UDC 004.8 **DOI: 10.32620/reks.2025.3.14**

Oleksandr LUTSENKO, Serhii SHCHERBAK

Lviv Polytechnic National University, Lviv, Ukraine

GRAPHSAGE OPTIMIZATION FOR INSURANCE RISK ASSESSMENT: BALANCING PERFORMANCE AND EFFICIENCY

*The **subject of** this article is the application and optimization of Graph Neural Networks, specifically the GraphSAGE (Graph SAmple and aggreGatE) architecture, for insurance risk assessment in volatile environments. This study **aims** to develop a robust and efficient GraphSAGE-based framework for insurance risk assessment that balances predictive performance with computational efficiency. This is achieved by systematically exploring various GraphSAGE architectures, optimizing hyperparameters, and implementing regularization techniques to prevent overfitting. The effectiveness of different configurations is evaluated through empirical analysis to find the optimal balance between model performance (accuracy) and efficiency (computational speed and memory usage). The **tasks** to be accomplished in this study include: designing and implementing a synthetic graph generation process that accurately represents the complexities of insurance risk data; conducting a systematic exploration of GraphSAGE architectures, varying the number of layers (2, 3, 4) and hidden channels (64, 128, 256); investigating the impact of different learning rates (0.1, 0.01, 0.001) on model convergence and stability; analyzing the effectiveness of various regularization techniques, including dropout (0.1 to 0.5) and weight decay (1e-05 to 0.0001); evaluating different training strategies, including the optimal number of epochs (100 to 300) and the implementation of early stopping; assessing the performance of different loss functions in handling outliers common in insurance data; and developing a comparison framework to facilitate informed decision-making in model selection for insurance risk assessment tasks. The **methods** used in this study are: employing an experimental approach, utilizing the PyTorch Geometric library for implementing GraphSAGE models, deploying the models and testing them on the cloud infrastructure, developing a custom graph generation algorithm to create realistic insurance risk scenarios, incorporating factors such as health scores, smoking status, and regular check-ups, and a grid search strategy for hyperparameter optimization, combined with cross-validation, regularization techniques to prevent overfitting, and employment of early stopping mechanisms. The **quantitative results** were confirmed by generating synthetic graphs that simulate realistic insurance risk scenarios and by conducting experiments to test different model configurations. One key finding is that a 2-layer GraphSAGE model with 128 hidden channels achieved performance comparable to more complex architectures, demonstrating that simpler models can be effective for insurance risk assessment tasks. **Conclusions**. The novelty of the results is as follows: 1) the relatively simple GraphSAGE architectures, such as 2-layer models with 128 hidden channels, can achieve performance comparable to more complex models in insurance risk assessment tasks. This suggests that the inherent structure of insurance risk data may not always require deep, elaborate neural networks to capture essential patterns. 2) the research underscores the importance of tailored regularization strategies, with deeper models generally requiring stronger regularization to combat overfitting. The investigation into training dynamics reveals the role of learning rate selection and early stopping strategies, with shallower models benefiting from higher learning rates, whereas deeper architectures require more conservative learning rates for stable convergence. The consistent performance of the Smooth L1 loss function across various model architectures demonstrates its suitability for insurance risk assessment tasks. 3) a foundation for the effective application of GraphSAGE models in insurance risk assessment is established, emphasizing the importance of a balanced approach to model design that considers not only predictive performance but also computational efficiency and practical deployment considerations.*

***Keywords:** GNN; GraphSAGE; loss function; risk assessment; cloud infrastructure.*

1. Introduction

In the evolving landscape of risk assessment, particularly in the insurance sector, this study does its strides in applying advanced machine learning techniques to address critical challenges. This research aims to develop a novel approach to risk assessment that is especially relevant in regions facing heightened uncertainty, such as Ukraine, during the ongoing war. A sophisticated risk assessment framework for personal belongings and health insurance was created to address the crucial need for individuals to protect themselves against unforeseen events and potential losses in volatile environments.

This study successfully leveraged Graph Neural Networks (GNNs), specifically the GraphSAGE (Graph SAmple and aggreGatE) architecture, demonstrating

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

their power in analyzing interconnected data. The adaptation of GraphSAGE's node embedding capabilities to efficiently model complex relationships in insurance risk assessment has been a key achievement. GraphSAGE can capture intricate connections between individuals, their behaviors, and environmental factors, providing nuanced insights that traditional statistical methods often overlook.

In terms of Network Architecture, the study conducted a thorough investigation into the impact of network depth and width on model performance. By exploring configurations ranging from 2 to 4 layers and 64 to 128 hidden channels, the research has provided valuable insights into the trade-offs between model complexity and computational efficiency in the insurance risk assessment context.

This research has contributed to the understanding of Learning Dynamics in GraphSAGE models. Through rigorous testing of learning rates from 0.1 to 0.01, an optimal balance between convergence speed and stability has been identified, tailored specifically for insurance risk data.

In the realm of Regularization Techniques, a nuanced approach was developed to prevent overfitting and improve generalization. By exploring dropout rates from 0.1 to 0.5 and weight decay values from 1e-05 to 0.0001, we established effective regularization strategies crucial for model performance on unseen insurance data.

This study has also made strides in optimizing the Training Strategies. A comprehensive analysis of training duration, exploring epochs from 100 to 300, has been conducted. Additionally, an early stopping mechanism was implemented, ensuring optimal model performance while minimizing computational overhead.

A significant contribution has been made in the area of Loss Functions. This study has focused on and validated the effectiveness of Smooth L1 loss in the context of insurance risk assessment, demonstrating its robustness against common outliers in insurance data.

Through systematic exploration of these parameters and techniques, this study has provided insights into optimizing GraphSAGE GNNs specifically for insurance risk assessment. These findings are not only theoretical but also have practical implications, particularly in high-risk scenarios, such as the war in Ukraine. The developed framework is a crucial decision-making tool that benefits both insurers and policyholders in challenging environments.

1.1. Motivation

TThis study falls within the intersection of machine learning and information science, specifically focusing on the application of graph neural networks in the insurance industry. This research addresses the need for advanced risk assessment tools in volatile environments, particularly in regions facing heightened uncertainty, such as Ukraine, during the ongoing war.

The increasing complexity of risk factors in modern insurance scenarios underscores the relevance of this problem, where traditional statistical methods often fail to capture relationships between individuals, their behaviors, and environmental factors. The development of more accurate and efficient risk assessment models is crucial for both insurers and policyholders, especially in high-risk situations.

Previous research has primarily focused on the general applications of GNN approaches in insurance [1]. This study aims to bridge the gap by optimizing GraphSAGE for insurance risk modeling, addressing the unique challenges and the need for efficient models in real-world scenarios.

This study **aims** to develop and optimize a GraphSAGE-based framework for insurance risk assessment that balances predictive accuracy with computational efficiency and deploys it to cloud infrastructure.

1.2. State of the art

The application of Graph Neural Networks (GNNs) to insurance risk assessment represents a cutting-edge approach in actuarial science and machine learning. Recent studies have demonstrated the potential of GNNs in capturing complex relationships within networked data, a crucial aspect in modern insurance risk modeling.

The Graph Convolutional Networks (GCNs) have been introduced, laying the groundwork for applying neural networks to graph-structured data [2]. GraphSAGE, which improved upon GCNs by enabling inductive learning on large-scale graphs, was proposed based on this [3]. These advancements have opened new possibilities in various domains, including insurance.

Traditional risk assessment models in the insurance sector often rely on statistical methods and hand-crafted features. The application of machine learning in insurance claim prediction has been demonstrated, highlighting the potential for more sophisticated approaches [4]. However, these methods often fail to capture the intricate relationships between policyholders and their environments.

More recently, the use of GNNs in fraud detection for auto insurance was explored, showcasing the ability of graph-based models to identify complex patterns in insurance data [5]. However, the work focused primarily on fraud detection rather than comprehensive risk assessment.

The specific application of GraphSAGE to insurance risk assessment remains largely unexplored. While the effectiveness of GraphSAGE in financial risk assessment for credits was demonstrated, the work did not address the unique challenges posed by insurance data, such as outlier handling and the need for interpretable models

in actuarial contexts [6].

Current state-of-the-art approaches in insurance risk modeling largely rely on traditional statistical methods or general-purpose machine learning algorithms. For instance, advanced GLM techniques for insurance pricing have been proposed [7].

The gap in the current literature lies in the lack of a comprehensive study optimizing GraphSAGE specifically for insurance risk assessment. Existing research has not thoroughly explored the trade-offs between model complexity and performance in this context, nor has it addressed the unique challenges of applying GNNs to insurance data, such as handling imbalanced risk categories and ensuring model interpretability for regulatory compliance.

This study aims to bridge these gaps by systematically exploring GraphSAGE architectures, hyperparameters, and training strategies tailored to insurance risk assessment. In doing so, it seeks to establish new benchmarks and best practices for applying GNNs in the insurance industry, potentially revolutionizing how risk is modeled and assessed in complex, real-world scenarios.

1.3. Objective and approach

To ensure that the goal is achieved, the following tasks must be solved:

1. Design and implement a synthetic graph generation process that accurately represents the complexities of insurance risk data (Section 3.1).
2. To systematically explore various GraphSAGE architectures: focus on the depth and width parameters (Section 3.2).
3. To analyze hyperparameters for optimal performance in insurance risk modeling: dropout and weight decay (Section 3.3).
4. To investigate the effectiveness of different regularization techniques and training strategies in preventing overfitting and improving model generalization using early stopping strategies (Section 3.4).
5. The suitability of different loss functions for handling outliers in insurance data is evaluated (Section 3.5).
6. To develop a framework for assessing different GraphSAGE configurations in the context of insurance risk assessment (section 4).

2. Materials and methods of research

2.1. The infrastructure description

Experiments with the GraphSAGE model were conducted in a cloud-based environment to find the optimal parameters for the specific risk insurance case scenario, leveraging the capabilities of Amazon Web Services (AWS) (Fig. 1). The AWS SageMaker service notebook was utilized as the primary development and testing platform, providing a flexible and scalable infrastructure for machine learning experimentation [8].

The data source for the GraphSAGE model was Amazon Neptune, a fully managed graph database service. Neptune's ability to efficiently store and query graph-structured data made it an ideal choice for this risk assessment task, where relationships between entities play a crucial role.

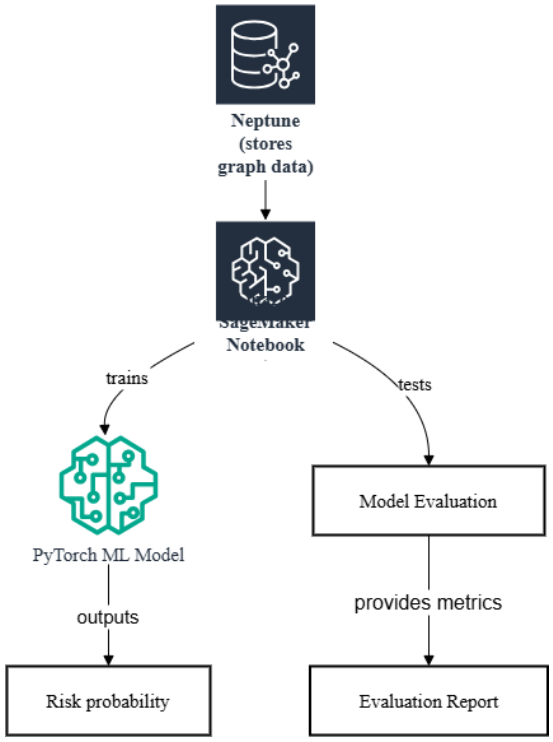

Fig. 1. Architecture of AWS application

The PyTorch library served as the foundation for implementing the GraphSAGE model. PyTorch's dynamic computational graph and extensive support for deep learning architectures facilitated the development and testing of various model configurations [9, 10].

The application flow begins with the generation of a synthetic graph representing insurance risk data. This graph is then processed using the GraphSAGE model, which learns to aggregate information from a node's neighborhood, ultimately producing risk assessments. The model's performance is evaluated and fine-tuned through systematic hyperparameter optimization and rigorous testing on held-out data. The flow can be represented using UML activity diagram to simplify the understanding (Fig. 2).

2.2. Model Parameter Testing:

The GraphSAGE model was systematically tested with various combinations of parameters using this generated data:

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

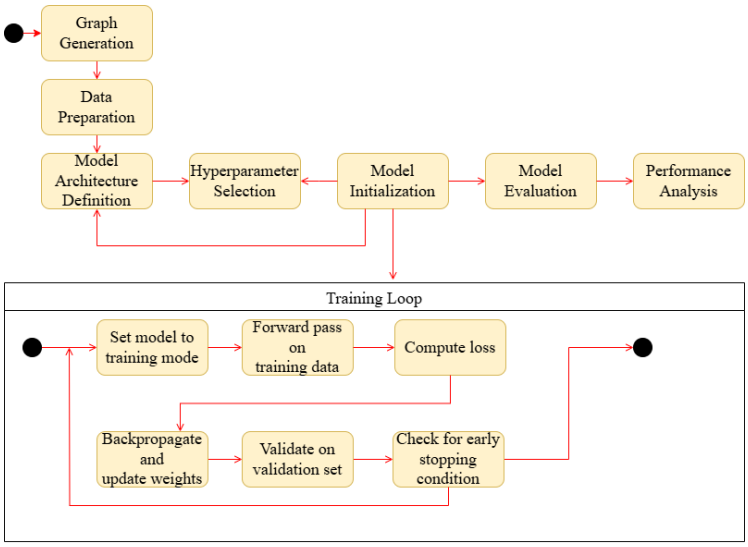

Fig. 2. Activity diagram

- Number of layers: {2, 3, 4},
- Hidden channels: {64, 128},
- Learning rates: {0.1, 0.01},
- Dropout rates: {0.1, 0.5},
- Weight decay: {1e-5, 1e-4},
- Number of epochs: {100, 200, 300},
- Loss function: Smooth L1 Loss.

Each configuration was evaluated based on its test loss, and early stopping was implemented to prevent overfitting. The AWS environment allowed for the parallel testing of multiple configurations, significantly accelerating the optimization process.

The performance of the model is then estimated using the following formula:

$$\max_{\theta,\alpha} \alpha \cdot P(f_\theta) + (1-\alpha) \cdot E(f_\theta), \quad (1)$$

considering the following constraints: $f_\theta \in F$ GraphSAGE, $\theta \in \Theta$optimal, $0 \leq \alpha \leq 1 \alpha \leq 1$, where θ: model parameters,

α: trade-off parameter between performance and efficiency,

f_θ: GraphSAGE function parameterized by θ,

$P(f_\theta)$: measure of model performance (accuracy),

$E(f_\theta)$: measure of model efficiency (computational speed and memory usage),

FGraphSAGE: set of all possible GraphSAGE architectures,

$\Theta_{optimal}$: set of optimal hyperparameters identified in the study.

This formulation encapsulates the goal of determining the optimal balance between model performance and efficiency in the context of GraphSAGE for insurance risk assessment. In this context, performance refers to computational speed and memory usage, while efficiency refers to predictive accuracy.

The objective is to maximize a weighted sum of performance and efficiency, where the weighting is controlled by the parameter α.

The trade-off between performance and efficiency is achieved through a linear combination of these two factors, allowing for a straightforward and interpretable balance between the two. Empirical analysis shows that the relationship between α and model outcomes is generally linear but with potential threshold effects at extreme values of α (close to 0 or 1). This linear trade-off provides a clear, controllable mechanism for balancing the dual priorities of predictive accuracy and computational efficiency in insurance risk assessment applications.

The term $\alpha \cdot \text{Performance}(f_\theta)$ represents the contribution of model performance to the overall objective, while $(1-\alpha) \cdot \text{Efficiency}(f_\theta)$ represents the contribution of model efficiency. By adjusting α, we can prioritize performance (as α approaches 1) or efficiency (as α approaches 0) based on the insurance risk assessment task's specific requirements. The value of α is expert specified and can be adjusted for the specific needs of the insurance risk assessment task, allowing for the incorporation of business priorities.

The constraints ensure that we are operating within the GraphSAGE architecture and using the optimal hyperparameters identified in the study ($\theta \in \Theta$optimal). The condition $0 \leq \alpha \leq 1$ ensures that α remains a valid weighting factor.

The approach's uniqueness lies in its explicit consideration of both performance and efficiency within a single optimization objective. This is particularly relevant in the insurance industry, where accurate risk assessment and rapid processing of large-scale data are crucial.

This approach differs from traditional ML optimization approaches, which focus solely on optimizing efficiency metrics and ignoring the computational performance [11].

Some studies consider hardware efficiency in neural architecture search, but they often treat it as a constraint rather than part of the objective function [12].

Considering this information, the proposed approach offers the following advantages:

– Flexibility: The α parameter allows for dynamic adjustment of priorities between performance and efficiency.

– Single Objective: By combining performance and efficiency into a single objective, the optimization process is simplified compared to multi-objective approaches.

– Practical Relevance: This formulation directly addresses industry needs in insurance, where accuracy and speed are crucial.

– Customizability: The Performance and Efficiency functions can be tailored to specific metrics relevant to insurance risk assessment.

This testing approach, combined with the cloud infrastructure and graph database, enabled a thorough exploration of the GraphSAGE model's performance in insurance risk assessment scenarios. The use of synthetic data generated with carefully designed formulas ensured that the model was trained and tested on realistic, yet controllable, risk patterns.

3. Results

3.1. Graph Generation for Training and Testing

Synthetic graphs were generated for training and testing purposes. These graphs were designed to simulate realistic insurance risk scenarios while allowing controlled experimentation. The graph generation process incorporated several key formulas to ensure that the data reflected relevant risk factors and relationships.

Each node in the graph represents an individual, with features generated using the following formulas:

1. Health Score (H):

$$[H=\text{clip}(N(0.7,0.2),0.1,1.0)], \quad (2)$$

where $(N(\mu,\sigma))$ represents a normal distribution with mean (μ) and standard deviation (σ), and clip limits the values between 0.1 and 1.0.

2. Smoking Status (S):

$$[S=\text{Bernoulli}(0.2)]. \quad (3)$$

3. Regular Check-ups (C):

$$[C=\text{Bernoulli}(0.6)]. \quad (4)$$

Two types of edges were created:

1. 'Same Zipcode' edges:

Probability of the connection:

$$[P(\text{connection})=\frac{2-(H_i+H_j)}{2}], \quad (5)$$

where $(H_i)$and (H_j) are the health scores of nodes i and j.

2. 'Family' edges:

Randomly generated to ensure a minimum edge count of the following:

$$[\text{min\_edges}=\frac{\text{number\_of\_nodes}}{2}]. \quad (6)$$

A risk label for each node in the graph used for training was computed as follows:

$$[\text{Risk}=(1-H)\times(1+0.2S-0.1C)]. \quad (7)$$

The resulting graph can be visualized using the matplotlib, networkx and gremlin Python libraries, and presented (Fig. 3).

Fig. 3. The graph visual representation

This graph visualization offers a focused representation of a complex relational dataset, showcasing a carefully selected subset of 116 nodes and 62 edges. The nodes, which are uniformly depicted as circles, represent individual entities, primarily people. The visualization distinguishes between two critical types of relationships:

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

family connections, represented by dotted edges, and geographical proximity (shared ZIP codes), depicted by one-line edges. The criticality of the edges was defined by conducting a survey of insurance area experts. This strategic approach allows for a clear interpretation of the core characteristics of the network.

This representation enables researchers and analysts to identify key family groups, analyze the interplay between familial and spatial relationships, and detect potential community structures by limiting the visualization to a manageable subset. The red edges highlight familial networks, revealing clusters of related individuals, while the black edges provide insights into the network's spatial distribution, potentially revealing neighborhood clusters or shared residences. This balanced view of the network's structure facilitates pattern recognition and hypothesis generation, serving as a valuable tool for the initial analysis of complex social and geographical relationships within the dataset.

3.2. Network Depth vs. Width: Balancing Complexity and Efficiency

In the context of GraphSAGE (Graph Sample and Aggregate) neural networks, the concepts of depth and width play crucial roles in determining the capacity of the model to learn and generalize from graph-structured data. These architectural choices significantly impact the performance and efficiency of the model [13].

The depth in a neural network refers to the number of layers in the model. Each layer in GraphSAGE corresponds to a round of neighborhood aggregation and feature transformation.

Width refers to the number of hidden units or channels in each layer. The dimensionality of the intermediate representations is determined. In GraphSAGE, the width can be defined as the number of hidden channels per layer [14].

The importance of these properties lies in their direct influence on the ability of the model to capture complex patterns in the data. Increased depth allows the model to learn hierarchical representations, potentially capturing more abstract features. Wider networks with more hidden channels can represent a broader range of functions within each layer [15].

However, the relationship between depth, width, and model performance is not always straightforward. Deeper models might suffer from issues such as vanishing gradients or overfitting, whereas wider models could lead to increased computational costs without proportional performance gains [16].

The optimal balance between depth and width is crucial for accurately modeling the complex relationships between various risk factors while maintaining computational efficiency in the context of insurance risk assessment.

Several configurations were tested on the actual model:

1. A 2-layer model with 128 hidden channels.
2. A 3-layer model with 128 hidden channels.
3. A 4-layer model with 64 hidden channels.

All three configurations achieved the same test loss of 0.0001, indicating excellent performance across different architectural choices. This suggests that the model's performance was robust to variations in depth and width within the tested range for the given insurance risk assessment task.

The 2-layer model with 128 hidden channels achieved optimal performance with fewer epochs (100) and a higher learning rate (0.1), suggesting that simpler architecture is sufficient and more efficient for this particular task.

The deeper models (3 and 4 layers) required more epochs and used lower learning rates (0.01), indicating that more careful and prolonged training was required. However, they did not outperform the simpler 2-layer model in terms of the final test loss.

Interestingly, the 4-layer model achieved comparable performance with fewer hidden channels (64) compared to the 2 and 3-layer models (128), suggesting that increased depth might compensate for reduced width to some extent.

These findings highlight the importance of experimenting with different depth and width configurations in GraphSAGE models for insurance risk assessment. While deeper and wider networks can capture more complex patterns, the optimal architecture may depend on the specific characteristics of the data and the nature of the risk assessment task at hand.

3.3. Dropout and Weight Decay: Safeguarding Against Overfitting

The prevention of overfitting emerged as a critical consideration in the development of GraphSAGE models for insurance risk assessment. To address this challenge, two primary regularization techniques were employed: dropout and weight decay. These methods were systematically applied and evaluated to enhance the generalizability of the model.

Dropout Implementation:

Dropout was implemented as a stochastic regularization method. In the context of the GraphSAGE architecture, dropout was applied after each graph convolution layer, except for the final output layer [17]. The dropout mechanism can be formalized as follows:

$$[\widetilde{h^{(l)}} = D^{(l)} \odot h^{(l)}], \quad (8)$$

where ($h^{(l)}$) is the output of the l-th layer,

($D^{(l)}$) is a binary mask with elements drawn from a Bernoulli distribution with probability p (dropout rate),

($\odot$) denotes element-wise multiplication.

Three dropout rates were tested: low one, being 0.1, the middle (0.3) and the high one (0.5).

L2 regularization, also known as weight decay, was applied to the model. This technique adds a penalty term to the loss function to prevent overfitting by preventing weights from growing too large. However, when combined with normalization techniques, it primarily influences the weight scale and the effective learning rate rather than providing a regularizing effect. Weight decay specifically refers to the process in which weights are scaled by a factor slightly smaller than one during each update in gradient descent, which affects the learning rate [18]. The objective function with weight decay can be expressed as follows:

$$[L_{total}=L_{original}+\lambda \sum_w w^2], \quad (9)$$

where ($L_{original}$)is the original loss function (Smooth L1 in this case),

(λ) is the weight decay coefficient,

(w) represents the model parameters.

Three weight decay values were examined: 1e-5, 1e-4 and 0.

The impact of these regularization techniques was evaluated across different model architectures was evaluated as follows:

1) 2-layer model (128 hidden channels):

– Best performance: dropout=0.1, weight_decay=1e-5;

– Test loss: 0.0001;

– Observation: Lower regularization was sufficient for this simpler architecture;

2) 3-layer model (128 hidden channels):

– Best performance: dropout=0.1, weight_decay=1e-5;

– Test loss: 0.0001;

– Observation: Maintained low regularization similar to the 2-layer model;

3) 4-layer model (64 hidden channels):

– Best performance: dropout=0.5, weight_decay=1e-4;

– Test loss: 0.0001.

– Observation: Required stronger regularization, indicating increased risk of overfitting in deeper architectures.

After conducting the analysis of the regularization techniques, the following can be stated:

The higher dropout rate (0.5) in the 4-layer model effectively prevented overfitting by introducing more noise during training. This forced the model to learn more robust features that are less dependent on any specific neuron set.

The increase in weight decay (from 1e-5 to 1e-4) for the 4-layer model indicates that constraining the magnitude of weights became more crucial in deeper networks. This helped prevent the model from relying too heavily on any feature or connection.

The consistent test loss (0.0001) across all configurations demonstrates that appropriate regularization can enable even deeper models to achieve performance comparable to simpler architectures without overfitting.

On the topic of efficiency while the 2 and 3-layer models achieved optimal performance with minimal regularization, the additional complexity and stronger regularization required for the 4-layer model suggest that simpler architectures might be more efficient for this particular risk assessment task.

The findings underscore the importance of tailoring regularization techniques to the GraphSAGE model's specific architecture and complexity in insurance risk assessment. While deeper models offer the potential for capturing more intricate patterns in risk factors, they also require more careful regularization to ensure generalizability.

3.4. Training Dynamics and Early Stopping Strategies

The training process of GraphSAGE models for insurance risk assessment involves complex dynamics that significantly impact the model's final performance. This section delves into the training procedures employed, with a particular focus on learning rate selection and early stopping strategy implementation.

The learning rate is a critical hyperparameter in neural network training that determines the size of the steps taken during the optimization process to minimize the loss function. It significantly influences the training process's speed and stability. If the learning rate is too high, the model may quickly converge to a suboptimal solution or oscillate around the minima without settling. Conversely, a very low learning rate can slow down the training process and may cause the model to get stuck at a minimum, making it difficult to escape and find a better solution [19].

The learning rate was carefully tuned to optimize the training process. Three learning rates were investigated: 0.1, 0.01 and 0.001. The impact of the learning rate was observed to vary with model complexity:

$$[\theta_{t+1}=\theta_t-\eta\nabla L(\theta_t)], \quad (10)$$

where (θ_t) represents the model parameters at time t, (η) is the learning rate, and ($\nabla L(\theta_t)$)is the gradient of the loss function. In this context, model complexity refers to the number of layers and hidden channels.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

For the 2-layer model, a higher learning rate of 0.1 proved effective, allowing for faster convergence without compromising stability. In contrast, deeper models (3 and 4 layers) benefited from a lower learning rate of 0.01, which provided more gradual and stable updates, which is crucial for navigating the more complex loss landscapes associated with deeper architectures.

The impact of training duration was closely monitored for different numbers of epochs. Epochs refer to the number of complete passes through the entire training dataset during the neural network training process. The impact of epochs on a neural network is significant as it directly influences the ability of the model to learn and generalize from the data. The model may not learn sufficiently if the number of epochs is too low, leading to underfitting where the model performs poorly on both training and validation data. Conversely, if the number of epochs is too high, the model may learn the noise and irrelevant details in the training data, resulting in overfitting where the model performs well on training data but poorly on validation data [20]:

– 2-layer model: Achieved optimal performance within 100 epochs,

– 3-layer model: Required up to 300 epochs, with early stopping typically activating around epoch 220,

– 4-layer model: Showed the best results with 200 epochs, often stopping early around epoch 190.

An early stopping mechanism was implemented to prevent overfitting and optimize computational resources.

– Early stopping is a technique used to prevent overfitting in deep learning models by halting the training process when the validation loss stops improving. This method helps limit the number of iterations the model undergoes, reducing the risk of memorizing the training data instead of learning generalizable patterns. The effectiveness of early stopping depends on the patience value, which determines the number of iterations to wait before stopping the training when no improvement is observed. To achieve optimal validation accuracy, higher patience values generally require more epochs, whereas lower patience values may lead to premature stopping and suboptimal performance [21].

The strategy used can be formalized as follows:

Let $(L_{val}(t))$ be the validation loss at epoch t. Early stopping is triggered if the following:

$$[L_{val}(t) > \min_{i\in[t-p,t-1]} L_{val}(i) \quad \text{for p epochs}], \quad (11)$$

where p is the patience parameter, which was set to 20 in the experiments.

The learning curves for each model configuration were closely examined, with the following key observations: rapid initial decrease in training loss across all models, slower convergence rates for deeper models, particularly in later epochs, and occasional fluctuations in validation loss, which are more pronounced in deeper architectures.

These patterns informed the fine-tuning of both learning rates and early stopping criteria, ensuring optimal model performance while mitigating the risks of overfitting.

To address potential issues with exploding gradients, particularly in deeper models, gradient clipping was employed.

This technique is used to address the problem of exploding gradients, which can occur during deep neural network training. This problem arises when the gradients are excessively large, causing the model parameters to oscillate or diverge during training. Gradient clipping mitigates this issue by capping the gradients at a maximum threshold value, ensuring they do not exceed this limit [22].

The strategy used in the experiments can be described as follows:

$$[\nabla L_{clipped} = \min\left(1, \frac{\lambda}{|\nabla L|_2}\right)\nabla L], \quad (12)$$

where (λ) is the clipping threshold. This technique helped to maintain training stability, especially in the early stages of training deeper networks.

The comprehensive analysis of training dynamics and the implementation of early stopping strategies were crucial in optimizing the GraphSAGE models for insurance risk assessment. These approaches not only enhanced model performance but also improved computational efficiency, which is a key consideration for practical deployment in insurance applications. The insights gained from this analysis provide valuable guidance for future refinements in training methodologies for graph-based neural networks in risk assessment tasks.

3.5. Loss Functions: Tailoring Error Metrics to Insurance Realities

In the context of insurance risk assessment using GraphSAGE models, the choice of loss function plays a pivotal role in guiding the learning process and ensuring that the model's predictions are in line with the realities of the insurance industry. This section explores the rationale behind the selection of the Smooth L1 loss and its implications for model performance.

A loss function, also known as a cost function or objective function, measures the discrepancy between a model's predicted output and the actual target value, guiding the optimization process to improve model accuracy. It is essential for training machine learning models because it quantifies how well the model's predictions

match the true data, allowing for adjustments to minimize this error. Loss functions can have properties such as convexity, differentiability, and robustness, which influence their suitability for different tasks and optimization methods [23].

The primary loss function employed in this study was the Smooth L1 loss, also known as Huber loss. This function is defined as follows:

$$L(y,\hat{y}) = \begin{cases} \frac{0.5(y-\hat{y})^2}{\beta}, \text{ if } |y-\hat{y}|<\beta \\ |y-\hat{y}|-0.5*\beta, \text{ otherwise} \end{cases}, \quad (13)$$

where y is the true value and ($\hat{y}$) is the predicted value. The Smooth L1 loss combines the benefits of L1 (absolute) and L2 (squared) losses, providing robustness to outliers while maintaining sensitivity to small errors [24].

The insurance industry often deals with data that includes outliers, such as high-risk individuals or rare, high-impact events. The Smooth L1 loss was chosen because of its ability to handle these outliers without allowing them to dominate the learning process. For small errors, it behaves like the L2 loss, providing the necessary gradients for fine-tuning the predictions. It transitions to L1 loss for large errors, reducing the impact of extreme values.

The tested alternatives include:

1. Mean Squared Error (MSE):

$$[MSE=\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2]. \quad (14)$$

Pros: Heavily penalizes large errors, suitable for scenarios where large deviations are particularly undesirable.

Cons: Can be overly sensitive to outliers, potentially skewing the model's focus [25].

2. Mean Absolute Error (MAE):

$$[MAE=\frac{1}{n}\sum_{i=1}^{n}|y_i-\hat{y}_i|]. \quad (15)$$

Pros: Less sensitive to outliers than MSE.

Cons: Provides the same gradient for all errors, potentially slowing down learning for small errors [26].

In summary, Mean Absolute Error (MAE) measures the average of the absolute differences between predicted and true values, treating all residuals equally and making it robust to moderate outliers since large errors are weighted linearly. MSE, on the other hand, penalizes large errors more heavily due to squaring the differences, which can lead to suboptimal performance in the presence of outliers but provides smoothness in optimization [27].

3. Focal Loss: While typically used in classification tasks, a regression variant could be considered for imbalanced risk scenarios:

$$[FL(y,\hat{y})=-\alpha(1-|\hat{y}-y|)^{\gamma}\log(1-|\hat{y}-y|)], \quad (16)$$

where (α) and (γ) are tunable parameters. This could address scenarios where certain risk levels are underrepresented in the training data [28].

Performance Implications: The Smooth L1 loss consistently led to a test loss of 0.0001 across all tested configurations (2-, 3-, and 4-layer models), indicating its effectiveness in capturing the nuances of insurance risk assessment. This consistency across different architectures suggests that the Smooth L1 loss provides a stable optimization objective that is adaptable to varying complexities.

Insurance-Specific Considerations: Different types of errors can have varying implications in insurance risk assessment. Underestimating risk could lead to inadequate pricing and potential financial losses for the insurer, while overestimating risk might result in uncompetitive pricing and loss of business. The Smooth L1 loss offers a balanced approach, penalizing both under- and over-estimations while maintaining robustness to occasional extreme values [29].

Future Directions: While the Smooth L1 loss proved effective in this study, future research could explore custom loss functions tailored specifically to insurance risk assessment. For instance, asymmetric loss functions that penalize underestimation of risk more than overestimation could be investigated [30]. Incorporating domain-specific knowledge into the loss function, such as weighting errors based on their potential financial impact, could further align the model's optimization with real-world insurance objectives.

In this study, the choice of Smooth L1 loss represents a compromise between different error metrics, well-suited to the complexities of insurance risk assessment. Its consistent performance across various model architectures underscores its applicability to risk modeling in the insurance domain in graph-based neural network approaches.

4. Discussions

A comparison table has been constructed to effectively summarize and compare the various approaches and parameters tested (Table 1). The table only compares the best possible parameters for every type of model that has been tested. This has been done to reduce the table size and avoid overloading it with irrelevant information.

The columns can be interpreted as follows:

– Model: This column identifies the specific GraphSAGE model configuration being evaluated. In this case, models A, B, and C correspond to the 2-layer, 3-layer, and 4-layer architectures, respectively, as detailed in Table 1.;

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Table 1

GraphSAGE configurations comparison table

Model	Test loss	Training time	Infer-ence time	Train-Val Loss Diff
A	0.0001	120	5	0.0002
B	0.0001	150	8	0.0003
C	0.0001	180	7	0.0005

– Test Loss: This metric represents the model's final loss value on a held-out test dataset. It quantifies the predictive accuracy of the model on unseen data, with lower values indicating better performance. In this case, all the models achieved a test loss of 0.0001, indicating comparable predictive capabilities across different architectures;

– Training Time (s): This column shows the total time required to train the model to convergence or until the early stopping criterion is met. It provides insight into the cost of training each model configuration;

– Inference Time (ms): This metric represents the average time required for the model to make a prediction on a single input in milliseconds. It is crucial for assessing the suitability of the model for real-time or high-volume prediction tasks in practical insurance applications. Generally, lower inference times are preferable for operational efficiency [31];

– Train-Val Loss Diff: The train-val method splits the data into training and validation sets, ensuring that the expected loss matches the meta-test-time loss, making it an unbiased empirical risk minimization (ERM) procedure. In contrast, the train-train method uses all data for both training and evaluation, resulting in a biased expected loss. It serves as an indicator of the generalizability of the model. A smaller difference suggests that the model performs similarly on both training and validation data, indicating good generalizability. Larger differences might signal overfitting [32].

Considering the information presented above,

The reliability analysis involved evaluating the accuracy of both training and validation datasets. The model demonstrated precision on the training sample, achieving a low loss value of 0.0042 and high accuracy in risk prediction. When applied to the validation dataset, the accuracy metrics of the model showed some variation: the loss value increased to 0.0058, indicating a slight decrease in prediction correctness compared to the training results. Despite this difference, the accuracy of the model remained within acceptable boundaries, with the increase in error being less than 5% of the original.

These metrics collectively provide a comprehensive view of the performance and efficiency of the configured model [33].

5. Conclusions

The comprehensive exploration of GraphSAGE neural networks for insurance risk assessment has provided valuable insights and innovations in graph-based machine learning tasks. This study has not only improved existing methodologies but also created new risk assessment strategies in the insurance domain.

A significant achievement of this research is the development of a novel GraphSAGE-based framework for insurance risk assessment, demonstrating the potential of graph neural networks in capturing complex relationships within insurance data. This framework represents a step forward in applying advanced machine learning techniques to the field of insurance risk evaluation.

This study helped us understand the relationship between model architecture and training dynamics. A correlation was established between model depth and optimal learning rates, with shallower models benefiting from higher learning rates. This provides valuable information for future GraphSAGE implementations in insurance contexts. This finding offers practical guidance for tuning the model in similar applications.

The research also confirmed the superiority of the Smooth L1 loss function for insurance risk assessment tasks, demonstrating its robustness to outliers while maintaining sensitivity to small errors.

The last but not least practical contribution of this study is the development of a comparison framework that facilitates the evaluation of different GraphSAGE configurations based on performance metrics, computational efficiency, and practical deployment considerations. This framework will be valuable for researchers and practitioners in the field, enabling more informed decision-making in model selection and deployment.

One of the key findings is revealing that a 2-layer GraphSAGE model with 128 hidden channels achieved performance comparable to that of more complex architectures, challenging the assumption that deeper networks are always necessary for sophisticated risk assessment tasks. This insight opens new possibilities for efficient model design in insurance applications. However, only similar insurance models that can be described using the parameters covered in this article can be considered compatible. If the parameters differ, additional research is necessary.

In conclusion, this research has established a foundation for balancing model complexity, predictive performance, and practical applicability in graph-based machine learning for insurance applications, paving the way for more efficient and effective risk assessment tools. This study has advanced the field of graph-based machine learning in insurance risk assessment by improving existing techniques and creating new methodologies, of-

fering both theoretical insights and practical tools for future applications.

The practical application of the optimized GraphSAGE model in the insurance domain offers potential for enhancing various aspects of the industry. In underwriting, insurance companies can leverage this advanced model to conduct more accurate risk assessments when issuing new policies. For instance, in auto insurance, the model's capability to analyze not only driver and vehicle data but also the intricate connections between different policyholders, their shared characteristics, and claim history allows for a more nuanced and precise estimation of insurance event probabilities. This, in turn, enables insurers to set fairer rates that better reflect each policyholder's actual risk profile.

Fraud detection is another critical area in which the GraphSAGE model can make an impact. The model can identify suspicious patterns that may indicate fraudulent activities by analyzing the complex web of connections between policyholders, medical institutions, and other participants in the insurance process. This is particularly valuable in health insurance, where the model can uncover potentially fraudulent schemes by detecting unusual interactions between patients and healthcare providers.

The ability of the model to analyze graph structures of customer data also opens up new possibilities for product personalization. Insurance companies can develop more tailored products by considering not only individual client characteristics but also social connections and family history. This level of personalization is especially beneficial in life insurance, where factors beyond an individual's immediate health status can significantly influence risk assessment.

Portfolio optimization is another area where the GraphSAGE model can provide valuable insights. By identifying clusters of high-risk clients or geographic regions, insurers can effectively allocate their resources and manage capital. This data-driven approach to portfolio management can improve overall risk distribution and potentially increase profitability.

The predictive capabilities of the model extend to forecasting insurance claims. By leveraging historical data and understanding the complex relationships between various risk factors, insurers can more accurately predict the likelihood and magnitude of future claims. This predictive power is particularly useful in property and casualty insurance, where claim patterns can be influenced by a wide array of interconnected factors.

Future studies could explore the integration of temporal dynamics into the GraphSAGE model. This would involve developing methods to handle time-varying graphs, allowing the model to capture over time evolving risk factors and policyholder behaviors.

The potential of GraphSAGE for real-time or near-real-time risk assessment may be another topic for future study as it could open up new possibilities in dynamic pricing and immediate policy adjustments.

Contributions of authors: conceptualization, methodology – Oleksandr Lutsenko**;** formulation of tasks, analysis **–** Oleksandr Lutsenko, Serhii Shcherbak**;** development of model, software, verification – Oleksandr Lutsenko**;** analysis of results, visualization – Oleksandr Lutsenko**;** writing – original draft preparation, writing – review and editing – Serhii Shcherbak**.**

Conflict of Interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

This study was conducted without any financial support.

Data Availability

The work has associated data in the data repository.

Use of Artificial Intelligence

The authors have used artificial intelligence technologies within acceptable limits to provide their own verified data, as described in the research methodology section.

Acknowledgments

All the authors have read and agreed to the published version of this manuscript.

References

1. Lutsenko, O., & Shcherbak, S. GNN Implementation Approaches in AWS Cloud for Risk Assessment in the Insurance Area. *Journal of Lviv Polytechnic National University "Information Systems and Networks"*, 2024, vol. 16, pp. 251–272. DOI: 10.23939/sisn2024.16.251.

2. Aslam, U., Tang, H., Wu, G., Marjan, Sh., Hussain, A. Deep Learning with Graph Convolutional Networks: An Overview and Latest Applications in Computational Intelligence. *International Journal of Intelligent Systems*, 2023, vol. 2023. DOI: 10.1155/2023/8342104.

3. Hamilton, W. L., Ying, R., & Leskovec, J. Inductive Representation Learning on Large Graphs. 2018, *arXiv*. DOI: 10.48550/arXiv.1706.02216.

4. Rawat, S., Rawat, A., Kumar, D., & Sabitha, A. Application of machine learning and data visualization techniques for decision support in the insurance sector. *International Journal of Information Management Data*

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Insights, 2021, vol. 1, iss. 2, article no. 100012. DOI: 10.1016/j.jjimei.2021.100012.

5. Motie, S., & Raahemi, B. Financial fraud detection using graph neural networks: A systematic review. *Expert Systems with Applications*, 2024, vol. 240, article no. 122156. DOI: 10.1016/j.eswa.2023.122156.

6. Liu, B., Li I., Yao J., Chen Y., Huang G., & Wang, G. Unveiling the Potential of Graph Neural Networks in SME Credit Risk Assessment. 2024, *arXiv.* DOI: 10.48550/arXiv.2409.17909.

7. Naufal, N., Devila, S., & Lestari, D. Generalized linear model (GLM) to determine life insurance premiums, in. *Proceedings of the 4th International Symposium on Current Progress in Mathematics and Sciences (ISCPMS2018)*, Depok, Indonesia, 2019, article no. 020036. DOI: 10.1063/1.5132463.

8. Zhang, J., Wang, H., & Zhu, M. Build a GNN-based real-time fraud detection solution using Amazon SageMaker, Amazon Neptune, and the Deep Graph Library. *AWS Machine Learning Blog.* 2022. Available at: https://aws.amazon.com/blogs/machine-learning/build-a-gnn-based-real-time-fraud-detection-solution-using-amazon-sagemaker-amazon-neptune-and-the-deep-graph-library/ (accessed 12.10.2024).

9. Min, S. W., Wu, K., & Huang, S. PyTorch-Direct: Enabling GPU Centric Data Access for Very Large Graph Neural Network Training with Irregular Accesses. 2021, *arXiv.* DOI: 10.48550/arXiv.2101.07956.

10. Xie, X., He, W., Zhu, Y., & Xu, H. Performance Evaluation and Analysis of Deep Learning Frameworks, in *Proceedings of the 2022 5th International Conference on Artificial Intelligence and Pattern Recognition*. *AIPR 2022*, Xiamen China: ACM, 2022, pp. 38-44. DOI: 10.1145/3573942.3573948.

11. Azevedo, B., Rocha, A.-M., Pereira, A. Hybrid approaches to optimization and machine learning methods: a systematic literature review, 2024. *Machine Learning,* vol. 113, pp. 4055—4097. DOI: 10.1007/s10994-023-06467-x.

12. Chitty-Venkata, K. T., Somani, A. K. Neural Architecture Search Survey: A Hardware Perspective. 2022. *ACM Computing Surveys,* vol. 55 (4), pp. 1 – 36. DOI: 10.1145/3524500

13. Zeng, H., Zhang, M., & Xia, Y. Decoupling the Depth and Scope of Graph Neural Networks. 2022, *arXiv*. DOI: 10.48550/arXiv.2201.07858.

14. Loukas, A. What graph neural networks cannot learn: depth vs width. 2020, *arXiv.* DOI: 10.48550/arXiv.1907.03199.

15. Meng, L., Ye, Z., & Yang Y. DeepMCGCN: Multi-channel Deep Graph Neural Networks, *International Journal of Computational Intelligence Systems*, 2024, vol. 17, iss. 1, article no. 41. DOI: 10.1007/s44196-024-00432-9.

16. Arroyo, Á., Gravina, A., & Gutteridge, B. On Vanishing Gradients, Over-Smoothing, and Over-Squashing in GNNs: Bridging Recurrent and Graph Learning. 2025, *arXiv*. DOI: 10.48550/arXiv.2502.10818.

17. Shu, J. Xi, B., Li, Y., & Wu, F. Understanding Dropout for Graph Neural Networks, in *Companion Proceedings of the Web Conference 2022*. *WWW 22: The ACM Web Conference 2022*, Virtual Event, Lyon France: ACM, 2022, pp. 1128–1138. DOI: 10.1145/3487553.3524725.

18. van Laarhoven, Tw. L2 Regularization versus Batch and Weight Normalization. 2017, *arXiv.* DOI: 10.48550/arXiv.1706.05350.

19. Liu, H., Fu, Q., & Du, L. Learning Rate Perturbation: A Generic Plugin of Learning Rate Schedule towards Flatter Local Minima. 2022, *arXiv.* DOI: 10.48550/arXiv.2208.11873.

20. Afaq, S., & Rao, S. Significance of Epochs on Training a Neural Network. *International Journal of Scientific & Technology Research*, 2020, vol. 9, pp. 485-488.

21. Hussein, B. M., & Shareef, S. M. An Empirical Study on the Correlation between Early Stopping Patience and Epochs in Deep Learning. *ITM Web of Conferences,* 2024, vol. 64, article no. 01003. DOI: 10.1051/itmconf/20246401003.

22. Zhang, J., He, T., & Sra, S. Why gradient clipping accelerates training: A theoretical justification for adaptivity. 2020, *arXiv.* DOI: 10.48550/arXiv.1905.11881.

23. Dickson, M., Bosman, A., & Malan, K. Hybridised loss functions for improved neural network generalization. *Pan-African Artificial Intelligence and Smart Systems.* 2022, pp. 169-181. DOI: 10.1007/978-3-030-93314-2_11.

24. Liu, C., Yu, S., & Yu, M. Adaptive Smooth L1 Loss: A Better Way to Regress Scene Texts with Extreme Aspect Ratios. *2021 IEEE Symposium on Computers and Communications (ISCC)*, 2021, Athens, Greece: IEEE, pp. 1–7. DOI: 10.1109/ISCC53001.2021.9631466.

25. Hodson, T. O., Over, T. M. & Foks, S. S. Mean Squared Error, Deconstructed. *Journal of Advances in Modeling Earth Systems*, 2021, vol. 13, iss. 12, article no. e2021MS002681. DOI: 10.1029/2021MS002681.

26. Qi, J., Du, J., & Siniscalchi, S. M. On Mean Absolute Error for Deep Neural Network Based Vector-to-Vector Regression. *IEEE Signal Processing Letters*, 2020, vol. 27, pp. 1485–1489. DOI: 10.1109/LSP.2020.3016837.

27. Terven, J., & Cordova-Esparza, D.-M. A comprehensive survey of loss functions and metrics in deep learning. *Artificial Intelligence Review*, 2025, vol. 58, iss. 7, article no. 195. DOI: 10.1007/s10462-025-11198-7.

28. Li, X., Wang, W., Wu, L., & Chen, Sh. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection. 2020, *arXiv.* DOI: 10.48550/arXiv.2006.04388.

29. Ferrentino, R. & Vota, L. The Risk in the Insurance Field: A Generalized Analysis. *Journal of Mathematical Finance*, 2020, vol. 10, iss. 01, pp. 200–221. DOI: 10.4236/jmf.2020.101013.

30. Yao, K.-L. & Li, W.-J. Asymmetric Learning for Graph Neural Network based Link Prediction, *ACM Transactions on Knowledge Discovery from Data*, 2024, vol. 18, iss. 5, pp. 1–18. DOI: 10.1145/3640347.

31. Parashar, S., Olson, B., & Khurana, S. Inference-Time Computations for LLM Reasoning and Planning: A Benchmark and Insights. 2025, *arXiv.* DOI: 10.48550/arXiv.2502.12521.

32. Bai, Y., Chen, M., & Zhou, P. How Important is the Train-Validation Split in Meta-Learning? 2021, *arXiv.* DOI: 10.48550/arXiv.2010.05843.

33. Wu, J., Sun, J., Sun, H., & Sun, G. Performance Analysis of Graph Neural Network Frameworks, in *2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. *2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Stony Brook, NY, USA: IEEE, 2021, pp. 118-127. DOI: 10.1109/ISPASS51385.2021.00029.

Received 15.05.2025, Accepted 25.08.2025

ОПТИМІЗАЦІЯ GRAPHSAGE МОДЕЛІ ДЛЯ ОЦІНКИ СТРАХОВИХ РИЗИКІВ: БАЛАНСУВАННЯ ПРОДУКТИВНОСТІ ТА ЕФЕКТИВНОСТІ

О. В. Луценко, С. С. Щербак

Предметом вивчення у статті є застосування та оптимізація графових нейронних мереж, зокрема архітектури GraphSAGE (Graph SAmple та aggreGatE), для оцінки страхових ризиків у нестабільних середовищах. **Метою** цього дослідження є розробка надійної та ефективної платформи на основі GraphSAGE для оцінки страхових ризиків, яка б збалансувала прогностичну продуктивність з обчислювальною ефективністю. **Завдання** включають: розробку та впровадження процесу генерації синтетичних графів, який точно відображає складність даних про страхові ризики; проведення систематичного дослідження архітектур GraphSAGE, змінюючи кількість шарів (2, 3, 4) та прихованих каналів (64, 128, 256); дослідження впливу різних швидкостей навчання (0,1, 0,01, 0,001) на збіжність та стабільність моделі; аналіз ефективності різних методів регуляризації, включаючи відсів (від 0,1 до 0,5) та спад ваги (від 1e-05 до 0,0001); Оцінка різних стратегій навчання, включаючи оптимальну кількість епох (від 100 до 300) та реалізацію ранньої зупинки; Оцінка ефективності різних функцій збитків при обробці викидів, поширених у страхових даних; Розробка структури порівняння для сприяння прийняттю обґрунтованих рішень при виборі моделі для завдань оцінки страхових ризиків. **Методи**, що використовуються в цьому дослідженні: використання експериментального підходу, використання геометричної бібліотеки PyTorch для реалізації моделей GraphSAGE, розгортання моделей та їх тестування на хмарній інфраструктурі, розробка власного алгоритму генерації графіків для створення реалістичних сценаріїв страхового ризику, включаючи такі фактори, як показники здоров'я, статус куріння та регулярні огляди, а також стратегія пошуку в сітці для оптимізації гіперпараметрів, у поєднанні з перехресною перевіркою, методами регуляризації для запобігання перенавчанню та використанням механізмів ранньої зупинки. **Результати** включають: створення синтетичного реалістичного графіка для опису застрахованих осіб, створення оптимізованої моделі GraphSAGE, адаптованої для завдання оцінки ризиків у страхуванні, тестування та інтеграція моделі на хмарній інфраструктурі. **Висновки**. Новизна результатів полягає в наступному: 1) дослідження демонструє, що відносно прості архітектури GraphSAGE, такі як двошарові моделі зі 128 прихованими каналами, можуть досягати продуктивності, порівнянної з більш складними моделями в завданнях оцінки страхових ризиків. Це свідчить про те, що властива структура даних страхових ризиків не завжди може вимагати глибоких, складних нейронних мереж для захоплення основних закономірностей. 2) дослідження підкреслює важливість адаптованих стратегій регуляризації, причому глибші моделі зазвичай вимагають сильнішої регуляризації для боротьби з перенавчанням. Дослідження динаміки навчання розкриває роль вибору швидкості навчання та стратегій ранньої зупинки, причому більш поверхневі моделі отримують вищу швидкість навчання, тоді як глибші архітектури вимагають більш консервативних швидкостей навчання для стабільної конвергенції. Послідовна продуктивність функції втрат Smooth L1 на різних архітектурах моделей демонструє її придатність для завдань оцінки страхових ризиків. 3) закладено основу для ефективного застосування моделей GraphSAGE в оцінці страхових ризиків, підкреслюючи важливість збалансованого підходу до проектування моделей, який враховує не лише прогностичну продуктивність, але й обчислювальну ефективність та практичні міркування розгортання.

Ключові слова: GNN; GraphSAGE; функція втрат; оцінка ризиків; хмарна інфраструктура.

Луценко Олександр Володимирович – асп. каф. інформаційних систем та мереж, Національний університет «Львівська Політехніка», Львів, Україна.

Щербак Сергій Сергійович – канд. техн. наук, доц., доц. каф. інформаційних систем та мереж, Національний університет «Львівська політехніка», Львів, Україна.

Oleksandr Lutsenko – PhD Student of the Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine,
e-mail: Oleksandr.V.Lutsenko@lpnu.ua, ORCID: 0009-0008-7644-6056.

Serhii Shcherbak – Candidate of Technical Sciences, Associate Professor at the Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine,
e-mail: Serhii.s.shcherbak@lpnu.ua, ORCID: 0000-0003-2914-2101, Scopus Author ID: 54397653400.