From Pixels to Graphs: Grid and Superpixel Graph Construction for Drone-Bird Image Classification Using Graph Neural Networks*

Zofia Wawrzos

Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, POLAND

Abstract

In this paper, we address the drone-bird classification problem using Graph Neural Networks (GNNs). The increasing use of unmanned aerial vehicles (UAVs) and the diversity of their applications create a growing need for their identification in the airspace. In addition to comparing the classification results between Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT), we investigated how the image segmentation method affects the final performance. We approached the problem by applying the SLIC algorithm, which generates an image composed of superpixels, and a segmentation method based on bilinear interpolation, which produces a grid-like image. The experimental results indicate that the GCN model achieves higher precision when using the SLIC algorithm, while the GAT network shows better classification accuracy for images downsampled using the bilinear interpolation method. Furthermore, the SLIC method is characterized by greater versatility in terms of the choice of network architecture.

Keywords

GCN, GAT, GNN, Superpixels, drones, birds

1. Introduction

In recent years, image classification using artificial intelligence (AI) has become a widely discussed topic in both literature and practice [1,2]. This is because it is one of the most practical applications of AI, allowing computers to automatically recognize and label image elements without the need for human supervision. As a result, this field is developing dynamically and is used in areas such as medicine [3,4], the automotive industry [5], and security systems [6].

Due to the popularity of this subject, many algorithms have been developed for AI-based image classification. Among classical machine learning methods one that stands out is Support Vector Machines [7]. They search for the optimal boundary, known as a hyperplane, to separate different data classes. In the context of neural networks, Convolutional Neural Networks (CNN) are frequently used [8], as they automatically learn to detect edges, shapes, and textures without the need for manual feature extraction. Another approach is the use of Graph Neural Networks (GNN) [9], which are adapted to the analysis of irregular data that can be generated from images. For this purpose, a regular grid or superpixels can be used. In the case of GNN, many different models are available, such as Graph Attention Networks (GAT) or Graph Convolutional Networks (GCN) [10, 11].

In this article, we compare the performance of Graph Neural Networks using two different graph constructions based on images for the classification of birds and drones. Such classification can be useful in contexts such as airspace monitoring for industrial or military areas, as well as for statistical data collection. The main contribution of this paper are:

- two image-based graph construction methods - SLIC superpixel graphs and regular GRID graphs and apply them to GNN for bird and drone classification,
- a comparative analysis of different superpixel sizes and grid resolutions,
- experimentally demonstrate how the choice and granularity of graph construction affect the ability of GNNS in the aerial classification task.

*IVUS 2026: Information Society and University Studies 2026, May 12, Kaunas, Lithuania

zw310289@student.polsl.pl

Copyright © 2026 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-06

CEUR-WS.org/Vol-4317/paper24.pdf

2. Image-to-Graph Representation

When developing Graph Neural Networks, the fundamental requirement is a set that represents the vertices of our graph. In a theoretical image classification problem, these vertices could consist of the individual pixels of the image, with the edges defined by pixel adjacency. However, from a practical standpoint, such a set would be too large for time- and resource-efficient training. Therefore, we must resort to simplifying the images to ensure the network's learning process is as efficient as possible. One such simplification method is downsampling the image using bilinear interpolation [12].

To compute bilinear interpolation, one applies a series of linear interpolations across both dimensions. This is achieved by first interpolating linearly with respect to the x -coordinate:

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}), \tag{1}$$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \tag{2}$$

where $R_1 = (x, y_1), R_2 = (x, y_2)$.
Now we repeat the procedure along the y-coordinate to arrive at the final value:

$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2). \tag{3}$$

By adopting a coordinate system where f is defined at the four points $\mathrm{Q}_{11}(0,0), \mathrm{Q}_{12}(0,1), \mathrm{Q}_{21}(1,0)$ and $\mathrm{Q}_{22}(1,1)$, the interpolation formula can be simplified to:

$$f(x, y) \approx f(0,0)(1 - x)(1 - y) + f(1,0)x(1 - y) + f(0,1)(1 - x)y + f(1,1)xy \tag{4}$$

or equivalently, in matrix operations:

$$f(x, y) \approx [1 - x, x] \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1 - y \\ y \end{bmatrix} \tag{5}$$

The output is an image on a grid plan with sharp edges. This method allows you to choose the output resolution or downscale the image by a specific factor.

Another approach to simplifying images is to use the SLIC algorithm, which merges pixels that are spatially close and possess similar color values into a single entity known as a superpixel with an "averaged" color value. These superpixels are generated using the Simple Linear Iterative Clustering (SLIC) algorithm [13, 14], which operates within the labxy space. In this context, [lab] represents the color vector in the CIELAB color space, while xy denotes the pixel's coordinates on the image map.

To use this algorithm, the distances must be normalized because distances in the CIELAB space are bounded, unlike those in Euclidean space.

Let N be the number of pixels in the input image. We define a variable K as the number of superpixels forming the simplified image. Consequently, the approximate number of pixels per superpixel can be denoted as $\frac{N}{K}$.

Thus, we obtain K superpixel centroids $C_k = [L_k, a_k, b_k, x_k, y_k]^T, k \in [1, K]$, which are spaced at an interval of $S = \sqrt{\frac{N}{K}}$ (assuming the superpixels are of similar size). We can therefore establish that the area occupied by a single superpixel is approximately S^2, and the pixels belonging to a given superpixel will be located within a $2S \times 2S$ region around its centroid in Euclidean space.

We then define the normalized distance D_S as

$$D_s = d_{Lab} + \frac{m}{S} d_{xy} \tag{6}$$

where

$$d_{Lab} = \sqrt{(L_k - L_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}, \tag{7}$$

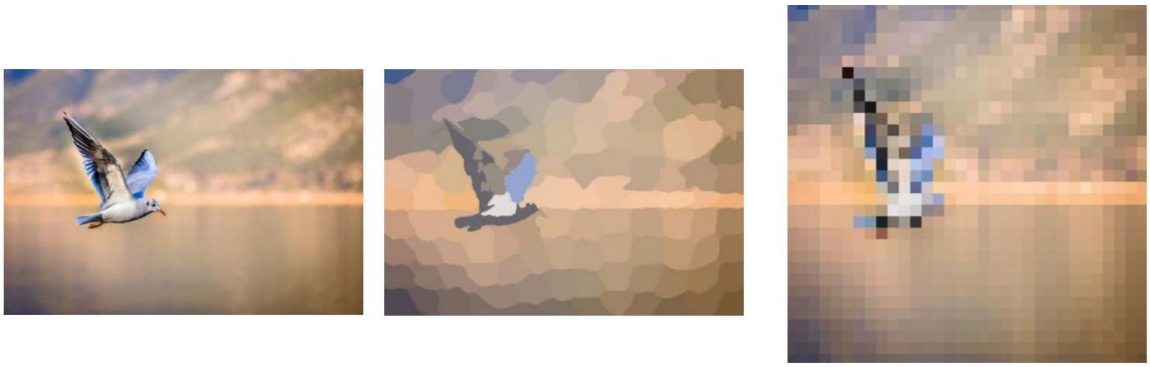
represents the distance in the CIELAB space,

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}, \tag{8}$$

represents the distance in Euclidean space, $m \in [1,20]$ is a variable responsible for the compactness of the superpixel (the higher the value of m, the more compact the set becomes).

Regardless of the chosen algorithm, the final result is a simplified image where the number of superpixels is small enough to serve as the vertices of a graph. Connections between these vertices are established based on the adjacency of the superpixels. In the case of the grid method, adjacency is defined by the superpixel's position on the simplified image map. Conversely, when using the SLIC algorithm, we must manually define the criteria for establishing connections between the resulting superpixels. This allows us to select the appropriate number of neighbors for each superpixel according to the specific problem and to define the metric used to calculate the distances between them.

In this article, we assume that the adjacency between superpixels generated by the SLIC algorithm will be determined using the K-Nearest Neighbors (KNN) algorithm [15]. To measure the distance between superpixel centroids, we will employ the Euclidean metric.



(a) Original image (b) SLIC image (c) Grid image

Figure 1: Different image representation methods.

3. Graph Neural Networks

Graph Neural Networks (GNNs) are designed to solve problems that naturally adopt a graph structure. A graph is a mathematical construct that defines relationships between various objects. For instance, if we represent bus stops as points on a map and bus routes as lines connecting them, we obtain a graph where the vertices represent the stops and the lines form the edges. Since an image grid consisting of pixels can also be represented as a graph, GNNs can be effectively utilized for image classification tasks.

Before the graph can be used for training, it needs to be transformed into an appropriate structure. The most straightforward way to represent a graph for neural network processing is by creating an adjacency matrix. However, this form of representation only captures the connectivity between vertices, often ignoring critical information such as vertex and edge features or the global context of

the graph. Furthermore, particularly for graphs with a large number of vertices, an adjacency matrix is not an optimal method for storing structural information. Since the degree of each vertex can vary significantly, there is a high probability that the resulting matrix will be sparse (containing mostly zeros), making it highly space-inefficient. Moreover, a single graph problem can be represented by multiple different adjacency matrices, which could lead to inconsistent results in machine learning tasks.

For these reasons, an adjacency list is often a superior choice for representing graph structures. This method represents a connection between vertices n_i and n_j via an edge e_k as a tuple (i, j) at the k-th position of the list. Since we expect the number of edges in a graph to be significantly lower than the number of elements in an adjacency matrix $\left(n_{\text{nodes}}^2\right)$, we eliminate the need to store information regarding elements that are not connected.

The generated adjacency list is then fed into the GNN, where it is processed using a mechanism known as message passing. This process facilitates network learning by exchanging information between adjacent vertices. Consequently, the classified images can be analyzed within a global context, rather than being restricted to the local pixel neighborhoods typical of convolutional neural networks. To implement message passing, we calculate the mean of the features from neighboring neurons and subsequently normalize this value using the following formula:

$$\frac{\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} \tag{9}$$

Subsequently, we update all nodes using the calculated values:

$$h_v^{(k)} = \sigma\left(W^{(k)} \cdot \frac{\sum_{u \in \mathcal{N}(v)} h_u^{(k-1)}}{|\mathcal{N}(v)|} + B^{(k)} \cdot h_v^{(k-1)}\right), \tag{10}$$

where $W^{(k)}$ is the weight matrix, in which weights determine the importance of a given feature, $B^{(k)}$ is the bias, $\sigma(\cdot)$ is a non-linear activation function, which enables the model to handle non-linear problems.

Over the years, many different types of GNNs have been developed. In this article, we will take a closer look at Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT).

In a GNN, each node in a graph can have a different degree (meaning a different number of neighbors), which may lead to excessively large values in subsequent layers. To address this issue, GCNs employ normalization, and the information propagation operation in the k-th layer of the network can be written in matrix form:

$$H^{(k)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}(A + I_N)\tilde{D}^{-\frac{1}{2}}H^{(k-1)}W^{(k-1)}\right), \tag{11}$$

where $\tilde{D}$ denotes the vertex degree matrix, $W^{(k)}$ is the weight matrix of the learned parameters, $\sigma(\cdot)$ is the activation function, A is the adjacency matrix and I_N is the identity matrix.

In GAT, aggregation of information in the neighborhood is enhanced by the attention mechanism. For this calculation, we introduce an attention coefficient for a pair of neighboring vertices:

$$e_{vu} = \text{LeakyReLU}(\mathbf{a}^T[Wh_v \| Wh_u]) \tag{12}$$

that is normalized as:

$$\alpha_{vu} = \frac{\exp\left(e_{vu}\right)}{\sum_{k \in \mathcal{N}(v)} \exp\left(e_{vk}\right)}. \tag{13}$$

The updated representation of the vertex is defined as follows:

$$h_v^{(k)} = \sigma\left(\sum_{u\in\mathcal{N}(v)} \alpha_{vu} W^{(k)} h_u^{(k-1)}\right). \tag{14}$$

For the purposes of further analysis, two graph network architectures were modeled - with GAT and GCN (see Tab. 1).

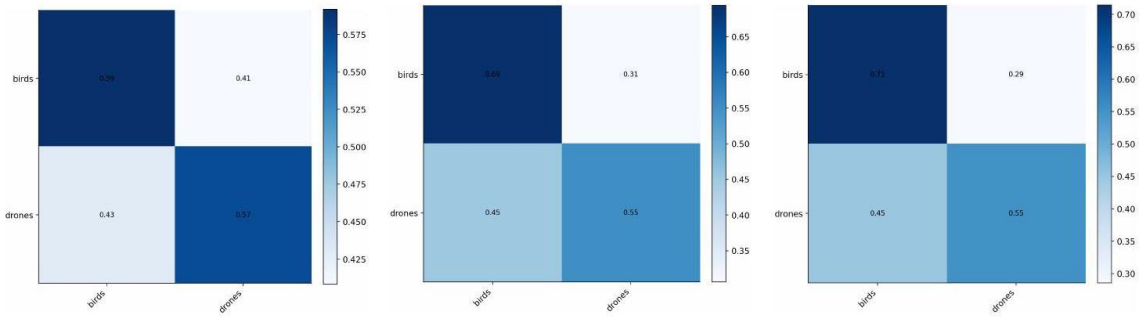
Table 1
Proposed architectures of GCN and GAT

GCN		GAT	
Layer	Description	Layer	Description
Input	Node feature matrix	Input	Node feature matrix
GCNConv 1	Graph convolution	GATConv 1	Graph attention layer (1 head)
ReLU	Non-linear activation	ReLU	Non-linear activation
GCNConv 2	Graph convolution	GATConv 2	Graph attention layer (2 heads)
ReLU	Non-linear activation	ReLU	Non-linear activation
GCNConv 3	Graph convolution	Global Mean Pool	Node-to-graph aggregation
ReLU	Non-linear activation	Linear 1	Fully connected layer
Global Mean Pool	Node-to-graph aggregation	ReLU	Non-linear activation
Linear 1	Fully connected layer	Linear 2	Class prediction
ReLU	Non-linear activation		
Linear 2	Class prediction		

4. Experiments

For our experiments, we relied on the data provided by the website https://www.kaggle.com/code/stpeteishii/drone-bird-classify-fasiai/notebook. The dataset comprises 165 images of birds and drones each. These were captured in various weather conditions, at different times of day, and from multiple perspectives to best simulate the natural observation of birds and drones.

We will now compare the results of the experiments conducted for the GCN and GAT network architectures described in Table 1. We will focus primarily on the outcomes of experiments using both the bilinear interpolation and SLIC algorithm segmentation methods. Subsequently, we will examine how the choice of segmentation method affects the performance of each specific architecture.



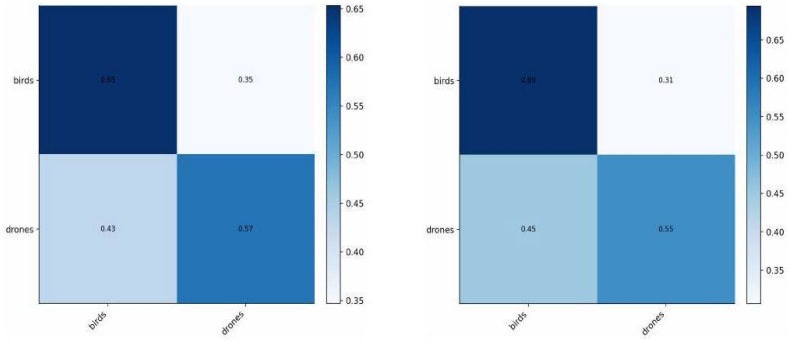
(a) 60 superpixels (b) 120 superpixels (c) 240 superpixels

Figure 2: Comparison of GCN performance for different numbers of superpixels.

Figure 2 above shows a comparison of three normalized confusion matrices for an experiment conducted using a GCN network with SLIC segmentation and the parameter $K = 8$, where K represents the number of neighbors for each superpixel. The X-axes of the presented matrices represent the network's predictions for each image, while the Y-axes indicate the ground truth. The color scale shows the percentage of responses for the bird and drone classes. The experiments differed in the number of superpixels per image.

Considering both the bird and drone classes, we can observe that the best overall results were achieved by the model trained on images containing 240 superpixels (Fig. 2c). Nevertheless, matrix 2b yielded very similar performance, in contrast to matrix 2a.

We can also compare how the number of neighbors, K, for each superpixel affects the network's performance (Fig. 3). Below is a comparison of the results, presented as normalized confusion matrices for 120 superpixels; however, it indicates that the value of the parameter K has a very minimal impact on the network's results.



(a) $k = 4$ (b) $k = 8$

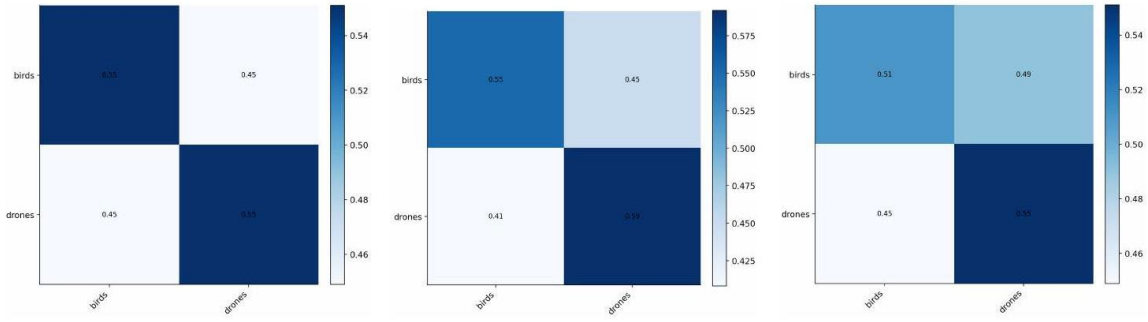
Figure 3: Comparison of GCN performance for different values of K parameter.

Let us now move on to comparing how the resolution of images simplified via bilinear interpolation affects GCN training. Figure 4 compares the results for three networks, showing that the best performance was achieved by the model trained on 16×16 images. However, it is worth noting that, in general, GCNs trained on images simplified through bilinear interpolation performed worse than those based on images segmented with the SLIC algorithm.

Experiments conducted for the bilinear interpolation downsampling method on GAT networks show significantly higher drone classification results than on the GCN network. This network achieved very similar results for resolutions of 16×16 and 32×32, which in turn significantly outperform the results for the 8×8 resolution.

Table 2 shows a comparison of the performance of GCN and GAT networks for the segmentation algorithm using the bilinear interpolation method (resolution: 16×16) and the SLIC algorithm (120 superpixels and 8 neighbours for each superpixel). The Precision columns specify what proportion of samples assigned to a given class is correct, the Recall columns inform us of what proportion of actual examples of a given class was correctly detected, while the F1-score columns are equal to the harmonic mean of Precision and Recall, representing a compromise between them.

The results in the indicated table clearly show that the GCN network best classified the



(a) 8×8 (b) 16×16 (c) 32×32

Figure 4: Comparison of GCN performance for different image resolutions.

images processed by the SLIC algorithm, whereas the GAT network performed better with the grid-structured images.

If we consider the choice of architecture in terms of detecting the highest number of drones (e.g., in aerial defense systems), the GAT network would be a significantly better choice. Although the Precision for drones is only about 58% regardless of the image segmentation type, they achieve much higher Recall values than GCN networks, while maintaining a high F1-score. On the other hand, GCN networks using the SLIC segmentation algorithm are much better suited for the bird recognition problem, achieving an F1-score nearly 10% higher than all other methods and a Recall value close to 70%.

Considering all metrics, the best choices would be GCN networks trained on SLIC-processed images or GAT with bilinear interpolation, depending on whether the primary focus is bird or drone detection.

Table 2
Comparison of GCN and GAT classification results including the SLIC method and bilinear interpolation.

	GCN			GAT		
Metric	Precision	Recall	F1-score	Precision	Recall	F1-score
Grid						
birds	0.5745	0.5510	0.5625	0.6667	0.4490	0.5366
drones	0.5686	0.5918	0.5800	0.5846	0.7755	0.6667
accuracy			**0.5714**			**0.6122**
macro avg	0.5715	0.5714	0.5713	0.6256	0.6122	0.6016
weighted avg	0.5715	0.5714	0.5713	0.6256	0.6122	0.6016
SLIC						
birds	0.6071	0.6939	0.6476	0.6389	0.4694	0.5412
drones	0.6429	0.5510	0.5934	0.5806	0.7347	0.6486
accuracy			**0.6224**			**0.6020**
macro avg	0.6250	0.6224	0.6205	0.6098	0.6020	0.5949
weighted avg	0.6250	0.6224	0.6205	0.6098	0.6020	0.5949

5. Conclusions

As can be inferred from the conducted experiments, the choice of architecture should depend on the specific problem at hand. In situations where the main focus is on drone detection-for instance, in the protection of areas such as airports or military zones, or in the monitoring of mass events and private properties - the best choice is to use the GAT network trained on images downsampled using bilinear interpolation. Conversely, if the focus is more on bird detection - for example, for gathering statistical data or detecting birds flying over an airport-the most effective option is to employ the GCN network utilizing images segmented with the SLIC algorithm.

Regarding the comparison of the SLIC algorithm to the bilinear interpolation method, we are unable to definitively state which of these two image segmentation methods generates overall better results. Based on the data in Table 2, it can be inferred that the SLIC algorithm is more versatile than the Grid method; however, this does not mean that it is always the best choice.

As experiments using the SLIC algorithm indicate, a larger number of superpixels in the image positively affects the network's learning process. At the same time, the number of neighbors for each superpixel does not have a significant impact on learning quality, which may suggest that a more accurate representation of elements within the image layout is more important than the

neighborhood relationships between them. Analyzing the impact of image resolution on the learning process, it can be observed that the network achieves better results for images with higher resolutions (e.g., 16×16 or 32×32). However, beyond a certain level, increasing the resolution does not lead to a significant improvement in performance.

6. Declaration on Generative AI

The authors have not employed any Generative AI tools.

7. References

[1] M. E. Fathy, S. A. Mohamed, M. I. Awad, H. E. Abd El Munim, A vision transformer based cnn for underwater image enhancement vitclaritynet, Scientific Reports 15 (2025) 16768.

[2] A. Jaszcz, D. Połap, End-to-end food condition system via wavelet attention enhanced network, Neurocomputing (2025) 131248.

[3] J. Li, Q. Xu, X. He, Z. Liu, D. Zhang, R. Wang, R. Qu, G. Qiu, Cfformer: Cross cnntransformer channel attention and spatial feature fusion for improved segmentation of heterogeneous medical images, Expert Systems with Applications 295 (2026) 128835.

[4] K. Prokop, Feature fusion using deep learning modules and fuzzy c-means for medical disease recognition, in: 2025 IEEE International Conference on Fuzzy Systems (FUZZ), IEEE, 2025, pp. 1-6.

[5] A. Medina, P. Ponce, R. Ramirez-Mendoza, Automotive embedded image classification systems, in: 2022 International Symposium on Electromobility (ISEM), IEEE, 2022.

[6] A. Jaszcz, D. Połap, G. Srivastava, Multimoora-guided federated learning with kd stabilization for health monitoring devices as protection against poisoning attacks, IEEE Internet of Things Journal (2025).

[7] K. K. Ray, A. Kumari, S. Kumar, R. Machavaram, I. Shekh, S. M. Deshmukh, P. Tadge, Guava leaf disease detection using support vector machine (svm), Smart Agricultural Technology (2025) 101190.

[8] P. Żerdziński, Fantastic fishes and how to classify them, in: International Conference on Information and Software Technologies, Springer, 2024, pp. 26-37.

[9] M. Songailaitė, J. Mandravickaitė, V. Bryskina, M. Bondar, T. Krilavičius, A graph network approach to disinformation detection in social media, in: International Workshop on Modelling and Mining Networks, Springer, 2025, pp. 141-156.

[10] A. Yılmaz, R. Das, A novel hybrid approach combining gcn and gat for effective anomaly detection from firewall logs in campus networks, Computer Networks 259 (2025) 111082.

[11] Y. Huo, M. He, X. Tan, K. Chen, Gcn and gat-based interpretable knowledge tracing model, Complex & Intelligent Systems 11 (2025) 300.

[12] P. S. Parsania, P. V. Virparia, A comparative analysis of image interpolation algorithms, International Journal of Advanced Research in Computer and Communication Engineering 5 (2016) 29-34.

[13] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, et al., Slic superpixels, Technical Report, Technical report EPFL, 2010.

[14] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, Slic superpixels compared to state-of-the-art superpixel methods, IEEE transactions on pattern analysis and machine intelligence 34 (2012) 2274-2282.

[15] H. Bojda, D. Gala, Analysis the performance of naive bayes and k-nearest neighbor classifiers, in: CEUR Workshop Proc, volume 3885, 2024, pp. 366-373.