0.7988	11.58	IM	0.7096	11.58	SB	0.9814	12.96	NBRW	0.4707	6.58
SRW	0.8001	12.00	FFE	0.7184	11.83	LE	0.9768	13.71	SRW	1.1622	7.54
SB	0.7945	12.58	PR	0.7038	12.38	BRW	0.9771	13.96	SB	2.2136	9.04
FFE	0.7936	12.88	FG	0.7077	12.58	IM	0.9751	14.08	RE	24.3170	9.08
CM	0.7928	13.08	LE	0.7156	12.62	CM	0.9735	14.12	FR	0.8484	9.08
IM	0.7954	13.29	NBRW	0.7174	13.62	FF	0.9766	14.25	FF	2.0439	10.54
FR	0.7953	13.54	LP	0.7080	14.54	LD	0.9787	14.71	PE	11.1198	11.17
AC	0.7874	13.67	AC	0.6902	14.83	LP	0.9809	14.75	BRW	7.0073	14.25
DG	0.7933	13.71	RN	0.6923	14.83	ML	0.9785	15.08	FFE	156.1304	14.67
FF	0.7914	13.79	FR	0.7103	15.12	FR	0.9806	15.42	BRWE	1674.4743	16.73
LP	0.7962	13.83	BRW	0.7060	15.54	FG	0.9731	15.50	RWE	1414.9614	17.46
BRW	0.7907	14.42	SB	0.7066	15.58	FFE	0.9725	15.50	CSP	29.2928	18.21
VC	0.7239	14.46	FF	0.7072	16.12	DG	0.9749	15.79	KC	137.1062	18.33
PR	0.7867	15.62	DG	0.7036	16.29	SRW	0.9776	16.12	CM	37.7419	18.67
RN	0.7809	16.71	SG	0.6752	16.42	BWT	0.9348	16.33	LP	10.4145	19.46
LV	0.7502	16.92	BWT	0.6832	16.83	PR	0.9596	18.04	ML	15.8310	21.12
BWT	0.7635	17.71	VC	0.6315	16.96	CBE	0.9656	18.21	LE	32.1415	21.33
SG	0.7453	19.04	N2V	0.6624	18.33	SC	0.9558	18.42	LD	14.6479	21.38
N2V	0.7403	19.54	BFS	0.6884	18.67	SG	0.9052	19.04	PR	14.9387	22.00
BFS	0.7749	19.75	CBE	0.6708	19.04	BFS	0.9427	19.33	FG	79.1674	22.33
CBE	0.7596	19.83	TG	0.5960	19.42	RN	0.9647	19.96	CBE	33.3202	22.67
SC	0.7676	20.29	SC	0.6731	19.50	MHRW	0.9338	20.12	AC	22.6338	23.46
MHRW	0.7626	23.54	MHRW	0.6721	20.92	BRWE	0.6594	21.50	VC	35.1726	23.95
TG	0.6647	24.58	LV	0.6202	22.54	N2V	0.9149	22.08	IM	50.8693	25.71
DM	0.7098	25.12	DM	0.6280	22.62	DM	0.8668	22.67	SC	53.5910	26.17
RE	0.6217	25.88	RE	0.5391	23.25	RE	0.7964	22.88	LV	28.1999	26.18
BRWE	0.5089	26.17	CSP	0.5743	24.83	RWE	0.6490	24.08	BWT	398.4134	28.38
RWE	0.5015	28.38	PE	0.4927	26.25	CSP	0.8495	24.71	TG	1174.3853	31.17
CSP	0.6646	28.92	BRWE	0.4267	26.54	TG	0.8153	26.54	N2V	621.3610	32.08
PE	0.5734	29.96	RWE	0.4198	28.00	PE	0.7068	28.58	DM	1273.9833	32.22
KC	0.4959	32.83	KC	0.4390	29.67	KC	0.6315	32.00	SG	2581.5605	33.67

identify the most representative nodes within each community.

- In the graph traversal family, NBRW and SRW are among the top performing in terms of average effectiveness scores and reduction rates, with FR, FF and BRW following closely behind. This highlights the potential of random walks in identifying the most important nodes by traversing the graph.
- Concerning the graph coarsening techniques, LV, VC and AC are ranked highly in reduction rates (top 3) and moderately in terms of average effectiveness scores, denoting their potential to achieve high reduction rates. However, their moderate effectiveness ranks could be biased due to their high reduction rates.

- The SB and CM are among the top-performing techniques in the node sampling family, with the majority of node sampling techniques ranking in the bottom half of the table.
- KC, CSP and several edge sampling techniques, such as PE, RWE, RE and BRWE, are among the worst performing techniques in terms of average effectiveness scores and reduction rates, denoting edge sampling techniques may not be as suitable for smaller datasets. On the other hand, FFE is the only edge sampling technique that is one of the top-performing techniques, with a vast disparity between FFE and other edge sampling techniques. Specifically for KC, the potential underperformance can be attributed to its method of iteratively removing nodes (and their associated edges) that do not meet a certain degree threshold. This process can disproportionately affect smaller datasets by eliminating nodes crucial for preserving the core structural properties and connectivity, leading to a significant degradation in the quality of the graph representation.
- The spread of ranks across different techniques implies that no single technique is universally superior in all metrics (i.e., no techniques below rank 10 in effectiveness scores), emphasizing the need for tailored approaches based on specific dataset characteristics and desired outcomes.
- BFS, RN, and DG excel in graph reduction speed, achieving near-instantaneous times with the best ranks of 2.92, 3.29, and 5.12, respectively.
- While MHRW shows a considerably higher average reduction time compared to the top-performing techniques in this category, its high rank indicates a fast reduction across all datasets. A similar trend is observed with FFE, BRWE, and RWE, where their average reduction times are significantly high, yet their rankings suggest faster performance on specific types of datasets.
- Simpler techniques, which require less computation, generally achieve faster graph reduction times. In contrast, community detection-based and graph coarsening techniques, which involve more complex computations, rank lower, indicating longer reduction times. Nonetheless, most reduction techniques do not exhibit excessively high average reduction times for smaller datasets. The rankings for graph reduction times vary widely, from 2 to 33, highlighting the disparity in efficiency across different techniques.

Table 6 presents the evaluation results for small datasets, detailing performance metrics such as accuracy (AC), F1 score ($F1$), and ROC-AUC score (AUC), along with their respective rankings and baseline values. The baseline values are established by directly applying machine learning models to the datasets without any graph reduction. Key insights derived from the table include:

- SRW, TG, ML, LD, PR, and CM stand out with strong performance scores, achieving the highest ranks close to the baseline scores.
- Many techniques achieve higher accuracy and F1 scores than the baseline, indicating that the reduction techniques may have removed potentially noisy data, thereby enhancing performance.
- Although several techniques, such as KC, PE, MHRW, and RE, show higher average accuracy, their ranks are considerably low, suggesting poorer performance across all datasets. A similar trend is observed in techniques such as CSP, FF, RE, LP, MHRW, and KC in terms of F1 score, and CSP, MHRW, BRWE, NBRW, and KC in terms of ROC-AUC score.

Table 6 Average test performance metrics obtained by the graph reduction techniques over small datasets

Method	$AC \uparrow$	Rank	Method	$F1 \uparrow$	Rank	Method	AUC ↑	Rank
SRW	0.8729	11.17	PR	0.8062	12.17	TG	0.8531	12.38
ML	0.8749	12.38	FFE	0.8037	13.04	ML	0.8952	13.79
LD	0.8746	12.42	TG	0.7590	13.08	SRW	0.8974	13.79
PR	0.8740	12.58	SRW	0.8011	13.46	LD	0.8951	13.96
LE	0.8794	12.92	ML	0.8008	13.92	CM	0.8975	14.08
FG	0.8752	12.96	CM	0.8025	13.92	FFE	0.9004	14.29
TG	0.8311	13.04	BFS	0.8022	14.00	BFS	0.8969	14.75
BFS	0.8737	13.58	SG	0.7730	14.21	DG	0.8977	14.83
RN	0.8768	13.83	LD	0.7980	14.33	FG	0.8983	15.25
CM	0.8756	14.12	RN	0.8039	14.67	LE	0.8983	15.38
SG	0.8557	14.46	N2V	0.7922	15.08	PR	0.8989	15.38
FFE	0.8763	14.62	DG	0.8002	15.46	RN	0.8965	15.42
DG	0.8728	14.67	FG	0.7956	15.62	SG	0.8784	15.50
N2V	0.8665	14.96	KC	0.8140	15.67	IM	0.8992	15.79
NBRW	0.8727	15.54	IM	0.8047	15.71	BRW	0.8946	15.83
BWT	0.8751	15.62	LE	0.8076	16.21	BWT	0.8948	16.04
FR	0.8715	15.71	PE	0.8052	16.21	NBRW	0.9005	16.25
LP	0.8738	15.96	NBRW	0.8013	16.33	N2V	0.8946	16.42
PE	0.8810	16.25	BWT	0.7967	16.46	RWE	0.8892	16.46
BRW	0.8723	16.33	BRW	0.7964	17.08	PE	0.8968	16.62
IM	0.8770	16.42	MHRW	0.8126	17.29	KC	0.9016	16.79
FF	0.8755	16.88	LP	0.8006	17.62	LP	0.8986	17.08
SB	0.8743	16.92	FR	0.7977	17.71	FR	0.8953	17.54
KC	0.8822	17.00	RE	0.8079	18.33	FF	0.8997	17.58
RWE	0.8650	17.08	FF	0.8045	18.62	RE	0.8989	19.08
SC	0.8701	18.25	RWE	0.7935	18.67	SC	0.8907	19.25
AC	0.8517	18.50	CSP	0.8087	18.92	MHRW	0.9019	19.38
MHRW	0.8816	18.88	AC	0.7626	18.92	CSP	0.9017	19.58
RE	0.8814	19.04	DM	0.7665	18.96	AC	0.8752	19.71
CBE	0.8585	20.00	SB	0.7998	19.08	CBE	0.8933	19.88
CSP	0.8784	20.25	SC	0.7844	19.38	SB	0.8975	20.21
DM	0.8376	20.96	CBE	0.7843	19.75	DM	0.8537	20.29
VC	0.7487	21.17	VC	0.6545	22.71	BRWE	0.8067	20.50
BRWE	0.7980	21.71	BRWE	0.7278	22.96	VC	0.7482	22.42
LV	0.7683	24.62	LV	0.6390	26.79	LV	0.7678	26.71
Baseline	0.8721	-	-	0.7933	-	-	0.8940	-

- The range of rankings is relatively narrow, from the highest at 11 to the lowest at 26, indicating close competition among the techniques, with most being relatively close to the baseline values.

Table 7 shows the percentage distribution of distance metrics derived from the best-performing results for each dataset. Cosine similarity displays a higher discriminative percentage in smaller datasets such as Heart Disease, Diabetes, and German, demonstrating its effectiveness in smaller-dimensional spaces. Conversely, the Euclidean metric is predominant in the Yeast and Titanic datasets, suggesting its ability to capture the intrinsic structures

Table 7 Percentage of Distance Metrics Distribution over Small Datasets

Dataset	# Of instances	Cosine	Euclidean	Manhattan
Heart Disease	303	0.4571	0.3286	0.2143
HCC	615	0.3429	0.3714	0.2857
Stroke Prediction	749	0.3571	0.3714	0.2714
Diabetes	768	0.4000	0.2571	0.3429
German Credit	1000	0.4143	0.3000	0.2857
Contraceptive	1473	0.3857	0.2571	0.3571
Yeast	1484	0.2188	0.4531	0.3281
Car	1728	0.3286	0.3000	0.3714
Titanic	2201	0.1857	0.4857	0.3286
Segment	2310	0.2857	0.3143	0.4000
Splice	3190	0.3571	0.3714	0.2714
Chess End-Game	3196	0.3857	0.2857	0.3286
Abalone	4168	0.3906	0.3125	0.2969
Spam-base	4601	0.4571	0.2571	0.2857
Banana	5300	0.2000	0.3143	0.4857
Phoneme	5404	0.2714	0.4143	0.3143
Page-Blocks	5473	0.2353	0.3529	0.4118
Texture	5500	0.2647	0.2647	0.4706
Opt-Digits	5620	0.3286	0.3714	0.3000
Satellite	6435	0.2941	0.2353	0.4706
Thyroid	7200	0.4571	0.3571	0.1857
Two-norm	7400	0.2647	0.2647	0.4706
Ring-norm	7400	0.2286	0.3143	0.4571
Coil-20	9822	0.3857	0.3286	0.2857

of these datasets. Manhattan distance shows notable discriminative percentages in datasets like Banana, Texture, Satellite, Two-norm, and Ring-norm, highlighting its reliability in certain contexts. These findings underscore the importance of selecting a distance metric tailored to the specific characteristics of the dataset, as no single "best" distance metric universally suits all datasets.

5.2 Large datasets

Table 8 presents the evaluation results for large datasets in terms of accuracy effectiveness (E_{AC}), F1 effectiveness (E_{F1}), reduction rate (R) and reduction time (T_R), alongside their respective rankings. Observing the table, we can point out the following insights:

- ML achieves the highest accuracy effectiveness with a top rank of 4.60, indicating it performs exceptionally well in terms of both accuracy and reduction rate.
- Close contenders such as RN, LP, and PR also demonstrate high accuracy effectiveness and reduction rates, showcasing their ability to maintain accuracy post-reduction.
- BWT, PE, and LP lead in F1 effectiveness with the highest ranks, suggesting their capability to balance precision and recall while reducing dataset size.
- ML ranks first in reduction rate with a score of 0.9916, highlighting its efficiency in reducing the dataset size while preserving essential information, with RN, FFE, TG, PR and LP following closely behind.
- Node and community detection-based sampling techniques appear to be top performers

Table 8 Average Test Effectiveness, Reduction Rate and Reduction Time Obtained by the Graph Reduction Techniques over Large Datasets

Method	$E_{AC} \uparrow$	Rank	Method	$E_{F1} \uparrow$	Rank	Method	$R \uparrow$	Rank	Method	$T_R \downarrow$	Rank
ML	0.8555	4.60	BWT	0.6508	6.80	ML	0.9916	6.80	BFS	0.0911	1.80
RN	0.8526	6.00	PE	0.6437	8.20	RN	0.9890	8.80	RN	0.0176	2.00
LP	0.8503	6.20	LP	0.6631	9.00	FFE	0.9853	8.80	NBRW	0.4465	3.40
PR	0.8509	6.80	FR	0.6716	10.60	TG	0.9810	9.20	SRW	0.7608	4.20
LD	0.8500	8.80	CM	0.6290	11.20	PR	0.9871	9.60	DG	0.1291	5.00
FG	0.8472	9.80	FG	0.6606	11.80	LP	0.9796	9.80	BRW	13.9121	7.00
TG	0.8486	10.60	DG	0.6504	12.80	VC	0.9163	10.80	FR	94.7064	9.60
DG	0.8380	11.40	RN	0.6673	13.00	LD	0.9864	11.40	FF	179.5922	9.80
FR	0.8450	12.60	N2V	0.5391	13.80	FR	0.9809	11.60	PE	15.9691	10.20
AC	0.8375	14.00	CBE	0.5961	14.00	SRW	0.9707	12.60	PR	10.7218	10.60
CM	0.8113	14.40	AC	0.6393	14.00	AC	0.9533	13.00	RE	25.0594	11.20
FF	0.8004	14.60	IM	0.6151	14.40	LV	0.7327	13.80	LD	20.5830	13.20
BFS	0.8249	15.00	ML	0.6688	14.80	FG	0.9803	13.80	LP	24.5948	15.00
IM	0.7923	16.00	LD	0.6575	15.00	DG	0.9803	15.40	BRWE	246.5536	15.60
LE	0.6474	16.40	RE	0.6197	15.20	FF	0.9371	16.20	KC	25.9202	16.00
BWT	0.8004	17.00	BFS	0.6443	15.60	DM	0.8398	17.00	ML	20.7337	17.60
SRW	0.7857	17.40	PR	0.6637	15.80	BRW	0.9617	17.40	MHRW	249.7027	17.80
PE	0.8197	18.20	TG	0.6626	16.20	CBE	0.9190	17.40	RWE	592.0134	18.80
VC	0.7588	18.60	SB	0.6054	17.20	IM	0.9095	18.40	TG	67.0947	20.00
CBE	0.7746	18.60	SRW	0.6043	17.20	BFS	0.9371	18.40	FG	156.1750	21.40
DM	0.7226	19.80	LE	0.4659	18.00	CM	0.9458	19.60	SB	2599.3683	22.00
RE	0.7704	20.20	FF	0.6097	18.40	LE	0.7752	20.00	CM	191.7594	22.80
FFE	0.7651	20.60	SC	0.5895	19.00	PE	0.9424	21.00	FFE	874.5884	22.80
BRW	0.7622	21.00	SG	0.5350	20.40	BWT	0.9210	21.60	CSP	365.6283	23.60
SG	0.6666	22.00	BRWE	0.5570	21.00	BRWE	0.9420	21.80	CBE	407.7802	23.60
SC	0.7638	23.60	CSP	0.5679	21.20	RE	0.8749	22.00	VC	209.4701	23.80
SB	0.7555	23.80	NBRW	0.5809	22.40	NBRW	0.9546	22.40	LE	239.4902	24.25
NBRW	0.7469	24.40	DM	0.5740	22.40	SB	0.8357	22.80	LV	289.9625	24.75
BRWE	0.7404	24.40	BRW	0.5798	23.20	SC	0.8918	23.60	BWT	1325.5210	26.40
LV	0.5793	25.80	FFE	0.5606	24.40	SG	0.7513	25.20	SG	2001.7654	28.20
CSP	0.6754	26.60	VC	0.4695	26.60	RWE	0.6550	26.00	AC	361.6786	28.40
N2V	0.6770	27.20	MHRW	0.4068	29.00	CSP	0.7861	28.20	IM	442.5221	29.60
RWE	0.4645	27.60	RWE	0.3617	30.80	N2V	0.7582	29.20	SC	1561.0224	30.60
MHRW	0.4672	31.00	LV	0.3320	31.60	MHRW	0.5120	31.40	DM	11093.0091	31.80
KC	0.0000	34.60	KC	0.0000	34.40	KC	0.0000	34.60	N2V	3175.3562	33.00

in terms of accuracy effectiveness and reduction rate, indicating their effectiveness. In contrast, there is a diverse set of techniques among the top contenders for F1 effectiveness.

- BFS and RN stand out for graph reduction speed, holding the top ranks, which is crucial for time-sensitive large-scale data applications. It is also observable that the top-performing techniques in small datasets are also top performers for large datasets.
- There is a huge range difference in terms of graph reduction time rankings, ranging from 1 to 33, indicating the difference in complexity and efficiency of techniques.

Table 9 presents the evaluation results for large datasets, detailing performance metrics such as accuracy (AC), F1 score ($F1$), and ROC-AUC score (AUC), along with their respective rankings and baseline values. Key insights derived from the table include:

- MHRW takes the lead in accuracy with an impressive score and a top rank of 2.00, demonstrating its effectiveness in large datasets. However, this high accuracy might primarily be due to the low reduction rates achieved by the technique. Similarly, SG and CSP also show strong accuracy performance, securing the 2nd and 3rd ranks, respectively, but these are accompanied by low reduction rates as indicated in Table 8. This trend is observed in the F1 metric as well. It presents an interesting pattern for large datasets:

Table 9 Average test performance metrics obtained by the graph reduction techniques over large datasets

Method	AC ↑	Rank	Method	$F1$ ↑	Rank	Method	AUC ↑	Rank
MHRW	0.9343	2.00	MHRW	0.7936	4.60	N2V	0.9393	5.60
SG	0.9252	8.00	BWT	0.7621	7.80	RE	0.9501	8.60
CSP	0.9341	8.20	CSP	0.7945	8.00	IM	0.9450	9.20
BWT	0.9203	9.20	N2V	0.7490	8.00	CM	0.9465	9.60
PE	0.9349	9.80	SG	0.7704	9.60	BWT	0.9411	10.00
RE	0.9351	10.60	SB	0.7673	9.80	CSP	0.9514	10.60
N2V	0.9120	10.60	PE	0.7956	10.40	MHRW	0.9500	10.80
IM	0.9249	10.80	CM	0.7725	11.40	PE	0.9518	11.20
SB	0.9241	11.40	RE	0.7959	11.60	SG	0.9344	12.60
BFS	0.9251	12.20	IM	0.7713	13.20	SB	0.9450	13.20
CM	0.9250	12.60	BFS	0.7712	15.20	SC	0.9134	15.40
SC	0.9099	14.20	CBE	0.6999	16.00	DG	0.9453	16.00
DG	0.9227	14.40	DG	0.7657	16.00	BFS	0.9505	16.40
FG	0.9237	15.20	FR	0.7557	16.20	FF	0.9468	16.60
RWE	0.8745	15.40	SC	0.7230	16.20	LE	0.7410	16.80
CBE	0.8722	15.60	RWE	0.7317	16.60	LD	0.9469	17.20
AC	0.8890	17.60	AC	0.6871	17.20	CBE	0.9252	17.60
LE	0.7173	17.80	FG	0.7651	17.60	RWE	0.9274	18.20
LD	0.9259	19.00	LP	0.7757	18.00	AC	0.9140	18.80
FF	0.9240	19.40	FF	0.7714	18.40	NBRW	0.9249	19.20
FR	0.9194	19.80	RN	0.7722	19.00	TG	0.9439	19.80
LP	0.9256	20.00	BRWE	0.7324	21.20	FG	0.9452	19.80
PR	0.9279	20.60	TG	0.7712	21.60	PR	0.9468	20.00
RN	0.9271	21.20	LE	0.5495	22.00	LP	0.9478	20.20
TG	0.9246	21.80	PR	0.7792	22.20	DM	0.9238	20.40
VC	0.8390	21.80	ML	0.7668	22.80	FR	0.9444	21.40
BRWE	0.8751	22.80	DM	0.6991	23.00	RN	0.9475	21.80
DM	0.8915	23.80	LD	0.7709	23.00	VC	0.8433	23.40
ML	0.9216	23.80	VC	0.5442	23.40	BRWE	0.9267	24.00
NBRW	0.8706	27.20	SRW	0.7013	24.80	SRW	0.9188	24.00
SRW	0.8693	27.80	BRW	0.6867	25.60	ML	0.9352	24.60
BRW	0.8606	28.60	NBRW	0.7105	26.00	BRW	0.9195	26.80
FFE	0.7936	30.20	FFE	0.5915	30.00	FFE	0.8588	26.80
KC	0.9343	31.60	KC	0.7943	31.40	KC	0.9493	30.80
LV	0.6441	32.00	LV	0.3952	32.20	LV	0.6290	32.60
Baseline	0.9395	-	-	0.7870	-	-	0.9387	-

lower reduction rates often correlate with higher accuracy or F1 scores, which is an expected outcome.

- LD, FG, and DG are among the techniques that achieve moderate accuracy rankings while obtaining high accuracy effectiveness rankings, indicating a balanced trade-off between data reduction and maintaining performance.
- N2V exhibits the best discriminative ability with an AUC rank of 5.60, indicating its potential to distinguish between classes effectively. RE and IM follow closely, showcasing their efficiency in model classification post-graph reduction.
- Although LV and FFE achieved high reduction ranks, their corresponding performance metrics rank among the lowest, suggesting that while they reduce data significantly, they may compromise too much on the essential information needed for accurate classification.

Figure 4 displays the distribution of the selected number of neighbours for constructing k-neighbours graphs in large datasets. The average number of neighbours used across datasets is approximately 26, indicating a preference for moderately dense neighbourhoods. The range of selected neighbours varies from 2 to 50, with a standard deviation of roughly 15, which implies considerable variation in the chosen number of neighbours across different datasets. This variation in neighbourhood size is influenced by the specific characteristics of each dataset, underscoring the necessity of parameter tuning to optimize performance.

6 Discussions

In this discussion, we discuss graph reduction techniques applied across various datasets in our study. We aim to analyze the impact of these techniques on graph build and reduction times and explore how different data reduction rates influence test accuracy across a selection of techniques. Additionally, we will provide a visual case study of the reduced datasets to illustrate the practical effects of graph reduction.

Figure 5 presents box plots of graph build times across datasets and their respective number of instances, with the background differentiated by the method used to build the graph,

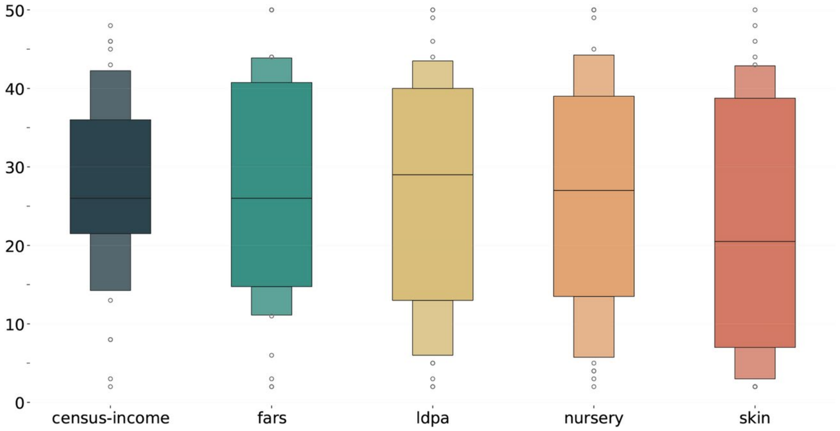


Fig. 4 Distribution of number of neighbours for k-Neighbours graph over large datasets

either through a similarity matrix or a KNN graph. There are minimal variations in graph build times for smaller datasets, particularly those with fewer than 4000 instances, with most being constructed in mere seconds. In contrast, larger datasets show a noticeable variation in build times, ranging from 40 to 120 s. However, the median build time for smaller datasets remains below 40 s. This considerable variability in build time for smaller datasets primarily stems from two factors: (1) the choice of distance metric and (2) the applied similarity threshold. The choice of distance metric significantly impacts the similarities detected between instances in each dataset, as we have observed that certain metrics perform better for some datasets, more effectively identifying similarities between instances. For graph building using distance metrics, we applied a “similarity threshold,” a percentile threshold ranging from 0.5 to 0.999. This threshold eliminates any nodes below this value based on the weights computed by the distance metrics, thereby speeding up the graph construction while removing less significant nodes due to their lower similarities in the dataset.

Regarding constructing graphs for large datasets, we opted for the k-Neighbours graph, primarily due to the computational and memory limitations of using distance metrics for large datasets. We acknowledge that using the KNN graph can be a significant limitation as it may not capture complex relationships between instances as effectively as distance metrics. However, given the constraints on time and resources, we believed this to be the best option. Surprisingly, the graph build time for large datasets using the KNN graph is considerably quick, with most datasets exhibiting a median build time of less than 20 s. The only dataset that experienced a notably high build time was the Census-Income dataset, which also has a relatively low median build time. We attribute this anomaly to the dataset’s many features (i.e., 41 features), likely contributing to the increased build time.

We acknowledge that one of the limitations of adopting graph techniques for non-graph-related tasks is the overhead incurred during graph construction. This may initially seem counterintuitive. However, it can be argued that graph construction is a one-time process, and if the results achieved are superior to those of non-graph techniques, then the use of graph methods becomes justifiable. Furthermore, with the multi-core capabilities of modern CPUs, it is possible to construct graphs more efficiently by dividing the datasets into overlapping, randomly selected chunks. This approach assumes that each chunk will somewhat represent the entire dataset. Our extensive evaluations have shown that the time required for construction can be significantly reduced with the appropriate selection of distance metrics and similarity thresholds.

Figure 6 displays box plots of graph reduction times across different graph reduction techniques, categorized by their methodological approach. To enhance visualization clarity, obvious outliers have been excluded from the figure, and the presence of outliers is indi-

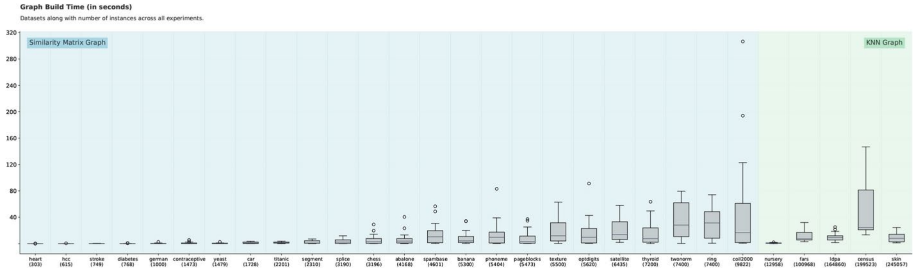


Fig. 5 Box plots of graph build times across datasets and their number of instances

cated by the following symbols next to the technique name: † (dagger for 30 min), * (star for 1 h), ** (two stars for 2 h), and *** (three stars for over three hours). We observe that node sampling, edge sampling, graph traversal, and graph sparsification techniques generally require the least time for reduction, with the exceptions of TG and RWE. These exceptions might be attributed to specific computational inefficiencies inherent to their algorithms.

In contrast, graph coarsening techniques exhibit comparable reduction times among themselves, which is expected due to the computational demands of identifying and merging nodes. Community detection-based sampling methods show varied reduction times; FG, LP, LE, LD, and ML have reduction times comparable to other techniques. However, BWT, DM, IM, and SG exhibit significantly higher reduction times. The extensive computation BWT requires to execute hierarchical clustering, DM to detect overlapping communities dynamically, and SG to optimise a complex energy function explains their prolonged durations. Node2Vec also demonstrates a high reduction time, mainly due to the extensive computations involved in generating walks and calculating transition probabilities, which are computationally intensive.

The shorter reduction times observed in node sampling, edge sampling, and graph traversal techniques can primarily be attributed to their simpler operational mechanisms. Specifically for graph traversal techniques, the process continues just until the desired sample size is reached, often resulting in quicker completion.

To better illustrate how reduction techniques perform over different reduction rates, we selected three varied datasets: German, Opt-Digits, and Census-Income, along with eight graph reduction techniques known for their fast reduction times across different categories: BFS, DG, LD, ML, PE, RE, RN, and SRW. Figure 7 presents a line chart of test accuracy across varying reduction percentages on the German dataset for the selected eight techniques, with the baseline value indicated by a red dashed line. We observe that for almost all techniques, up to a 90% reduction rate, the accuracy exceeds the baseline, reaching above

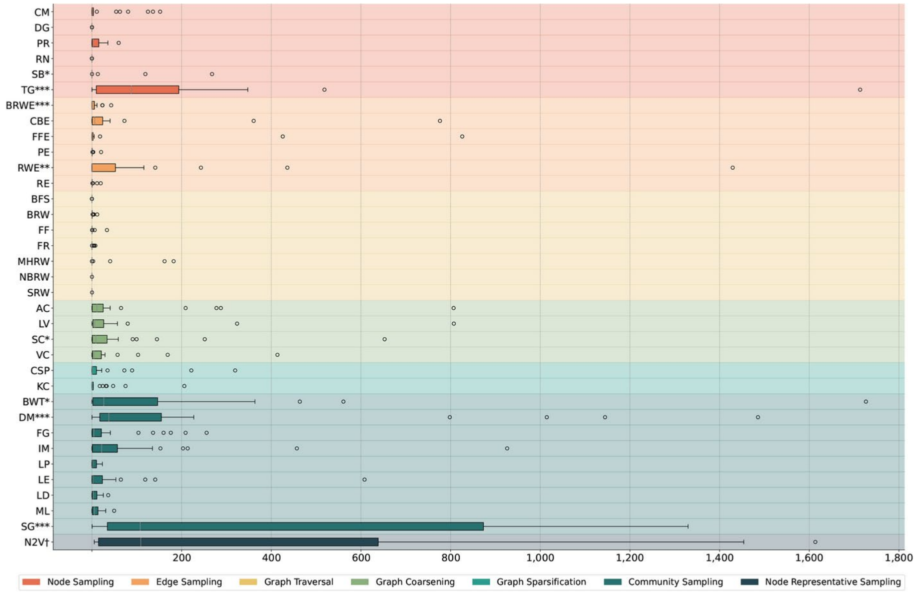


Fig. 6 Box plots of graph reduction times (in seconds) across graph reduction techniques

0.8. However, beyond the 90% reduction rate, some techniques, such as ML, LD, and DG, begin to outperform others, with LD maintaining accuracy above the baseline even at a 95% reduction rate. SRW sampling exhibits the poorest performance from the outset, even at lower reduction rates, and PE displays a highly erratic test accuracy at higher reduction rates.

Figure 8 shows a line chart of test accuracy across reduction percentages on the Opt-Digits dataset for the selected techniques. Unlike the German dataset, none of the techniques in this dataset achieve higher test accuracy at any reduction rate. Notably, PE and DG experience significant accuracy declines as the reduction rate increases, particularly after the 70% mark. However, other techniques show a more gradual reduction in accuracy as the reduction rate increases, with techniques such as LD, ML, and SRW experiencing acceptable declines even above a 95% reduction rate.

Figure 9 illustrates a line chart of test accuracy across reduction rates on the large Census-Income dataset for the selected reduction techniques. Here, we observe a gradual decrease in accuracy across techniques as the reduction rate increases, compared to the Opt-Digits dataset. No techniques show a drastic drop-off in performance; instead, they track closely with each other. Techniques such as BFS, PE, SRW, LD, and ML display slightly better performance at high reduction rates compared to others.

Figure 10 presents a line chart showing test accuracy across various reduction rates on the Abalone dataset for selected reduction techniques. An important characteristic of the Abalone dataset is its composition of 21 target classes coupled with significant class imbalance, which typically challenges models to achieve satisfactory results. From the results, it is evident that the highest reduction rate achieved was 90% by the LD sampling technique. Techniques such as ML and RN consistently outperformed the baseline in terms of accuracy, but their effectiveness only reaches up to 80% reduction rate. Conversely, PE, SRW, and DG underperformed, while RE only managed to achieve a maximum reduction rate of about 35%. Although these visualizations do not encompass all reduction techniques and

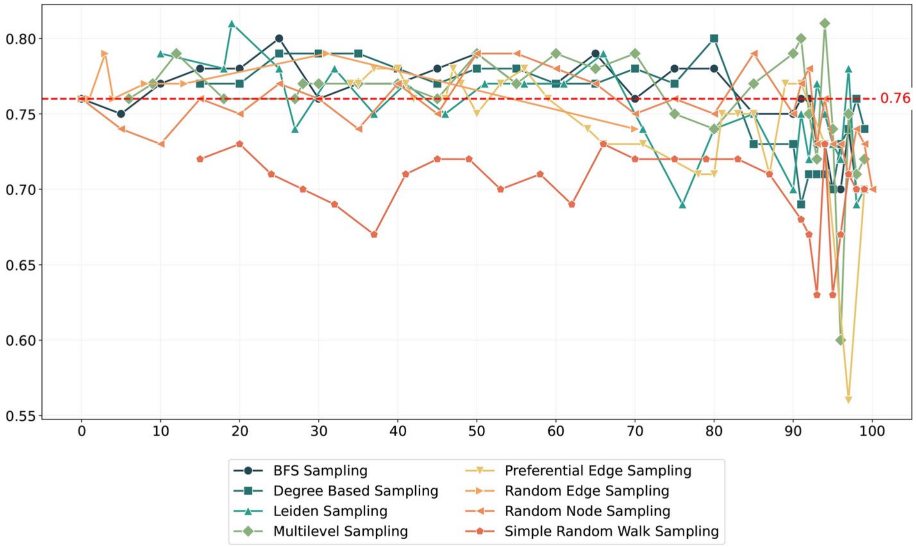


Fig. 7 Test accuracy over reduction percentages for the German dataset

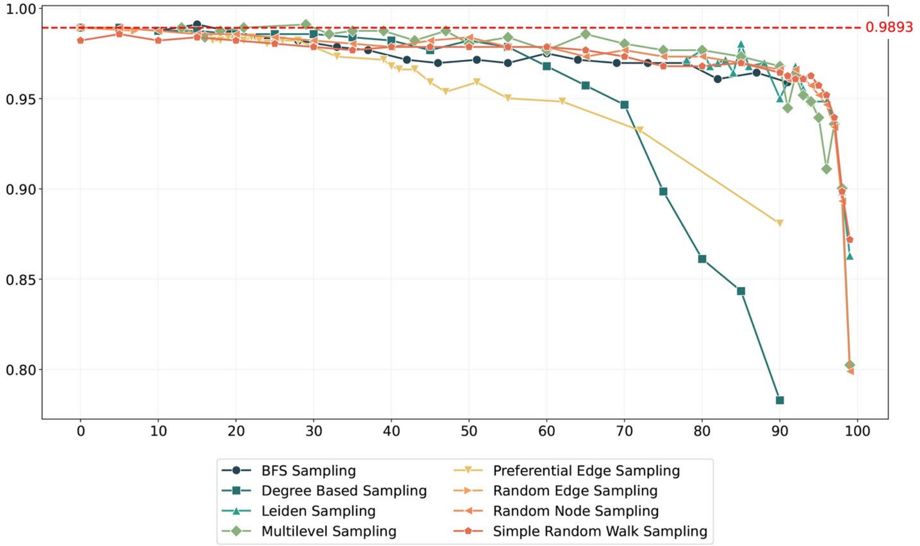


Fig. 8 Test accuracy over reduction percentages for the opt-digits dataset

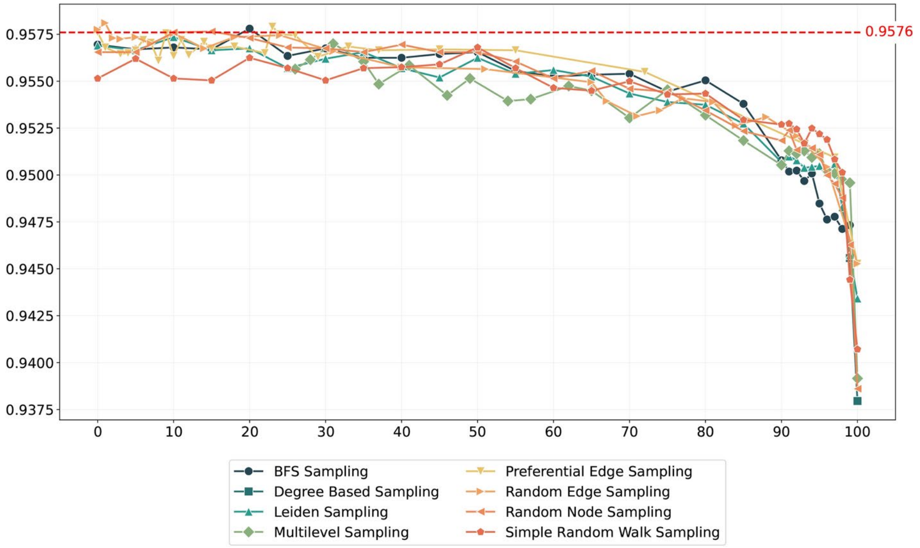


Fig. 9 Test accuracy over reduction percentages for the census income dataset

datasets, they serve to illustrate how some techniques perform across different reduction rates on selected datasets.

Table 10 presents the best-performing graph reduction techniques for each dataset considering effectiveness, reduction rate and efficiency score metrics. The following insights can be derived:

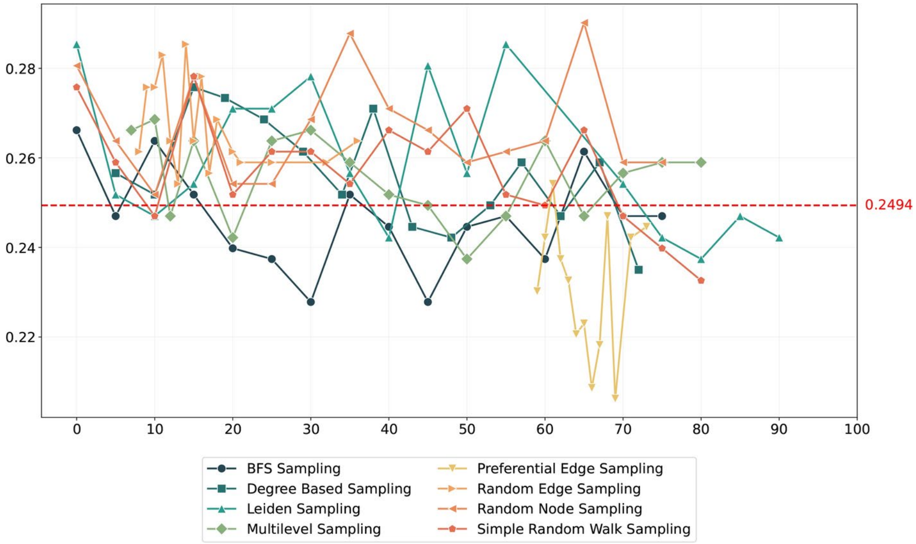


Fig. 10 Test accuracy over reduction percentages for the abalone dataset

- No single technique consistently outperforms others across all datasets and metrics, highlighting the importance of tailored selection based on specific dataset characteristics and performance priorities.
- For accuracy effectiveness (E_{AC}), techniques such as LD, FF, FG, SC, and LP frequently appear as the best performers across multiple datasets. This suggests that these techniques effectively preserve accuracy while achieving significant reduction rates. Random walk techniques show potential for smaller datasets, while community detection-based techniques perform better for larger datasets.
- In terms of F1 effectiveness (E_{F1}), techniques like CM, AC, and LD are often the top performers. These techniques strike a good balance between precision and recall, even after graph reduction.
- When considering the reduction rate (R), techniques such as NBRW, AC, and LV consistently achieve high reduction percentages across various datasets.
- For the efficiency score (ES), which balances test effectiveness with reduction time, techniques like FF, SRW, DG, and FFE frequently appear as the best performers. Although the efficiency score utilizes the accuracy test effectiveness, we can observe that in some datasets, the best-performing techniques are different from those of the test effectiveness techniques, denoting that when considering the reduction time, these techniques offer desirable performance.
- Datasets like Skin Segmentation, Ring-norm, Two-norm, and Thyroid show very high-efficiency scores, suggesting that graph reduction techniques are particularly effective for these types of data. Conversely, datasets such as Abalone and Yeast present lower scores, indicating potential challenges in domains with high-class numbers or significant imbalance ratios.
- Medical datasets often see strong performance from random walk-based techniques, while image-related datasets benefit from local structure-preserving methods. This sug-

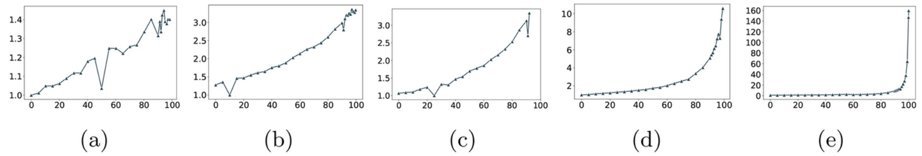
Table 10 Best performing techniques on each dataset over different evaluation metrics

Dataset	Technique	$E_{AC} \uparrow$	Technique	$E_{F1} \uparrow$	Technique	$R \uparrow$	Technique	$ES \uparrow$
Heart Disease	DM	90.0978	SRW	90.9608	NBRW	99.1803	SRW	180.1957
HCC	BRW	94.8238	BRW	89.9531	NBRW	99.5976	BRW	189.4485
Stroke Prediction	AC	82.0594	CM	73.5132	BFS	99.3399	DG	160.9594
Diabetes	RWE	77.4202	AC	64.6113	BRW	99.5169	RWE	154.8403
German Credit	FFE	80.1778	FFE	69.3907	BRWE	99.7531	FFE	160.1373
Contraceptive	FG	58.2884	FG	56.4510	BRWE	99.4966	FG	115.5161
Yeast	SC	54.7009	CBE	45.8618	AC	94.6533	SC	108.6541
Car Evaluation	FR	81.4620	LD	71.4067	NBRW	97.9986	FR	162.8041
Titanic	FFE	79.3252	FFE	61.9406	CM	99.8878	FFE	158.4733
Segment	MHRW	87.6051	SRW	87.4981	AC	97.5949	MHRW	175.1908
Splice	NBRW	86.7632	NBRW	86.8173	CSP	98.7224	NBRW	173.4221
Chess End-Game	VC	92.6608	VC	92.8907	SG	98.1453	FF	182.2280
Abalone	FG	22.3781	IM	15.6953	SC	87.5556	SC	42.2105
Spam-base	SC	89.6075	SC	86.8430	SB	99.1412	LD	177.5904
Banana	ML	87.5626	ML	86.6573	AC	99.9534	FR	169.3833
Phoneme	DG	81.4844	DG	71.8915	RWE	99.9543	DG	162.9590
PageBlocks	FF	94.6240	CM	80.4888	BRW	99.0523	FF	189.1794
Texture	LD	88.4935	LD	88.4926	LV	99.4613	RN	175.0793
Opt-Digits	VC	91.2191	VC	91.2054	FFE	98.0448	RN	178.4518
Satellite	FF	83.5231	AC	80.5294	NBRW	99.5970	FF	166.5468
Thyroid	LP	97.4656	LP	94.7947	LV	99.8628	RE	191.7635
Two-norm	AC	96.5181	AC	96.5036	SRW	99.8165	SB	191.8924
Ring-norm	LD	96.5019	LD	96.5376	AC	99.5162	BRW	190.5507
Coil-20	BFS	94.0287	BFS	3.3308	BFS	99.9246	BFS	188.0574
Nursery	SG	89.5498	CM	84.4306	LV	98.7232	FF	177.6322
Fars	ML	75.2371	FR	52.4768	SRW	97.5594	ML	149.7884
LDPA	LD	70.9745	PE	51.7703	VC	99.4226	LD	141.4242
Census-Income	LP	94.1581	CBE	45.4967	PR	99.9691	SRW	188.2451
Skin Segmentation	PE	99.2719	PE	99.4597	FFE	99.8081	PE	198.4351

gests that the underlying data structure influences the effectiveness of different reduction techniques.

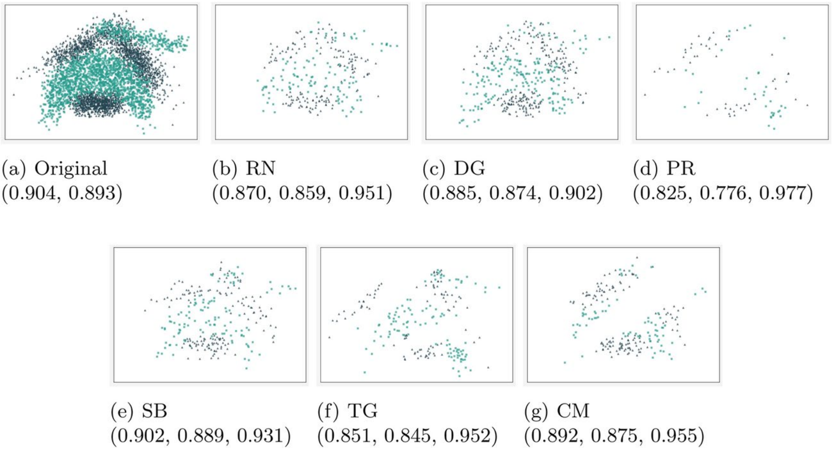
- For larger datasets (e.g., Census-Income), techniques that efficiently handle scale, such as LP and LD, tend to perform well, indicating their potential for big data applications.

Figure 11 presents a series of line plots illustrating model training time speedup over various reduction percentages across five datasets: Heart (11a), German (11b), Yeast (11c), Opt-Digits (11d), and Census Income (11e). Speedup is defined as the baseline training time (without graph reduction) divided by the training time on reduced datasets, with a higher value indicating greater efficiency. Observations show that the greater the data reduction, the higher the speedup, especially in larger datasets such as Census-Income, where the speedup reaches 160 times. This significant increase highlights the critical role of instance selection in enabling data analysis and modeling, particularly in scenarios where large datasets may otherwise hinder these processes.



(a) (b) (c) (d) (e)

Fig. 11 Model training time speed up over reduction percentages for datasets: **a** Heart; **b** German; **c** Yeast; **d** Opt-Digits; **e** Census-Income



(a) Original (0.904, 0.893)

(b) RN (0.870, 0.859, 0.951)

(c) DG (0.885, 0.874, 0.902)

(d) PR (0.825, 0.776, 0.977)

(e) SB (0.902, 0.889, 0.931)

(f) TG (0.851, 0.845, 0.952)

(g) CM (0.892, 0.875, 0.955)

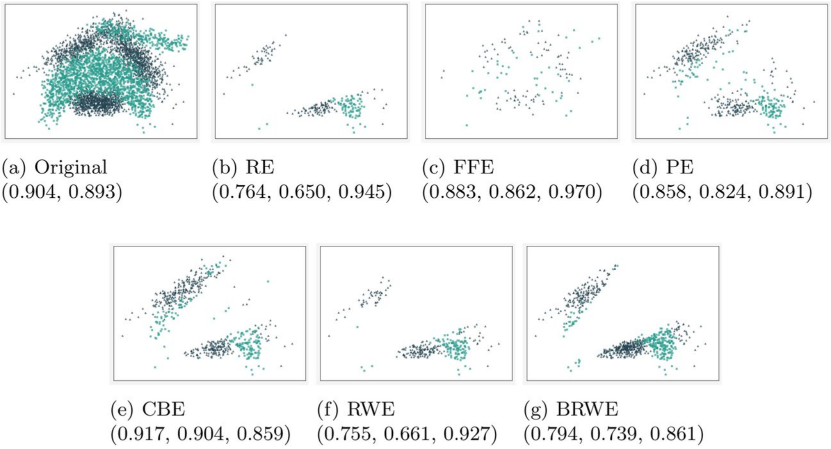
Fig. 12 Banana data subsets for node sampling techniques

6.1 Visualization of data subsets: a case study based on the banana dataset

We present a case study using the Banana dataset to understand better how each graph reduction technique impacts the data visually. This artificial dataset comprises two features, two classes, and 5300 instances arranged in well-defined clusters with some overlap between classes, making it particularly suitable for visualization. Each visualization displays the original training set alongside the graph reduction techniques in that category. The individual captions show the test accuracy, F1 score, and reduction rate in parentheses.

Figure 12 showcases visualizations of node sampling techniques: RN, DG, PR, SB, TG, and CM. It is observable that techniques such as RN, DG, and SB successfully preserve the original dataset's structure while reducing the dataset by up to 95%. This preservation is evident as, when the sampling technique retains the underlying structure, the performance metrics are comparable to those of the original set. Conversely, techniques like CM, PR, and TG show failures in capturing the essential structure, with TG notably displaying a skewed representation favouring one class over the other.

Figure 13 illustrates the visualizations for edge sampling techniques: RE, FFE, PE, CBE, RWE, and BRWE. In these visualizations, it is evident that there is a significant division of data into upper and lower halves, with a substantial gap between these clusters. This pattern indicates that edge sampling techniques prioritise the data's peripheral or boundary instances rather than those at the centre. As a result, these techniques generally fail


(a) Original
(0.904, 0.893)

(b) RE
(0.764, 0.650, 0.945)

(c) FFE
(0.883, 0.862, 0.970)

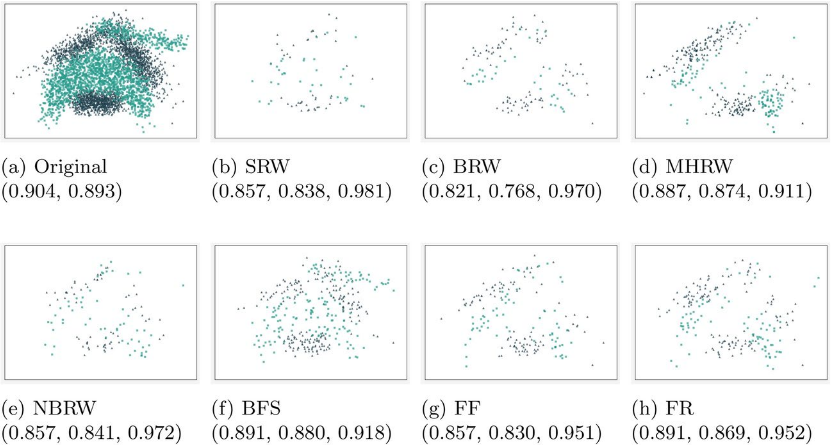
(d) PE
(0.858, 0.824, 0.891)

(e) CBE
(0.917, 0.904, 0.859)

(f) RWE
(0.755, 0.661, 0.927)

(g) BRWE
(0.794, 0.739, 0.861)

Fig. 13 Banana data subsets for edge sampling techniques


(a) Original
(0.904, 0.893)

(b) SRW
(0.857, 0.838, 0.981)

(c) BRW
(0.821, 0.768, 0.970)

(d) MHRW
(0.887, 0.874, 0.911)

(e) NBRW
(0.857, 0.841, 0.972)

(f) BFS
(0.891, 0.880, 0.918)

(g) FF
(0.857, 0.830, 0.951)

(h) FR
(0.891, 0.869, 0.952)

Fig. 14 Banana data subsets for graph traversal techniques

to capture the original dataset's structure effectively, except for FFE. Surprisingly, CBE achieved even higher accuracy and F1 scores than the baseline, although the reduction rate was limited to 85

Figure 14 displays the visualizations of the graph traversal techniques: SRW, BRW, MHRW, NBRW, BFS, FF, and FR. It is observable that techniques such as SRW, BFS, NBRW, FF, and FR successfully capture the structure of the original dataset. In contrast, BRW and MHRW exhibit visual structures similar to those seen with edge sampling techniques, highlighting their less effective representation of the dataset's central connections. The visualization for BFS sampling appears as a sparsified version of the original dataset, maintaining an almost identical structure but with fewer instances. Meanwhile, random walking techniques (SRW, MHRW, NBRW) generally retain the overall structure well but

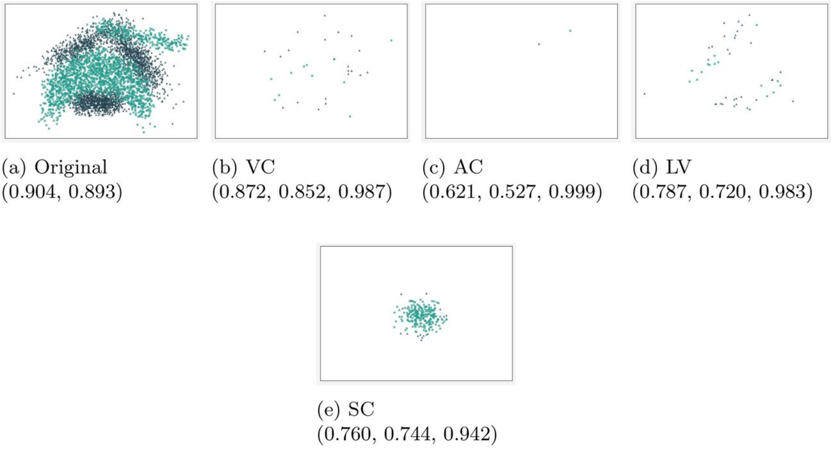


Fig. 15 Banana data subsets for graph coarsening techniques

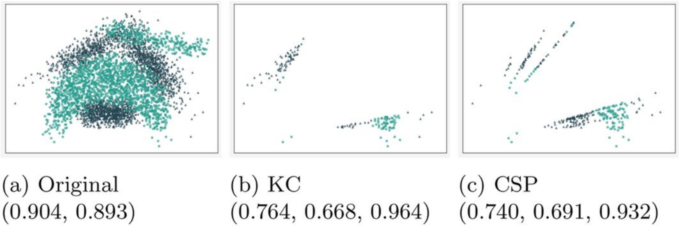


Fig. 16 Banana data subsets for graph sparsification techniques

have the drawback of capturing overlaps between classes, which include noisy instances, thereby potentially diminishing the clarity of the reduced dataset.

Figure 15 presents the visualisations for graph coarsening techniques: VC, AC, LV, and SC. Right off the bat, it is observable that SC struggles to fully capture the structural complexity of the original dataset. As a clustering technique, SC reduces the data into a circular cluster, which manifests a clear class imbalance. The techniques in this category achieve some of the highest reduction rates, with AC, for example, reducing to only two instances, consequently negatively impacting performance. On the other hand, VC demonstrates promising results with a 99% reduction rate, successfully preserving the essential structure of the original dataset and offering performance comparable to the baseline.

Figure 16 presents the visualizations for graph sparsification techniques: KC and CSP. These methods achieve high reduction rates but struggle to preserve the data's original structure. The resulting structure is similar to that observed with edge sampling techniques, which leads to comparably low-performance measures.

Figure 17 shows the visualizations for community detection-based sampling techniques: BWT, IM, LE, SG, ML, LP, LD, DM and FG. In this group, all techniques except DM successfully capture the underlying structure of the data. The failure of DM to perform comparably could be attributed to its focus on detecting overlapping communities, which may not effectively delineate the distinct clusters required for this dataset. Among the rest of the

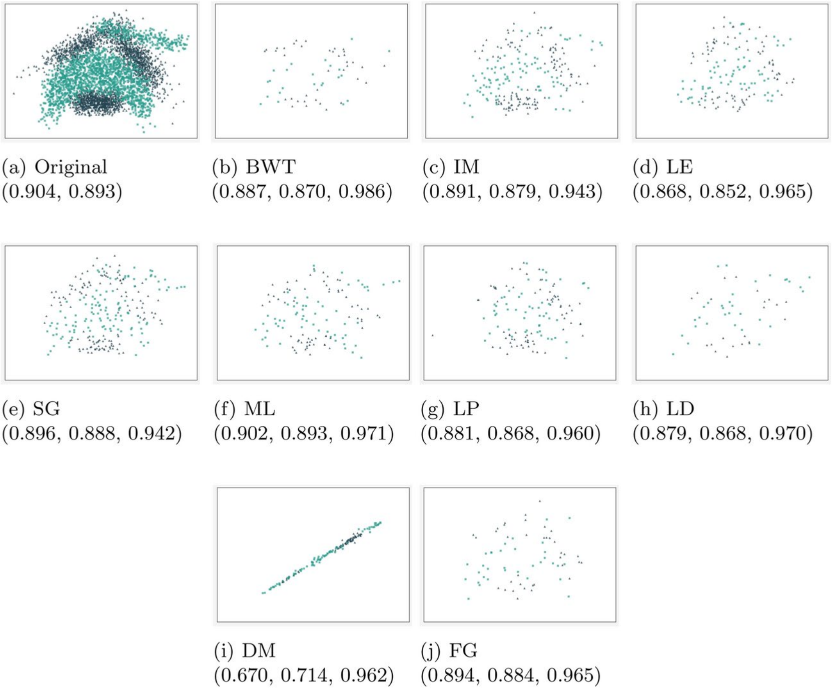
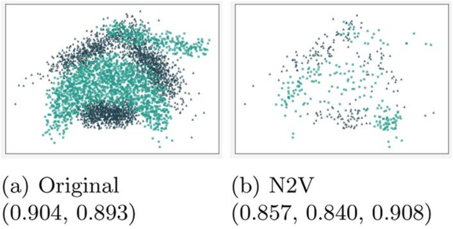


Fig. 17 Banana data subsets for community detection-based sampling techniques

Fig. 18 Banana data subsets for node representative sampling techniques



techniques, the performances are generally on par with the baseline, with reduction rates reaching up to 99%. Notably, ML, LD, and BWT exhibit promising structures in the reduced data with fewer overlaps and more distinct structural representations.

Figure 18 presents the visualization for the node representative sampling technique, N2V. Evidently, this technique successfully captures the overall structure of the data. However, there are noticeable overlaps among the classes, which could potentially introduce noise into the dataset. Nevertheless, this method shows a promising avenue for utilizing embeddings to improve instance selection.

7 Conclusions

In this study, we have conducted a comprehensive comparative analysis of graph reduction techniques for instance selection in machine learning tasks. Our goal was to leverage the strengths of graph-based methods and apply them to the relatively unexplored domain of instance selection. We identified, explained, and implemented 35 graph reduction techniques, which we categorized into seven distinct groups. To assess the performance of these techniques, we collected 29 datasets of varying sizes suitable for supervised machine learning classification tasks. We evaluated these techniques primarily based on effectiveness, reduction rate, accuracy, F1 score, and reduction time. We presented multiple visualizations comparing graph construction time, reduction time, and performance across reduction rates to enhance understanding. Most importantly, we included a case study that contrasts the original and reduced data for each technique using the Banana dataset.

Our study is not without limitations. It focuses exclusively on supervised learning classification datasets with clearly labelled instances. Each technique has been optimized primarily for accuracy effectiveness, which could limit performance on imbalanced datasets. We employed a KNN graph for large datasets, which may not adequately capture complex relationships between instances. Additionally, while the self-implemented techniques were developed in Python for feasibility, utilizing more performance-oriented programming languages such as C or C++ could yield more efficient results. Furthermore, incorporating larger datasets, possibly those containing millions of instances, could enhance the robustness and applicability of our findings.

Before concluding the paper, we offer some guidelines and suggest future directions for research in the field of graph reduction for instance selection. While several techniques generally performed well, it cannot be concluded that these techniques are universally the best. However, they can guide readers in selecting appropriate techniques for their specific problems. Future work could explore the application of distance metrics to large datasets by potentially processing the data in chunks with some overlaps to determine if this approach yields a significant difference compared to using a KNN graph. Additionally, this study could be expanded to evaluate how graph reduction techniques perform for instance selection in regression tasks, unsupervised learning tasks, and with other forms of data such as images, text, and sound. Moreover, there is potential to leverage the strengths of well-performing techniques to develop a new graph reduction method that combines the advantages of several techniques while overcoming their limitations.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10462-024-10971-4.

Author contributions ZR and NZ, AZ and RD contributed to the conceptual idea of the research work. CR, NZ, and JR contributed to the analysis and experimental work. ZR, JR, and NZ contributed to drafting the manuscript. All authors reviewed the manuscript and approved the submission of the paper.

Funding No funding was received for this study.

Data availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare that they have no potential conflicts or Conflict of interest.

References

Akiba T, Sano S, Yanase T, Ohta T, Koyama M (2024) Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 2623–2631. ACM. https://doi.org/10.1145/3292500.3330701 . https://dl.acm.org/doi/10.1145/3292500.3330701 Accessed 13 April 2024

Albelwi S, Mahmood A (2016) Analysis of instance selection algorithms on large datasets with deep convolutional neural networks. In: 2016 IEEE long Island systems, applications and technology conference (LISAT), pp. 1–5. IEEE. https://doi.org/10.1109/LISAT.2016.7494142 . http://ieeexplore.ieee.org/document/7494142/ Accessed 19 March 2024

Asuncion A, Newman D et al (2007) UCI machine learning repository. Irvine, CA, USA

Batagelj V, Zaversnik M (2003) An O(m) algorithm for cores decomposition of networks. arXiv. http://arxiv.org/abs/cs/0310049 Accessed 12 April 2024

Batson J, Spielman DA, Srivastava N, Teng S-H (2013) Spectral sparsification of graphs: theory and algorithms. Commun ACM 56(8):87–94. https://doi.org/10.1145/2492007.2492029

Blachnik M (2019) Ensembles of instance selection methods: a comparative study. Int J Appl Math Comput Sci 29(1):151–168. https://doi.org/10.2478/amcs-2019-0012

Blachnik M, Kordos M (2022) Comparison of instance selection and construction methods with various classifiers. Appl Sci 10(11):3933. https://doi.org/10.3390/app10113933

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech 2008(10):10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Bradley AP (1997) The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recogn 30(7):1145–1159. https://doi.org/10.1016/S0031-3203(96)00142-2

Cano JR, Herrera F, Lozano M (2003) Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. IEEE Trans. Evol. Computat. 7(6):561–575. https://doi.org/10.1109/TEVC.2003.819265

Chen J, Saad Y, Zhang Z (2022) Graph coarsening: from scientific computing to machine learning. SeMA J 79(1):187–223. https://doi.org/10.1007/s40324-021-00282-x

Chen Y, Ye H, Vedula S, Bronstein A, Dreslinski R, Mudge T, Talati N (2023) Demystifying graph sparsification algorithms in graph properties preservation. Proc VLDB Endow 17(3):427–440. https://doi.org/10.14778/3632093.3632106

Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70(6):066111. https://doi.org/10.1103/PhysRevE.70.066111. Accessed 3 April 2024

Coscia M, Rossetti G, Giannotti F, Pedreschi D (2014) Uncovering hierarchical and overlapping communities with a local-first approach. ACM Trans Knowl Discov Data 9(1):1–27. https://doi.org/10.1145/2629511. Accessed 3 April 2024

Coscia M, Rossetti G, Giannotti F, Pedreschi D (2024) DEMON: a local-first discovery method for overlapping communities. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 615–623. ACM. https://doi.org/10.1145/2339530.2339630 . https://dl.acm.org/doi/10.1145/2339530.2339630 Accessed 3 April 2024

Cunha W, Viegas F, França C, Rosa T, Rocha L, Gonçalves MA (2023) A comparative survey of instance selection methods applied to non-neural and transformer-based text classification. ACM Comput Surv 55(13):1–52. https://doi.org/10.1145/3582000

Derrac J, Garcia S, Sanchez L, Herrera F (2015) Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Mult Valued Logic Soft Comput 17:255–287

Eppstein D, Galil Z, Italiano GF, Nissenzweig A (1997) Sparsification-a technique for speeding up dynamic graph algorithms. J ACM 44(5):669–696. https://doi.org/10.1145/265910.265914

Fawcett T (2006) An introduction to ROC analysis. Pattern Recogn Lett 27(8):861–874. https://doi.org/10.1016/j.patrec.2005.10.010

Frey BJ, Dueck D (2007) Clustering by passing messages between data points. Science 315(5814):972–976. https://doi.org/10.1126/science.1136800

Gao X, Yu J, Jiang W, Chen T, Zhang W, Yin H (2024) Graph condensation: a survey. arXiv. http://arxiv.org/abs/2401.11720 Accessed 19 March 2024

Garcia S, Derrac J, Cano JR, Herrera F (2012) Prototype selection for nearest neighbor classification: taxonomy and empirical study. IEEE Trans Pattern Anal Mach Intell 34(3):417–435. https://doi.org/10.1109/TPAMI.2011.142

Hashemi M, Gong S, Ni J, Fan W, Prakash BA, Jin W (2024) A comprehensive survey on graph reduction: sparsification, coarsening, and condensation. arXiv. http://arxiv.org/abs/2402.03358 Accessed 19 March 2024

Hastings WK (1970) Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57(1):97–109. https://doi.org/10.1093/biomet/57.1.97

Leskovec J, Faloutsos C (2006) Sampling from large graphs. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 631–636. ACM. https://doi.org/10.1145/1150402.1150479 . https://dl.acm.org/doi/10.1145/1150402.1150479 Accessed 10 April 2024

Liu Y, Safavi T, Dighe A, Koutra D (2019) Graph summarization methods and applications: a survey. ACM Comput Surv 51(3):1–34. https://doi.org/10.1145/3186727

Malekipirbazari M, Aksakalli V, Shafqat W, Eberhard A (2021) Performance comparison of feature selection and extraction methods with random instance selection. Expert Syst Appl 179:115072. https://doi.org/10.1016/j.eswa.2021.115072

Malhat M, Menshawy ME, Mousa H, Sisi AE (2020) A new approach for instance selection: algorithms, evaluation, and comparisons. Expert Syst Appl 149:113297. https://doi.org/10.1016/j.eswa.2020.113297

Mazurowski MA, Malof JM, Tourassi GD (2011) Comparative analysis of instance selection algorithms for instance-based classifiers in the context of medical decision support. Phys Med Biol 56(2):473–489. https://doi.org/10.1088/0031-9155/56/2/012

Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21(6):1087–1092. https://doi.org/10.1063/1.1699114

Mohr N, Hürtgen H (2018) Achieving business impact with data. Digital McKinsey, New York

Newman MEJ (2006) Finding community structure in networks using the eigenvectors of matrices. Phys Rev E 74(3):036104. https://doi.org/10.1103/PhysRevE.74.036104. Accessed 3 April 2024

Olvera-López JA, Carrasco-Ochoa JA, Martínez-Trinidad JF, Kittler J (2010) A review of instance selection methods. Artif Intell Rev 34(2):133–143. https://doi.org/10.1007/s10462-010-9165-y

Ozturk Kiyak E, Ghasemkhani B, Birant D (2023) High-level K-nearest Neighbors (HLKNN): a supervised machine learning model for classification analysis. Electronics 12(18):3828. https://doi.org/10.3390/electronics12183828

Page L, Brin S, Motwani R, Winograd T et al (1999) The pagerank citation ranking: bringing order to the web

Pons P, Latapy M (2005) Computing communities in large networks using random walks. In: Yolum p, Güngör T, Gürgen F, Özturan C (eds.) Computer and information sciences - ISCIS 2005 vol. 3733, pp. 284–293. Springer. https://doi.org/10.1007/11569596_31. Series title: lecture notes in computer science. http://link.springer.com/10.1007/11569596_31 Accessed 3 April 2024

Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E 76(3):036106. https://doi.org/10.1103/PhysRevE.76.036106

Randall D, Tony RM (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38:257–286. https://doi.org/10.1023/A:1007626913721

Rehman SU, Khan AU, Fong S (2012) Graph mining: a survey of graph mining techniques. In: Seventh international conference on digital information management (ICDIM 2012), pp. 88–92. IEEE. https://doi.org/10.1109/ICDIM.2012.6360146 . http://ieeexplore.ieee.org/document/6360146/

Reichardt J, Bornholdt S (2006) Statistical mechanics of community detection. Phys Rev E 74(1):016110. https://doi.org/10.1103/PhysRevE.74.016110. Accessed 3 April 2024

Rosvall M, Bergstrom CT (2008) Maps of random walks on complex networks reveal community structure. Proc Natl Acad Sci USA 105(4):1118–1123. https://doi.org/10.1073/pnas.0706851105

Rosvall M, Axelsson D, Bergstrom CT (2009) The map equation. Eur Phys J Spec Top 178(1):13–23. https://doi.org/10.1140/epjst/e2010-01179-1

Rustamov J, Rustamov Z, Zaki N (2023) Green space quality analysis using machine learning approaches. Sustainability 15(10):7782. https://doi.org/10.3390/su15107782

Sestino A, Prete MI, Piper L, Guido G (2020) Internet of things and big data as enablers for business digitalization strategies. Technovation 98:102173. https://doi.org/10.1016/j.technovation.2020.102173

Shabani N, Wu J, Beheshti A, Sheng QZ, Foo J, Haghighi V, Hanif A, Shahabikargar M (2024) A comprehensive survey on graph summarization with graph neural networks, pp. 1–21 https://doi.org/10.1109/TAI.2024.3350545. Accessed 19 March 2024

Shah S, Shabbir H, Rehman S, Waqas M (2020) A comparative study of feature selection approaches: 2016–2020. Int J Sci Eng Res 11(2):469

Spielman DA, Teng S-H (2011) Spectral sparsification of graphs. SIAM J Comput 40(4):981–1025. https://doi.org/10.1137/08074489X

Taffel S (2023) Data and oil: metaphor, materiality and metabolic rifts. New Media Soc 25(5):980–998. https://doi.org/10.1177/14614448211017887

Traag VA, Bruggeman J (2009) Community detection in networks with positive and negative links. Phys Rev E 80(3):036115. https://doi.org/10.1103/PhysRevE.80.036115. Accessed 3 April 2024

Traag VA, Waltman L, Van Eck NJ (2019) From louvain to leiden: guaranteeing well-connected communities. Sci Rep 9(1):5233. https://doi.org/10.1038/s41598-019-41695-z. Accessed 3 April 2024

Wagenseller P, Wang F, Wu W (2018) Size matters: a comparative analysis of community detection algorithms. IEEE Trans Comput Soc Syst 5(4):951–960. https://doi.org/10.1109/TCSS.2018.2875626

Warden P, Situnayake D (2020) TinyML: machine learning with Tensorflow lite on arduino, and ultra-low power micro-controllers. O'Reilly, Sebastopol

Xu H, Zhang L, Ma Y, Zhou S, Zheng Z, Jiajun B (2024) A survey on graph condensation. arXiv. http://arxiv.org/abs/2402.02000 Accessed 19 March 2024

Yang Z, Algesheimer R, Tessone CJ (2016) A comparative analysis of community detection algorithms on artificial networks. Sci Rep 6(1):30750. https://doi.org/10.1038/srep30750

Yang L, Zhu Q, Huang J, Wu Q, Cheng D, Hong X (2019) Constraint nearest neighbor for instance reduction. Soft Comput 23(24):13235–13245. https://doi.org/10.1007/s00500-019-03865-z

Zaki N, Krishnan A, Turaev S, Rustamov Z, Rustamov J, Almusalami A, Ayyad F, Regasa T, Iriho BB (2024) Node embedding approach for accurate detection of fake reviews: a graph-based machine learning approach with explainable AI. Int J Data Sci Anal. https://doi.org/10.1007/s41060-024-00565-2

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Zahiriddin Rustamov[1] · Nazar Zaki[1] · Jaloliddin Rustamov[1] · Ayham Zaitouny[2] · Rafat Damseh[1]

✉ Nazar Zaki
nzaki@uaeu.ac.ae

Zahiriddin Rustamov
700043167@uaeu.ac.ae

Jaloliddin Rustamov
700043175@uaeu.ac.ae

Ayham Zaitouny
ayham.zaitouny@uaeu.ac.ae

Rafat Damseh
rdamseh@uaeu.ac.ae

[1] Department of Computer Science and Software Eng., College of Info. Technology, United Arab Emirates University (UAEU), Al Ain, Abu Dhabi 15551, United Arab Emirates

[2] Department of Mathematical Sciences, College of Science, United Arab Emirates University (UAEU), Al Ain, Abu Dhabi 15551, United Arab Emirates